Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, July 31 - August 4, 2005

Incremental Learning for Online Face Recognition

Seiichi Ozawa, Soon Lee Toh, and Shigeo Abe
Graduate School of Science and Technology,
Kobe University, Kobe 657-8501, JAPAN
E-mail: {ozawasei, abe}@kobe-u.ac.jp

Shaoning Pang and Nikola Kasabov
Knowledge Engineering & Discover Research Institute
Auckland University of Technology, Private Bag 92006,
Auckland 1020, New Zealand
E-mail: {shaoning.pang, nik.kasabov}@aut.ac.nz

Abstract—**In this paper, a new approach to face recognition is presented in which not only a classifier but also a feature space of input variables is learned incrementally to adapt to incoming training samples. A benefit of this type of incremental learning is that the search for useful features and the learning of an optimal decision boundary are carried out in an online fashion. To implement this idea, an extended version of Incremental Principal Component Analysis (IPCA) and Resource Allocating Network with Long-Term Memory (RAN-LTM) are effectively combined. Using IPCA, a feature space is updated by rotating its eigen-axes and increasing the dimensions to adapt to a new training sample. In RAN-LTM, a small number of training samples called memory items are selected and they are utilized for retraining a classifier to realize an excellent incremental ability. To accommodate the classifier to the evolution of the feature space, we present a way to reconstruct the neural classifier without keeping all of the training samples given previously. In the experiments, the proposed incremental learning model is evaluated over a self-compiled face image database. As the result, we verify that the proposed model works well without serious forgetting and the test performance is improved as the learning stages proceed.**

I. Introduction

One of the main subjects in face recognition tasks is to enhance the robustness against the spatial and temporal variations of human faces due to the growth (or aging) and the changes in lighting conditions, face directions, expressions, make-up, and so forth [1], [2]. Conventional face recognition systems can achieve an excellent recognition performance when tested against a benchmark dataset. However, the performance could be dropped rather drastically when they are operated in a practical environment. This is because the initial training set of face images is either insufficient or inappropriate. Even if a large amount of face images are available when constructing a face recognition system, all the variations that will happen in future cannot be considered in advance; thus high recognition performance in practical situations can hardly be expected with only a static dataset. In addition, constructing a large database should impose a severe burden on not only system builders but also registered people. A solution to these problems is to make face recognition systems learn continuously to adapt to incoming training samples. This can be done by embedding an incremental learning ability into a face recognition system.

In conventional face recognition systems, the information processing is composed of the two parts: feature selection and classification. This means that when constructing an adaptive recognition system, we should consider two types of incremental learning: one is the incremental learning of feature space and the other is that of classifier. In many conventional approaches, these have been separately developed [3], [4], [5]. As for the feature selection, Principal Component Analysis (PCA) has been often adopted in face recognition tasks [6], [7], [8], [9]. Since PCA is not suited for incremental learning purposes, Hall and Martin have devised a method to update eigenvectors and eigenvalues (i.e., the update of an eigen-feature space) in an incremental way, called Incremental Principal Component Analysis (IPCA) [10]. On the other hand, Kasabov has proposed a fast on-line clustering method called Evolving Clustering Method (ECM) [11]. Both IPCA and ECM are categorized into *one-pass* incremental learning algorithms, in which the given training samples can be discarded after the training; thus, a distinctive property of these algorithms is that they do not consume so much large memory to enhance their performances. In this context, we have recently proposed a new efficient scheme for pattern recognition in which IPCA and ECM are effectively combined to realize a one-pass incremental learning of a feature space and a classifier [12], [13]. In this scheme, IPCA is used for learning a feature space and ECM is used for learning prototype vectors in the k-nearest neighbor classifier.

In this paper, we present another incremental approach in which a neural classifier, Resource Allocating Neural Network with Long-Term Memory (RAN-LTM) [14], [15], is adopted in place of ECM. The motivation of using neural classifiers is that we can expect the high generalization properties in classification. However, since the feature space is updated by IPCA every time when a new training sample is given, the eigen-axes are always rotated and the dimensions of the feature space could be increased. This means that RAN-LTM should also be updated to adapt to the evolution of a feature space. We give a solution to this problem.

This paper is organized as follows. Section II gives a brief explanation about an extended IPCA algorithm and RAN-LTM. In Section III, after giving an overview of our face recognition system, we describe how to accommodate a neural classifier (i.e., RAN-LTM) to the evolution of an eigenspace updated by IPCA. In Section IV, some experiments are conducted to evaluate the incremental learning performance, and the concluding remarks are given in Section V.

II. Incremental Learning Model

It is well known that the learning of neural networks becomes difficult in the situations where only a small number

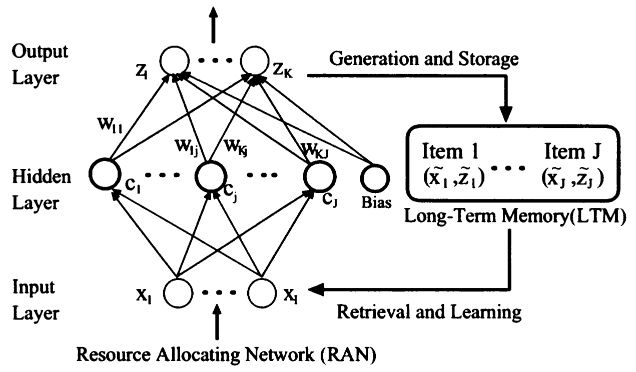

Fig. 1. The architecture of RAN-LTM.

of training data are presented at a time and all of the given training data are not able to keep due to the system limitations (i.e., the retraining of all given data is not permitted). In such situations, the input-output relations acquired in the past are easy to be collapsed by the learning of new data. This disruption in neural networks is called "forgetting" or "catastrophic interference" that is caused by the excessive adaptation of connection weights to new data [16].

A way of overcoming this problem is that only representative training data are kept in memory and some of them are learned with newly given training data. In this context, we have proposed an incremental learning model called *Resource Allocating Network with Long-Term Memory* (RAN-LTM) [14]. Let us briefly explain it in the following.

A. Resource Allocating Network with Long-Term Memory

Figure 1 shows the architecture of RAN-LTM which consists of two parts: Resource Allocating Network (RAN) [17] and Long-Term Memory (LTM). RAN is an extended model of Radial Basis Function (RBF) network [18] in which the allocation of hidden units is automatically carried out.

Let us denote the number of input units, hidden units, and output units as I, J, K, respectively. Moreover, let the inputs be $\boldsymbol{x} = \{x_1, \cdots, x_I\}^T$, the outputs of hidden units be $\boldsymbol{y} = \{y_1, \cdots, y_J\}^T$, and the outputs be $\boldsymbol{z} = \{z_1, \cdots, z_K\}^T$. The calculation in the forward direction is given as follows:

$$y_j = \exp(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_j\|^2}{2\sigma_j^2}) \quad (j = 1, \cdots, J), \tag{1}$$

$$z_k = \sum_{j=1}^{J} w_{kj} y_j + \xi_k \quad (k = 1, \cdots, K) \tag{2}$$

where $\boldsymbol{c}_j = \{c_{j1}, \cdots, c_{jI}\}^T$ and σ_j^2 are the center and variance of the jth hidden unit, w_{kj} is the connection weight from the jth hidden unit to the kth output unit, and ξ_k is the bias of the kth output unit.

The items stored in LTM are called "memory items" that correspond to the representative input-output data. These data can be selected from training samples, and they are learned with newly given training data to suppress forgetting. In the learning algorithm, a memory item is created when a hidden unit is allocated: that is, an RBF center and the corresponding output are stored in a memory item.

The learning algorithm of RAN-LTM is divided into two phases: the allocation of hidden units (i.e., the selection of RBF centers in an incremental fashion) and the calculation of connection weighs between hidden and output units. The procedure in the former phase is the same as that in the original RAN [17], except that memory items are created at the same time. Once hidden units are allocated, the centers are fixed afterwards. Therefore, the connection weights $\boldsymbol{W} = \{w_{kj}\}$ are only parameters that are updated based on the output errors. To minimize the errors based on the least squares method, it is well known that the following linear equalities should be solved [19], [20]:

$$\boldsymbol{\Phi} \boldsymbol{W} = \boldsymbol{D} \tag{3}$$

where $\boldsymbol{D}$ is the matrix whose column vectors correspond to the target outputs. Suppose that a training sample $(\boldsymbol{x}, \boldsymbol{d})$ is given and M memory items $(\tilde{\boldsymbol{x}}_m, \tilde{\boldsymbol{z}}_m)$ $(m = 1, \cdots, M)$ have already been created, then the target matrix $\boldsymbol{D}$ are formed as follows: $\boldsymbol{D} = \{\boldsymbol{d}, \tilde{\boldsymbol{z}}_1, \cdots, \tilde{\boldsymbol{z}}_M\}^T$. Furthermore, $\boldsymbol{\Phi} = \{\varphi_{mj}\}$ is calculated from these target vectors as follows:

$$\varphi_{1j} = \exp(\frac{\|\boldsymbol{x} - \boldsymbol{c}_j\|}{\sigma_j^2}), \quad \varphi_{mj} = \exp(\frac{\|\tilde{\boldsymbol{x}}_m - \boldsymbol{c}_j\|}{\sigma_j^2})$$
$$(j = 1, \cdots, J; \ m = 1, \cdots, M).$$

To solve $\boldsymbol{W}$ in Eq. (3), Singular Value Decomposition (SVD) can be used. The learning algorithm of RAN-LTM [5] is summarized as follows.

```
[Learning Algorithm]
1) Find the nearest center $c^*$ to an input $x$ and then calculate the output error $E$.
2) If $E > \varepsilon$ and $\|x - c^*\| > \delta$, then allocate a hidden unit (i.e., $J \leftarrow J + 1$) and create a memory item $(\tilde{x}_M, \tilde{z}_M)$ as follows:
     [Hidden Unit]   $w_J = d - z$,   $c_J = x$,
     [Memory Item]   $\tilde{x}_m = x$,   $\tilde{z}_m = d$.
   Then, increment the number of memory items as $M \leftarrow M + 1$ and go to Step 6. Otherwise, go to Step 3.
3) Calculate hidden outputs for the training sample $(x, d)$ and memory items $(\tilde{x}_m, \tilde{z}_m)$ $(l = 1, \cdots, M)$, and calculate $\Phi$ in Eq. (3).
4) Using SVD, decompose $\Phi$ as follows: $\Phi = UHV^T$ where $U$ and $V$ are orthogonal matrices, and $H$ is a diagonal matrix. Then, calculate the weight matrix as follows: $W = VH^{-1}U^T D$.
5) Give the input $x$ to RAN-LTM again, and calculate the output error $E$. If $E > \varepsilon$, add a hidden unit and generate a memory item $(\tilde{x}_M, \tilde{z}_M)$ in the same way of Step 2.
6) If a new training sample is given, go back to Step 1.
```

B. Extended IPCA Algorithm

In IPCA, an eigen-feature space is updated through the following two operations: the rotation of eigen-axes and the dimensional augmentation. The dimensional augmentation is carried out whenever the norm of a residue vector is larger than a threshold value. However, this is not a good criterion in practice because a suitable threshold can be varied depending on the magnitude and variance of input values. If the threshold is too small, a redundant high-dimensional feature space tends to be obtained; this will lead to deteriorating the generalization performance and the computational efficiency. To obtain an informative feature space with small-dimensions, we have proposed an extended IPCA algorithm in which the accumulation ratio is used as a criterion of the dimensional augmentation [13]. Let us explain this extended IPCA briefly.

Assume that N training samples $\boldsymbol{x}_i \in \mathcal{R}^n$ $(i = 1, \cdots, N)$ have been presented so far, and an eigenspace model $\Omega = (\bar{\boldsymbol{x}}, \boldsymbol{U}, \boldsymbol{\Lambda}, N)$ is constructed by calculating the eigenvectors and eigenvalues from the covariance matrix of $\boldsymbol{x}_i$, where $\bar{\boldsymbol{x}}$ is a mean input vector, $\boldsymbol{U}$ is an $n \times l$ matrix whose column vectors correspond to the eigenvectors, and $\boldsymbol{\Lambda}$ is an $l \times l$ matrix whose diagonal elements correspond to the eigenvalues. Here, l is the number of dimensions of the current eigenspace.

Let us consider the case that the $(N+1)$th training sample $\boldsymbol{y}$ is presented. The addition of $\boldsymbol{y}$ will lead to the changes in both of the mean vector and covariance matrix; therefore, the eigenvectors and eigenvalues should also be recalculated. The mean input vector $\bar{\boldsymbol{x}}$ is easily updated as follows:

$$\bar{\boldsymbol{x}}' = \frac{1}{N+1}(N\bar{\boldsymbol{x}} + \boldsymbol{y}). \tag{4}$$

The problem is how to update the eigenvectors and eigenvalues.

When the eigenspace model Ω is reconstructed to adapt to a new sample, we must check if the dimensions of the eigenspace should be changed or not. If the new sample has almost all energy in the current eigenspace, the dimensional augmentation is not needed in reconstructing the eigenspace. However, if it has some energy in the complementary space to the current eigenspace, the dimensional augmentation cannot be avoided. This can be checked by the accumulation ratio whose incremental representation is given as follows [13]:

$$A(l) = \frac{N(N+1)\sum_{i=1}^{l}\lambda_i + N\|\boldsymbol{U}^T(\boldsymbol{y} - \bar{\boldsymbol{x}})\|^2}{N(N+1)\sum_{i=1}^{n}\lambda_i + N\|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2} \tag{5}$$

If $A(l)$ is smaller than a threshold value θ, a new eigen-axis is added to the current eigenspace along the following residue vector:

$$\boldsymbol{h} = (\boldsymbol{y} - \bar{\boldsymbol{x}}) - \boldsymbol{U}\boldsymbol{g} \tag{6}$$

where

$$\boldsymbol{g} = \boldsymbol{U}^T(\boldsymbol{y} - \bar{\boldsymbol{x}}). \tag{7}$$

It has been shown that the eigenvectors and eigenvalues can be updated based on the solution of the following intermediate eigenproblem [10]:

$$\left(\frac{N}{N+1}\begin{bmatrix} \boldsymbol{\Lambda} & \boldsymbol{0} \\ \boldsymbol{0}^T & 0 \end{bmatrix} + \frac{N}{(N+1)^2}\begin{bmatrix} \boldsymbol{g}\boldsymbol{g}^T & \gamma\boldsymbol{g} \\ \gamma\boldsymbol{g}^T & \gamma^2 \end{bmatrix}\right)\boldsymbol{R} = \boldsymbol{R}\boldsymbol{\Lambda}' \tag{8}$$

where $\gamma = \tilde{\boldsymbol{h}}^T(\boldsymbol{y} - \bar{\boldsymbol{x}})$, $\boldsymbol{R}$ is an $(l+1) \times (l+1)$ matrix whose column vectors correspond to the eigenvectors obtained from the above intermediate eigenproblem, $\boldsymbol{\Lambda}'$ is the new eigenvalue matrix, and $\boldsymbol{0}$ is an l-dimensional zero vector. Using $\boldsymbol{R}$, we can calculate the new $n \times (l+1)$ eigenvector matrix $\boldsymbol{U}'$ as follows:

$$\boldsymbol{U}' = [\boldsymbol{U}, \ \hat{\boldsymbol{h}}]\boldsymbol{R} \tag{9}$$

where

$$\hat{\boldsymbol{h}} = \begin{cases} \boldsymbol{h}/\|\boldsymbol{h}\| & \text{if } A(l) < \theta \\ \boldsymbol{0} & \text{otherwise.} \end{cases} \tag{10}$$

Here, θ is a threshold value.

III. Face Recognition System

A. System Overview

The proposed face recognition system mainly consists of the following functional modules: face detection, face recognition, face image verification, and incremental learning. The operations in these modules must be done online without any human intervention. Figure 2 shows the overall process in the proposed system. In the face detection part, at each time frame, face regions are localized based on the information of skin color and edges. Thereafter three types of facial features (eye, nose, mouth) are searched for within the localized regions through raster operations. In each raster operation, a small sub-image is extracted from a localized region, then the eigen-features of the sub-image are given to a *Detection Neural Network* (DNN) to verify if it corresponds to any one of the facial features. These eigen-features are obtained using PCA. After all raster operations are done, face candidates are generated by combining the identified facial features based on some geometrical constraints. The output of the face detection part is the center positions of the face candidates.

Next, in the face recognition part, some rectangular regions of the face candidates can be extracted from the original image, and then each of the extracted regions is transformed into an eigen-feature vector using IPCA in order to reduce the dimensions. This eigen-feature is given to *Recognition Neural Network* (RNN) that is implemented with RAN-LTM, and then a recognition result is obtained.

The misclassified images are collected to use as training samples for incremental learning and sent to the face verification part. Since the perfect face detection cannot be ensured,

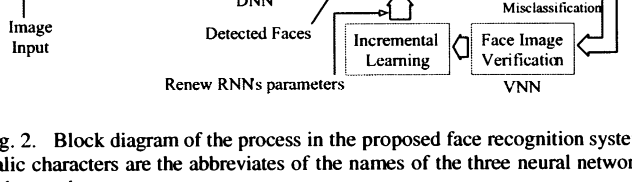

Fig. 2. Block diagram of the process in the proposed face recognition system. Italic characters are the abbreviates of the names of the three neural networks to be used.

there is a possibility that non-face images happen to be mixed with the misclassified face images to be learned. Apparently the training of these non-face images will deteriorate the recognition performance of RNN. Thus, another neural network called *Verification Neural Network* (VNN) is introduced into this part in order to filter out non-face images.

After the verification process, the learning for the misclassified face images is carried out by RNN in the incremental learning part. Note that RNN should be reconstructed when the eigen-feature space is updated using IPCA.

B. Reconstructing Neural Classifier with Incremental Update of Feature Space

As described in the previous subsection, the update of a feature space by IPCA can lead to the rotation of eigen-axes and the dimensional augmentation. This means that the inputs of a neural classifier (i.e., RNN) can be changed in not only their values but also the number of input variables whenever the feature space is update due to the presentation of a new training sample.

This operation is not easily realized in a one-pass incremental learning model since the past training samples are already discarded. Hence, there is no way of retraining the neural classifier to ensure that all previously trained samples can be correctly classified again after the update of the feature space. This problem can be circumvented if a minimum number of representative samples are properly selected and used for retraining the classifier. As easily expected, RAN-LTM is suitable for this purpose.

To actualize this idea, we need to devise an efficient way to adapt the memory items in RAN-LTM to an updated eigenspace. Let an input vector of the mth memory item $\mathcal{M}_m$ be $\tilde{\boldsymbol{x}}_m \in \mathcal{R}^l$ and let its target vector $\tilde{\boldsymbol{z}}_m$: $\mathcal{M}_m = \{\tilde{\boldsymbol{x}}_m, \tilde{\boldsymbol{z}}_m\}$. Furthermore, let the original vector associated with $\tilde{\boldsymbol{x}}_m$ in the input space be $\boldsymbol{x}_m \in \mathcal{R}^n$. These two vectors have the following relation: $\tilde{\boldsymbol{x}}_m = \boldsymbol{U}^T(\boldsymbol{x}_m - \bar{\boldsymbol{x}})$.

After the current eigenspace model $\Omega = (\bar{\boldsymbol{x}}, \boldsymbol{U}, \Lambda, N+1)$ is updated with a new sample $\boldsymbol{y}$ using IPCA, assume that a new model $\Omega' = (\bar{\boldsymbol{x}}', \boldsymbol{U}', \Lambda', N+1)$ is obtained. Then the updated memory item $\tilde{\boldsymbol{x}}'_m$ should strictly satisfy with the following equation:

$$\begin{aligned} \tilde{\boldsymbol{x}}'_m &= \boldsymbol{U}'^T(\boldsymbol{x}_m - \bar{\boldsymbol{x}}') \\ &= \boldsymbol{U}'^T(\boldsymbol{x}_m - \bar{\boldsymbol{x}}) + \frac{1}{(N+1)}\boldsymbol{U}'^T(\bar{\boldsymbol{x}} - \boldsymbol{y}) \quad (11) \end{aligned}$$

where $\bar{\boldsymbol{x}}'$ and $\boldsymbol{U}'$ are given by Eqs. (4), (9). The second term in the right-hand side of Eq. (11) is easily calculated. To calculate the first term strictly, however, the information on $\boldsymbol{x}_m$ that is already discarded is needed. Therefore, to circumvent this problem, we should consider the approximation to this term.

Let us assume that a new dimension is added. Substituting Eq. (9) into the first term on the right-hand side of Eq. (11), the first term is given as follows:

$$\boldsymbol{U}'^T(\boldsymbol{x}_m - \bar{\boldsymbol{x}}) = \boldsymbol{R}^T \begin{bmatrix} \tilde{\boldsymbol{x}}_m \\ \hat{\boldsymbol{h}}^T(\boldsymbol{x}_m - \bar{\boldsymbol{x}}) \end{bmatrix} \quad (12)$$

Even if we attempt to calculate an estimate of $\boldsymbol{x}_m$ from $\tilde{\boldsymbol{x}}_m$ using a pseudo inverse of the matrix $\boldsymbol{U}$, the estimated vector $\boldsymbol{U}^{T+}\tilde{\boldsymbol{x}}_m$ still lack the information in the complimentary subspace that $\hat{\boldsymbol{h}}$ spans; the information was lost during the dimensional reduction process. However, recalling a fact that $\hat{\boldsymbol{h}}$ is orthogonal to every vector in the subspace spanned by $\boldsymbol{U}$, we can approximate the term $\hat{\boldsymbol{h}}^T(\tilde{\boldsymbol{x}}_m - \bar{\boldsymbol{x}})$ to zero. Thus Eq. (11) is reduced to

$$\tilde{\boldsymbol{x}}'_m \approx \boldsymbol{R}^T \begin{bmatrix} \tilde{\boldsymbol{x}}_m \\ 0 \end{bmatrix} + \frac{1}{(N+1)}\boldsymbol{U}'^T(\bar{\boldsymbol{x}} - \boldsymbol{y}). \quad (13)$$

Now that all the information in Eq. (13) is available; then we can update the memory items after the update of eigenspace.

After the update, the neural classifier RAN-LTM undergoes the retraining with a new sample and the updated memory items. That is to say, the memory items are set to the centers of hidden units, and the connection weights between the hidden units and output units are obtained based on the learning algorithm stated in II-A.

IV. Performance Evaluation

A. Experimental Conditions and Evaluation Method

The dataset used here is divided into two sets: training dataset and test dataset. The former is used for assessing the performance of incremental learning in RNN through an online recognition process. For this dataset, training samples are given to RNN one by one, and the recognition and incremental learning are carried out online. The latter dataset is used for assessing the generalization performance of RNN. Hence, the images in the test dataset are not used in the training.

These datasets contain video clips of seven people: four people (2 males and 2 females) are chosen as registered persons and the other three people (2 males and a female) are non-registered persons. The video clips have durations of $5 \sim 15$ seconds and they are taken over two weeks such that some changes in facial appearances are included. The training dataset consists of 654 images detected from the video clips of the first week; on the other hand, the test dataset consists of 499 images detected from the clips of the second week. Since the extraction of face images is automatically done in the detection part, some non-face images could inevitably be included in the datasets. However, with regard to the training dataset, almost all data are face images since non-face images are presumably filtered out by VNN (see Fig. 2). Figure 3 shows the examples of detected face images for four registered persons and three non-registered persons.

To simulate real-life consecutive learning, the training dataset is further divided into six subsets. These subsets are given to the face recognition system in turn, and the misclassified images are collected and learned incrementally in RNN. The number of images in each subset is as follows: 139 (*initial training* dataset) $\rightarrow$ 107 $\rightarrow$ 75 $\rightarrow$ 98 $\rightarrow$ 106 $\rightarrow$ 129. Hence, five stages of incremental learning are carried out. In this experimental setup, only the images of the four registered people are included in the initial training dataset. The effectiveness of incremental learning is evaluated in terms

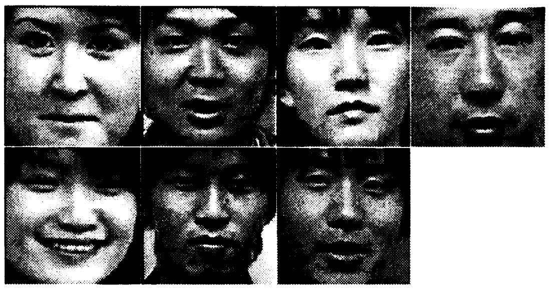

Fig. 3. Examples of detected face images for four registered persons (upper) and three non-registered persons (lower).

TABLE I
EVOLUTION OF THE EIGENSPACE MODEL ESM3.

Stage	Training Samples	Dimensions	Accumulation Ratio
Initial	139	139	1
1	204	153	0.892
2	204	153	0.892
3	204	153	0.892
4	266	169	0.891
5	277	173	0.891

of the recognition accuracy for all subsets given so far (*online* dataset). This evaluation is conducted to check if the forgetting occurred in the incremental learning. The number of evaluated images in the online dataset increases as the learning stage proceeds: 139 $\rightarrow$ 246 $\rightarrow$ 321 $\rightarrow$ 419 $\rightarrow$ 525 $\rightarrow$ 654.

As a security system, it is also important to evaluate the false-positive rate (the rate of the cases where non-registered faces or non-faces are classified as registered faces). To do this, we evaluate the recognition performance using another set of 3311 images, which consists of 1748 non-face images and 1563 face images that are collected from various databases. This dataset is referred to as *FP* dataset.

The experiment is preformed in the following environment: MS Windows XP and Athlon(tm) XP 2200+.

B. Experimental Results

In order to evaluate the effectiveness of learning a feature space using IPCA, the classification performance of RAN-LTM is examined when it combines with three different eigenspace models: (1) a static eigenspace model obtained by PCA (*ESM1*), (2) an adaptive eigenspace model obtained by IPCA without the dimensional augmentation (*ESM2*), and (3) an adaptive eigenspace model obtained by IPCA (*ESM3*). In ESM1, the eigenspace is calculated from the initial training dataset and it is not updated afterward. In ESM2, we examines the effects of rotating the feature space without allowing the dimensional augmentation in IPCA. The parameters of RAN-LTM are fixed for all the experiments.

Table I shows the evolution of the eigenspace learning with ESM3 in which the feature dimensions and the accumulation ratio are examined. As you can see from Table I, the dimensions of feature spaces increase such that the accumulation ratio is maintained over the threshold value $\theta = 0.891$. This means that we can control the dimensions of an eigenspace by specifying a proper accumulation ratio.

Figure 4 shows the experimental results against four types of datasets: (a) training dataset, (b) online dataset, (c) test dataset, and (d) false positive dataset. As seen from Fig. 4(a), the performance of ESM3 is slightly better than the other eigenspace models ESM1 and ESM2. In addition, we can say that the training dataset at the 4th stage is quite different from those in other stages. Fig. 4(b) shows the performance against the online dataset which consists of all the training datasets given at the previous stages. The recognition rates are fluctuated a little but they are not seriously falling down. This means that RAN-LTM can learn incrementally in one-pass without serious forgetting even after the eigenspace model is simultaneously updated by IPCA. As seen from Fig. 4(c), the overall recognition rate gets better if IPCA is introduced in the incremental learning process. A large improvement in recognition rate can be seen from stage 3 to stage 4. This is again due to the fact that the training samples at the 4th stage is significantly different from those at the earlier stages.

An interesting result in Fig. 4(c) is that the recognition performance of ESM2 is better than that of ESM3, which means that IPCA without the dimensional augmentation is more suitable for enhancing the generalization performance than that with the dimensional augmentation. On the other hand, as seen from Fig. 4(d), the false-positive rate in ESM3 is suppressed as compared with ESM2. The result of ESM3 shows that the eigenspace possesses more information on the non-registered faces through the dimensional augmentation. Hence, this leads to a tradeoff between the generalization performance and false-positive rate.

V. CONCLUSIONS

In this paper, we have proposed a new approach to constructing adaptive face recognition systems in which a low-dimensional feature space and a classifier are incrementally learned in an online fashion. To learn a useful feature space incrementally, we adopt an extended version of Incremental Principal Component Analysis (IPCA) in which the augmentation of feature space dimensions is determined based on the accumulation ratio. When the feature space is dynamically changed over the learning stages, the inputs of a neural classifier could also be changed in their values and the number of input variables. To adapt to the evolution of a feature space, an extended model of Resource Allocating Network (RAN) called *RAN with Long-Term Memory* (RAN-LTM) is adopted as a classifier, and we propose an efficient way to reconstruct RAN-LTM after the update of the feature space.

To evaluate the incremental learning properties, a self-compiled face image database is applied to the proposed model. In the experiments, we verify that the proposed incremental learning works well without serious forgetting and the test performance is improved as the incremental learning stages proceed. However, the dimensional augmentation of a feature space leads to deteriorating its test performance, while it can decrease the false-positive rate.

Acknowledgment

The authors would like to thank Shinji Kita for his great support to developing the face recognition system and the face image database. This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (C) 16500130.

References

[1] A. J. Howell, "Introduction to Face Recognition," in L. C. Jain et al. (Eds.) *Intelligent Biometric Techniques in Fingerprint and Face Recognition*, CRC Press, pp. 217-284, 1999.

[2] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET Evaluation Methodology for Face-Recognition Algorithms," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1090-1104, 2000.

[3] J. Weng, C. H. Evans, and W.-S. Hwang, "An Incremental Learning Method for Face Recognition under Continuous Video Stream," *Proc. of Fourth IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pp. 251-256, 2000.

[4] C. Liu and H. Wechsler, "Learning the Face Space - Representation and Recognition," *Proc. of Int. Conf. on Pattern Recognition*, pp. 1249-1256, 2000.

[5] S. L. Toh and S. Ozawa, "A Face Recognition System Using Neural Networks with Incremental Learning Ability," *Proc. 8th Australian and New Zealand Conf. on Intelligent Information Systems*, pp. 389-394, 2003.

[6] E. Oja and J. Karhunen, "On Stochastic Approximation of the Eigenvectors and Eigenvalues of the Expectation of a Random Matrix," *J. Math. Analysis and Application*, vol. 106, pp. 69-84, 1985.

[7] T. D. Sanger, "Optimal Unsupervised Learning in a Single-layer Linear Feedforward Neural Network," *IEEE Trans. Neural Networks*, vol. 2, pp. 459-473, 1989.

[8] M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve Procedure for the Characterization of Human Faces," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 103-108, 1990.

[9] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

[10] P. Hall and R. Martin, "Incremental Eigenanalysis for Classification," *Proc. British Machine Vision Conference*, vol. 1, pp. 286-295, 1998.

[11] N. Kasabov, *Evolving Connectionist Systems: Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines*, Springer-Verlag, 2002.

[12] S. Pang, S. Ozawa, and N. Kasabov, "One-pass Incremental Membership Authentication by Face Classification," in Biometric Authentication, D. Zhang and A. K. Jain (Eds.), Lecture Notes in Computer Science, Springer-Verlag, pp. 155-161, 2004.

[13] S. Ozawa, S. Pang, and N. Kasabov, "A Modified Incremental Principal Component Analysis for On-line Learning of Feature Space and Classifier," in PRICAI 2004: Trends in Artificial Intelligence, C. Zhang, H. W. Guesgen, and W. K. Yeap (Eds.), Lecture Notes in Artificial Intelligence, Springer-Verlag, pp. 231-240, 2004.

[14] M. Kobayashi, A. Zamani, S. Ozawa, and S. Abe, "Reducing Computations in Incremental Learning for Feedforward Neural Network with Long-Term Memory," *Proc. Int. Joint Conf. on Neural Networks*, vol. 3, pp. 1989-1994, 2001.

[15] K. Okamoto, S. Ozawa, and S. Abe, "A Fast Incremental Learning Algorithm of RBF Networks with Long-Term Memory," *Proc. Int. Joint Conf. on Neural Networks*, vol. 1, pp. 102-107, 2003.

[16] G. A. Carpenter and S. Grossberg, "The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network," *IEEE Computer*, vol. 21, no. 3, pp. 77-88, 1988.

[17] J. Platt, "A Resource-allocating Network for Function Interpolation," *Neural Computation*, vol. 3, no. 2, pp. 213-225, 1991.

[18] T. Poggio and F. Girosi, "Networks for Approximation and Learning," *IEEE Trans. on Neural Networks*, vol. 78, no. 9, pp. 1481-1497, 1990.

[19] M. J. L. Orr, "Introduction to Radial Basis Function Networks," *Tech. Report of Institute for Adaptive and Neural Computation*, Division of Informatics, Edinburgh University, 1996.

[20] S. Haykin, *Neural Networks - A Comprehensive Foundation* (2nd Ed.), Prentice Hall, 1999.

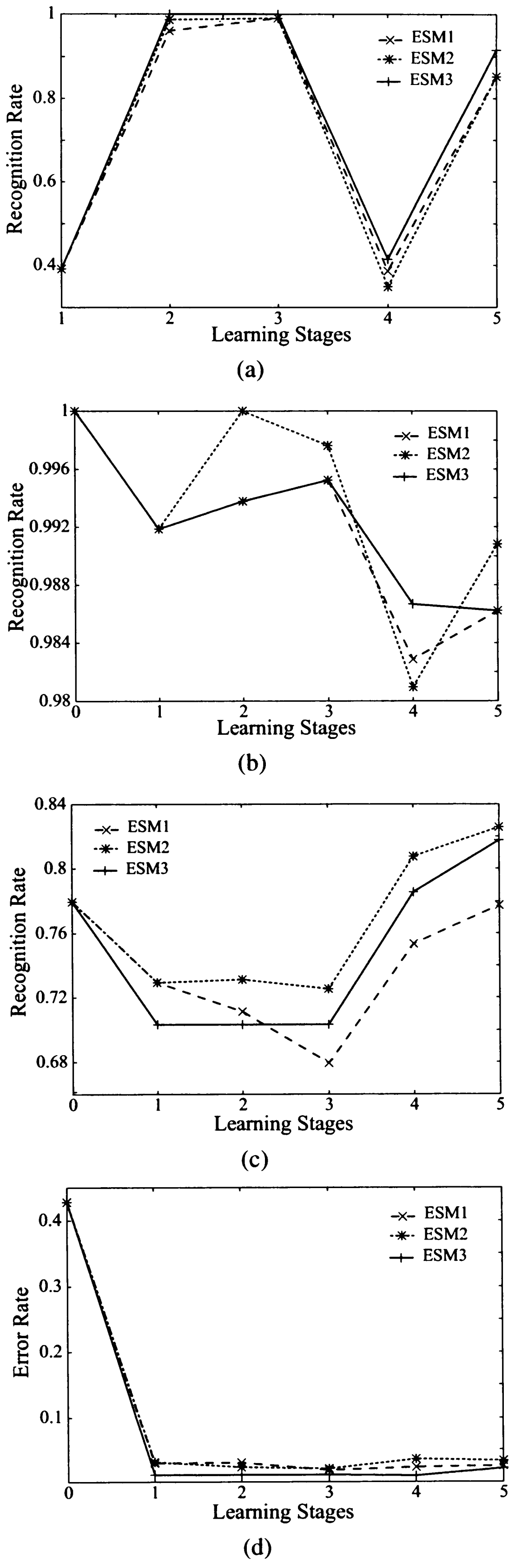

Fig. 4. The evolution of the recognition performance against (a) training dataset, (b) online dataset, (c) test dataset, and (d) false-positive dataset.