Successive Learning of Linear Discriminant Analysis: Sanger-type Algorithm

K. Hiraoka, K. Hidai, M. Hamahira, H. Mizoguchi, T. Mishima, and S. Yoshizawa
Saitama University
Department of Information and Computer Sciences
255 Shimo-okubo, Urawa, 338-8570, Japan
hira@ics.saitama-u.ac.jp

Abstract

Linear discriminant analysis (LDA) is applied to broad areas, e.g. image recognition. However, successive learning algorithms for LDA are not sufficiently studied while they have been well established for principal component analysis (PCA).

Recently, a successive leaning algorithm which does not need $N \times N$ matrices has been proposed for LDA, where N is the dimension of data. In the present paper, an improvement of this algorithm is examined based on Sanger's idea. By the original algorithm, we can obtain only the subspace which is spanned by major eigenvectors. On the other hand, we can obtain major eigenvectors themselves by the improved algorithm.

1. Introduction

Linear discriminant analysis (LDA) is applied to broad areas, e.g. image recognition [5]. However, successive learning algorithms for LDA are not sufficiently studied while they have been well established for principal component analysis (PCA) [8][9][10]. Conventional learning methods for LDA [6][1] have a disadvantage that $N \times N$ matrices must be kept and updated, where N is the dimension of data. This is a serious problem because N is often large for some applications such as image recognition tasks.

Recently, a successive leaning algorithm which does not need $N \times N$ matrices has been proposed for LDA [2][3][4]. In the present paper, an improvement of this algorithm is examined based on Sanger's idea [10]. By the original algorithm, we can obtain only the subspace which is spanned by major eigenvectors. On the other hand, we can obtain major eigenvectors themselves by the improved algorithm.

In section 2, the improved algorithm is proposed. The algorithm is shown in two forms. One is easy to understand and the other is efficient in calculation. They are different only on the expression and both forms yield the exactly same result. In section 3, the proposed algorithm is applied to the face recognition task and the performance is confirmed. In section 4, the conclusion is mentioned. The derivation of the proposed algorithm is explained in appendix A.

2. Sanger-type Algorithm

To estimate the class for a data vector $\boldsymbol{x}$ by LDA, we transform it to the "feature" vector $y = A^T \boldsymbol{x}$ and compare y with the mean feature vector of each class. In online LDA, the discriminant matrix A is updated every time a new datum is presented. The derivation of algorithm is shown in appendix.

2.1. Basic Form

At every time step $t = 1, 2, 3, \cdots$, a new pair $(\boldsymbol{x}(t), c(t))$ is presented, where $\boldsymbol{x}(t)$ is an N-dimensional data vector, $c(t) \in \{1, \cdots, M\}$ is the class of $\boldsymbol{x}(t)$, and M is the number of classes. Based on this pair, auxiliary variables are updated as follows:

$$t^c(t) = t^c(t-1) + \delta(c, c(t)), \tag{1}$$

$$\bar{\boldsymbol{x}}(t) = \left(1 - \frac{1}{t}\right) \bar{\boldsymbol{x}}(t-1) + \frac{1}{t}\boldsymbol{x}(t), \tag{2}$$

$$\bar{\boldsymbol{x}}^c(t) = \begin{cases} \left(1 - \frac{1}{t^c(t)}\right) \bar{\boldsymbol{x}}^c(t-1) + \frac{1}{t^c(t)}\boldsymbol{x}(t) & (c = c(t)), \\ \bar{\boldsymbol{x}}^c(t-1) & (c \neq c(t)), \end{cases} \tag{3}$$

$$v^c(t) = \bar{\boldsymbol{x}}^c(t) - \bar{\boldsymbol{x}}(t), \tag{4}$$

$$w(t) = \boldsymbol{x}(t) - \bar{\boldsymbol{x}}^{c(t)}(t), \tag{5}$$

1051-4651/00 $10.00 © 2000 IEEE

$$B(t) = \frac{1}{M}\sum_{c=1}^{M} v^c(t)v^c(t)^T, \qquad (6)$$

where $c = 1, \cdots, M$ and $\delta(c, c(t)) = 1\ (c = c(t)),\ 0\ (c \neq c(t))$. Then $N \times L$ discriminant matrix A is updated:

$$\begin{aligned} A(t) &= A(t-1) + \eta\Big[B(t)A(t-1) \\ &- \frac{1}{2}B(t)A(t-1)\mathcal{UT}\left(A(t-1)^T w(t)w(t)^T A(t-1)\right) \\ &- \frac{1}{2}w(t)w(t)^T A(t-1)\mathcal{UT}\left(A(t-1)^T B(t)A(t-1)\right)\Big], \end{aligned} \qquad (7)$$

where the learning coefficient η is a small positive number, and

$$\mathcal{UT}(S) = \begin{pmatrix} s_{11} & s_{12} & s_{13} & \cdots & s_{1L} \\ 0 & s_{22} & s_{23} & \cdots & s_{2L} \\ 0 & 0 & s_{33} & \cdots & s_{3L} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & s_{LL} \end{pmatrix} \qquad (8)$$

for a matrix $S = (s_{ij})$. The number L of features is less than or equal to $\min(N, M-1)$. Note that the covariance matrix $W(t) = \frac{1}{t}\sum_{\tau=1}^{t} w(\tau)w(\tau)^T$ within classes is replaced with the instantaneous value $w(t)w(t)^T$. Such replacement is justified by the theory of stochastic approximation [7]. As for the initial values, $t^c(0) = 0$, $\bar{x}(0)$ and $\bar{x}^c(0)$ are arbitrary vectors, and $A(0)$ is an arbitrary matrix which satisfies $\mathrm{rank}A(0) = L$.

2.2. Procedures for efficient calculation

In order to avoid $N \times N$ matrices, the following procedures are recommended instead of the basic form in the previous subsection:

$$y^c(t) = A(t-1)^T v^c(t), \qquad (9)$$

$$z(t) = A(t-1)^T w(t), \qquad (10)$$

$$F(t) = \frac{1}{M}\sum_{c=1}^{M} v^c(t)y^c(t)^T, \qquad (11)$$

$$G(t) = \frac{1}{M}\sum_{c=1}^{M} y^c(t)y^c(t)^T, \qquad (12)$$

and

$$\begin{aligned} A(t) &= A(t-1) \\ &+ \eta\Big(F(t) - \frac{1}{2}F(t)\mathcal{UT}\left(z(t)z(t)^T\right) - \frac{1}{2}w(t)z(t)^T\mathcal{UT}(G(t))\Big). \end{aligned} \qquad (13)$$

Note that the result of the updating is exactly same as the basic form, and the $N \times N$ covariance matrix B between classes is not used any more. The updating procedures in this subsection is efficient for the case $N \gg M$, which is typical in image recognition tasks since the number of pixels are often very large.

3. Experimental Results

3.1. Adaptivity

We have applied the proposed algorithm for face identification task. The face image consists of 10×10 pixels, and hence $N = 100$. Each pixel has 256 level gray scale value from -1 to $+1$. The number of persons is $M = 2$ in the first part of the experiment In the latter part, new person is added and $M = 3$. The parameters of learning are $\eta = 0.01$, $L = 1$ (when $M = 2$) and $L = 2$ (when $M = 3$). Each element of the initial matrix $A(0)$ is generated by the uniform distribution on $[-0.1, +0.1]$. In this experiment, ww^T is replaced by $ww^T + \epsilon_W I$ so that stable result is obtained [4], where $\epsilon_W = 0.01$ and I is the identity matrix. The result is shown in Fig. 1. The step t in the figure is equal to the total number of presented images. The proposed algorithm successfully adapts to the new situation.

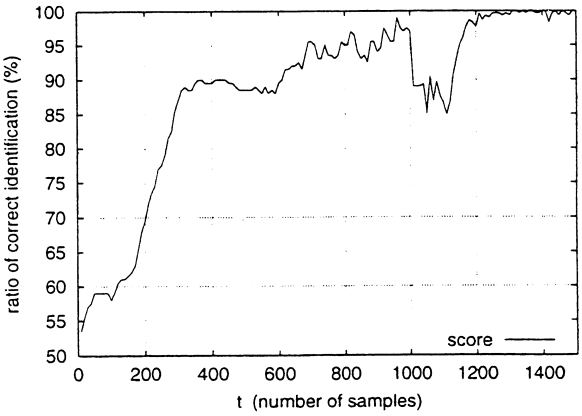

Figure 1. Learning curve of proposed algorithm. *Until $t = 1000$, images of two persons (A) and (B) are presented in random order, e.g. A, A, B, A, A, A, B, B, A, $\cdots$. At $t = 1000$, a new class of the third person (C) is added. After $t = 1000$, images of three persons are presented in random order. The ratio of correct identification is evaluated at every 10 steps. Different data sets are used for the learning and for the evaluation. The proposed algorithm adapts to the new situation.*

3.2. Comparison with [2][3][4]

We have also compared the proposed algorithm with the algorithm in [2][3][4]. The number of persons is $M = 5$. Other parameters are $N = 100$, $\eta = 0.01$, $\epsilon_W = 0.01$. First, 1500 step learning is executed with $L = 4$. Then, the last 2 columns of $A = (a_1, a_2, a_3, a_4)$ are deleted and the performance of the identification with this "reduced" $A = (a_1, a_2)$ is evaluated. In this experiment, 100 images are prepared for each person and they are used repeatedly. The result is shown is Fig. 2. The deterioration of the performance is small for the proposed algorithm, while it is sometimes large for the algorithm in [2][3][4]. This can be understood in the following way. Let $p_1, \cdots, p_4$ be the eigenvectors which correspond to 4 largest eigenvalues $\lambda_1 \geq \cdots \geq \lambda_4$, respectively:

$$Bp_i = \lambda_i W p_i, \qquad (i = 1, \cdots, 4), \tag{14}$$

where B and W are covariance matrices between classes and within classes, respectively. In the proposed algorithm, the deleted columns are p_3 and p_4, and the remaining columns are p_1 and p_2. Consequently, two most important components remain every time. On the other hand, in the algorithms in [2][3][4], the columns of A are only guaranteed that they are an W-orthonormal basis of the subspace S which is spanned by $p_1, \cdots, p_4$. Thus, the directions of remaining column vectors a_1 and a_2 in S change every time according to the initial value of A. An example of learning curves are shown in Fig. 3.

4. Conclusion

An improvement of successive learning algorithm of LDA [2][3][4] was examined based on Sanger's idea [10]. By the original algorithm, we can obtain only the subspace which is spanned by major eigenvectors. On the other hand, we can obtain major eigenvectors themselves by the improved algorithm. The performance of the proposed algorithm was experimentally studied.

One problem in online LDA is selection of the learning coefficient η. A guideline on selection or automatic adjustment of η is desired to use online LDA for extensive applications. In general, the smaller η is, the more precise the final result is. However, the convergence is slow if η is small. On the other hand, if η is large, $A(t)$ keeps moving around the best $A = A_0$ although the speed of approach to a neighborhood of A_0 is fast.

Of course, annealing method can be applied for the present algorithm, e.g. $\eta = \eta_0/t$. Then the convergence to the exact solution can be guaranteed by the theorem of stocastic approximation [7]. However, selecting a sufficiently small constant η is practically enough in most cases. Constant η is also desirable in terms of adaptivity.

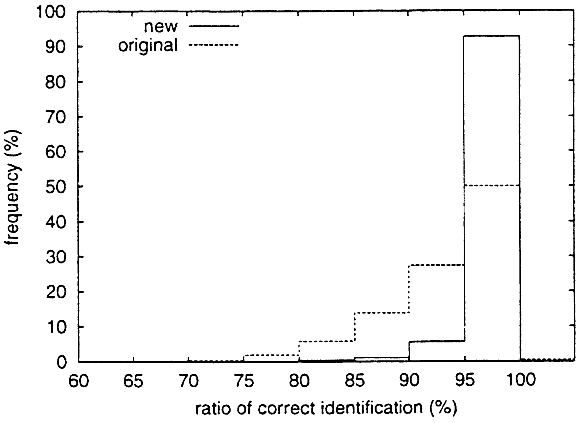

Figure 2. Comparison of proposed algorithm ("new") and algorithm in [2][3][4] ("original"). *The performance with "reduced" A ($L = 2$) is evaluated after learning with "full" A ($L = 4$). The result is shown as histogram for* 1291 *trials for each algorithm. The deterioration of the performance is small for the proposed algorithm.*

The authors wish to thank Dr. Takahashi for his helpful discussions and suggestions. This work has been partly supported by CREST of JST (Japan Science and Technology) 279102.

A. Appendix: Derivation of algorithm

The matrix dynamics in [2][3][4] can be written in vector form:

$$\begin{aligned} \dot{a}_k &= Ba_k - \frac{1}{2}B\left(\sum_{l=1}^{L} a_l a_l^T\right) W a_k \\ &\quad - \frac{1}{2}W\left(\sum_{l=1}^{L} a_l a_l^T\right) B a_k \qquad (15) \\ &= Ba_k - \frac{1}{2}B\left(a_1, \cdots, a_L\right)\begin{pmatrix} a_1^T W a_k \\ \vdots \\ a_L^T W a_k \end{pmatrix} \\ &\quad - \frac{1}{2}W\left(a_1, \cdots, a_L\right)\begin{pmatrix} a_1^T B a_k \\ \vdots \\ a_L^T B a_k \end{pmatrix}, \qquad (16) \end{aligned}$$

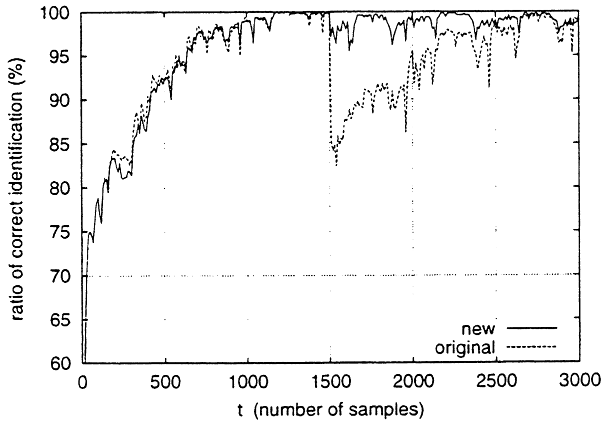

Figure 3. Comparison of proposed algorithm ("new") and algorithm in [2][3][4] ("original"). *Learning curve is shown for each algorithm. Until $t = 1500$, learning is executed with $L = 4$. At $t = 1500$, the last two columns of A are deleted. After $t = 1500$, learning is executed with $L = 2$. The deterioration of the performance is small for the proposed algorithm.*

where $A(t) = (\boldsymbol{a}_1(t), \cdots, \boldsymbol{a}_L(t))$ and $k = 1, \cdots, L$. By replacing L in (15) with k, we obtain

$$\begin{aligned}
\dot{\boldsymbol{a}}_k &= B\boldsymbol{a}_k - \frac{1}{2}B\left(\sum_{l=1}^{k} \boldsymbol{a}_l\boldsymbol{a}_l^T\right)W\boldsymbol{a}_k \\
&\quad - \frac{1}{2}W\left(\sum_{l=1}^{k} \boldsymbol{a}_l\boldsymbol{a}_l^T\right)B\boldsymbol{a}_k \qquad (17) \\
&= B\boldsymbol{a}_k \\
&\quad - \frac{1}{2}B\left(\boldsymbol{a}_1, \cdots, \boldsymbol{a}_k, \boldsymbol{a}_{k+1}, \cdots, \boldsymbol{a}_L\right)\begin{pmatrix} \boldsymbol{a}_1^T W\boldsymbol{a}_k \\ \vdots \\ \boldsymbol{a}_k^T W\boldsymbol{a}_k \\ 0 \\ \vdots \\ 0 \end{pmatrix} \\
&\quad - \frac{1}{2}W\left(\boldsymbol{a}_1, \cdots, \boldsymbol{a}_k, \boldsymbol{a}_{k+1}, \cdots, \boldsymbol{a}_L\right)\begin{pmatrix} \boldsymbol{a}_1^T B\boldsymbol{a}_k \\ \vdots \\ \boldsymbol{a}_k^T B\boldsymbol{a}_k \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \qquad (18)
\end{aligned}$$

It is written in matrix form as

$$\dot{A} = BA - \frac{1}{2}BAUT(A^TWA) - \frac{1}{2}WAUT(A^TBA). \qquad (19)$$

From this dynamics, the proposed algorithm is constructed.

References

[1] C. Chatterjee and V. P. Roychowdhury. On self-organizing alogorithms and networks for class-separability features. *IEEE Tr. NN*, 8(3):663–678, 1997.

[2] K. Hiraoka and M. Hamahira. On successive learning type algorithm for linear discriminant analysis. *IEICE Technical Report*, NC99(73):85–92, 12 1999 (in Japanese).

[3] K. Hiraoka, K. ichi Hidai, M. Hamahira, H. Mizoguchi, T. Shigehara, and T. Mishima. Derivation of online lda algorithm and its application for face identification. *Robotics Symposia*, 2000 (in Japanese), *submitted.*

[4] K. Hiraoka, S. Yoshizawa, K. ichi Hidai, M. Hamahira, H. Mizoguchi, and T. Mishima. Convergence analysis of online linear discriminant analysis. *International Joint Conference on Neural Networks (IJCNN)*, 2000, *submitted.*

[5] T. Kurita and S. Hayamizu. Gesture recognition using hlac features of parcor images and hmm based recognizer. *Proceedings of the third international conference on automatic face and gesture recognition*, pages 422–427, April 1998.

[6] J. Mao and A. K. Jain. Artificial neural networks for feature extraction and multivariate data projection. *IEEE Tr. NN*, 6(2):296–317, 1995.

[7] M. B. Nevel'son and R. Z. Has'minskiĭ. *Stochastic Approximation and Sequential Estimation.* Nauka, 1968 (in Russian; translated into Japanese by T. Kitagawa and K. Tajima, 1983).

[8] E. Oja. A simplified neuron model as a principal component analyzer. *J. Math. Biol.*, 15:267–273, 1982.

[9] E. Oja, H. Ogawa, and Wangviwattana. Principal component analysis by homogeneous neural networks, part I and part II. *J. Ieice Transactions On Information and Systems*, E75D:366–381, 1992.

[10] T. D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–473, 1989.