Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA, August 12-17, 2007

Multi-Stage Optimal Component Analysis

Yiming Wu, Xiuwen Liu, Washington Mio

Abstract— **Optimal component analysis (OCA) uses a stochastic gradient optimization process to find optimal representations for general criteria and shows good performance in object recognition applications. However, OCA often requires extensive computation for gradient estimation and linear representation updating. To significantly reduce the required computation, in this paper, a multi-stage learning process is proposed which decomposes the original optimization problem into several levels. As the learning process at each level starts with a good initial point obtained from next level, the multi-stage OCA algorithm can speed up the original algorithm significantly and make OCA learning feasible for many applications. We illustrate the effectiveness of the proposed method on the application of face classification.**

I. Introduction

Over the last ten years or so, 2-D image object recognition using statistical methods has become a popular area of research and one of the most successful applications of image analysis and understanding. However, as images are, in general, rather high dimension, the statistical method is difficult and often infeasible to apply and analyze with high dimensional data. On the other hand, it is well-known that images are generated by imaging processes with typically a small number of physical parameters such as lighting, orientation, camera distance, etc. This motivates a search for methods that can reduce image dimensions without severe loss in information. A commonly used method is to project images into a low dimensional subspace and use this projection for processing. For instance, let I be an image reshaped into $n \times 1$ vector and let U be an $n \times d$ orthogonal matrix denoting a basis of an orthogonal d-subspace of $\Re^n (n \gg d)$, the vector $\alpha(I) = U^T I \in \Re^d$, also called the image coefficients, provides a d-dimension subspace representation of I.

In recent years, many approaches have been brought to bear on linear representation problem. We will briefly review some of them, including Principle Component Analysis (PCA)[1][2], Independent Component Analysis (ICA)[3], Canonical Correlation Analysis (CCA)[4], and Linear Discriminative Analysis (LDA)[5], etc. PCA finds a set of the most representative projection vectors by eigen-decomposition of the data's covariance matrix. It is an optimal linear technique for data dimension reduction and data reconstruction. However, when applied to the classification problem, PCA is not optimal as it can not utilize the class information for class projection. ICA is a more general method than PCA which captures both second and higher-order statistics and projects the input data onto the basis vectors that are as statistically independent as possible. However, ICA suffers from computational expensiveness, which limits its application to high-dimension data classification. CCA is a way of measuring the linear relationship between two multidimensional variables. It finds two bases, one for each variable, that are optimal with respect to correlations and, at the same time, it finds the corresponding correlations. The LDA uses the class information and finds an optimal basis that maximize the between-class scatter while minimizing the within-class scatter. The optimal basis is readily computed by solving a generalized eigen-decomposition to the scatter matrices. A limitation of LDA lies in the fact that it assumes the conditional distribution is Gaussian with the same variance. Furthermore, the objective function of LDA requires that one of the scatter matrices be non-singular. However, in many real applications, such as face recognition and text classification, all scatter matrices in question can be singular since the dimension of data, in general, exceeds the number of data points. Thus, although these methods have been widely used and satisfy certain optimality criteria, they may not necessarily be optimal for a specific application at hand [6][7].

Unlike these methods, Optimal Component Analysis (OCA) [8][9] is a recently proposed stochastic method which gives a general optimality criterion for applications. The search for optimal linear representation, or an optimal subspace, is based on a stochastic optimization process which maximizes a pre-specified performance function over all subspaces of a particular dimension. [8][9] shows its good performance on object recognition.

However, as OCA search is a stochastic process, the searching time for optimal basis is high, which becomes a main obscure to apply OCA to real applications. In our previous work, a two-stage strategy is proposed to deal with this problem [11]. In this method, the input data is first reduced to a lower dimension using dimension reduction methods like PCA, LDA, etc., then OCA search is performed in the reduced space. As the searching space is reduced, the OCA will be significantly sped up. This method shows its good performance in certain applications. But when we reduce the data dimension to very low, the performance of this method is not guaranteed.

Based on the two-stage OCA method, in this paper, a multi-stage strategy is proposed to solve these problems. In this method, the data dimension is first hierarchically reduced to several lower levels through several shrinkage

Yiming Wu and Xiuwen Liu are with the Department of Computer Science, Florida State University, Tallahassee, FL 32306, E-mail: {ywu,liux}@cs.fsu.edu. Washington Mio is with the Department of Mathematics, Florida State University, Tallahassee, FL 32306, E-mail: mio@math.fsu.edu.

1-4244-1380-X/07/$25.00 ©2007 IEEE

matrices. The shrinkage matrices can be obtained using dimensional reduction methods such as PCA. Then the OCA search is performed in the lowest level with an initial basis obtained using common dimension reduction methods, such as PCA, LDA, etc. We denote the level with the original data dimension by level 0 and the lowest level by level L. Then the following idea is applied recursively in each level 1 < L: (i) optimize the recognition performance in the reduced space by performing OCA searching with an initial basis obtained from the next lower level, and (ii) expand the optimal subspace to the bigger space in a pre-specified way. By applying this decomposition procedure recursively, a hierarchy of layers is formed. We show the optimal performance obtained in the reduced space is maintained after the expansion. Thus, the learning process of each level starts with a good initial point obtained from the next level. Also, this speeds up the original algorithm significantly since the learning is performed mainly in reduced spaces, which makes OCA learning feasible for many applications.

The rest of the paper is organized as follows: Section II gives a overall review of OCA method and the proposed multi-stage OCA method is presented at Section III; The exprimental results on face data sets are presented in section IV; Section V concludes the paper with a brief discussion.

II. Optimal Component Analysis

The image classification problem can be formalized as following: given images belong to several classes, we want to find an optimal basis which clusters the images belongs to the same class while at the same time, separates image with those of other classes as far as possible. OCA is proposed based on this motivation [8]. Let $U \in \Re^{n\times d}$ be an orthonormal basis of a d-dimensional subspace of $\Re^n$, where n is the size of an image and d is the required dimension of the optimal subspace (generally n$\gg$d). For an image I, considered as a column vector of size n, the vector of coefficients is given by $\alpha(I,U) = U^T I \in \Re^d$. Let there be C classes to be recognized from the images; each class has k_{train} training images (denoted by $I_{c,1},\ldots,I_{c,k_{train}}$) and k_{cross} cross validation images (denoted by $I'_{c,1},\ldots,I'_{c,k_{cross}}$). A quantity ρ is defined as follows:

$$\rho(I'_{c,i},U) = \frac{\min_{c'\neq c,j} d(I'_{c,i},I_{c',j};U)}{\min_j d(I'_{c,i},I_{c,j};U)+\epsilon}\ . \qquad (1)$$

where

$$d(I_1,I_2;U) = \|\alpha(I_1,U)-\alpha(I_2,U)\| \qquad (2)$$

$\epsilon > 0$ is a small number to avoid division by zero. Here ρ measures how well the nearest-neighbor classifier applied to the data projected onto U identifies the elements $I_{c,i}$ as belonging to class c.

For easy handling, we scale the quality ρ using the performance function F. F is defined in the following way:

$$F(U) = \frac{1}{Ck_{cross}}\sum_{c=1}^{C}\sum_{i=1}^{k_{cross}} h(\rho(I'_{c,i},U)-1). \qquad (3)$$

where h$(\cdot)$ is a monotonically increasing bounded function is used to control bias with respect to particular classes in measurements of performance. In our implementation, $h(x) = 1/(1+\exp(-2\beta x))$ where β controls the degree of smoothness of $F(U)$. As stated in [8], when we let $\beta\to\infty$, F is precisely the recognition performance of the nearest neighbor classifier after projection to the subspace U.

Under this formulation, $F(U) = F(UH)$ for any $d\times d$ orthogonal matrix H as the distance $d(I_1,I_2;U) = d(I_1,I_2;UH)$; the choice of 2-norm in $d(I_1,I_2,U)$ allows for this equality. In other words, F depends on the subspace spanned by U but not on the specific basis chosen to represent that subspace. Therefore, our search for optimal representation(s) is on the space of d-dimensional subspace rather than on their bases. Let $\mathcal{G}_{n,d}$ be the set of all d-dimensional subspaces of $\Re^n$; it is called a Grassmann manifold. It is a compact, connected manifold of dimension $d(n-d)$. An element of this manifold, i.e., a subspace, can be represented by a basis. Let U be an orthonormal basis in $\Re^{n\times d}$ such that *span(U)* is the set of all the orthonormal bases of *span(U)*, i.e., $[U] = \{UH | H\in\Re^{d\times d}, H^TH = I_d\}\in\mathcal{G}_{n,d}$. The problem of finding optimal linear subspace for recognition becomes an optimal problem:

$$\hat{U} = \arg\max_{U\in\mathcal{G}_{n,d}} F(U)$$

In [8], a stochastic gradient-based algorithm is used to find an optimal subspace $\hat{U}$. Since Grassmann manifold $\mathcal{G}(n,d)$ is a curved space (i.e., let $u_1\in\mathcal{G}(n,d)$, $u_2\in\mathcal{G}(n,d)$, but $u_1 + u_2 \notin \mathcal{G}(n,d)$) , as opposed to being a (flat) vector-space, the gradient process has to account for its intrinsic geometry. The performance function F can be viewed as a scalar-field on $\mathcal{G}(n,d)$. A necessary condition for $\hat{U}$ to be a maximum is that, for any tangent vector at $\hat{U}$, the directional derivative of F in the direction of that vector should be zero. Based on this, in this algorithm, we first initialize a random basis U, then at each iteration, the gradient vector of F with respect to U, which is a skew-symmetric matrix, is computed. By following the gradient, a new solution is generated, which is used as a proposal and is accepted with a probability that depends on the performance improvement. If the performance of the new solution is better than the current solution, it is always accepted. Otherwise, the worse the new solution's performance, the lower the probability the solution is being accepted.

However, a brute force implementation of OCA is typically computational expensive and may prevent its wide use in certain applications. The computational complexity C_n of each iteration of this algorithm is

$$C_n = O(d\times(n-d)\times k_{test}\times k_{training}\times n\times d)$$

C_n is obtained by the following computation. $d\times(n-d)$ is the dimension of the gradient vector. For each dimension and for each test image, the closest images in all the classes need to be found to compute the ratio ρ in Eq.(1) and performance function in Eq.(3), this gives the product $k_{test}\times k_{training}$. The term $n\times d$ comes from Eq.(2). Therefore, we obtained

the complexity for one iteration as the expression. The overall computational complexity is $C_n \times T$ where T is the number of iterations.

From the above analysis, we see that the computation at each iteration depends on several factors and the complexity is $O(n^2)$. For typical applications, n, which is the number of pixels in an image, is relatively large. Also when n is large, the dimensional of the search space will also be large. (In the Grassmann manifold, it is $(n-d) \times d$, is large). Thus the algorithm can be time consuming.

III. Multi-Stage OCA

In our previous work, similar with the two-stage LDA methods presented in [10], a two-stage OCA method is proposed to deal with the heavy computational cost problem [11]. In the first stage of this method, the dimension of the input data is reduced using certain well-known dimensional reduction methods, such as PCA, ICA, LDA, RCA, QR, etc. In the second stage, OCA search is performed in the lower dimensional space. Obviously, as the data dimension is reduced, the search space will also be smaller and the OCA search time will be reduced accordingly. However, a hidden problem is when we reduce the dimension to very low, the good performance is not guaranteed. In this paper, we extend the two-stage OCA method and proposed a method called multi-stage OCA.

A. Multi-Stage Learning

The learning process of multi-stage OCA is illustrated in Figure 1. We pre-define the number of levels L, Level 0 is called the highest level and level L is the lowest level. Then we recursively shrink data I_l $(0 \leq l < L)$ to get dimension reduced data I_{l+1} by multiplying it with shrinkage matrix A_l at each level. We denote the shrinkage factor as m_l, then the size of data at level l is $\frac{n_{l-1}}{m_l} = \frac{n_0}{\prod_{i=1}^{l} m_i}$, and the size of shrinkage matrix A_l is $n_l \times n_{l+1}$, where n_l is the data size at level l.

Note that our goal is to find an optimal basis $\hat{U}_0$ of size $n_0 \times d$ at level 0. To fulfill this task, we hierarchically search the optimal basis at each level. The search begins from level L on $\mathcal{G}_{n_L,d}$ with images size n_L. The search can be effective since the learning space $\mathcal{G}_{n_L,d}$ is relatively small and the computational complexity of each iteration is low. After getting an optimal basis $\hat{U}_L$ at level L, we obtain basis $\bar{U}_{L-1}$ of level $L-1$ by expanding $\hat{U}_L$. Then we use $\bar{U}_{L-1}$ as an initial point to perform the search at level $L-1$ on $\mathcal{G}_{n_{L-1},d}$ with images of size n_{L-1}, As we will discuss later, the performance based on basis $\bar{U}_{L-1}$ will be consistent with that of $\hat{U}_L$. Thus, the search can be performed relatively effectively as it begins with a good initial point. The search result of this level will be used to obtain a basis $\bar{U}_{L-2}$ of level $L-2$, which is used as an initial point for further search at level $L-2$. This process is repeated until we have reached level 0. In summary, the search is performed from the lowest level L to the highest level 0. The lower the level, the more efficient the search. The search result of the lower level provides a good initial point for the next upper level. The recognition performance keeps on increasing at each level.

Algorithm 1:Multi-Stage OCA Algorithm

Input: Training data matrix $I_{train} \in \Re^{K_{train} \times n_0}$, shrinkage factors $m_1, ..., m_L$.
Output: optimal basis U_0 of level 0

1) Choose the dimension shrinkage matrices A_l of size $n_l \times n_{l-1}$, for $l = 1, \ldots, L$. Then shrink the data hierarchically by left-multiplying A_l with image in level l.
2) Learn starting from level L for the optimal basis $\hat{U}_L$ at level L on $\mathcal{G}_{n_L,d}$ with data size n_L.
3) For each $l = L-1, \ldots, 0$,
   BEGIN
   a) let $\bar{U}_l = A_l^T \hat{U}_{l+1}$,
   b) using $\bar{U}_l$ as the initial point, search for the optimal basis $\hat{U}_l$ at level l on $\mathcal{G}_{n_l,d}$ with data size n_l.
   END

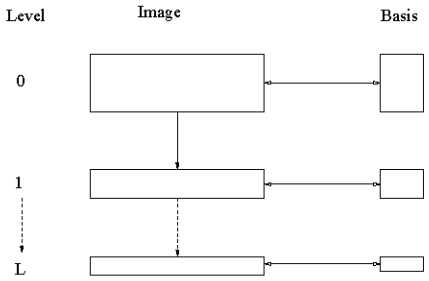

Fig. 1. Multi-stage search process.

B. Constraints on Shrinkage Matrix

Now let us have a look what is the constraint the shrinkage matrix A should have. Recall that the optimal basis $\hat{U}$ searched in OCA should have

$$\hat{U}_l^T \hat{U}_l = I_{d \times d} \tag{4}$$

where $\hat{U}_l \in \Re_{n_l \times d}$ is the optimal basis obtained at level l. When we up-sample the optimal basis to level $l-1$ using shrinkage matrix A, the optimal basis at level $l-1$ should also have

$$\begin{aligned} I_{d \times d} &= \bar{U}_{l-1}^T \bar{U}_{l-1} \\ &= (A\hat{U}_l)^T (A\hat{U}_l) \\ &= \hat{U}_l^T A^T A \hat{U}_l \end{aligned} \tag{5}$$

Combining Eq.(4) and Eq.(5), we have,

$$A^T A = I_{n_l \times n_l}$$

That mean, the shrinkage matrix should be orthogonal. Based on this constraints, PCA is a good choice to shrink data dimension as the transform matrix of PCA satisfies this constraints.

C. Recognition Performance Analysis

Now let us have a deep explore of the affect of performance when we expand basis from lower level to next higher level. Recall Eq.(3) and Eq.(2), the performance function F depends only on the distance between the representation of images $d(\cdot,\cdot;\cdot)$. We assume $\hat{U}_l \in \Re_{d\times n_l}$ and $\bar{U}_{l-1} \in \Re_{d\times n_{l-1}}$ are the optimal basis at level l and initial optimal basis at $l-1$ respectively, and I_1^l, I_2^l is two image data at level l, I_1^{l-1}, I_2^{l-1} are the image data at level $l-1$, then we have

$$\begin{aligned} d(I_1, I_2; \hat{U}_l) &= \|\alpha(I_1, \hat{U}_l) - \alpha(I_2, \hat{U}_l)\| \\ &= \|\hat{U}_l^T I_1 - \hat{U}_l^T I_2\| \\ &= \|\bar{U}_{l-1}^T A I_1 - \hat{U}_{l-1}^T A I_2\| \\ &= \|\bar{U}_{l-1}^T I_1^{l-1} - \bar{U}_{l-1}^T I_2^{l-1}\| \\ &= \|\alpha(I_1^{l-1}, \bar{U}_{l-1}) - \alpha(I_2^{l-1}, \bar{U}_{l-1})\| \\ &= d(I_1^{l-1}, I_2^{l-1}; \bar{U}_{l-1}). \end{aligned}$$

Therefore, $F(\hat{U}_l) = F(\bar{U}_{l-1})$, which means the performance is kept when we expand basis.

D. Computational Analysis

Now let us have a look of how much we gain in term of computational cost by using the multi-stage OCA algorithm. For each iteration, the computational complexity with images of size n_0 is $C_{n_0} = O(d\times(n_0-d)\times k_{test}\times k_{training}\times n_0\times d)$. While the computational complexity with images of size n_l is

$$\begin{aligned} C_{n_l} &= O(d\times(n_l-d)\times k_{test}\times k_{training}\times n_l\times d) \\ &= \frac{(n_l-d)\times n_l}{(n_0-d)\times n_0} C_{n_0} \end{aligned}$$

and the total search time for the multi-stage OCA will be

$$\begin{aligned} C_{total} &= \Sigma_{l=1}^{L} C_{n_l} \\ &= \Sigma_{l=1}^{L} \frac{(n_l-d)\times n_l}{(n_0-d)\times n_0} C_{n_0} \\ &= \Sigma_{l=1}^{L} \frac{(n_0-\prod_{i=1}^{l} m_i d)}{(n_0-d)(\prod_{i=1}^{l} m_i)^2} C_{n_0} \end{aligned} \quad (6)$$

Usually we select $2 \le L \le 5$, $10 < d < 100$ for many applications and $m_i > 1$, Eq.(6) will lead to $C_{total} \ll C_{n_0}$, thus it is much more efficient to learn using the multi-stage OCA than the original OCA.

IV. Experimental Results

We evaluate the effectiveness of the multi-stage OCA algorithm on face classification applications. The face data sets used for our experiments are presented in subsection A; In subsection B, two experiments are performed to illustrate the performance of multi- stage OCA algorithm. In the first experiment, we study its performance in term of classification accuracy and efficiency. In the second experiment, we compare the performance of multi-stage OCA with other classification methods, which includes PCA, PCA+LDA, LDA/QR, original OCA, two-stage OCA etc. In these experiments, K-Nearest Neighbor(KNN) algorithm is used as the classifier. The C program is running on a workstation with a Intel 3.00GHz CPU, 8.0G RAM.

A. Data Sets

We have three face data sets for our performance evaluation:

- ORL face data set [12]. It contains 400 face images of 40 individuals. The image size is 92×112. We use the whole image as an instance, i.e.,the dimension of an instance is 92×112 = 10,304.
- PIE face data set [13]. It contains 66 person with 21 images each. The image size is 100×100 = 10,000.
- AR face data set[14]. It is a large face image data set. We use a subset of AR containing 1,638 face images of 126 individuals. Its image size is 768×576. We first crop the image from the row 100 to 500 and the column 200 to 550, and then sub-sample the cropped images with sample step 4×4. The dimension of each instance is reduced to $101 \times 88 = 8,888$.

TABLE I
Statistics for our real test data sets

Data set	# of class	# of data per class	# of training per set	# of test per set
ORL	40	10	5	5
PIE	66	21	10	11
AR	126	13	6	7

B. Experimental Results

This part includes two experiments. In the first experiment, we evaluate the performance of multi-stage OCA algorithm in terms of recognition accuracy and efficiency. Table I summarizes the data setting used in the experiment. Here PCA is used to compute the shrinkage matrix and the shrinkage level is 3. OCA search is performed in each level except level 0 for 500 iterations[1]. The dimension of subspace d is set to 10. After we get the final optimal basis in level 0, we perform the classification using K-NN method. Table II gives the classification accuracy of the multi-stage OCA algorithm. From the Table, we can have the following important observations:

- KNN with K = 1 usually performs the best by all algorithms on these face data sets. These is a clear tread of decrease in accuracy for each data set as K increases.
- We can see the multi-stage OCA can give good performance in terms of classification accuracy. Specially, it can achieve 100% on ORL and PIE data sets. This is mainly due to the relatively small within-class variations in these data.
- For the AR data set, as the data sets are relatively difficult to classify, the classification accuracy is a little worse, but is still has better or comparable performance when compared with other methods[15][16][17].

Now let us have a look of the computational efficiency gained by the multi-stage OCA algorithm. The running time and classification accuracy of multi-stage OCA original OCA

[1]Although we can also search on level 0, we do not do so as the recognition accuracy already very high till now.

TABLE II

RECOGNITION ACCURACY(%) OF MULTI-STAGE OCA ON SEVERAL FACE DATA SETS

KNN	ORL	PIE	AR
1	100	100	97.95
3	98.50	99.17	95.24
4	98.00	98.35	94.90
5	95.50	94.36	92.97

TABLE III

TIME COST (S) AND RECOGNITION ACCURACY (%) OF MULTI-STAGE OCA AND ORIGINAL OCA

Data Set	Measure Quantity	Level 3	Level 2	Level 1	Total	Original
ORL	d	50	100	199	-	10,304
	t	0.75	1.39	4.55	6.69	173
	accuracy	98.00	100	100	100	100
PIE	d	20	100	600	-	10,000
	t	4.28	6.16	15.23	25.67	224
	accuracy	99.86	100	100	100	100
AR	d	200	1,000	2,000	-	8,888
	t	0.78	5.40	40.22	46.40	421
	accuracy	81.97	96.37	97.34	97.95	98.03

are shown on Table III. For each data set, the first row shows the data dimensions in each level, the second row is the searching time used in each iteration, and the third row is corresponding recognition accuracies. The total searching time is the sum of searching time in the 3 levels. From this Table, we can see

- The total searching time is greatly reduced compared with the original OCA algorithm. Take ORL for an example, when we perform search on the original space, the running time is about 3 minutes for one iteration, while it only takes 6.56 seconds with multi-stage OCA;
- The running time of a higher level is larger for those in lower levels. This is easy to see, as we the level becomes higher, the searching space becomes larger, which increases the searching time.

More convincing illustration of performance of multi-stage OCA is shown in Figure 2. This figure shows the evolution of recognition accuracy and performance function F of multi-stage OCA algorithm of 3 levels on these data sets. Here OCA search runs for 500 iterations. The left figures in each row show the evolution of recognition accuracy of testing data. We can see in each level the recognition accuracy is increased and the recognition accuracy of final iteration in the previous level is consistent with the initial point of next level. Take the ORL data set for an example, the performance reaches 100% in level 2. In this case, we do not need to do the further search. However, we still list the search performance in all 3 levels in order to easily compare. The right figures of each row show the evolution of performance function F. This proves the good performance of multi-stage OCA method.

Our second experiment is to compare the performance of multi-stage OCA method with other classification methods, including PCA, PCA+LDA, QR/LDA, original OCA and two-stage OCA. The data set used here are the ORL and AR face data sets. For a fair comparison, we use the same experiment setting and experimental results as in [10], The relevant parameters are as follows: p = 100 principle components in PCA and the PCA stages of PCA+LDA and first stage of two-stage OCA algorithm. The classification accuracies are estimated by 10-fold cross-validation. For the proposed multi-stage OCA, we use 3 levels and the dimension of each level is set the same as experiment one. Table IV shows the recognition accuracy results of different classification methods on ORL and AR data sets. We can see that for different number of neighborhood methods, the proposed multi-stage OCA method can overcome other methods such as PCA, PCA+LDA, QR/LDA easily. For the ORL data set, we can see the recognition accuracies of all OCA type of algorithms (original OCA, two-stage OCA and multi-stage OCA) can achieve 100%, while others methods can not. For AR data set, as it is more difficult to classify, thus the recognition performance is a little worse than ORL data set. But we can see the OCA type of algorithm also can overcome the other methods. We can see the recognition accuracy of multi-stage OCA is very close to that original OCA algorithm, but the searching time is greatly reduced. Compared with two-stage OCA, Table IV shows multi-stage OCA improves recognition accuracy.

TABLE IV

RECOGNITION ACCURACY(%) OF DIFFERENT CLASSIFICATION METHODS ON ORL AND AR DATA SET

Data Set	KNN	PCA	PCA+ LDA	QR/LDA	OCA	2-stage OCA	M-stage OCA
ORL	1	97.25	95.00	98.25	100	100	100
	3	94.50	94.75	98.00	100	100	100
	5	92.25	95.50	98.25	100	100	100
AR	1	65.30	92.45	92.24	100	99.21	100
	3	59.05	90.72	90.63	96.49	94.78	96.83
	5	57.49	88.50	89.53	95.24	94.11	94.33

V. DISSCUSSION

Optimal component analysis (OCA) provides a general subspace formulation and gives an optimal solution for data classification. However, computational cost is a serious problem which is a main obscure to apply OCA to real applications. In order to deal with this problem, in this paper, we proposed a multi-stage OCA algorithm which extend the two-stage OCA algorithm by fist reducing the data to several levels of lower dimension using corresponding shrinkage matrices. Then OCA search is performed at the lowest level. After we get the optimal basis at the lowest level, we expand the basis to a higher level and use this expanded basis as initial basis and perform OCA search in the higher level. As each level provides a good initial basis for the searching of the higher level, then the searching time will be reduced greatly and the performance will be kept.

The optimal basis of OCA is set-valued rather than being point-valued, which accounts for the improvement of multi-

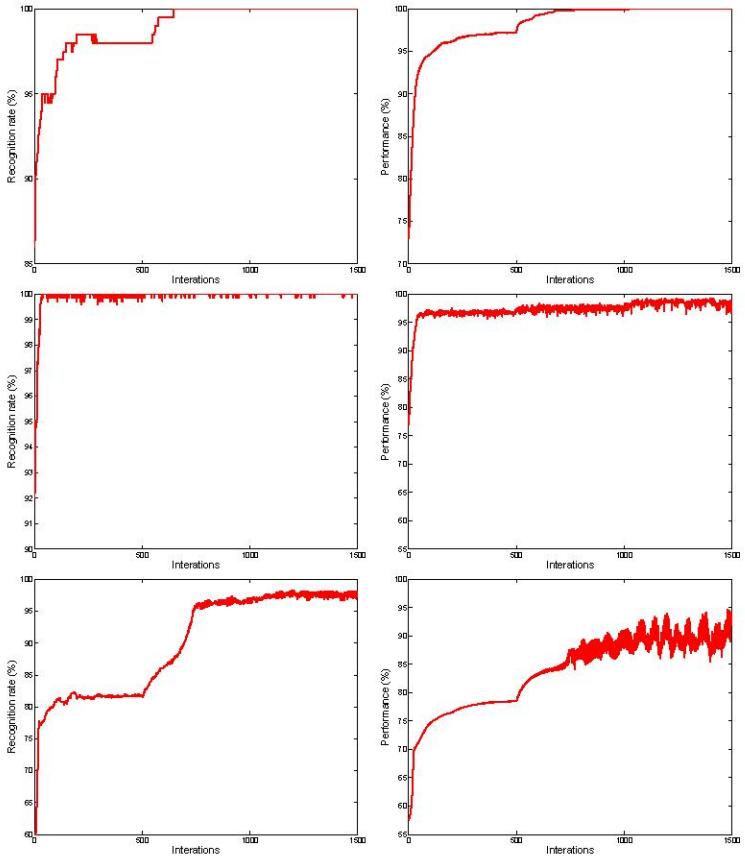

Fig. 2. Evolution of recognition accuracy and performance function F on ORL, PIE, AR face data sets. Left row: recognition accuracy of test images; Right row: corresponding performance function F on these data sets.

stage OCA search in each level. While for other linear dimension reduction methods, such as PCA and LDA, as the optimal basis of these method is unique, they will not lead to improvements when we use multi-stage strategy. This also shows the advantage of OCA algorithm over other algorithms.

VI. Acknowledgment

The authors would like to thank the anonymous reviewers for their comments and suggestions. Thanks also go to the producers of the ORL, PIE and AR data sets for making them public. This research was supported in part by NSF grants CCF-0514743, IIS-0307998, ACI-0324944 and ARO grant W911NF-04-01-0268.

References

[1] I. T. Jolliffe, *Principal Componment Analysis*, Springer-Verlag, 1986.
[2] M. Turk and A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuro-science*, vol. 3, pp. 71–86, 1991.
[3] A. Hyvarinen, J. Karhunen, and E. Oja, *Indenpendent Component Analysis*, John Wiley and Son, 2001.
[4] M. Barker and W. Rayens, "Partial Least Squares for Discrimination", *Journal of Chemometrics*, vol. 17, pp. 166-173, 2003
[5] R. O. Duda, P. E. Hart, and D. Stock, *Pattern Classification*, Wiley, 2000.
[6] P. N. Belhumeour, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection", *IEEE Transacation on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.
[7] A. M. Martinez and A. C. Kak, "PCA versus LDA", *IEEE Transacation on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228–233, Feb. 2001.
[8] X. Liu, A. Srivastava, and K. A. Gallivan, "Optimal Linear Representation of Images for Object Recognition", *IEEE Transacation on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 662–666, 2004.
[9] A. Srivastava and X. Liu, "Tools for Application-Driven Dimensional Reduction", *Neuro Computation*, vol. 67, pp. 136–160, 2005.
[10] J. Ye and Q. Li, "A Two-Stage Linear Discriminant Analysis via QR-Decomposition", *IEEE Transacation on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 929–941, 2005.
[11] Y. Wu, X. Liu, W. Mio, and K. A. Gallivan, "Two-stage optimal component analysis", in *In Proceeding of IEEE International Conference on Image Processing*, Oct. 2006.
[12] F. Samaria and A. Harter, "Parameterisation of a Stochastic Model for Human Face Identification", in *Proceedings of 2nd IEEE Workshop on Applications of Computer Vision*, Sarasota FL, December 1994.
[13] T. Sim, S. Baker, and M. Bsat, "The cmu pose, illumination, and expression (pie) database of human faces", Tech. Rep. CMU-RI-TR-01-02, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, January 2001.
[14] A. M. Martinez and R. Benavente, "The AR Face Database", *CRC Technicle Reprot 24*, June 1998.
[15] X. Lu and A. K. Jain, "Resampling for Face Recognition", *AVBPA03*, pp. 869–877, 2003.
[16] A. M. Martinez, "Recognition Imprecisely Localized, Parially Occluded and Experession Variant Faces from a Single Sample per Class", *IEEE Transacation on Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 748–763, 2002.
[17] D. Roth, M. Yang, and N. Ahuja, "Learning to Recognition Objects", *Neural Computation*, vol. 14, no. 5, pp. 1071–1104, 2002.