Statistical Search for Hierarchical Linear Optimal Representations of Images

Qiang Zhang, Xiuwen Liu and Anuj Srivastava
Department of Computer Science, Florida State University, Tallahassee, FL 32306-4530
Email: {zhang, liux}@cs.fsu.edu
Department of Statistics, Florida State University, Tallahassee, FL 32306
Email: {anuj}@stat.fsu.edu

***Abstract*— Although linear representations of images are widely used in appearance-based recognition of objects, the frequently used ideas, such as PCA, ICA, and FDA, are often found to be suboptimal. A stochastic search algorithm has been proposed recently [4] for finding representations that are optimal for specific tasks and datasets. However, this search algorithm is computationally efficient only when the image size is relatively small. Here we propose a hierarchical learning algorithm to speed up the search. The proposed approach decomposes the original optimization problem into several stages according to a hierarchical organization. In particular, the following idea is applied recursively: (i) reduce the image dimension using a shrinkage matrix, (ii) optimize the recognition performance in the reduced space, and (iii)expand the optimal subspace to the bigger space in a pre-specified way. We show that the optimal performance is maintained in the last step. By applying this decomposition procedure recursively, a hierarchy of layers is formed. This speeds up the original algorithm significantly since the search is performed mainly in reduced spaces. The effectiveness of hierarchical learning is illustrated on a popular database, where the computation time is reduced by a large factor compared to the original algorithm.**

I. Introduction

Due to its simplicity, analytical tractability, and reported success on datasets for appearance-based object recognition, recently linear subspaces have become a popular choice for recognition. However, commonly used representations such as principal component analysis (PCA), Fisher discriminant analysis (FDA), and independent component analysis (ICA) are not optimal for a given application. Theoretically, PCA and ICA are designed for optimal reconstruction and statistical independence and thus are not directly related to recognition performance. While FDA is designed for optimal recognition performance, the optimality is valid only when the underlying probability distributions are Gaussian and linear discrimination functions are used[4]. As probability distributions of natural images are very different from Gaussian [7] and nonlinear discrimination functions are almost exclusively used in recognition applications, FDA is not optimal for recognition using images. This has been supported also by empirical evidence as contradictory conclusions are reported in the literature (e.g. [5], [1]). As recognition performance is critical for many applications, finding linear representations with optimal recognition performance for applications using images becomes an important topic. By formulating the problem of linear representation of images for recognition on a Grassmann manifold, Liu et al. [4] proposed a Markov chain Monte Carlo (MCMC) type algorithm that has been shown to be effective on all the used datasets and demonstrated significant improvement over commonly used bases. However due to its complexity, the algorithm is computationally effective only when the image size is relatively small. When datasets in typical applications are used, the algorithm is very time consuming as the search is done in a high dimensional space.

In this paper, we propose an algorithm that can speed up the original one [4] significantly. The key idea is to perform the search in shrinked image space and expand it back in such a way that the optimal performance is preserved. To be effective, the requirement of the dimension shrinkage is that the optimal performance in the reduced space should be as high as possible. In this paper, this is achieved through a heuristic-based algorithm, which is named as *adaptive K-means algorithm*. The procedure can be repeated and a hierarchy can be formed. Thus the method is named as *hierarchical learning*.

This paper is organized as follows. In Section *II* we overview the problem of optimizing the recognition performance over the set of subspaces. The analytical result of hierarchical learning is shown in Section *III* and experimental results are given in Section IV. Section V concludes the paper with a discussion.

II. Computing Optimal Linear Representations

Obtaining the optimal linear representation of images in terms of actual application performance requires the search in the linear representation space. Here we assume that the recognition performance depends only on the linear subspace and does not depend on the choice of bases in the given subspace. In other words, we assume the recognition performance on U is the same for any basis in the following set

$$[U] = \{UO | O \in I\!R^{d\times d},\ O^T O = I_d\} \quad \in \mathcal{G}_{n,d}\ ,$$

where U is a d-dimensional orthogal basis and O a $d \times d$ rotation matrix. For a given d, all the d-dimensional subspace in $I\!R^n$ is called a Grassmann manifold [2], denoted by $\mathcal{G}_{n,d}$. Thus the problem of finding optimal linear subspaces becomes an optimization on the Grassmann manifold. While there exist different optimization algorithms to solve this problem, a stochastic optimization process that maximizes the perfor-

mance function over all subspaces is used in [4], given by,

$$\hat{U} = \underset{U \in \mathcal{G}_{n,d}}{\operatorname{argmax}} F(U) \ , \tag{1}$$

in which $F : \mathcal{G}_{n,d} \mapsto I\!R_+$ is the performance function, and U is one basis of the given subspace.

Since the set $\mathcal{G}_{n,d}$ is compact, if F is a smooth function, then the optimizer $\hat{U}$ is well defined. The search is performed in a probabilistic framework by defining a probability density function

$$f(X) = \frac{1}{Z(T)} \exp(F(X)/T) \ , \tag{2}$$

where $T \in I\!R$ plays the role of temperature and f is an induced density on the set $\mathcal{G}_{n,d}$.

The algorithm requires an estimation of the recognition performance F for a given subspace. Because the recognition performance is not a smooth function, a smoothed performance measure related to the recognition performance using the nearest neighbor rule is introduced in [4]. Specifically, 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_{test} test images (denoted by $I'_{c,1}, \ldots, I'_{c,k_{test}}$) to evaluate the recognition performance measure.

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

where $h(\cdot, \cdot)$ is a monotonically increasing and bounded function in its first argument, and

$$\rho(I'_{c,i}, U) = \frac{\min_{c' \neq c, j} d(I'_{c,i}, I_{c',j}; U)}{\min_j d(I'_{c,i}, I_{c,j}; U) + \epsilon} \ . \tag{4}$$

Here $d(I_1, I_2; U) = \|\alpha(I_1, U) - \alpha(I_2, U)\|$, $\|\cdot\|$ denotes the 2-norm, $\alpha(I, U) = U^T I$ and $\epsilon > 0$ is a small number to avoid division by zero. In our implementation, following [4], $h(x, \beta) = 1/(1 + \exp(-2\beta x))$ is used, where β controls the smoothness of F. When we let $\beta \to \infty$, F is precisely the recognition performance of the nearest neighbor classifier [4].

In [4], a Markov chain Monte Carlo (MCMC) type gradient-based algorithm is used to find the optimal subspace $\hat{U}$. 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. In the original implementation used in [4], the computation complexity C_n of each iteration of this algorithm is $C_n = O(d \times (n-d) \times k_{test} \times k_{training} \times n \times d)$, where $d \times (n-d)$ 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 in Eqn. 4 and to compute the performance F in 3. The gradient calculation further requires computation of performances using perturbed subspaces. The complexity for one iteration can be obtained easily according to Eqn. 3. The overall computation complexity is $C \times t$ where t is the number of iterations.

III. Hierarchical Learning

From the above analysis, the computation at each iteration depends on several factors and the complexity is $O(n^2)$ in terms of n. For typical applications, n, which is the number of pixels in the image, is relative large. For example, in the face dataset (See Sect. IV) n is 2,576. Thus the algorithm can be very time consuming. Also when n gets larger, the dimension of the Grassmann manifold gets larger, resulting a search space with larger dimension. As the other factors in the complexity can not be avoided, here we propose an algorithm by shrinking the dimension of the search space first and performing the search in the reduced dimension. This reduces the complexity at each iteration and also the dimension of the search space.

There are several issues that need to be addressed. First, we are interested in finding optimal linear subspaces in the original space not the reduced space. How can we use the optimal subspace obtained in the reduced space? This question is answered by an analytical result which states that there exists a way to expand that optimal subspace to the larger original space in such a way that the performance is preserved in the expansion. In other words, if we can find an acceptable solution in the reduced space, we can immediately get a solution in the original space. The success of this procedure depends on whether the highest performance achievable in the reduced space is acceptable or not. Ideally, one would require a search over different shrinkage matrices to obtain the desired performance (see below). However, in this paper, the search is replaced by finding a good shrinkage matrix based on some heuristics that involve an adaptive K-means algorithm. Even under this setting, the optimal performance achieved in the reduced space may not be acceptable. In that case, one can use the result from this expansion as an initial condition for the search in the original space thereby reducing the number of required iterations.

In this section, we first show an analytical result between the performance in reduced space and the one in the original space. Then, we present an adaptive K-means algorithm to find a shrinkage matrix with a good performance.

A. Hierarchical Learning Algorithm

Before we present the hierarchical learning algorithm, we first state the following result that gives a specific way to expand a basis in a reduced space to the original space so that the performance is maintained.

Theorem: *Suppose* $N_1 = m \times N_2$ *where* $N_1, N_2, m \in Z^+$. *Let* $\hat{U} \in I\!R^{N_2 \times d}, U \in I\!R^{N_1 \times d}, d \in \mathcal{N}, I_1 \in I\!R^{N_1}, I_2 \in I\!R^{N_1}, \hat{I}_1 \in I\!R^{N_2}, \hat{I}_2 \in I\!R^{N_2}, A = [a_{ij}]_{N_2 \times N_1} \in I\!R^{N_2 \times N_1}$, *in which* $a_{ij} \in \{0, 1\}$ *for* $i = 1, \ldots, N_2; j = 1, \ldots, N_1$;

$$\sum_{j=1}^{N_1} 1(a_{ij}) = m, \sum_{i=1}^{N_2} 1(a_{ij}) = 1.$$

$$1(x) = \begin{cases} 0 & \textit{if } x = 0 \\ 1 & \textit{otherwise} \end{cases}$$

If $\hat{I}_i = \frac{1}{m} A I_i$, *for* $i = 1, 2$; $U = \frac{1}{m} A^T \hat{U}$.
Then $d(I_1, I_2; U) = d(\hat{I}_1, \hat{I}_2; \hat{U})$.

Proof: Using definitions of α and $d(\cdot, \cdot; \cdot)$, we get

$$\begin{aligned} \alpha(I_i, U) &= U^T I_i \\ &= \frac{1}{m} \hat{U}^T A I_i \\ &= \hat{U}^T \hat{I}_i \\ &= \alpha(\hat{I}_i, \hat{U}), \end{aligned} \tag{5}$$

for $i = 1, 2$.

Therefore,

$$\begin{aligned} d(I_1, I_2; U) &= \|\alpha(I_1, U) - \alpha(I_2, U)\| \\ &= \|\alpha(\hat{I}_1, \hat{U}) - \alpha(\hat{I}_2, \hat{U})\| \\ &= d(\hat{I}_1, \hat{I}_2; \hat{U}). \end{aligned} \tag{6}$$

EOP.

In the above theorem, we get the result without using the conditions $\sum_{j=1}^{N_1} 1(a_{ij}) = m$ and $\sum_{i=1}^{N_2} 1(a_{ij}) = 1$. The reason we put them there is that the adaptive kmeans algorithm(see below) requires A be that form.

Now let's consider how to compute the optimal basis $U \in I\!R^{N_1 \times d}$ in a more efficient way. Given the training and test images of size N_1, we can shrink their size to N_1/m by left multiplying matrix $(1/m)A$ with original training and test images, where A is a matrix that satisfies the conditions stated in the theorem. We call the dimension reduction process *shrinking the dimension* by a *shrinking factor* m through a shinkage matrix A. With the resulting new training and test images, we apply the optimization algorithm to search for $\hat{U} = \text{argmax}_{U \in \mathcal{G}_{N_1/m,d}} F(U)$. After $\hat{U} \in I\!R^{N_1/m \times d}$ is computed, we get $U \in I\!R^{N_1 \times d}$ by the equation $U = (1/m) A^T \hat{U}$. We call it *expanding basis* $\hat{U}$ to U. As the theorem states, $d(I_1, I_2; U) = d(\hat{I}_1, \hat{I}_2; \hat{U})$. As long as the performance function F is invariant with the distance between the representation of training and test images, the recognition rate will be unchanged. In this paper, F satisfies the above condition, because the nearest neighbor rule is used as the classifier. Thus we can get the basis $U \in I\!R^{N_1 \times d}$ by searching for basis $\hat{U} \in I\!R^{N1/m \times d}$.

For each iteration, the computation complexity with images of size N_1 is $C_{N_1} = O(d \times (N_1 - d) \times k_{test} \times k_{training} \times N_1 \times d)$. While the computation complexity with images of size N_1/m is

$$\begin{aligned} C_{N_2} &= O(d \times (\tfrac{N_1}{m} - d) \times k_{test} \times k_{training} \times \tfrac{N_1}{m} \times d) \\ &= \tfrac{N_1 - md}{m^2 (N_1 - d)} C_{N_1} \\ &\approx \tfrac{1}{m^2} C_{N_1}, \end{aligned} \tag{7}$$

considering the fact $N_1 >> d$.

From the above analysis we see that we can save time significantly on the search if we perform dimension shrinking on the original images using a large shrinking factor m. However, the best achievable performance using these reduced images can be much lower than that in the original space. To get the optimal basis, we have to search again with images of original size, which is potentially time consuming if the performance in the reduced space is considerably low.

Inspired by [6], to overcome this problem, we propose to perform the search hierarchically at different levels. The basic idea is as follows. Instead of choosing large m to shrink dimension only once, we can choose relatively smaller shrinking factor $\hat{m}$ to shrink the original images k times to get the training and test images at level k, where $k = 1, ..., L$. The size of images at level k is $\frac{n_0}{\hat{m}^k}$ where n_0 is the size of the original images. The search begins from level L with dimension-reduced images of size $\frac{n_0}{\hat{m}^L}$. The computational complexity at this level is $\frac{1}{\hat{m}^{2L}} C_{n_0}$ for each iteration. The search can be effectively done. After getting an optimal basis U_L at level L, we obtain basis $\bar{U}_{L-1}$ of level $L-1$ by expanding U_L. Based on the above discussion we know that $\bar{U}_{L-1}$ and U_L have the same performance. If we use $\bar{U}_{L-1}$ as an initial point to perform the search at level $L-1$ with images of size $\frac{n_0}{\hat{m}^{L-1}}$, the search can be performed relatively

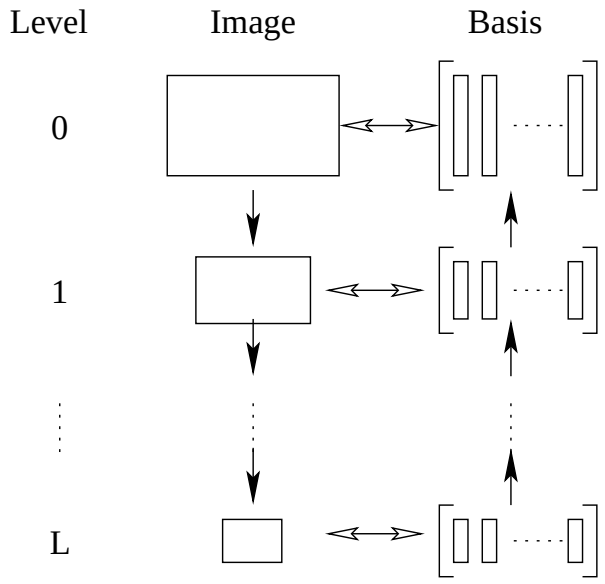

Fig. 1. Hierarchical search process. Image size is reduced as level number grows.Firstly, the optimal basis U_L is got at level L. We obtain basis $\bar{U}_{L-1}$ of level $L-1$ by expanding U_L. $\bar{U}_{L-1}$ and U_L have the same performance. We use $\bar{U}_{L-1}$ as an initial point to perform the search at level $L-1$ with images of size $\frac{n_0}{\hat{m}^{L-1}}$. The 'search – expand basis – search' process goes on till we get the optimal basis for level 0.

effectively. The computational complexity at this level is $\frac{1}{\hat{m}^{2(L-1)}}C_{n_0}$ for each iteration. 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. This process is illustrated in Fig. 1. The hierarchical learning algorithm is given below.

Hierarchical Learning Algorithm: Suppose we have $L+1$ levels, and the shrinking factor is m. The original image size is n_0. Our aim is to find the optimal basis of level 0.

1) Choose the dimension shrinkage matrices A_k, for $k = 1, \ldots, L$. Then prepare new training and test images at each level i: shrinking the image dimension on the training and test images for i times by multilplying $\frac{1}{m^i}\prod_{k=i}^{1} A_k$ with the original images.
2) Search for the optimal basis U_L at level L with training and test images of size $\frac{n_0}{m^L}$.
3) For each $k = L-1, \ldots, 0$, let $\bar{U}_k = \frac{1}{m}A_{k+1}^T U_{k+1}$, using $\bar{U}_k$ as the initial point, search for the optimal basis U_k at level k with training and test images of size $\frac{n_0}{m^k}$.

B. Pixels Grouping Through Adaptive K-means

For the hierarchical search algorithm to be effective, the best achievable performance in the original space should be kept as high as possible. There are many different ways to group pixels, i.e to choose dimension shrinkage matrix A, when performing dimension shrinkage. As an illustrative example, Fig. 2 shows two methods on two 4×4 images. In this simple example, one method is very effective in that any performance achievable in the 4×4 image space can be achieved in the reduced space and the other one is useless in that the reduced images do not provide any information for recognition any more.

Based on the example, we argue that we should group pixels with similar values in the original image together and represent them in the dimension reduced images by their mean. To achieve this, we propose a pixel grouping algorithm called adaptive K-means, which is an adaped version of original K-means algorithm [3]. This method is used in the hierarchical learning algorithm to generate shrinkage matrix A_i, which is desighed such that pixels that are clustered together contribute to the same coordinate in the reduced space. As all the training images should be shrunk in the same way, we put all of them in a matrix M of size $n \times N_1$ with each row of M is an image where n is the number of training images. N_1 is the size of the image, $N_1 = mN_2$ where m is the shrinking factor. We want to get a matrix $\overline{M}$ of size $n \times N_2$ where each column is the mean of m columns of M.

Let $M = [M_1 M_2 ... M_{N_1}]$ where $M_i (i = 1, .., N_1)$ is the i - th column of M. Treating M_i as a point in $\mathbb{R}^n$, we give the following algorithm to group the N_1 points $M_i (i = 1, .., N_1)$ to N_2 clusters $\overline{M}_i (i = 1, \ldots, N_2)$.

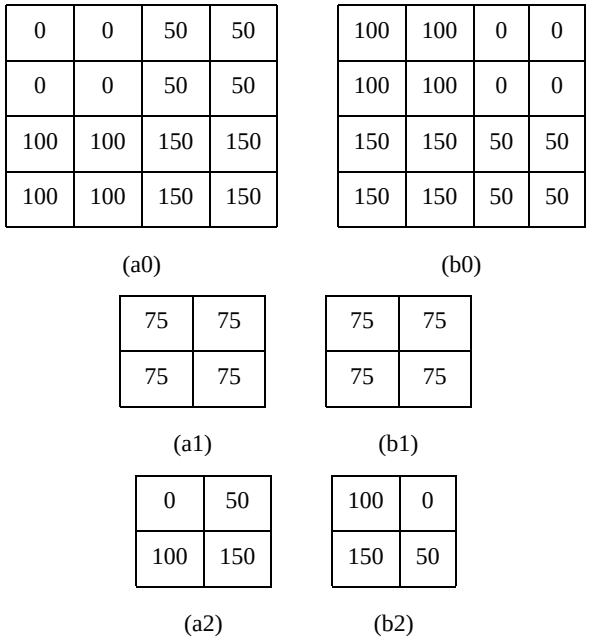

Fig. 2. An illustrative example for comparing two ways of shrinking. Two 4×4 images (a0) and (b0) are reduced using two different methods, giving two different 2×2 images. The recognition performance using the two images given by (a1) and (b1) is worse than that using the two images given by (a2) and (b2).

Adaptive K-means Algorighm: Let $S = \{M_i | i = 1, \ldots, N_1\}$. Randomly choose N_2 points $\overline{M}_i \in \mathbb{R}^n, i = 1, \ldots, N_2$.

1) For each $\overline{M}_i$, choose the nearest m points in S, group them into cluster $\overline{M}_i$. Remove the chosen points from S.
2) For each cluster $\overline{M}_i$, compute variance $\overline{V}_i$ and mean $(\overline{M}_i)_{new}$, set $\overline{M}_i = (\overline{M}_i)_{new}$.
3) Let $S = \{M_i | i = 1, \ldots, N_1\}$. Sorting according to $\overline{V}_i$, get list $L : \overline{V}_{i_1} \leq \overline{V}_{i_2} \leq \cdots \leq \overline{V}_{i_{N_2}}$, in which $(i_1, \cdots, i_{N_2})$ is a permutation of $(1, \cdots, N_2)$.
4) According to L, for each cluster $\overline{M}_{i_k}$ we choose the nearest m points in S and put them into this cluster. Remove the chosen points from S.
5) For each cluster $\overline{M}_{i_k}$, compute variance $\overline{V}_{i_k}$ and mean $(\overline{M}_{i_k})_{new}$. If $(\overline{M}_{i_k})_{new} = \overline{M}_{i_k}$ stop, else go to step 3.

IV. Experimental Results

The database that has been used in our experiments: the ORL face recognition dataset[1]. The ORL dataset consists of faces of 40 different subjects with 10 images each.

We plot the recognition rate versus iteration and the distance between X_t and X_0 versus t, in which X_t is the basis of the subspace at iteration t. The distance between any two subspaces U_1 and U_2 is computed as: $\|U_1 U_1^T - U_2 U_2^T\|$.

1) We run the hierarchical learning algorithm on the ORL dataset with L=3, original image size n is chosen to

[1]http://www.uk.research.att.com/facedatabase.html

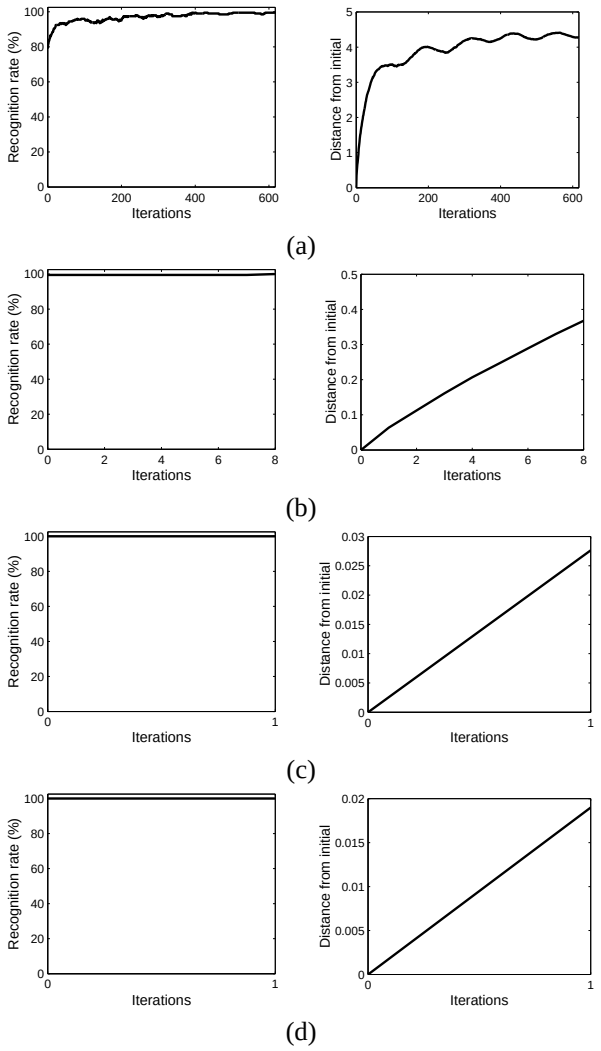

Fig. 3. Plots of recognition rate (left) and distance of X_t from X_0 (right) versus t for different level using hierarchical search algorithm. (a) level 3, (b) level 2, (c) level 1, (d) level 0. For these curves, $L = 3$, $m = 4$, $n = 2576$, $d = 10$, $k_{train} = 5$, and $k_{test} = 5$. Database is ORL.

be 2576, with the shrinking factor $m = 4$, $d = 10$, $k_{train} = 5$, and $k_{test} = 5$. Fig. 3 shows cases at different levels. We can see that it costs 600 iterations to achieve a perfect performance at level 3 (a). Due to numerical error, when we get the initial basis at level 2 by expanding the optimal basis got at level 3 the performance goes down a little bit. It takes 8 iterations again at level 2 to achieve perfect performance (b). ¿From (c) and (d) we see that it takes only one step to get the optimal basis at level 1 (c) and level 0 (d) respectively. This means that the performance is kept when expanding the optimal basis got at higher level to a basis at lower level. It's also worthy to note the plots of distance of X_t from X_0. At lower levels, it takes a small number of iterations with small moves from the initial point to get the optimal basis. Expending the basis from a higher level as the initial basis at the lower level does give a good guess of the optimal basis at the lower level.

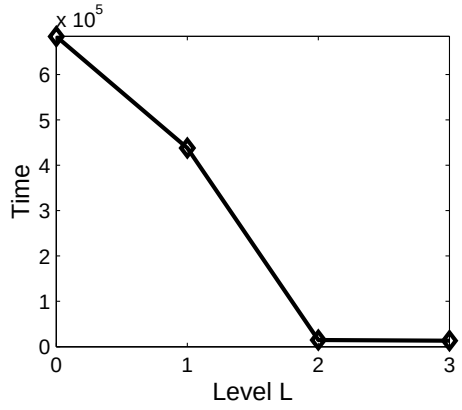

Fig. 4. Plot of searching time versus L. $n = 2576$, $m = 4$, $d = 10$, $k_{train} = 5$, and $k_{test} = 5$.

2) Next, we have studied the time for searching for the optimal basis versus the level number L. Fig. 4 shows the result, which highlights the effectiveness of hierarchical learning algorithm. In this experiment, the ORL database is used. n is chosen to be 2576, shrinking factor $m = 4$, $d = 10$, $k_{train} = 5$, and $k_{test} = 5$.
3) Fig. 5 shows the different cases when using a different level number L. In these experiments, we use the ORL dataset, n is chosen to be 2576, with the shrinking factor $m = 4$, $d = 10$, $k_{train} = 5$, and $k_{test} = 5$. All the cases converge to a perfect classification solution regardless of the level number. One point to mention is that after doing the dimension shrink, the recognition rate may be even higher than that of the original images. For instance, in Fig. 5 the initial recognition rate is 73% with dimension reduced images(c), but the initial recognition rate is 60% with the original images (a).

V. Discussion

In this paper, we have proposed a hierarchical learning algorithm for finding optimal linear representation. By grouping pixels using an adaptive K-means algorithm, we speed up the original search on the dataset significantly. This makes it computationally feasible and effective to find optimal linear representations for typical applications with large images. The significance of this work is that now it is possible to study generalization and other fundamental issues in linear representations systematically.

Note that the efficiency is gained by decomposing the search on a large Grassmann manifold to a number of hierarchically organized Grassmann manifolds with small dimensions. The effectivenness of the algorithm depends on the best achievable performance in the reduced dimension. While the proposed adaptive K-means algorithm is shown to be effective for the dataset we have used, its effectiveness is not guaranteed. The conditions under which the algorithm is effective needs to be further investigated. Another future research issue is to find better pixel grouping algorithms to generate the shrinkage

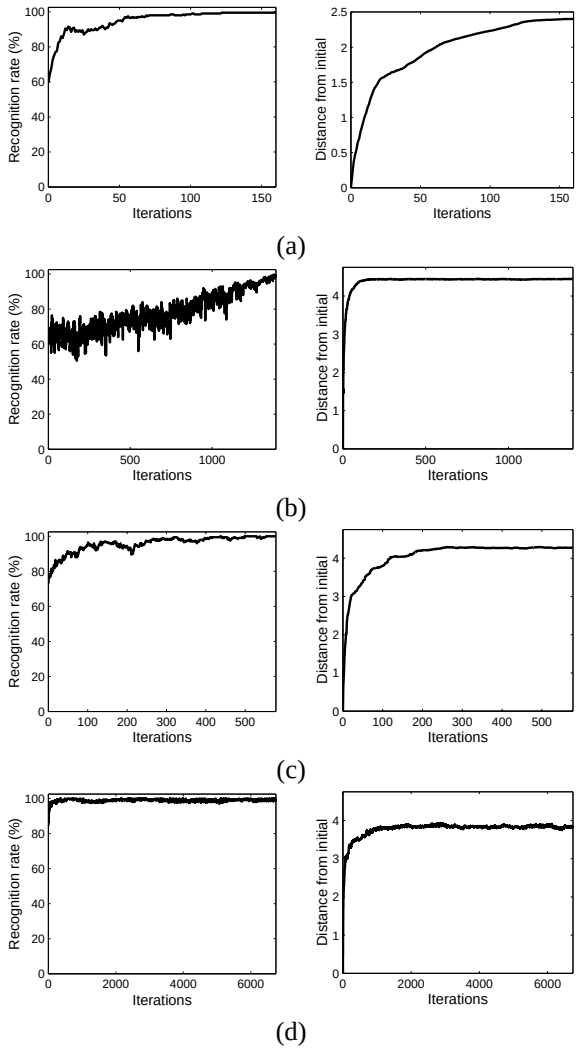

Fig. 5. Plots of recognition rate (left) and distance of X_t from X_0 (right) versus t at the highest level for different L. (a) L = 0, level 0, (b) L = 1, level 1, (c) L = 2, level 2, (d) L = 3, level 3. For these curves, $n = 2576$, $m = 4$, $d = 10$, $k_{train} = 5$, and $k_{test} = 5$.

matrix A.

Acknowledgments This research was supported in part by an NIMA grant (NMA201-01-2010).

REFERENCES

[1] P. N. Belhumeur, J. P. Hepanha and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, " *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19(7), pp. 711–720, 1997.

[2] W. M. Boothby, *An Introduction to Differential Manifolds and Riemannian Geometry*, Academic Press, 1986.

[3] D. A. Forsyth and J. Ponce, *Computer Vision, A Modern Approach*. pp. 315-317. Prentice Hall, 2003.

[4] X. Liu, A. Srivastava and K. Gallivan, "Optimal linear representations of images for object recognition, " in Proceedings of *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2003.

[5] A. M. Martinez and A. C. Kak, "PCA versus LDA, " *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23(2), pp. 228–233, 2001.

[6] A. Rosenfeld, "Image analysis: problems, process and prospects, " *Readings in computer vision*, pp. 3–12, Morgan Kaufmann Publishers, 1987.

[7] A. Srivastava, A. B. Lee, E. P. Simoncelli and S. C. Zhu. "On advances in statistical modeling of natural images, " *Journal of Mathematical Imaging and Vision*, vol. 18(1), 2003.