Sixth International Conference on Machine Learning and Applications

Scalable Optimal Linear Representation for Face and Object Recognition

Yiming Wu, Xiuwen Liu
Department of Computer Science
Florida State University
{ywu, liux}@cs.fsu.edu

Washington Mio
Department of Mathematics
Florida State University
mio@math.fsu.edu

Abstract

Optimal Component Analysis (OCA) is a linear method for feature extraction and dimension reduction. It has been widely used in many applications such as face and object recognitions. The optimal basis of OCA is obtained through solving an optimization problem on a Grassmann manifold. However, one limitation of OCA is the computational cost becoming heavy when the number of training data is large, which prevents OCA from efficiently applying in many real applications. In this paper, a scalable OCA (S-OCA) that uses a two-stage strategy is developed to bridge this gap. In the first stage, we cluster the training data using K-means algorithm and the dimension of data is reduced into a low dimensional space. In the second stage, OCA search is performed in the reduced space and the gradient is updated using an numerical approximation. In the process of OCA gradient updating, instead of choosing the entire training data, S-OCA randomly chooses a small subset of the training images in each class to update the gradient. This achieves stochastic gradient updating and at the same time reduces the searching time of OCA in orders of magnitude. Experimental results on face and object datasets show efficiency of the S-OCA method, in term of both classification accuracy and computational complexity.

1. Introduction

In the past decades, scalable algorithm have been widely used on the applications such as speech recognition [1], video processing [2]. To our knowledge, there are two streamline in dealing with the large data problem. One is using hardware accelerators, such as FPGA [3], supercomputer [4]. The other is based on software development, such as Hidden Markov Model (HMM) [5], parallel algorithms [6] and wavelet transform [7]. Recently, several novel scalable algorithms based on linear representation are proposed. Based on the discovery that natural images exhibit structure in a low-dimensional subspace, Chennubhotla etc. developed a sparse Principal Component Analysis (S-PCA) algorithm which achieves the recognition of larger number of data [8]. Yan etc. proposed a novel scalable algorithm for supervised subspace learning method called as supervised kampong measure (SKM) [9]. It assigns data points as close as possible to their corresponding class mean, simultaneously assign data points to be as far as possible from other class mean in the transformed lower dimensional subspace, thus theoretically shows the algorithm is not limited by the number of classes or the singularity problem faced by LDA. Ye etc. proposed a two-stage Linear Discriminant Analysis (LDA) method named LDA/QR [10]. The first stage of this method applies QR decomposition on a small matrix involving class centroid. The second step applies LDA to the "reduced" scatter matrices resulting from the first stage. which shows good scalability in terms of both the number of original data dimensions and the number of training data points. However, an implicit assumption in the first stage of this method is the data in the same class is with a single Gaussian distribution. Although we can take the assumption in most cases, it is not a accurate representation for some datasets where large variance between data in the same class exits.

Optimal Component Analysis (OCA) [11] is a linear presentation algorithm that addresses problem of learning an optimal linear representation for a particular classification task. The search for an optimal basis subspace is based on a stochastic gradient process that seeks to maximize a specified performance function over all subspaces of a Grassmann manifold. A solution is obtained by conducting a search over the Grassmannian with a stochastic searching algorithm. OCA provides a computational framework for finding optimal linear representations for particular applications and its effectiveness has been demonstrated in many real applications.

Although OCA shows good recognition performance on many datasets, it suffers from the fact that the computational cost is heavy when the number of training data is large, which prevents OCA from efficiently applying in many real datasets. In this paper, a scalable version OCA algorithm,

0-7695-3069-9/07 $25.00 © 2007 IEEE
DOI 10.1109/ICMLA.2007.110

named S-OCA, is proposed to bridge this gap. We treat the recognition process in two stages. In the first stage, we cluster the training images in same class using K-means algorithm. Then, by using the transformation matrix which is obtained by applying Singular Value Decomposition (SVD) on the centroid of each cluster, we reduce data into low dimension space. In the second step, we run the OCA search in the low dimensional Grassmann manifold. The gradient is updated using a numerical approximation. However, instead of using all of the training data in each class in the original OCA, we update the gradient by using a small subset of training data which are random chose from the entire training data. This achieves stochastic gradient updating and at the same time the optimal basis searching time is reduced dramatically.

We apply the proposed algorithm on ORL face dataset and $COIL$ object dataset. Experiment shows that S-OCA achieves good performance in term of both recognition accuracy and computational complexity.

2 Optimal Component Analysis

2.1 Optimization for recognition

Optimal Component Analysis is a dimension reduction technique that finds an optimal subspace (of a prescribed dimension) of feature space that optimizes the ability of the nearest neighbor classifie to index and classify images or more general data. The measurement of optimality is based on training data and the algorithm yields an orthonormal basis of the estimated optimal subspace.

More specifically, let $U \in \Re^{n \times d}$ be a matrix whose columns form an orthonormal basis of a d-dimensional subspace of $\Re^n$, where n is the size of the input image and d is the dimension of the desired subspace (generally $n \gg d$). For an image I, considered as a column vector of size n, the vector of coefficients is given by $\alpha(I,U) = U^T I \in \Re^d$ and represents the orthogonal projection of I onto the subspace S_U spanned by the columns of U. Suppose the training data consists of representatives of C classes of images, with each class represented by k_{train} training images (denoted by $I_{c,1}, \ldots, I_{c,k_{train}}$) and k_{cross} cross validation images (denoted by $I'_{c,1}, \ldots, I'_{c,k_{cross}}$). Let

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

The numerator is the distance from $I'_{c,i}$ to the closest training image not in its class and the denominator is the distance from $I'_{c,i}$ to the closest training image in the same class. Here, D denotes Euclidean distance; that is,

$$D(I_1, I_2; U) = \|\alpha(I_1, U) - \alpha(I_2, U)\|,$$

where $\|\cdot\|$ is the usual 2-norm. In Eq. (1), $\epsilon > 0$ is a small number introduced to avoid division by zero. Note that large values of ρ are desirable, since this means that $I'_{c,i}$ will be closer to its class than to other classes in the subspace S_U. A performance function F is defined to essentially measure the average value of ρ over all cross-validation images, as follows:

$$F(U) = \frac{1}{Ck_{cross}} \sum_{c=1}^{C} \sum_{i=1}^{k_{cross}} h(\rho(I'_{c,i}, U) - 1) \,, \qquad (2)$$

where h(·) is a monotonically increasing bounded function used to control bias with respect to particular classes in measurements of performance. In our implementation, we use $h(x) = 1/(1 + \exp(-2\beta x))$, where β is a parameter that controls the degree of smoothness of $F(U)$. Thus, F is a quantifier of the ability of the nearest neighbor classifier to discern the C classes after projection onto S_U. Moreover, as $\beta \to \infty$ and $\epsilon \to 0$, F gives precisely the recognition performance of the nearest neighbor classifier after projection to the subspace given by U [11].

Under this formulation, $F(U) = F(UH)$ for any $d \times d$ orthogonal matrix H. This is the case because F depends only on distances in S_U and right multiplication by H changes the orthonormal basis, but not the subspace S_U. Therefore, our search for optimal representation can be viewed as an optimization problem on the space of d-dimensional subspaces rather than the space of orthonormal frames. The Grassmann manifold, $\mathcal{G}(n,d)$, is the set of all d-dimensional subspaces of $\Re^n$. It is a compact, connected manifold of dimension $d(n-d)$, which can be represented either by a basis (non-uniquely) or by a projection matrix (uniquely). Choosing the former, let U be an $n \times d$ matrix whose columns are an orthonormal basis for the given subspace of $\Re^n$ and let $[U]$ denote the set of all the orthonormal bases of S_U, i.e., $[U] = \{UH | H \in \Re^{d \times d}, H^T H = I_d\} \in \mathcal{G}(n,d)$. The remarks above imply that F is a function of $[U]$, not just U. Unlike the actual recognition performance, $F([U])$ is smooth and thus allows us to use a gradient-type algorithm to solve the optimization problem. An optimal d-dimensional subspace for the given classification problem from the viewpoint of the available data is given by

$$[\hat{U}] = \underset{[U] \in \mathcal{G}_{n,d}}{\operatorname{argmax}} F([U]) \qquad (3)$$

2.2 Optimal basis search

In [11], an optimization algorithm utilizing the geometric properties of the manifold is presented. A Monte Carlo version of a stochastic gradient-based algorithm with simulated annealing is used to find an optimal subspace $\hat{U}$. Since the gradient search is conducted over a Grassmann manifold, the process has to account for its intrinsic geometry.

We now review the MCMC-type simulated annealing process presented in [11].

Let J be the $n \times d$ matrix given by the first d columns of the $n \times n$ identity matrix. Complete the orthonormal set U to an orthonormal basis of $\Re^n$ and let Q be the corresponding $n \times n$ orthogonal matrix. Then, the gradient of F at $[U]$ is given by $A([U])J$, where

$$A([U]) = Q \sum_{i=1}^{d} \sum_{j=d+1}^{n} \alpha_{ij}(U) E_{ij} \in \Re^{n \times n}, \tag{4}$$

where

$$\alpha_{ij}(U) = \lim_{\epsilon \to 0} \frac{F(Qe^{\epsilon E_{ij}} J) - F(U)}{\epsilon} \tag{5}$$

is the directional derivatives of F in the directions given by E_{ij}. Here E_{ij} is an $n \times n$ skew-symmetric matrix

$$E_{ij}(k,l) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } k = i,\ l = j, \\ -\frac{1}{\sqrt{2}} & \text{if } k = j,\ l = i, \\ 0 & \text{otherwise}, \end{cases}$$

$1 \leq i \leq d$ and $d < j \leq n$. The matrices $E_{ij}J$ represent an orthonormal basis of the vector space tangent to $\mathcal{G}(n,d)$ at $[J]$. The deterministic gradient flow is a solution of the equation

$$\frac{dU_t}{dt} = A(U_t)J\,, \tag{6}$$

where U_t is the solution at time t. Computationally, we discretely update U_t according Eq. (6) and at each iteration, the gradient vector of F with respect to U_t is computed. This gives rise to a deterministic gradient optimization algorithm that is intrinsic to the Grassmann manifold, i.e., every new solution is guaranteed to be on the manifold given that U_0 is. This algorithm shares the limitations of all deterministic gradient algorithms and it will not be able to escape a local maximum. To overcome this problem, in [11] stochastic optimization is used by first perturbing the gradient randomly and then using a Markov chain Monte Carlo process. A proposed subspace is accepted with a probability that depends on the performance improvement and an annealing parameter. If the performance on the new subspace is better than that of the current solution, it is always accepted; otherwise, the worse the performance, the lower the probability of the subspace being accepted. This guarantees that a global optimal solution[1] can be reached given that the Markov chain is sufficiently long. For details, see [11].

3 Scalable OCA

OCA shows good performance on many datasets, however, when the number of training data is large, the computational complexity is heavy and it becomes an obscure for its efficient applications on these data sets. This motivates the idea of S-OCA algorithm where a two-stage strategy is used. In the first stage, we cluster the training data into several clusters using K-means algorithm. Then, we reduce the data dimension into low dimension space, by using transform matrix obtained through applied SVD on the centroid of each cluster. In the second stage, the OCA searching is performed in the low dimensional Grassmann manifold. However, in the process of gradient updating, instead of using all the training images in each class, only a small subset of training image in each class is randomly chosen to update the gradient. It automatically brings stochastic property to the gradient updating process of OCA searching. Furthermore, compared with the original OCA algorithm, S-OCA speeds up the search process dramatically, which enables OCA applying on large datasets.

3.1 First stage: K-means clustering and dimensional reduction

In [10], a two-stage LDA method proposed to overcome the singularity problem of LDA. The first stage of this method maximizes the separation between different classes via QR decomposition on a small between-class scatter matrix involving the class centroid. This method gains efficiency in both time and space costing.

Similarly, in the first stage of S-OCA algorithm, we first cluster images in each class using K-means clustering algorithm. Without loss of generality, we represent each class by K clusters instead of one cluster which is used in LDA/QR, as many datasets do not satisfy the assumption that the data in same class is with one single Gaussian distribution. Object in $COIL$ data sets is an example, where there are 72 images for each object which were take at pose intervals of 5 degree. Obviously, it is not accurate to use the centroid of each class to represent all the data in each class. However, by using K-means algorithm, we cluster the data in the one class into several clusters, and the centroid of each cluster gives a more accurate representation of the data in one class.

After we cluster data in each class, we reduce dimension of training and test data from original high dimension space into a low dimensional space. We define the the between-cluster scatter matrix as follows:

$$S_b = \frac{1}{N} \Sigma_{i=1}^{k} N_i (m_i - m)(m_i - m)^T = H_b H_b^T$$

And the precursor H_b of the between-cluster scatter matrices is computed as follows:

$$H_b = \frac{1}{\sqrt{N}} [\sqrt{N_1}(m_1 - m), \cdots, \sqrt{N_K}(m_K - m)] \tag{7}$$

here N is the number of clusters in the total training set, m_i is the centroid of the ith cluster, m is the global centroid of

[1] Note that the solution of Eq.(3) can be a set rather than a unique subspace.

the training data set and K is the number of clusters. Then, the general linear dimensional reduction methods, such as Singular Value Decomposition (SVD) is applied in H_b.

$$H_b = ASV^T \tag{8}$$

where $A \in \Re^{n_0 * n_1}$ is the eigenvector matrix, n_0 is the original data dimension, n_1 is the reduced data dimension in the first stage, S is the diagonal matrix consisting of the set of all eigenvalues of H_b's covariance matrix C. V is the matrix consisting of the set of all eigenvectors of C. After we get the eigenvector matrix A, we reduce the original data from original dimension n_0 to low dimension n_1.

3.2 Second stage: OCA search on the low dimensional Grassmann manifold

In the second stage of S-OCA, OCA is applied on the low dimensional Grassmann manifold. For global optimization, in [11], a stochastic component is added to gradient updating in Eq.(6). In the S-OCA algorithm, the gradient of F is numerically approximated using a finite differences:

$$\alpha_{ij} = \frac{F(\tilde{U}) - F(U)}{\epsilon}, 1 \leq i \leq d, d < j \leq n, \tag{9}$$

for a small $\epsilon > 0$. Here, the matrix $\tilde{U} = Q^T e^{\epsilon E_{ij}} J$ is an $n \times d$ matrix that differs form U in only the i^{th}-column which is now given by

$$\tilde{U_i} = \cos(\epsilon) U_i + \sin(\epsilon) V_j, \tag{10}$$

where U_i, V_j are the i^{th} and j^{th} columns of U and V, respectively, with $V \in \Re^{n \times (n-d)}$ is any matrix such that $VV^T = I_{n-d}$ and $U^T V = 0$.

For a step size $\delta > 0$, we will denote the search process at discrete times $U(t\delta)$ by U_t. Then, a discrete approximation of the solution of Eq.(6) is given by,

$$U_{t+1} = Q_t^T exp(\delta A_t) J, \tag{11}$$

where $A_t = \sum_{i=1}^{d} \sum_{j=d+1}^{n} \alpha_{ij}(U_t) E_{ij}$ and $Q_{t+1} = exp(-\delta A) Q_t$.

Note that in the original OCA, the performance function F is computed as the average of function h on all the training images in all class. That mean, in order to update F, we should compute h on all images in all the class. This is inefficient when the number of training data is in large scale. In S-OCA, instead of update F on all the training images in each class. We random choose a small subset image in each class and update F accordingly. That is,

$$F(U) = \frac{1}{Ck_{subtrain}} \sum_{c=1}^{C} \sum_{i=1}^{k_{subtrain}} h(\rho(I'_{c,i}, U) - 1) \tag{12}$$

Algorithm 1: Scalable OCA Algorithm

Input: Data matrix $I = I_{train} + T_{test}$, where $I \in \Re^{n_0 \times N}$; the number of clusters in each class K; the reduced dimension in the first stage n_1; the OCA search iteration T; OCA optimal subspace d;
Output: Reduced data matrix I^L

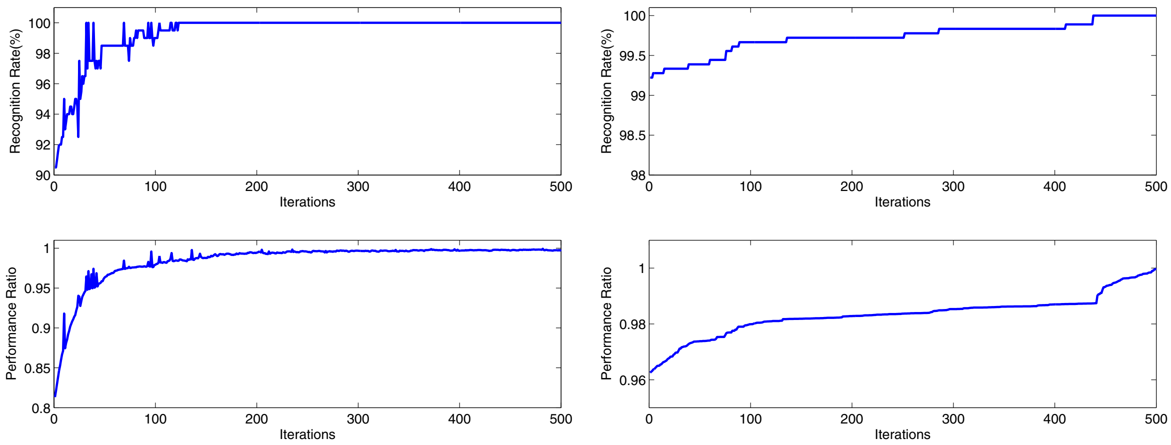
```
Stage I:
  1. Apply K-Means clustering on $I_{train}$.
  2. Construct the matrix $H_b$ as Eq.(7), where $H_b \in \Re^{n_0 \times K}$.
  3. Apply $SVD$ to $H_b$ as $H_b = ASV^T$ and obtains $A \in \Re^{n_0 \times n_1}$.
  4. Get low dimensional data $I_1 \leftarrow A^T I$, where $I_1 \in \Re^{n_1 \times N}$.
Stage II:
  5. Let $U_0 \in \mathcal{G}(n_1, d)$ be any initial basis.
  6. for t = 1:T-1,
      (a) Compute $F(U_t)$ according to Eq.(1) and (12).
      (b) Using the value of $U_t$, generate a candidate value $Y$ according to Eq.(11).
      (c) Compute F(Y) according to Eq.(1) and (12).
      (d) If $F(Y)) > F(U_t)$, $U_{t+1} = Y$, otherwise, $U_{t+1} = U_t$.
     end for
  7. $\hat{U}^T \leftarrow U_{t+1}$
  8. $I_L \leftarrow \hat{U}^T I_L$
```

where $k_{subtrain}$ is the number of training images chosen for each class.

As the images in each class is randomly selected, it adds stochastic gradient updating for F, which is important for avoiding local maxima of OCA. Furthermore, the OCA search will be sped up, as only a subset of training images in each class is selected for gradient updating. The pseudocode for this algorithm is given in **Algorithm 1**.

3.3 Time complexity analysis

Now we have a look of the computational complexity of OCA and S-OCA. For S-OCA, we only consider the time consuming in the second step as the time consuming in the first stage is ignorable when compared with that of the second stage. The computational complexity of each iteration of the 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 observations. The dimension of the gradient vector is $d \times (n-d)$, which can be seen from Eq. (6) (as there are $d \times (n-d)$ E_{ij}'s). For each E_{ij}, in order to compute $\alpha_{ij}(U)$, we need to compute $F(e^{\epsilon E_{ij}} U)$, which requires to compute the ratio (Eq.



(a) (b)

Figure 1. Evolution of performance $F(U_t)$ and recognition accuracy versus t on (a) ORL dataset; (b) $COIL$ dataset

(1)) for each test image, which again requires a search of the closest training image in the same class and the closest in other classes. Therefore estimating the gradient requires the given computational complexity. By exploiting the structure of $A(U)$, an $O(n)$ updating algorithm can be achieved and thus it can be ignored. The overall computational complexity is therefore $C_n \times T$, where T is the number of iterations.

Note that the OCA algorithm requires solving an optimization problem with dimension of $d \times (n - d)$; for recognition applications based on images, n is typically on the order of $10,000$. In [11], OCA has been implemented and demonstrated on recognition problems with n of $10,000$. However, due to its computational complexity, OCA has not been used widely for recognition applications.

The search space for the S-OCA, however, is reduced to the low Grassmann manifold with dimension of $d \times (n_1 - d)$, and we use a subset of training image instead of all the training images in each class to update F, thus, the computational complexity for each iteration of the scalable OCA is $C_1 = O(d \times (n_1 - d) \times k_{test} \times k_{subtrain} \times n_1 \times d)$. Compared with original OCA, we can expect a factor of $\frac{n_0 \times k_{train}}{n_1 \times k_{subtrain}}$ improvement in terms of computational complexity.

4 Experimental results

We have applied the scalable OCA algorithm to the search for optimal linear basis successfully on a variety of datasets. Due to the limited space, we restrict to two datasets: the ORL face recognition data set [2] and one object dataset: the $COIL$ dataset [3]. Table 1 shows the experiment setting of these two data sets. For both datasets, we use half images from each class for training and half for testing.

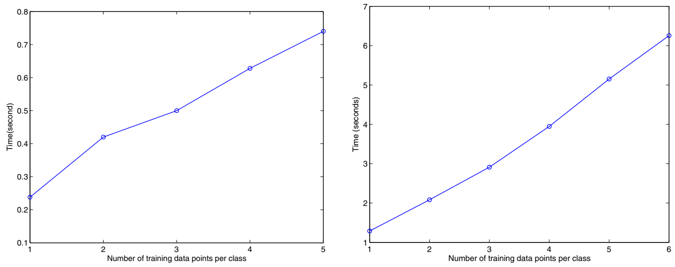
Dataset	n_0	n_1	C	N	K	d
ORL	10,304	40	40	400	5	10
COIL	1,024	20	100	7200	6	10

Table 1. Statistics for our Real Test Data Sets

Figure 1 shows the evolution of performance function $F(U_t)$ and the recognition rate. As the space is limited, in this paper, we only show the case when $k_{subtrain}$=1 for both datasets. We can see the F can achieve near 1 through 500 iterations running and the classification accuracy reaches 100% in both datasets. Compared to the recognition performance of original OCA, we can see the recognition performance is not decreased with a smaller number of training set is used. Figure 2 shows the time cost of each iteration running of the S-OCA algorithm with respect to the number of the subset of training data being randomly chosen. We can observe that the running time is nearly linear with the number of training images. These results confirm the theoretical complexity analysis in section 3.3. When compared with the original OCA, the running time of S-OCA is reduced in the order of magnitude. Take the experimental result on ORL data sets for example, it takes original OCA about 173 seconds for one iteration, while for the S-OCA, it only takes 0.1 second for one iteration, with the same experimental setting.

[2] http://www.cl.cam.ac.uk/research/dtg/attarchive/facesataglance.html

[3] http://www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php



(a) (b)

Figure 2. Scalability of scalable OCA with respect to the number of subset of training images on (a) ORL dataset; (b) $COIL$ dataset

5 Conclusion

OCA obtains the optimal basis through a stochastic search process on a Grassmann manifold and shows good recognition performance on the many applications. However, the computational complexity is heavy when OCA is applied to the datasets with large number of training data. In this paper, a scalable OCA method using a two stage strategy is proposed. In the first stage, the data in each class is clustered into several clusters by using K-means algorithm and the dimension of training and test data is reduced into a low dimension space. In the second stage, OCA search is performed in the low dimension space where a numerical approximation method is used for gradient updating. In the process of optimal basis searching, instead of using all the training images in each class, a small subset of image from the training set of each class is randomly chosen to update the gradient of performance function F. When compared with the original OCA, the computational cost is reduced in magnitude order while the recognition performance is kept.

The experimental results on ORL and $COIL$ datasets show promising performance of the S-OCA algorithm, in term of both of the recognition accuracy and running time. More experiments on complex datasets will be conducted in the future.

References

[1] W. M. Campbell, K. T. Assaleh, and C. C. Broun, "Speaker recognition with polynomial classifiers", *IEEE Transaction on Speech and Audio Processing*, vol. 10, pp. 205-212, 2002.

[2] C. Hentschel, R. Braspenning and M. Gabrani, "Scalable algorithms for media processing", in *Proceedings of IEEE International Conference on Image Processing*, vol.3, pp. 342-345, 2001.

[3] R. Sivilotti, C. Young, S. Wen-King, D. Cohen and B. Bray, "Scalable network based FPGA accelerators for an automatic target recognition application", *Proceeding of IEEE Symposium Digital Object Identifer*, vol. 10, pp. 282-283, 1998.

[4] H. Wang, A. Nicolau, S. Keung and K. Siu, "Computing programs containing band linear recurrences on vector supercomputers", *IEEE Transaction on Parallel and Distributed Systems*, vol. 7, pp. 769-782, 1996.

[5] S. Chakrabartty, G. Singh, and G. Cauwenberghs, "Hybrid Support Vector Machine/Hidden Markov Model approach for continuous speech recognition", *IEEE Midwest Symposium on Circults and Systems*, vol. 2, pp. 828-831, 2000.

[6] L. H. Jamieson, E. J. Delp, S. E. Hambrusch, A. A. Khokhar, G. W. Cook, F. Hameed, J. N. Patel and S.Ke, "Parallel scalable libraries and algorithms for computer vision", in *Proceedings of IEEE International Conference on Pattern Recognition*, vol.3, pp. 223-228, 1994.

[7] H. Danyali and A. Mertins, "Flexible, highly scalable, object-based wavelet image compression algorithm for network applications", in *IEE Proceedings on Vision, Image and Signal Processing*, vol.151, pp. 498-510, 2004.

[8] C. Chennubhotla, A. Jepson, "Sparse PCA: Extracting multi-Scale structure from data", in *Proceeding of IEEE International Conference on Computer Vision*, vol. 1, pp. 641-647, 2001.

[9] J. Yan, N. Liu, B. Zhang, Q. Yang, S. Yan and Z. Chen, "A novel scalable algorithm for supervised subspace learning", in *Proceeding of IEEE International Conference on Data Mining*, 2006.

[10] J. Ye and Q. Li, "A two-stage linear discriminant analysis via QR-decomposition", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 929-941, 2005.

[11] X. Liu, A. Srivastava, and K. A. Gallivan, "Optimal linear representation of images for object recognition", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 662-666, 2004.