A Two-Phase Test Sample Sparse Representation Method for Use With Face Recognition

Yong Xu, *Member, IEEE,* David Zhang, *Fellow, IEEE,* Jian Yang, and Jing-Yu Yang

***Abstract*—In this paper, we propose a two-phase test sample representation method for face recognition. The first phase of the proposed method seeks to represent the test sample as a linear combination of all the training samples and exploits the representation ability of each training sample to determine M "nearest neighbors" for the test sample. The second phase represents the test sample as a linear combination of the determined M nearest neighbors and uses the representation result to perform classification. We propose this method with the following assumption: the test sample and its some neighbors are probably from the same class. Thus, we use the first phase to detect the training samples that are far from the test sample and assume that these samples have no effects on the ultimate classification decision. This is helpful to accurately classify the test sample. We will also show the probability explanation of the proposed method. A number of face recognition experiments show that our method performs very well.**

***Index Terms*—Computer vision, face recognition, pattern recognition, sparse representation, transform methods.**

I. Introduction

TRANSFORM methods, such as linear [1]–[5] and nonlinear [6]–[14] methods, have been widely used in face recognition. The principal component analysis (PCA) [1]–[3] and the linear discriminant analysis (LDA) [4], [5] are two typical examples of linear transform methods. The widely used kernel PCA [6]–[9] and kernel LDA [10]–[12] are two typical examples of nonlinear transform methods. When transforming samples into a new lower-dimensional space, different transform methods have different goals. For example, the PCA transforms samples into a space where samples have the maximum variance, whereas the LDA transforms samples into a space where the centers of different classes have the maximum distances. Transform methods usually use the entire set of training samples to obtain transform axes and then project each test and training sample onto the transform axes to produce a representation for the sample. They then compute the distance between the representation of the test sample and that of the training sample, and exploit the distance and a classifier to classify the test sample.

Local transform methods that have very different motivations in comparison with conventional transform methods have also been proposed. The local transform methods exploit local training samples rather than all of the training samples to produce transform axes. For example, Sugiyama integrated the ideas of the LDA and locality preserving projection to generate a local transform method for a problem where the samples in a class are multimodal [13]. Vural proposed to use the local dependencies of samples for classification [14]. Liu *et al.* showed that a local PCA was preferable to a global PCA for feature extraction [15]. Other attempts to use the relationship between samples in a local region can be found in [16]–[20]. As local and conventional transform methods all first depend on a transform process to produce a representation for the training and test samples, and then use a classifier to classify the test sample, we refer to them as transform-based sample representation methods.

Recently, a method that addresses classification problems from a novel viewpoint has been proposed. This method addresses classification problems by evaluating representation ability on a test sample of each class. For example, the method proposed in [21] imposes a sparse constraint on local discriminant projections, and requires that the test sample be sparsely represented by the training samples. The methods in [22]–[28] first use a sparse linear combination of the training samples to represent the test sample. "Sparse" and "sparsely" imply as follows: when a method uses a linear combination of all the training samples to represent the test sample, the coefficients of some training samples are equal to zero. These methods then calculate the contribution, in representing the test sample, of the training samples of each class. They ultimately classify the test sample into the class that has the greatest contribution. These methods exhibit good performance in some pattern recognition problems such as face recognition [21]–[28]. However, it is not very clearly known why the sparse representation method is able to obtain good performance and what makes up the underlying theoretical foundation.

Manuscript received July 13, 2010; revised February 14, 2011; accepted March 18, 2011. Date of publication April 7, 2011; date of current version September 2, 2011. This work was supported in part by the Key Laboratory of Network Oriented Intelligent Computation, the Program for New Century Excellent Talents in University, under Grant NCET-08-0156, the National Nature Science Committee of China, under Grants 61071179, 60973098, 60803090, 60902099, and 60975006, the Fundamental Research Funds for the Central Universities (HIT.NSRIF. 2009130), as well as the CERG Fund from the Hong Kong Special Administrative Region Government and the Central Fund from Hong Kong Polytechnic University. This paper was recommended by Associate Editor F. Lavagetto.

Y. Xu is with the Bio-Computing Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: laterfall2@yahoo.com.cn).

D. Zhang is with the Department of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: csdzhang@comp.polyu.edu.hk).

J. Yang and J.-Y. Yang are with the School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: csjyang@mail.njust.edu.cn; yangjy@mai.njust.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2011.2138790

1051-8215/$26.00 © 2011 IEEE

In this paper, we propose a two-phase test sample representation (TPTSR) method for face recognition. The first phase of this method represents the test sample as a linear combination of all the training samples, and exploits the representation ability of each training sample to determine the M nearest neighbors for the test sample. The second phase represents the test sample as a linear combination of all the M nearest neighbors and uses the representation result to perform classification. The proposed method has the following rationales: its first phase identifies a number of training samples that are the most similar to the test sample and takes the class labels of the identified training samples as candidates for the class label of the test sample. As the class labels of the identified training samples are usually a subset of those of all the training samples, ultimate classification becomes a problem of determining the class label from a smaller number of candidates. Under the conditions where the genuine class label of the test sample is really one of those of the identified training samples, this will be very helpful for the second phase to perform an accurate classification. In this paper, we will also show the probability explanation of the proposed method. The experimental results show that this method achieves very good performance in face recognition problems. Our work has the following contributions: first, it proposes, for the first time, the idea and method of two-phase sample representation. Second, it clearly presents the rationales and theoretical foundations of the proposed method. Third, it provides a very large number of experiments and the experimental results show that the proposed method is very competitive.

The remainder of this paper is organized as follows. In Section II, we describe our proposed method. Section III provides an analysis of our method. Section IV presents the experimental results. Finally, Section V offers our conclusion.

II. Two-Phase Test Sample Representation (TPTSR) Method

In the section, we will present the details of the proposed TPTSR method. We assume that there are L classes and n training samples, $x_1 ... x_n$. If a training sample is from the jth class ($j = 1, 2, ..., L$), we take j as the class label of this training sample.

A. The First Phase of the TPTSR

The first phase of the TPTSR uses all of the training samples to represent each test sample and exploits the representation result to identify the M nearest neighbors of the test sample from the set of the training samples. It first assumes that the following equation is approximately satisfied:

$$y = a_1 x_1 + ... + a_n x_n \tag{1}$$

where y is the test sample and a_i ($i = 1, 2, ..., n$) are the coefficients. We can rewrite (1) into the following equation:

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

where $A = [a_1 ... a_n]^T$, $X = [x_1 ... x_n]$. $x_1 ... x_n$ and y are all column vectors. If X is a nonsingular square matrix, we can solve A by using $A = X^{-1}y$; otherwise, we can solve it by using $A = (X^T X + \mu I)^{-1} X^T y$, where μ is a small positive constant and I is the identity matrix.

Equation (1) shows that every training sample makes its own contribution to representing the test sample. The contribution that the ith training sample makes is $a_i x_i$. The contribution, in representing the test sample, of the ith training sample x_i can be also evaluated by the deviation between $a_i x_i$ and y, i.e., $e_i = ||y - a_i x_i||^2$. e_i can also be somewhat viewed as a measurement of the distance between the test sample and the ith training sample. We consider that a small e_i means that the ith training sample has a great contribution in representing the test sample. We exploit e_i to identify the M training samples that have the M greatest contributions, in representing the test sample and denote them by $x_1, \ldots, x_M$. We refer to these samples as the M nearest neighbors of the test sample. Let $C = \{c_1, c_2, ..., c_d\}$, a set of some numbers, stand for the set of class labels of the M nearest neighbors. If a nearest neighbor is from the jth class ($j = 1, 2, ..., L$), we take j as the class label of this nearest neighbor. C must be one subset of set $\{1, 2, ..., L\}$, i.e., $C \subseteq \{1, 2, ..., L\}$. If no neighbor is from the pth class, then the number p must be not an element of C. Consequently, the TPTSR will not ultimately classify the test sample into the pth class.

B. The Second Phase of the TPTSR

The second phase of the TPTSR seeks to represent the test sample as a linear combination of the determined M nearest neighbors and uses the representation result to classify the test sample. This phase assumes that the following equation is approximately satisfied:

$$y = b_1 \tilde{x}_1 + ... + b_M \tilde{x}_M \tag{3}$$

where $\tilde{x}_i$ ($i = 1, 2, ..., M$) are the identified M nearest neighbors and b_i ($i = 1, 2, ..., M$) are the coefficients. We rewrite (3) into

$$y = \tilde{X} B \tag{4}$$

where $B = [b_1 ... b_M]^T$, $\tilde{X} = [\tilde{x}_1 ... \tilde{x}_M]$. If $\tilde{X}$ is a nonsingular square matrix, we can solve B by using $B = (\tilde{X})^{-1} y$; otherwise, we can solve it by using

$$B = (\tilde{X}^T \tilde{X} + \gamma I)^{-1} \tilde{X}^T y \tag{5}$$

where γ is a positive constant and I is the identity matrix. After we obtain B, we refer to $\tilde{X}B$ as the representation result of our method. We can convert the representation result into a 2-D image with the same size as the original sample image.

Since the neighbors might be from different classes, we calculate the sum of the contribution to represent the test sample of the neighbors from each class and exploit the sum to classify the test sample. For example, if all the neighbors from the rth ($r \in C$) class are $\tilde{x}_s \ \ldots \ \tilde{x}_t$, then the sum of the contribution to represent the test sample of the rth class will be

$$g_r = b_s \tilde{x}_s + ... + b_t \tilde{x}_t. \tag{6}$$

We calculate the deviation of g_r from y by using

$$D_r = ||y - g_r||^2, r \in C. \quad (7)$$

We can also convert g_r into a 2-D matrix with the same size as the original sample image. When we do so, we also refer to this matrix as a 2-D image that stands for the contribution of the rth class. A smaller deviation D_r means a greater contribution to representing the test sample. Thus, we classify y into the class that produces the smallest deviation. In summary, the main steps of the TPTSR are as follows.

Step 1. Use the first phase to determine the M nearest neighbors for the test sample.

Step 2. Exploit the M nearest neighbors of the test sample to construct (3) and solve this equation.

Step 3. Use (7) to compute deviation D_r which is generated from the rth class, $r \in C$.

Step 4. Classify the test sample into the class that has minimum deviation. In other words, if $D_q = \min D_r(q, r \in C)$, the test sample will be classified into the qth class.

Using the second phase, the TPTSR determines the contribution in representing the test sample of different M nearest neighbors and classifies the test sample into the class that makes the largest contribution. The contribution of one nearest neighbor is mainly associated with the corresponding coefficient. If the ith nearest neighbor has a large similarity with the test sample, then the corresponding coefficient probably has a large absolute value. As a result, the ith nearest neighbor probably also makes a large contribution to representing the test sample.

III. Analysis of Our Method

This section analyzes the ideas and rationale of the TPTSR and compares it with other methods.

A. Methodology Comparison Between the TPTSR and Other Methods

In this section, we first present the sparse representation method proposed in [22] and [23]. This method obtains promising performances in face recognition. As presented earlier, the method in [22] and [23] proposes to represent the test sample as a sparse linear combination of all the training samples. We note that apart from classification problems, the sparse representation method can be also applied to clustering [27] and motion segmentation problems [28]. In [22]–[28], the extent of sparseness is evaluated by the l_1 norm of the coefficient vector of the linear combination and it is considered that a smaller l_1 norm means a higher extent of "sparseness." The method in [22] and [23] uses a multiobjective programming algorithm to obtain its solution and has a very high time complexity. In addition, it is almost impossible for the algorithm to simultaneously achieve minimum deviation and l_1 norm.

The TPTSR can be also regarded as a supervised sparse representation method. The sparseness is reflected by the following fact: when we rewrite the linear combination of the second phase of the TPTSR as a linear combination of all the training samples, the training sample that is not one of the M nearest neighbors indeed has a coefficient of zero. "Supervised" means that the TPTSR uses the first phase to produce the sparseness, and the sparse extent of the coefficients (i.e., the number of zero coefficients) and which coefficients of the linear combination are zero are known. However, as for the sparse representation method proposed in [22] and [23], it is not known which coefficients of the linear combination are equal or close to zero. We refer to this method as the unsupervised sparse method.

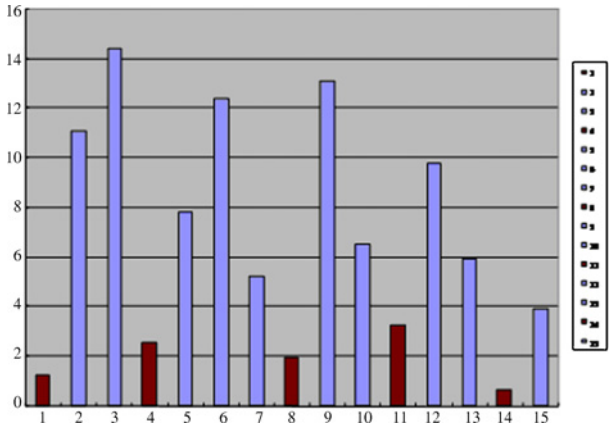

Fig. 1. Possible distribution of $p(T_i|y), i = 1, ..., n$. The horizontal coordinate shows the number of training samples and the vertical coordinate shows the probability $p(T_i|y)(\%)$. A brown bar denotes a small $p(T_i|y)$.

The TPTSR has both differences and similarities with local transform methods. The TPTSR and local transform methods are similar in that they only use a subset of the training samples. Local transform methods such as the local Fisher discriminant analysis proposed in [13] are able to exploit the local structure of patterns to make a good classification decision. When the TPTSR uses only the M nearest neighbors to represent and classify the test sample, it has the following underlying rationale: the test sample and its nearest neighbors usually belong to the same class. It seems that the k nearest neighbor classifier has a similar idea; however, this classifier exploits the neighbors of the test sample too coarsely (it simply counts the number of class labels of the nearest neighbors and classifies the test sample into the class that contains the most neighbors) and usually produces a bad classification result when k has a large value. When the TPTSR uses a linear combination of the M nearest neighbors to represent and classify the test sample, it specifically takes the relationship between the test sample and neighbors into account by setting different coefficients for different neighbors.

B. Probability Explanation

The classification rule of our method can be described as follows: let $c_1, \ldots, c_L$ denote the L classes, respectively. The first phase of the TPTSR can be explained as a phase that exploits e_i to evaluate the probability that the test sample y and the ith training sample are from the same class. Let $p(T_i|y)$ denote this probability. T_i stands for the event that y and the ith training sample are from the same class. The first phase of the TPTSR indeed assumes that $p(T_i|y) \propto e'_{\max} - e_i$, where $e'_{\max}$ is

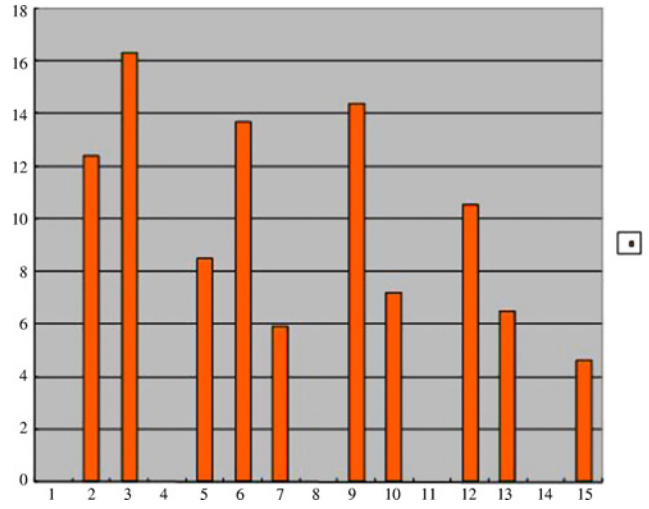

Fig. 2. Possible modification of $p(T_i|y)$, $i = 1, ..., n$, shown in Fig. 1, made by the second phase of the TPTSR. The $p(T_i|y)$ that corresponds to the brown bars in Fig. 1 has been set to zero, whereas the other $p(T_i|y)$ has been increased.

used to denote a number greater than the maximum value of $e_i = ||y - a_i x_i||^2$, $i = 1, ..., n$. Fig. 1 visually shows that $p(T_i|y)$ might severely vary with i. If a training sample has a small $p(T_i|y)$ (shown by the brown bar), the TPTSR will regard that it has no effect on the ultimate classification of the test sample. Actually, the TPTSR does not take this kind of training sample into account and exploits only the remaining training samples to implement the second phase. This implies that the TPTSR reassigns zero to $p(T_i|y)$ for this kind of training sample. As the value of $\sum_i p(T_i|y)$ is not variable, the TPTSR indeed implicitly converts Fig. 1 into Fig. 2, in which the $p(T_i|y)$ that originally corresponds to the brown bars in Fig. 1 is set to zero, whereas the other $p(T_i|y)$ is increased. Fig. 2 can be viewed as the probability explanation of the fact that the second phase of the TPTSR neglects some training samples that are far from the testing sample and exploits only the remaining training samples to represent the testing sample.

The second phase of the TPTSR assumes that the probability of test sample y being from class c_i, is directly related to $||y - g_i||^2$, that is

$$p(c_i|y) \propto D_{\max} - ||y - g_i||^2, c_i \in C \tag{8}$$

where $D_{\max}$ stands for a number greater than the maximum value of $||y - g_i||^2$. A smaller $||y - g_i||^2$ means a greater posterior probability $p(c_i|y)$. The classification rule of the TPTSR indeed classifies the test sample into the class that has the greatest posterior probability. It is known that $C \subset \{1, 2, ..., L\}$. Thus, if c, is an element of $\{1, 2, ..., L\}$ but not an element of C, it is impossible for TPTSR to classify the test sample into the c_jth class. Actually, in this case, the TPTSR assumes that $p(c_j|y) = 0$. In other words, it is assumed that if one class does not contain any of the M nearest neighbors of the test sample, this class should have a posterior probability of zero, i.e., $p(c_j|y) = 0$.

In contrast to the TPTSR, its global version works in a different way. It uses all the training samples to represent

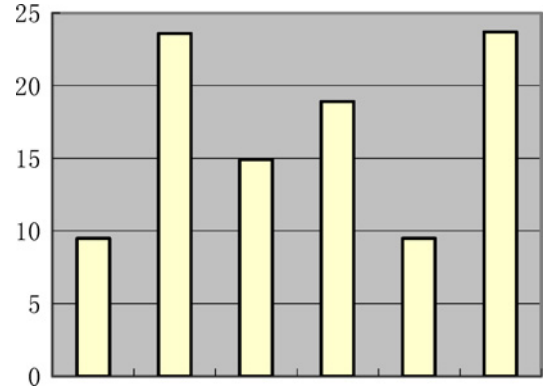

Fig. 3. Possible distribution of the posterior probability, determined by the global version of the TPTSR. The horizontal coordinate shows there are six classes and the vertical coordinate shows the posterior probability (%). The posterior probability ($p(c_j|y)$) shown in this figure is a consequence of $p(T_i|y)$ shown in Fig. 1.

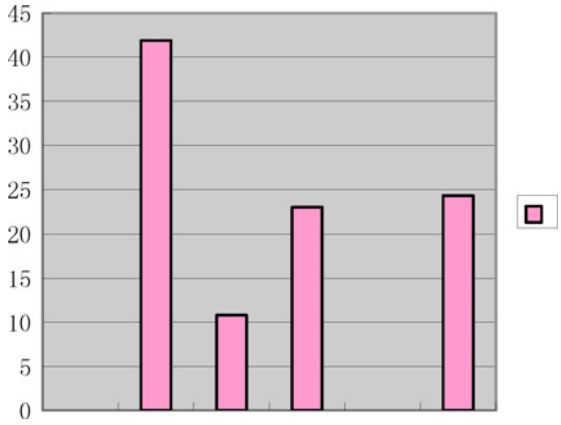

Fig. 4. Possible distribution of the posterior probability determined by the TPTSR. The horizontal coordinate shows there are six classes and the vertical coordinate shows the posterior probability (%). The posterior probability ($p(c_j|y)$) shown in this figure is a consequence of $p(T_i|y)$ shown in Fig. 2.

and classify the test sample. As a result, it seems that every $p(c_j|y)$, $j = 1, ..., L$ has a non-zero value. The global version of the TPTSR should identify the maximum posterior probability from all of the L posterior probabilities and classify the test sample into the corresponding class.

As the TPTSR sets some posterior probabilities to zero, it also increases the others. As a result, the TPTSR probably produces a sharper distribution of the posterior probability than its global version. This is shown in Figs. 3 and 4. We suppose that in Figs. 3 and 4, the test sample is true from the second class. It is possible that due to the interference of the noise, the global version of the TPTSR erroneously classifies the test sample into the sixth class. However, the TPTSR probably correctly classifies the test sample into the second class with a very high posterior probability. Actually, the TPTSR has the following advantage: in a complex case where there is interference caused by noise, the TPTSR is able to achieve a higher accuracy than its global version. As a result, the TPTSR provides us with a good way to deal with complex classification problems, such as face recognition (as there are variations in lighting, facial expression, and pose).

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

Fig. 6. Some face images from the ORL database.

IV. Experimental Results

We conducted a number of experiments on the ORL [30], Feret [31], [32], and AR [33] face databases. The ORL database includes 400 face images taken from 40 subjects, with each subject providing 10 face images. For some subjects, the images were taken at different times, with varying lighting, facial expressions (open/closed eyes, smiling/not smiling), and facial details (glasses/no glasses). From the AR face database, we used 3120 gray images from 120 subjects with each subject providing 26 images. These images were taken in two sessions [33]. From the Feret face database, we only used a subset made up of 1400 images from 200 individuals with each subject providing seven images [31]. This subset is 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 s samples of the n samples per class are used for training and the remaining samples are used for testing, there are $C_n^s = \frac{n(n-1)...(n-s+1)}{s(s-1)...1}$ possible combinations. As a result, there are C_n^s training sets and C_n^s testing sets. For the Feret database, we used four images of each subject as the training samples and took the remaining images as test samples. As a result, we tested different methods by using 35 training and testing sample sets from the Feret database. For the ORL database, we tested different methods in two cases. In the first case, we used five images of each subject as the training samples and there were 252 training and testing sample sets. In the second case, we used six images of each subject as the training samples and there were 210 training and testing sample sets. As the AR face database contained too many samples, we used only the first eight samples from each class as the training samples and the others as the test samples. Figs. 5 and 6 show some of the face images from the AR and ORL databases, respectively. We then resized each face image from the AR database to a 40 by 50 matrix by using the down-sampling algorithm presented in [29]. The face images of the ORL and Feret databases were also resized by using the same algorithm. When implementing our method, we solved A and B by using $A = (X^T X + \mu I)^{-1} X^T y$ and $B = (\tilde{X}^T \tilde{X} + \gamma I)^{-1} \tilde{X}^T y$, respectively. Both μ and γ were set to 0.01. We also conducted experiments of PCA, LDA and the sparse method proposed in [23]. All of the PCA and LDA experiments used the nearest neighbor classifier to perform classification.

We refer to g_r ($r \in \{1, 2, ..., L\}$) in (6) as the reconstruction result of the testing sample, generated from the rth class.

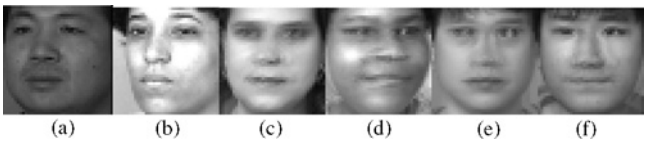

Fig. 7. Result of the global version of the TPTSR on a test sample from the Feret database. (a) Original test sample. (b)–(f) Constructed images generated from the five classes that produce the first five smallest deviations from the testing sample, respectively. In the figure below, (b), (c), (d), (e), and (f) also stand for the five constructed images with the first five smallest deviations, respectively.

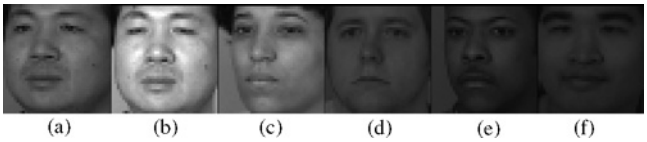

Fig. 8. Result of the TPTSR on a test sample from the Feret database. (a) Same original test sample shown in Fig. 7. (b)–(f) Five constructed images.

As shown earlier, if g_s has minimum deviation from the testing sample, then our method classifies the testing sample into the sth class. We can convert each g_r into a matrix I_r with the same size as the matrices of the face images that are directly used in the experiment. I_r is also referred to as the reconstruction image of the rth class. Since g_r is indeed a linear combination of some training samples of the rth class, it is certain that I_r also looks like a face image of this class. On the other hand, if the testing sample is from the sth class and the reconstruction image generated from the class with minimum deviation also looks like a sample from the sth class, then the method (TPTSR or its global version) will correctly classify the testing sample. Otherwise, the method will erroneously classify the testing sample. As a result, we can easily determine whether a testing sample can be correctly classified by judging whether the reconstruction image generated from the class with minimum deviation from the testing sample looks like the testing sample. We use Figs. 7 and 8 to visually show some original face images and reconstruction images.

Figs. 7 and 8 show the results of the TPTSR and its global version on a test sample from the Feret database, respectively. It is clear that the TPTSR correctly classifies the test sample, whereas its global version fails to do so. Fig. 9 shows the test samples of a subject from the ORL database and the images generated from the representation results of

TABLE I

MEAN OF THE CLASSIFICATION ERROR RATES OF THE PCA, LDA, AND SPARSE METHODS ON THE AR DATABASE

Method and number of the transform axes used	PCA(50)	PCA(100)	PCA(150)	PCA(200)	LDA (119)	The sparse method
Mean of the rates of classification errors	45.3%	42.7%	42.1%	41.9%	34.2%	46.7%

PCA(50), PCA(100), PCA(150), and PCA(200) indicate that PCA used 50, 100, 150, and 200 transform axes for feature extraction, respectively. LDA(119) means that LDA used 119 transform axes for feature extraction. Tables II and III show the method and number of transform axes used in the same way.

TABLE II

MEAN OF THE CLASSIFICATION ERROR RATES OF THE PCA, LDA, AND SPARSE METHODS ON THE FERET DATABASE

Method and number of the transform axes used	PCA(50)	PCA(100)	PCA(150)	PCA(200)	LDA (199)	The sparse method
Mean of the rates of classification errors	38.0%	36.4%	36.2%	36.0%	36.3%	31.0%

Fig. 9. Test samples of a subject from the ORL database and images of the representation result of the test sample. The first row shows the original test sample. For the three images in the same column, the first, second, and third images are the original test sample, and the images of the representation results obtained using the TPTSR and its global version, respectively. The first five images of each subject are used as the training samples and the others are used as the test samples.

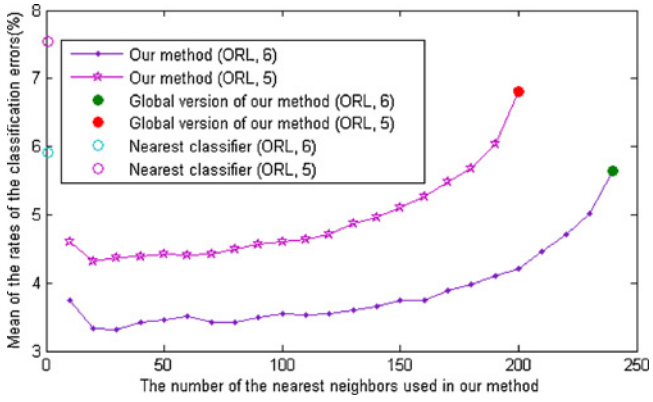

Fig. 10. Means of the classification error rates (%) of our method (i.e., the TPTSR), the global version of our method and the nearest-neighbor classifier on the ORL face database.

the test sample. The second and third rows of Fig. 9 show the representation results obtained using the TPTSR and its global version, respectively. As presented earlier, vector $\tilde{X}B$ is the representation result obtained using the TPTSR. On the other hand, the representation result obtained using the TPTSR is calculated as follows: first A is obtained by solving (2). Then, XA is taken as the representation result.

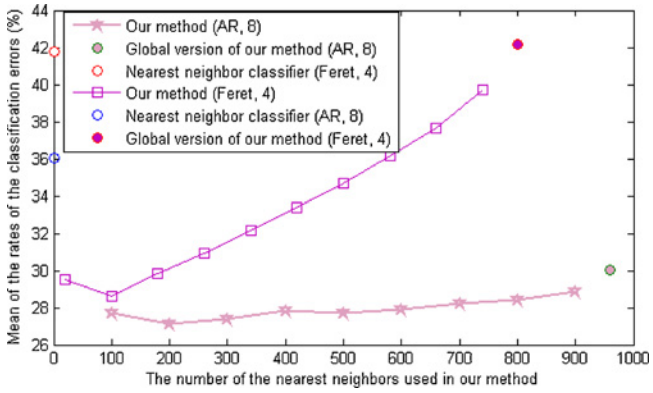

Fig. 11. Means of the classification error rates (%) of our method, the global version of our method and the nearest-neighbor classifier on the AR and Feret face databases.

Figs. 10 and 11 show the means of the classification error rates (%) of our method and its global version. These two figures clearly show that our method is always able to obtain a much lower classification error rate than the global version. For the ORL and AR databases, our method also classifies more accurately than the nearest neighbor classifier. For the Feret database, the maximum rate of the classification errors obtained using our method is close to the mean of the classification error rates obtained using the nearest neighbor classifier. These figures also show that our method can achieve a very low classification error rate when it uses a suitable number of nearest neighbors to represent the test sample.

Tables I to III show the classification results of the PCA, LDA and sparse methods. These tables and Figs. 10 and 11 clearly show that the TPTSR outperforms the PCA and LDA. In the experiments, we noted that the within-class scatter matrix S_w of the LDA is singular. As a result, we changed it to $S_w + 0.001I$, where I is the identity. These tables and Figs. 10 and 11 tell us that the TPTSR can classify much more accurately than the PCA. For example, when the PCA is applied to the Feret database, it obtains a minimum error rate of 36.0%. The TPTSR achieves a minimum error rate of 28.6%, which is 7.4% lower than that of the PCA. Moreover, in all cases, when the TPTSR is applied to the AR database, it obtained a much lower error rate than the LDA. When the LDA is applied to the AR database, the error rate obtained is 34.2%, whereas the TPTSR obtains a minimum error rate of 27.1% and a maximum error rate of 30.1%, which is also the

TABLE III
MEAN OF THE CLASSIFICATION ERROR RATES OF THE PCA, LDA, AND SPARSE METHODS ON THE ORL DATABASE

Method Number of the Transform Axes Used	Mean of the Rates of Classification Errors	
	Five Training Samples Per Class	Six Training Samples Per Class
PCA(50)	7.2%	5.2%
PCA(100)	7.9%	6.0%
PCA(150)	7.8%	6.1%
PCA(200)	7.6%	6.0%
LDA(39)	4.8%	3.7%
The sparse method	5.4%	3.5%

As the sparse method is too time consuming, we randomly selected 50 sets of training samples and the corresponding sets of testing samples to conduct experiments on the sparse method.

error rate of its global version. In most cases, when the TPTSR is applied to the Feret database, it obtains a lower error rate than the LDA. The minimum error rate of the TPTSR on the ORL database is also lower than that of the LDA. Through the experiments, we also find that more training samples usually result in a lower error rate.

V. CONCLUSION

The first and second phases of the method proposed in this paper made coarse to fine classification decisions for the test samples, respectively. The first phase first expressed the test sample as a linear combination of all the training samples and then uses the representation result to determine the M nearest neighbors of the test sample. The second phase seeks to represent the test sample as a linear combination of the determined nearest neighbors and then exploits the representation result to perform classification. According to the devised classification rule, the coarse classification produced by the first phase will allow the test sample to be ultimately classified into only one of the class labels of the M nearest neighbors. The fine classification produced by the second phase ultimately determines the most suitable class label for the test sample.

The proposed method is a supervised sparse representation method and uses a very reasonable way to determine the sparseness. Our analysis shows that the proposed method has solid theoretical foundations. A large number of face recognition experiments show a good performance of our method. The two-phase classification framework proposed for the first time in this paper seems to be also useful for improving other face recognition methods. In the future, we will explore the applications of this framework on other methods.

REFERENCES

[1] M. Kirby and L. Sirovich, "Application of the KL phase for the characterization of human faces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 1, pp. 103–108, Jan. 1990.

[2] Y. Xu, D. Zhang, J. Yang, and J.-Y. Yang, "An approach for directly extracting features from matrix data and its application in face recognition," *Neurocomputing*, vol. 71, nos. 10–12, pp. 1857–1865, 2008.

[3] J. Yang, D. Zhang, A. F. Frangi, and J.-Y. Yang, "Two-dimensional PCA: A new approach to appearance-based face representation and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 131–137, Jan. 2004.

[4] Y. Xu and D. Zhang, "Represent and fuse bimodal biometric images at the feature level: Complex-matrix-based fusion scheme," *Opt. Eng.*, vol. 49, no. 3, 2010.

[5] S. W. Park and M. Savvides, "A multifactor extension of linear discriminant analysis for face recognition under varying pose and illumination," *EURASIP J. Adv. Signal Process.*, vol. 2010, no. 158395, p. 11, 2010.

[6] M. Debruyne and T. Verdonck, "Robust kernel principal component analysis and classification," *Adv. Data Anal. Classification*, vol. 4, no. 2, pp. 151–167, Sep. 2010.

[7] Y. Xu, D. Zhang, F. Song, J.-Y. Yang, Z. Jing, and M. Li, "A method for speeding up feature extraction based on KPCA," *Neurocomputing*, vol. 70, nos. 4–6, pp. 1056–1061, Jan. 2007.

[8] M. E. Tipping, "Sparse kernel principal component analysis," in *Neural Information Processing Systems*, T. K. Leen, T. G. Dietterich, and V. Tresp, Eds. Cambridge: MIT Press, 2000, pp. 633–639.

[9] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Proc. ICANN*, 1997, pp. 583–588.

[10] B. Schölkopf and A. Smola, *Learning With Kernels*. Cambridge: MIT Press, 2002.

[11] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Mar. 2001.

[12] D. Tao and X. Tang, "Kernel full-space biased discriminant analysis," in *Proc. IEEE ICME*, Jun. 2004, pp. 1287–1290.

[13] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local Fisher discriminant analysis," *J. Mach. Learn. Res.*, vol. 8, pp. 1027–1061, May 2007.

[14] V. Vural, G. Fung, B. Krishnapuram, J. G. Dy, and B. Rao, "Using local dependencies within batches to improve large margin classifiers," *J. Mach. Learn. Res.*, vol. 10, pp. 183–206, Feb. 2009.

[15] Z. Y. Liu, K. C. Chiu, and L. Xu, "Improved system for object detection and star/galaxy classification via local subspace analysis," *Neural Netw.*, vol. 16, nos. 3–4, pp. 437–451, 2003.

[16] A. Ahmadi, S. Omatu, T. Fujinaka, and T. Kosaka, "Improvement of reliability in banknote classification using reject option and local PCA," *Inf. Sci.*, vol. 168, nos. 1–4, pp. 277–293, 2004.

[17] Y. Zeng, Y. Yang, and L. Zhao, "Nonparametric classification based on local mean and class statistics," *Expert Syst. Appl.*, vol. 36, no. 4, pp. 8443–8448, 2009.

[18] H. Ryu, J.-C. Yoon, S. S. Chun, and S. Sull, "Coarse-to-fine classification for image-based face detection," in *Proc. CIVR*, 2006, pp. 291–299.

[19] S. Gangaputra and D. Geman, "A design principle for coarse-to-fine classification," in *Proc. IEEE Comput. Soc. Conf. CVPR*, Jun. 2006, pp. 1877–1884.

[20] X. Fan, "Learning a hierarchy of classifiers for multi-class shape detection," Ph.D. dissertation, Dept. Electric. Comput. Eng., Johns Hopkins Univ., Baltimore, MD, 2006.

[21] Z. Lai, Z. Jin, J. Yang, and W. K. Wong, "Sparse local discriminant projections for feature extraction," in *Proc. ICPR*, 2010, pp. 926–929.

[22] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, 'Sparse representation for computer vision and pattern recognition," *Proc. IEEE*, vol. 98, no. 6, pp. 1031–1044, Jun. 2010.

[23] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[24] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," in *Proc. Adv. NIPS*, vol. 21. 2009.

[25] Y. Shi, D. Q. Dai, C. C. Liu, and H. Yan, "Sparse discriminant analysis for breast cancer biomarker identification and classification," *Prog. Natural Sci.*, vol. 19, no. 11, pp. 1635–1641, 2009.

[26] M. Dikmen and T. Huang, "Robust estimation of foreground in surveillance videos by sparse error estimation," in *Proc. Int. Conf. Pattern Recog.*, Dec. 2008.

[27] E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 2790–2797.

[28] S. Rao, R. Tron, R. Vidal, and Y. Ma, "Motion segmentation via robust subspace separation in the presence of outlying, incomplete, and corrupted trajectories," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[29] Y. Xu and Z. Jin, "Down-sampling face images and low-resolution face recognition," in *Proc. 3rd Int. Conf. Innovative Comput. Inform. Control*, Jun. 2008, pp. 392–395.

[30] [Online]. Available: http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[31] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 10, pp. 1090–1104, Oct. 2000.
[32] P. J. Phillips, *The Facial Recognition Technology (FERET) Database* [Online]. Available: http://www.itl.nist.gov/iad/humanid/feret/feret_master.html
[33] [Online]. Available: http://cobweb.ecn.purdue.edu/~aleix/aleix_face_DB.html

Yong Xu (M'06) was born in Sichuan, China, in 1972. He received the B.S. and M.S. degrees in 1994 and 1997, respectively, and the Ph.D. degree in pattern recognition and intelligence system from the Nanjing University of Science and Technology, Nanjing, China, in 2005.

Currently, he is with the Bio-Computing Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China. His current research interests include pattern recognition, biometrics, machine learning, image processing, and video analysis.

David Zhang (F'08) graduated in computer science from Peking University, Beijing, China, in 1974. He received the M.S. degree in computer science and the Ph.D. degree from the Harbin Institute of Technology (HIT), Harbin, China, in 1982 and 1985, respectively, and the second Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 1994.

From 1986 to 1988, he was a Post-Doctoral Fellow with Tsinghua University, Tsinghua, China, and then an Associate Professor with Academia Sinica, Beijing. Currently, he is the Chair Professor with the Department of Computing, Hong Kong Polytechnic University, Kowloon, Hong Kong, where he is the Founding Director of the Biometrics Technology Center supported by the Hong Kong SAR Government in 1998. He serves as a Visiting Chair Professor with Tsinghua University, and an Adjunct Professor with Shanghai Jiao Tong University, Shanghai, China, Beihang University, Beijing, HIT, and the University of Waterloo. He is the author of more than 10 books and 160 journal papers.

Dr. Zhang is the Founder and Editor-in-Chief of the *International Journal of Image and Graphics*, a Book Editor of the *Springer International Series on Biometrics*, was an Organizer of the International Conference on Biometrics in 2004 and 2006, an associate editor of more than ten international journals, including IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS-A/SYSTEMS MAN AND CYBERNETICS-C/*Pattern Recognition*, the Chair of the IEEE/CIS Technical Committee on Intelligent Systems Application. He is a Croucher Senior Research Fellow, a Distinguished Speaker of the IEEE Computer Society, and a Fellow of the International Association of Pattern Recognition.

Jian Yang received the B.S. degree in mathematics from Xuzhou Normal University, Xuzhou, China, in 1995, the M.S. degree in applied mathematics from Changsha Railway University, Changsha, China, in 1998, and the Ph.D. degree in the subject of pattern recognition and intelligence systems from the Nanjing University of Science and Technology (NUST), Nanjing, China, in 2002.

In 2003, he was a Post-Doctoral Researcher with the University of Zaragoza, Zaragoza, Aragon, Spain. From 2004 to 2006, he was a Post-Doctoral Fellow with the Biometrics Center, Hong Kong Polytechnic University, Kowloon, Hong Kong. From 2006 to 2007, he was a Post-Doctoral Fellow with the Department of Computer Science, New Jersey Institute of Technology, Newark. Currently, he is a Professor with the School of Computer Science and Technology, NUST. He is the author of more than 50 scientific papers on pattern recognition and computer vision. His journal papers have been cited more than 1200 times on the ISI Web of Science, and 2000 times on Google Scholar. His current research interests include pattern recognition, computer vision, and machine learning.

Dr. Yang was awarded the RyC Program Research Fellowship sponsored by the Spanish Ministry of Science and Technology in 2003. Currently, he is an Associate Editor of *Pattern Recognition Letters* and *Neurocomputing*.

Jing-Yu Yang received the B.S. degree in computer science from the Nanjing University of Science and Technology, Nanjing, China.

From 1982 to 1984, he was a Visiting Scientist with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana. From 1993 to 1994, he was a Visiting Professor with the Department of Computer Science, Missouri University, Columbia. In 1998, he was a Visiting Professor with Concordia University, Montreal, QC, Canada. He is with the School of Computer Science and Technology, Nanjing University of Science and Technology. His current research interests include the areas of pattern recognition, image processing, computer vision, and artificial intelligence.