User Adaptive Handwriting Recognition by Self-Growing Probabilistic Decision-Based Neural Networks

Hsin-Chia Fu, *Member, IEEE*, Hung—Yuan Chang, Yeong Yuh Xu, and H.-T. Pao

***Abstract*—It is generally agreed that, for a given handwriting recognition task, a user dependent system usually outperforms a user independent system, as long as a sufficient amount of training data is available. When the amount of user training data is limited, however, such a performance gain is not guaranteed. One way to improve the performance is to make use of existing knowledge, contained in a rich multiuser data base, so that a minimum amount of training data is sufficient to initialize a model for the new user. We mainly address the user adaption issues for a handwriting recognition system. Based on self-growing probabilistic decision-based neural networks (SPDNNs), user adaptation of the parameters of SPDNN is formulated as incremental reinforced and antireinforced learning procedures, which are easily integrated into the batched training procedures of the SPDNN. In this study, we developed 1) an SPDNN based handwriting recognition system; 2) a two-stage recognition structure; and 3) a three-phase training methodology for a) a global coarse classifier (stage 1); b) a user independent hand written character recognizer (stage 2); and c) a user adaptation module on a personal computer. With training and testing on a 600-word commonly used Chinese character set, the recognition results indicate that the user adaptation module significantly improved the recognition accuracy. The average recognition rate increased from 44.2% to 82.4% in five adapting cycles, and the performance could finally increase up to 90.2% in ten adapting cycles.**

***Index Terms*—Decision-based neural networks (DBNNs), handwriting recognition, self-growing, self-growing probabilistic DBNN, supervised learning, user adaptation (UA).**

I. Introduction

ADAPTIVE learning of neural networks for handwriting recognition is of great interest for both theoretical and practical purposes. Adaptive learning techniques have been applied to both online or offline handwriting recognition to handle situations not seen or dealt with during the batch training phase. This includes effects due to varying writing tools, writing styles, and sizes of characters. In this paper, we focus on adaptive learning techniques for incremental training data. It is general agreed that, for a given recognition task, a user-dependent (UD) system usually outperforms a user-independent (UI) system, as long as a sufficient amount of training data is available to obtain user-dependent models. When the amount of user-specific training data is limited, however, such a performance gain is not guaranteed. One way to improve the performance is to make use of existing knowledge, contained in a database from a rich multiuser pool, so that a minimum of additional amount of training data is sufficient to initialize a recognition model for a new-user. Such a training procedure is often referred to as user adaptation (UA) when the prior knowledge is derived from a multiuser data set.

User adaptation can be formulated in a number of ways, including 1) *user transformation,* in which a well-trained model for one user is converted into a model for a new user using a small amount of user-specific training data; 2) *user adaptation,* in which a multiuser model is adapted to a new user who provides specific training data; and 3) *incremental adaptation,* in which user-specific training data are acquired over time, and the user-dependent model is adapted incrementally every time new training data is acquired. Here we focus on *incremental adaptation* for neural-network character recognizers.

In this paper, we propose a three-phase training methodology for a two-stage handwriting recognition system. The three-phase training is designed for 1) a global coarse classifier (stage 1); 2) a user independent handwritten character recognizer (stage 2); and 3) a user adaptation module. In particular, a self-growing probabilistic pattern recognition decision-based neural network is proposed to implement the kernel of the proposed personal handwriting recognition system. We present an EM algorithm based batch and incremental learning procedures for obtaining the self-growing probabilistic decision-based neural networks (SPDNNs) parameters. Two features of the SPDNN make it suitable for implementing not only in personal handwriting recognition systems, but also in other adaptive systems. These features are the following.

- *Architecture Feature:* SPDNN adapts the modular one class in one network (OCON) structure, which devotes one of its subnets to representation of a particular character. This kind of structure is beneficial not only for training and recognition performance, but also for hardware implementation.
  1) *The system is easy to train and maintain*. Training a SPDNN-based character recognition system is relatively straightforward. In other words, by adding or modifying one or a few clusters in a subnet of SPDNN [using self-growing (cf. Section III-B2)

Manuscript received June 18, 1999; revised June 22, 2000. This work was supported in part by the National Science Council under Grant NSC 87-2213-E009-092.

H.-C. Fu, H.-Y. Chang, and Y. Y. Xu are with the Department of Computer Science and Information Engineering, National Chiao Tung University, Hsin-Chu, Taiwan, R.O.C.

H.-T. Pao is with the Department of Management Science, National Chiao Tung University, Hsin-Chu, Taiwan, R.O.C.

Publisher Item Identifier S 1045-9227(00)09848-9.

or incremental learning (cf. Section III-B3) processes], the recognition system can adapt a person's handwriting style. A centralized system, in contrast, would have to include global updating.

2) *A distributed computing principle is adopted.* For example, the number of total characters in Chinese is very large;[1] thus, the computing hardware requirements for recognition system are greater. Due to its modular architecture, an SPDNN-based character recognition system is relatively easy to implement on parallel computers.

- *Performance Feature:* The discriminant function of SPDNN is in a form of probability density. This yields an effective adaptation process and very accurate recognition. In addition, by applying the proposed personal adaptation process, the system recognition performance can be improved from 44.2% to 90.2% in ten adaptive learning cycles (cf., Section IV-C).

The organization of this paper is as follows: In Section II, an overview of an SPDNN-based handwriting recognition system is presented. The structural properties and learning rules of SPDNN are described in Section III. In Section IV, the implementation of a two-stage handwriting recognition architecture and three phase learning methodologies are introduced first, and then the recognition performance of each stage in the proposed SPDNN system are described and discussed. Conclusions and future works are given in Section V.

II. SPDNN Handwriting Recognition System

An SPDNN-based handwritten Chinese major hybrid character recognition system is being developed in the Neural Networks Laboratory of National Chiao-Tung University. The system configuration is depicted in Fig. 1. All the major processing modules, including preprocessing and feature selection modules, a coarse classifier, a character recognizer, and a personal adaptation module, are implemented on a personal computer.

The system built upon the proposed SPDNN model has been demonstrated to be applicable under reasonable variations of character orientation, size, and stroke width. This system also has been shown to be very robust in recognizing characters written using various tools, such as pencils, ink pens, marking pens, and Chinese calligraphy brushes. As to the processing speed of the prototype system, the whole recognition process (including image preprocessing, feature selection, and character recognition) consumes approximately 0.14 s/character on a Pentium-II-based personal computer, without using a hardware accelerator or coprocessor.

A. Image Preprocessing and Feature Selection

Image Preprocessing: After a binary image of a page of handwriting is obtained from a scanner, it is then ready for character segmentation. Character segmentation of cursive handwriting is a very difficult task; thus, segmentation and

[1]More than 10 000 Chinese characters are used, and the number of commonly or daily used characters is reported to be 5401.

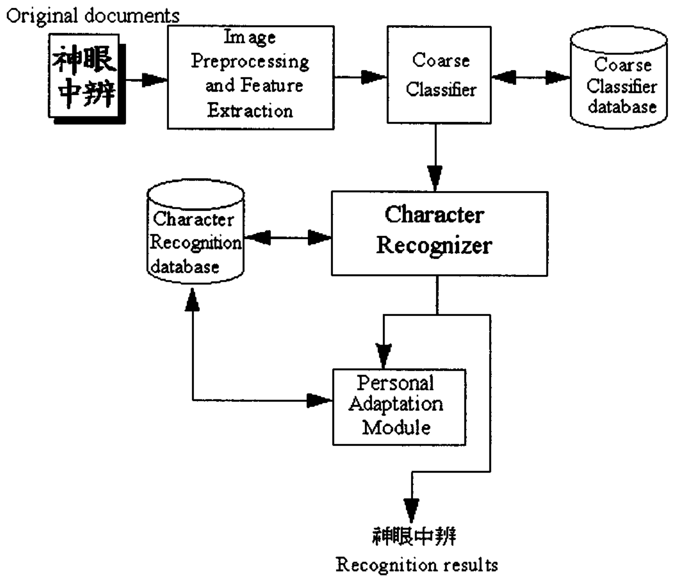

Original documents
神眼中辨
Image Preprocessing and Feature Extraction
Coarse Classifier
Coarse Classifier database
Character Recognition database
Character Recognizer
Personal Adaptation Module
神眼中辨
Recognition results

Fig. 1. System configuration of the personalized handwriting recognition system. The handwriting recognition system acquires images from a scanner. The coarse classifier determines an input character image which will be one of the predefined subclasses. The character recognizer matches the input character with a reference character. The personal adaptive module learns the user's own writing style so as to enhance the recognition accuracy.

recognition are performed interactively in some character recognition systems. Since handwriting recognition is already a complicated problem, interactive segmentation-recognition usually slows down the overall processing speed. Thus, iterative rule-based character segmentation [2] is applied to divide up a whole page of noncursive handwriting into a stream of character images. Basically, this method uses some heuristic rules to combine several nearby isolated stroke images into a character image. The binary images of a handwritten character are then passed through a series of image processing stages, including boundary smoothing, noise removing, space normalization, and stroke thinning operations.

Feature Selection: In general, a desired feature for character recognition purposes should partition a character space with small domain area for each class and a large center distance between classes. Based on this concept, we propose an evaluation criterion, *feature index* (I), to indicate the separation capability of a feature vector $\mathbf{x}$

$$I(\mathbf{x}) = \frac{D_{Inter}}{D_{Intra}} = \frac{\sum_{i=1}^{c}(m_i - m)^T(m_i - m)}{\sum_{i=1}^{c}\sum_{j=1}^{n_i}(x_j - m_i)^T(x_j - m_i)}$$

where

c	number of classes;
n_i	number of data in class i;
m	mean of all classes;
m_i	mean of ith class;
D_{Intra}	intraclass distance;
D_{Inter}	interclass distance.

Through simple mathematical derivation, we can prove that the feature index is mathematically equivalent to Devijver's $J2$ [3]

feature index, which is well known for its merit of computational simplicity for various pattern recognition applications. By evaluating the *feature index I* on most of the well-known features, we can select features with high index values, such as the *crossing count* (CCT), *belt shape pixel number* (BSPN), and *stroke orientation feature* (STKO), as candidate features for the proposed character recognition system. Features such as the *crossing count* (CCT) represent the stroke complexity of a character image. As shown in Fig. 2(a), a normalized character image is scanned horizontally and vertically. For each scanned line, the number of transitions from a white pixel (0) to a black pixel (1) is calculated and depicted in histogram form below and to the right of the character image. In order to reduce the number of feature values, each histogram is partitioned into *n* blocks, and the number of zero to one transitions in each block is then calculated and used as a feature value. In addition, the number of one to zero transitions can be obtained in a similar manner. Thus, a total of $2 \times 2 \times n$ feature values of CCT are collected for each character image. In order to represent the pixel distribution of a character pattern, BSPN counts the total number of pixels in each horizontal and vertical partition. As shown in Fig. 2(b), the histograms below and to the right of the character image represent the number of pixels in each horizontal and vertical scan line. Similar to CCT, each histogram of BSPN is also partitioned into n blocks, and the number of pixels in each block represents one of the feature values of BSPN. Considering both horizontal and vertical scan lines, there are $2n$ feature values in total. In [8], Kimura *et al.*, proposed the directional code histogram, which was successfully used for Chinese character recognition. By simplifying their feature selection process, the STKO feature can be selected by dividing a character image into $n \times n$ square partitions and counting the number of line segments in four quantized directions (0°, 45°, 90°, 135°), as shown in Fig. 3.

Partition of a Character Image: The partition number n in the previous discussion of feature selection is decided based on the feature index I. By computing the index value I of a type of feature with n being from two to ten, we can select the smaller partition number n, which corresponds to the highest index value I. Table I lists the selected feature sets, the *index* values, and their associated partition size n. For the selected features, CCT, BSPN, and STKO, the desired partition numbers are four, six, and 16, respectively. Without referencing the feature index (I), a proper partition size of n will be determined based on full implementation of the recognition system for each feature with various sizes of n, which will be very computer time consuming due to the large amount of training data involved.

B. Coarse Classification of Input Characters

Since there are as many as 5401 commonly used Chinese characters as well as 62 alphanumerics and symbols in a Chinese majored hybrid-language handwritten document, it is desirable to perform coarse classification to reduce the number of candidate characters for the *character recognition* process, so as to improve the overall recognition speed and recognition accuracy.

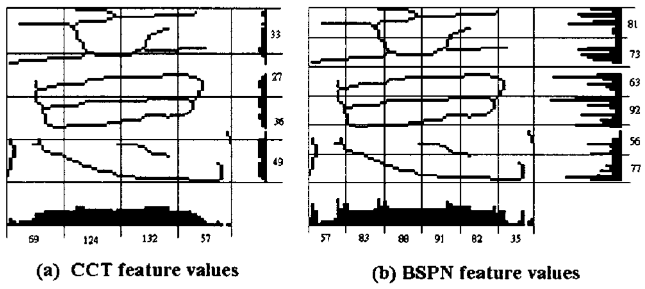


Fig. 2. Extracting the CCT and BSPN features of a Chinese character. (a) The histograms below and to the right of the character image represent the number of zero to one transitions in each horizontal and vertical scan line. The number in each partition block represents the total number of zero to one transitions in each block area. (b) Similar to CCT, the BSPN histograms represent the number of pixels in each horizontal and vertical scan line. The number in each partition represents the total pixels in the corresponding block.

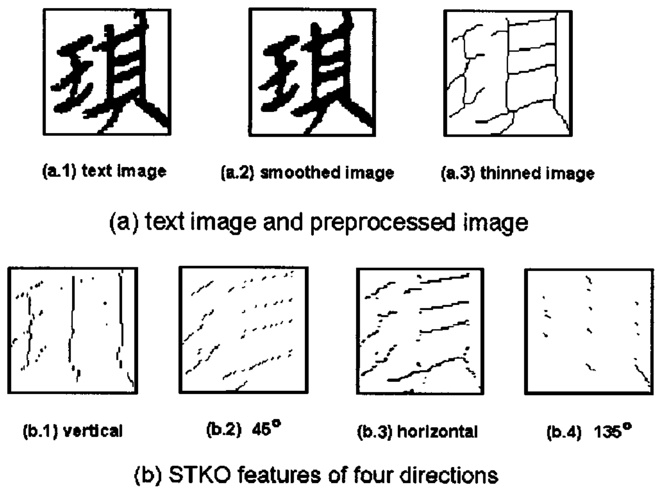


Fig. 3. Preprocessing and STKO feature extraction of a Chinese character.

TABLE I
INDEXES ANDDIMENSIONS OF THE SELECTED FEATURES

Number of coarse class	Ave. No. of characters in a class	Training Accuracy	Testing Accuracy
61	516	99.9 %	99.8%

C. Character Recognition

The design of the character recognizer is based on the SPDNN model. One SPDNN character recognizer is constructed for one coarse class. When a character image is identified as being in one of the coarse classifiers, it is then sent to the corresponding character recognizer. The feature used to perform character recognizer is different from the feature used by the coarse classifier. The CCT feature is used by the coarse classifier because of its small dimensionality and, thus, lower computational complexity. As far as the BSPN and STKO are concerned, these two features provide better separability between characters in a coarse class.

D. Personalization of the Recognition System

Most of the recent reports on handwriting recognition systems claim benchmarking recognition performance higher than 90%. However, when these systems are applied to personal

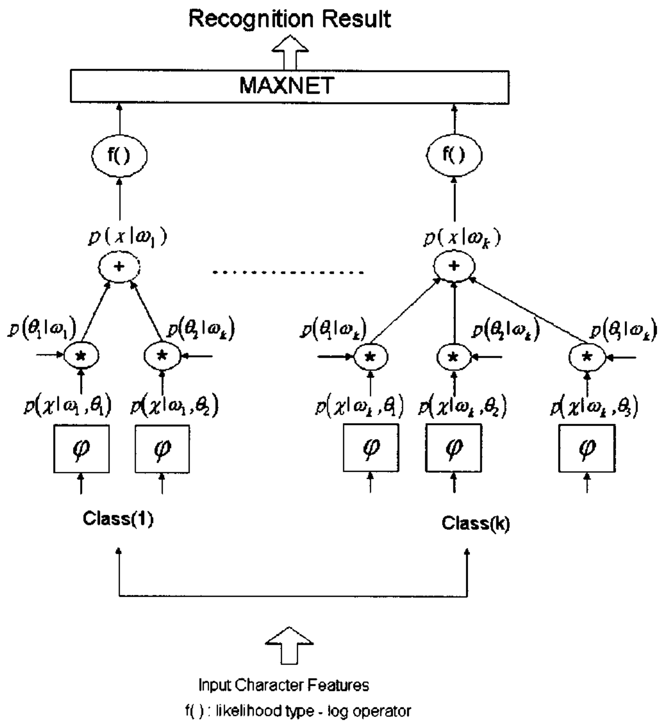


Fig. 4. Schematic diagram of a k-class SPDNN character recognizer.

freehand-writing, their recognition accuracy is usually between 40% and 50% [5]. Hence, we suggest a personal adaptation module to fine tune the parameters of the SPDNN character recognizer in order to adapt to the user's own writing style. The parameters or the decision boundaries of the corresponding subnets in the SPDNN are modified according to the proposed incremental learning EM algorithms (Section III-B3). When more and more freehand-written characters are presented to the system, the SPDNN recognizer gradually learns the user's personal writing style.

E. Training and Testing Character Data Generation

A modern handwritten document usually contains characters of hybrid languages. For example, a primarily Chinese handwritten draft usually includes some alpha-numerical characters. In this section, we present two main aspects of the training and testing data generation scheme for the hybrid-language handwriting recognizer.

1) *Database Character Set*: In order to assure sufficient diversity of Chinese handwriting styles in the training and testing sets, we use the CCL/HCCR1 [15] handwriting database for benchmark comparison. The CCL/HCCR1 database has been used by several handwriting recognition research groups [2], [4], [10]. The CCL/HCCR1 database contains more than 200 samples of 5401 frequently used Chinese characters. The samples were collected from 2600 people, including junior high school and college students as well as employees of ERSO/ITRI. Half of the randomly selected samples of each character are used for training, and the other half of the samples are used for testing. When we searched through major conference proceedings, technical journals and WWW sites, we were not able to find any performance test reports on multilinguistic handwriting. We, therefore, prepared combined database that combines the CCL/HCCR1 and CEDAR [6] databases. The training and testing data sets for the Chinese character part were selected in the same way as stated before. The CEDAR database contains various styles of handwritten alphanumerics, which were lifted from address blocks on envelopes from the United States. Among the data, 4000 alphanumerics were used for training and 2000 for testing.
2) *Run-Time Training Data Generation*: During the personal adaptation phase, the SPDNN based system also generates run-time training data. If the SPDNN falsely recognizes a character, then this particular character will be used to train the corresponding two subnets, i.e., antireinforced learning on the false subnet and reinforced learning on the supposed subnet. In the meantime, the system issues the current TOP 10 recognition candidates to the user and asks the user to write these characters as the training data to further improve the system performance.

III. Self-Growing Probabilistic Decision-Based Neural Network

As shown in Fig. 4, the proposed SPDNN is a new multivariate Gaussian neural network [7], [9]. Three learning phases are proposed in SPDNN: When each subnet is initialized with one cluster (see Section III-B-1) or is self-grown with a new cluster (Section III-B-2), the system enters the supervised learning (SL) phase. In the SL phase, teacher information is used to reinforce or antireinforce the decision boundaries obtained during the initialization or self-growing stages. When the supervised training process proceeds very slowly or is trapped in a paralysis state (local minimum), but the classification or recognition accuracy is not at a satisfactory level, the training process enters the self-growing (SG) phase. In the SG phase, an SPDNN creates a new cluster in a subnet according to the proposed SG rule. Then, the training process enters the supervised learning phase again. The batch learning procedure terminates when the training accuracy reaches a predefined satisfaction level. The system enters the third *incremental learning* phase (Section III-B-3), when a user corrects misclassified or misrecognized characters. In this learning phase, the system gradually adapts to the user's own personal handwriting style. A detailed description of the SPDNN model is given in the following sections.

A. Discriminant Functions of SPDNN

One of the major differences between multivariate Gaussian neural networks (MGNNs) [7], [9] and SPDNN is that SPDNN extends the fixed number of clusters in a subnet of MGNN to a flexible number of clusters in a subnet of the SPDNN. That is, the subnet discriminant functions of SPDNN are designed to model the log-likelihood functions of different complexed pixel distributions of handwritten characters. Given a set of independently identically distributed (i.i.d.) patterns $\mathbf{X}^+ =$

$\{x(t);\, t=1, 2, \cdots, N\}$, we assume that the likelihood function $p(\mathbf{x}(t)|\omega_i)$ for class ω_i (i.e., a character class) is a mixture of Gaussian distributions.

Define $p(x(t)|\omega_i, \Theta_{r_i})$ as one of the Gaussian distributions which comprise $p(x(t)|\omega_i)$, where Θ_{r_i} represents the parameter set $\{\mu_{r_i}, \mathbf{\Sigma}_{r_i}\}$ for a cluster r_i in a subnet i

$$p(\mathbf{x}(t)|\omega_i) = \sum_{r_i=1}^{R_i} P(\Theta_{r_i}|\omega_i)p(\mathbf{x}(t)|\omega_i, \Theta_{r_i})$$

where $P(\Theta_{r_i}|\omega_i)$ denotes the prior probability of the cluster r_i. By definition, $\sum_{r_i=1}^{R_i} P(\Theta_{r_i}|\omega_i) = 1$, where R_i is the number of clusters in ω_i.

The discriminate function of the multiclass SPDNN models the log-likelihood function

$$\begin{aligned}\varphi(\mathbf{x}(t), \mathbf{w}_i) &= \log p(\mathbf{x}(t)|\omega_i)\\ &= \log\left[\sum_{r_i=1}^{R_i} P(\Theta_{r_i}|\omega_i)p(\mathbf{x}(t)|\omega_i, \Theta_{r_i})\right] \qquad (1)\end{aligned}$$

where $\mathbf{w}_i \equiv \mu_{r_i}, \mathbf{\Sigma}_{r_i}, P(\Theta_{r_i}|\omega_i), T_i\}$. T_i is the output threshold of the subnet i.

In most general formulations, the basis function of a cluster should be able to approximate the Gaussian distribution with a full rank covariance matrix, i.e., $\varphi(\mathbf{x}, \omega_i) = -(1/2)\mathbf{x}^T\mathbf{\Sigma}_{r_i}^{-1}\mathbf{x}$, where $\mathbf{\Sigma}_{r_i}$ is the covariance matrix. However, for applications which deal with high-dimension data but a finite number of training patterns, the training performance and storage space requirements discourage such matrix modeling. A natural simplifying assumption is to assume uncorrelated features of unequal importance. That is, suppose that $p(\mathbf{x}(t)|\omega_i, \Theta_{r_i})$ is a D-dimensional Gaussian distribution with uncorrelated features:

$$\begin{aligned}&p(\mathbf{x}(t)|\omega_i, \Theta_{r_i})\\ &= \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}_{r_i}|^{1/2}} \cdot \exp\left[-\frac{1}{2}\sum_{d=1}^{D} \frac{(x_d(t)-\mu_{r_id})^2}{\sigma_{r_id}^2}\right] \qquad (2)\end{aligned}$$

where $\mathbf{x}(t) = [\mathbf{x}_1(t), \mathbf{x}_2(t), \cdots, \mathbf{x}_D(t)]^T$ is the input pattern, $\mu_{r_i} = [\mu_{r_i1}, \mu_{r_i2}, \cdots, \mu_{r_iD}]^T$ is the mean vector, and diagonal matrix $\mathbf{\Sigma}_{r_i} = \mathrm{diag}[\sigma_{r_i1}^2, \sigma_{r_i2}^2, \cdots, \sigma_{r_iD}^2]$ is the covariance matrix. As shown in Fig. 4, an SPDNN contains K subnets which are used to represent a K-category classification problem. Inside each subnet, an elliptic basis function (EBF) serves as the basis function for each cluster r_i

$$\varphi(\mathbf{x}(t), \omega_i, \Theta_{r_i}) = -\tfrac{1}{2}\sum_{d=1}^{D} \alpha_{r_id}(x_d(t)-\mu_{r_id})^2 + \theta_{r_i} \qquad (3)$$

where $\theta_{r_i} = -(D/2)\ln 2\pi + (1/2)\sum_{d=1}^{D}\ln \alpha_{r_id}$. After passing an exponential activation function, $\exp\{\varphi(\mathbf{x}(t), \omega_i, \Theta_{r_i})\}$ can be viewed as a Gaussian distribution, as described in (2), except for a minor notational change: $1/\alpha_{r_id} = \sigma_{r_id}^2$.

B. Learning Rules for SPDNN

1) Supervised Learning in Each Subnet: Since the number of clusters in a subnet of an SPDNN can be adjusted in the self-growing phase, each subnet is initialized with one cluster. Suppose that $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \cdots, \mathbf{x}_i(M_i)\}$ is a set of given training characters, which correspond to one of the L classes $\{\omega_i, i=1, \cdots, L\}$; the mean μ_i and covariance $\mathbf{\Sigma}_i$ of the first cluster in subnet i can be initialized as

$$\mu_i = \frac{1}{M_i}\sum_{m=1}^{M_i} \mathbf{x}_i(m) \qquad (4)$$

$$\mathbf{\Sigma}_i = \frac{1}{M_i-1}\sum_{m=1}^{M_i} (\mathbf{x}_i(m)-\mu_i)(\mathbf{x}_i(m)-\mu_i)^T. \qquad (5)$$

During the **supervised learning** phase, training data are used to fine tune the decision boundaries of each class. Each class is modeled by a subnet with discriminant functions, $\varphi(\mathbf{x}(t), \mathbf{w}_i),\ i = 1, 2, \cdots, L$. At the beginning of each supervised learning phase, use the still-being-trained SPDNN to classify all the training characters $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \mathbf{x}_i(2), \cdots, \mathbf{x}_i(M_i)\}$ for $i = 1, \cdots, L$. $\mathbf{x}_i(m)$ is put into class ω_i if $\varphi(\mathbf{x}_i(m), \mathbf{w}_i) > \varphi(\mathbf{x}_i(m), \mathbf{w}_k), \forall k \neq i$, and $\varphi(\mathbf{x}_i(m), \mathbf{w}_i) \geq T_i$, where T_i is the output threshold for subnet i. According to the classification results, the training characters for each class i can be divided into three subsets:

- $D_1^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \in \omega_i, \mathbf{x}_i(m)$ is classified into ω_i (the correctly classified set)$\}$;
- $D_2^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \in \omega_i, \mathbf{x}_i(m)$ is misclassified into another class ω_j (the false rejection set)$\}$;
- $D_3^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \notin \omega_i, \mathbf{x}_i(m)$ is misclassified into class ω_i (the false acceptance set)$\}$.

The following reinforced and antireinforced learning rules [9] are applied to the corresponding misclassified subnets

Reinforced Learning:
$$\mathbf{w}_i^{(m+1)} = \mathbf{w}_i^{(m)} + \eta\nabla\varphi(\mathbf{x}_i(m), \mathbf{w}_i). \qquad (6)$$
Antireinforced Learning:
$$\mathbf{w}_j^{(m+1)} = \mathbf{w}_j^{(m)} - \eta\nabla\varphi(\mathbf{x}_i(m), \mathbf{w}_j). \qquad (7)$$

In (6) and (7), η is a user defined learning rate, where $0 < \eta \leq 1$. For the data set D_2^i, reinforced and antireinforced learning are applied to classes ω_i and ω_j, respectively. As for the false acceptance set D_3^i, antireinforced learning is applied to class ω_i, and reinforced learning is applied to class ω_j, to which $x_i(m)$ belongs.

The gradient vectors in (6) and (7) are computed as follows:

$$\begin{aligned}&\left.\frac{\partial\varphi(\mathbf{x}_i(m), \mathbf{w}_i)}{\partial\mu_{r_id}}\right|_{\mathbf{w}_i=\mathbf{w}_i^{(j)}}\\ &\quad= h_{r_i}^{(j)}(m)\cdot\alpha_{r_id}^{(j)}\Big(x_{id}(m)-\mu_{r_id}^{(j)}\Big) \qquad (8)\\ &\left.\frac{\partial\varphi(\mathbf{x}_i(m), \mathbf{w}_i)}{\partial\alpha_{r_id}}\right|_{\mathbf{w}_i=\mathbf{w}_i^{(j)}}\\ &\quad= h_{r_i}^{(j)}(m)\cdot\frac{1}{2}\left[\frac{1}{\alpha_{r_id}^{(j)}} - \Big(x_{id}(m)-\mu_{r_id}^{(j)}\Big)^2\right]. \qquad (9)\end{aligned}$$

By applying the EM algorithm [17], the intermediate parameter, $h_{r_i}^{(j)}(m)$, can be computed as follows: At each epoch j

$$h_{r_i}^{(j)}(m) = \frac{P^{(j)}(\Theta_{r_i}|\omega_i)p^{(j)}(\mathbf{x}_i(m)|\omega_i, \Theta_{r_i})}{\sum_{k_i=1}^{R_i} P^{(j)}(\Theta_{k_i}|\omega_i)p^{(j)}(\mathbf{x}_i(m)|\omega_i, \Theta_{k_i})}.$$

Also, the cluster prior probabilities $P(\Theta_{r_i}|\omega_i)$ can be updated by

$$P^{(j+1)}(\Theta_{r_i}|\omega_i) = \frac{1}{M_i}\sum_{m=1}^{M_i} h_{r_i}^{(j)}(m), \tag{10}$$

where M_i is the total number of training samples $\mathbf{x}_i(m)$. To update the mean μ_{r_i} and the diagonal covariance matrix $\mathbf{\Sigma}_{r_i}$, we apply the gradient ascent approach

$$\begin{aligned}\mu_{r_i}^{(j+1)} &= \mu_{r_i}^{(j)} + \zeta_\mu \frac{1}{N_{D_2^i}} \sum_{m=1}^{N_{D_2^i}} h_{r_i}^{(j)}(m)\mathbf{\Sigma}_{r_i}^{-1(j)}\left[\mathbf{x}_i(m) - \mu_{r_i}^{(j)}\right] \\ &\quad - \zeta_\mu \frac{1}{N_{D_3^i}} \sum_{m=1}^{N_{D_3^i}} h_{r_i}^{(j)}(m)\Sigma_{r_i}^{-1(j)}\left[\mathbf{x}_i(m) - \mu_{r_i}^{(j)}\right]\end{aligned} \tag{11}$$

$$\begin{aligned}\mathbf{\Sigma}_{r_i}^{(j+1)} &= \mathbf{\Sigma}_{r_i}^{(j)} + \frac{1}{2}\zeta_\sigma \frac{1}{N_{D_2^i}} \sum_{m=1}^{N_{D_2^i}} h_{r_i}^{(j)}(m)\left(\mathbf{H}_{r_i}^{(j)}(m) - \mathbf{\Sigma}_{r_i}^{-1(j)}\right) \\ &\quad - \frac{1}{2}\zeta_\sigma \frac{1}{N_{D_3^i}} \sum_{m=1}^{N_{D_3^i}} h_{r_i}^{(j)}(m)\left(\mathbf{H}_{r_i}^{(j)}(m) - \mathbf{\Sigma}_{r_i}^{-1(j)}\right)\end{aligned} \tag{12}$$

where ζ_μ and ζ_σ are user-defined positive learning rates, $N_{D_2^i}$ and $N_{D_3^i}$ are the number of characters in D_2^i and D_3^i, and

$$\mathbf{H}_{r_i}^{(j)}(m) = \mathbf{\Sigma}_{r_i}^{-1(j)}\left[\mathbf{x}_i(m) - \mu_{r_i}^{(j)}\right]\left[\mathbf{x}_i(m) - \mu_{r_i}^{(j)}\right]^T \mathbf{\Sigma}_{r_i}^{-1(j)}.$$

Threshold Updating: The threshold value $\mathbf{T}_i$ of a subnet i in the SPDNN recognizer can also be learned by means of reinforced or antireinforced learning rules.

2) Self-Growing a New Cluster: The network enters the self-growing phase when the supervised learning reaches a saturated learning state but with unsatisfactory classification accuracy. In other words, the whole training set has been presented a few times, the train status (especially the recognition accuracy) remains unchanged or unimproved, i.e., the training process falls into a local minimum. To escape from the local minimum, an extra cluster is needed in order to reshape the energy distribution of the current SPDNN. It is suggested that the new cluster be created from the subnet which caused most of the misclassifications during the recent supervised learning processes. When a new cluster is created, its initial *center* and *covariance* values should be properly determined; otherwise, poor classification will result.

Assume that a training character $\mathbf{x}$ corresponding to class ω_i is presented to an SPDNN classifier: the cluster Θ_i is in class ω_i, and the cluster (say Θ_j) is in the class ω_j which corresponds to the largest response among the classes other than ω_i. Let o_i and o_j be the output of $\mathbf{x}$ from class ω_i and class ω_j, respectively. According to the retrieving scheme of the proposed SPDNN, if o_j is larger than or equal to o_i, the retrieving result for the training character $\mathbf{x}$ still could be wrong. Thus, as shown in Fig. 5(a), the best position for the center of the new cluster should be located at $\mathbf{x}$, i.e., $\mu_0 = \mathbf{x}$, so that the class with the new cluster Θ_i' will generate the maximal output o_i for the training character $\mathbf{x}$. To determine the covariance matrix $\mathbf{\Sigma}_0$, we first let $\mathbf{\Sigma} = \sigma\mathbf{I}$, and let σ be a positive constant (to be determined). As shown in Fig. 5(b), if the σ of the new cluster Θ_i' is not properly determined, then $o_i(\mathbf{x})$, the output of class ω_i with the new cluster, is the largest output of all the classes', but this output may still be smaller than the output $o_j(\mathbf{x})$ of a class ω_j. In other word, cluster Θ_i' is still *overwhelmed* by cluster Θ_j, where μ_j and $\mathbf{\Sigma}_j$ are the center and covariance of cluster Θ_j. To prevent the *overwhelming* problem, Fig. 5(c) presents a properly initiated new cluster Θ_i'. The following two constraints are suggested for a proper initial value of σ:

$$\begin{aligned}o_j(\mathbf{x}) &= \frac{P(\Theta_j|\omega_j)}{(2\pi)^{D/2}|\Sigma_j|^{1/2}} \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d(t)-\mu_{r_jd})^2}{\sigma_{r_jd}^2}\right] + \epsilon_j \\ &< o_i(\mathbf{x}) = \frac{P(\Theta_0|\omega_i)}{(2\pi\sigma)^{D/2}} + \epsilon_i\end{aligned} \tag{13}$$

$$\begin{aligned}o_i(\mu_j) &= \frac{P(\Theta_0|\omega_i)}{(2\pi\sigma)^{D/2}} \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d(t)-\mu_{r_jd})^2}{\sigma_{r_id}^2}\right] + \epsilon_i' \\ &< o_j(\mu_j) = \frac{P(\Theta_j|\omega_j)}{(2\pi)^{D/2}|\Sigma_j|^{1/2}} + \epsilon_j'\end{aligned} \tag{14}$$

where $P(\Theta_0|\omega_i)$ and $P(\Theta_j|\omega_j)$ are the prior probability of clusters Θ_i' and Θ_j, respectively. ϵ_i and ϵ_j represent the partial output of classes ω_i and ω_j from the clusters other than Θ_i and Θ_j at $\mathbf{x}$. ϵ_i' and ϵ_j' are the partial output at μ_j of the clusters other than Θ_i and Θ_j. The prior probability $P(\Theta_0|\omega_i)$ of cluster Θ_i' can be initialized as $(\sigma/\overline{\sigma}_i)P(\Theta_i|\omega_i)$, where $\overline{\sigma}_i = (1/R_i)\sum_{r_i=1}^{R_i}\sigma_{r_i}$, in which σ_{r_i} is the variance of cluster r_i in class ω_i. Since ϵ_i,ϵ_j, ϵ_i', and ϵ_j', are very small at $\mathbf{x}$ and μ_i, they are ignored in the following σ estimation.

These two constraints imply that cluster Θ_i' and cluster Θ_j will not overwhelm each other. To satisfy (13), σ is initialized so as to be less than

$$\left(\frac{P(\Theta_i|\omega_i)}{(2\pi)^{D/2}\overline{\sigma}_i o_j(\mathbf{x})}\right)^{2/(D-2)}.$$

Then, σ can be iteratively decreased by a small value η $(0 \le \eta \le 1)$ until (14) is satisfied; the final value of σ will be a proper initial value for the new cluster Θ_i'.

3) Incremental Learning for User Adaptation: In this section, we propose an incremental learning algorithm to further fine-tune the decision boundaries of an SPDNN algorithm for personal handwriting recognition. The SPDNN algorithms introduced in previous sections are based on batch learning; that is, the parameters are updated after all the data from the character databases are considered. The incremental SPDNN algorithm learns the parameters of the SPDNN from the presentation of a user input pattern $x_i(n)$. At the beginning of the incremental learning phase, we use the batch-trained SPDNN to classify the

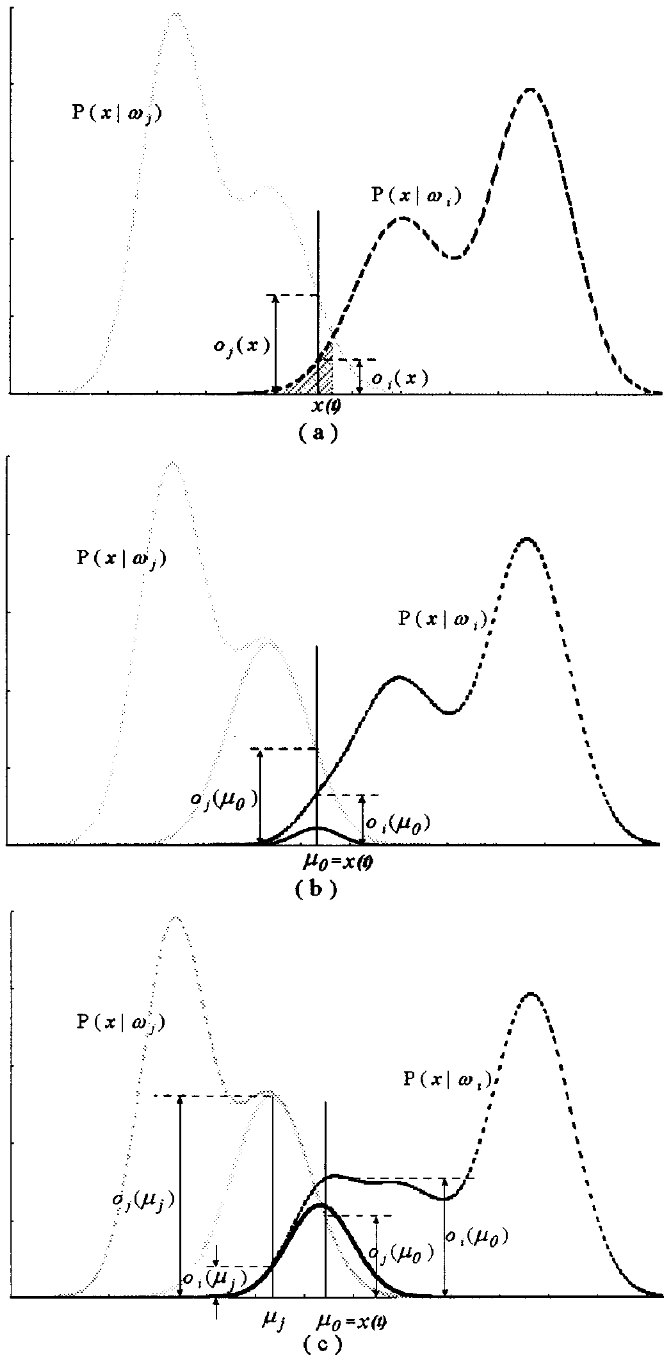


Fig. 5. Example of creating a new cluster in a mixture Gaussian distribution. (a) Suppose $x(t) \in \omega_i$; since $o_i(x(t))$ is smaller than $o_j(x(t))$, $x(t)$ is not correctly classified. A new cluster Θ_i' is needed in ω_i. (b) The new cluster Θ_i' is overwhelmed by the cluster Θ_j; i.e., $o_i(x(t))$ is still smaller than $o_j(x(t))$. (c) Through proper initialization of μ_0, σ_0, and $P(\Theta_0|\omega_i)$, the new cluster Θ_i' can sufficiently support class ω_i, such that $o_i(x(t))$ is larger than $o_j(x(t))$, and $o_i(\mu_j)$ is smaller than $o_j(\mu_j)$.

user input handwritten characters $x_i(n)$ for $n = 1, \cdots, N$, where N is the number of testing characters and the input character $x_i(n)$ is assumed to be in class ω_i. According to the classification results, each input character can be classified into one of the following three categories:

- $C_1^i = \{\mathbf{x}_i(n); \mathbf{x}_i(n) \in \omega_i, \mathbf{x}_i(n)$ is classified into ω_i (the correctly classified set)$\}$;
- $C_2^i = \{\mathbf{x}_i(n); \mathbf{x}_i(n) \in \omega_i, \mathbf{x}_i(n)$ is misclassified into other class ω_j (the false rejection set)$\}$;
- $C_3^i = \{\mathbf{x}_i(n); \mathbf{x}_i(n) \notin \omega_i, \mathbf{x}_i(n)$ is misclassified into class ω_i (the false acceptance set)$\}$.

As proposed by Jordan and Jacobs [7], by replacing $1/N$ with $\eta(N)$ and inserting the decay terms $\prod_{s=n+1}^{N} \lambda(s)$, into (11) and (12), we can convert these two batch learning equations into their incremental learning forms

$$\mu_{r_i}^{(j+1)} = \mu_{r_i}^{(j)} + \zeta_\mu \eta(N_{C_2^i}) \sum_{n=1}^{N_{C_2^i}} \left(\prod_{s=n+1}^{N_{C_2^i}} \lambda(s) \right) \cdot h_{r_i}^{(j)}(n) \Sigma_{r_i}^{-1(j)} \left[\mathbf{x}_i(n) - \mu_{r_i}^{(j)} \right] - \zeta_\mu \eta(N_{C_3^i}) \sum_{n=1}^{N_{C_3^i}} \cdot \left(\prod_{s=n+1}^{N_{C_3^i}} \lambda(s) \right) h_{r_i}^{(j)}(n) \Sigma_{r_i}^{-1(j)} \left[\mathbf{x}_i(n) - \mu_{r_i}^{(j)} \right] \tag{15}$$

$$\mathbf{\Sigma}_{r_i}^{(j+1)} = \mathbf{\Sigma}_{r_i}^{(j)} + \zeta_\sigma \eta(N_{C_2^i}) \sum_{n=1}^{N_{C_2^i}} \left(\prod_{s=n+1}^{N_{C_2^i}} \lambda(s) \right) h_{r_i}^{(j)}(n) \cdot \left(\mathbf{H}_{r_i}^{(j)}(n) - \mathbf{\Sigma}_{r_i}^{-1(j)} \right) - \zeta_\sigma \eta(N_{C_3^i}) \sum_{n=1}^{N_{C_3^i}} \cdot \left(\prod_{s=n+1}^{N_{C_3^i}} \lambda(s) \right) h_{r_i}^{(j)}(n) \left(\mathbf{H}_{r_i}^{(j)}(n) - \mathbf{\Sigma}_{r_i}^{-1(j)} \right). \tag{16}$$

The parameter $\lambda(s) \in [0, 1]$ is a decay parameter, which is introduced to forget the effect of the old posterior values obtained by employing the earlier inaccurate batch learning method. $N_{C_2^i}$ and $N_{C_3^i}$ are the numbers of characters in C_2^i and C_3^i, respectively. $\eta(N_{C_2^i})$ and $\eta(N_{C_3^i})$ are the normalization coefficients, and ζ_μ and ζ_σ both play a role like that of the learning rate for each incremental step. Since the latest training character should not decay due to its effect on the posterior values, in (15) and (16), when $n = N_{C_2^i}$ or $n = N_{C_3^i}$, we set $\prod_{s=n+1}^{N_{C_2^i}} \lambda(s) = 1$ or $\prod_{s=n+1}^{N_{C_3^i}} \lambda(s) = 1$, respectively.

The incremental learning of the mean and variance of (15) and (16) are performed after all N characters are recognized; however, some of them may be misclassified. Hence, the system needs to keep a record of the feature values of all the misclassified characters. Suppose one would like to update the mean and variance after each instance of misclassification is corrected; the following sequential algorithm is suggested.

Suppose the covariance matrix is in a diagonal form; one can formularize the sequential updating algorithm as follows:

$$\mu_{r_i d}^{(n+1)} = \mu_{r_i d}^{(n)} \pm \zeta_\mu \Delta \mu_{r_i d}^{(n)}, \qquad \text{for } 1 \le d \le D \tag{17}$$

$$\beta_{r_i d}^{(n+1)} = \beta_{r_i d}^{(n)} \pm \zeta_\sigma \Delta \beta_{r_i d}^{(n)}, \qquad \text{for } 1 \le d \le D. \tag{18}$$

According to the theorem and lemma presented in the Appendix, we let $M(n) = (\beta_{r_i}^{(n-1)}(x_i(n) - \mu_{r_i}^{(n-1)}))h_{r_i}^{(n)}$ and $f(N) = \Delta\mu_{r_i}^{(n)}$. Then, the incremental updating term, $\Delta\mu_{r_i}^{(n)}$, can be derived as follows:

$$\Delta\mu_{r_i}^{(n)} = (1-\eta(n))\Delta\mu_{r_i}^{(n-1)} + \eta(n)\left(\beta_{r_i}^{(n-1)}\left(\mathbf{x}_i(n) - \mu_{r_i}^{(n-1)}\right)\right)h_{r_i}^{(n)}. \quad (19)$$

Similarly, we let

$$M(n) = \left(\frac{1}{2}\left(\frac{1}{\beta_{r_i}^{(n-1)}} - \left(\mathbf{x}_i(n) - \mu_{r_i}^{(n-1)}\right)^2\right)\right)h_{r_i}^{(n)}$$

and $f(N) = \Delta\beta_{r_i}^{(n)}$; then

$$\Delta\beta_{r_i}^{(n)} = (1-\eta(n))\Delta\beta_{r_i}^{(n-1)} + \eta(n)\left(\frac{1}{2}\left(\frac{1}{\beta_{r_i}^{(n-1)}} - \left(\mathbf{x}_i(n) - \mu_{r_i}^{(n-1)}\right)^2\right)\right)h_{r_i}^{(n)}. \quad (20)$$

$\mu_{r_i}^{(0)}$ and $\beta_{r_i}^{(0)}$ are the mean and variance obtained throughout the previous batch learning of SPDNN.

IV. Three Phase Training in SPDNN for Handwriting Recognition

Based on SPDNN and its learning rules, the architecture of the proposed multistage SPDNN handwriting recognizer was designed as shown in Fig. 6.

A. Phase One: Global Training on Coarse Classification

In order to achieve balanced recognition performance in a multistage recognition system, the coarse classifier needs to maintain very high accuracy (e.g., $\geq$99.9%). Although this is a difficult task, we use the *CCT* feature and the K-means algorithm to initialize a set of SPDNN's for the coarse classifier. The K-means method adjusts the center of a cluster based on the distance $||x - \mu_i||$ of its neighboring patterns x. One limitation of the K-means algorithm is that the number of coarse classes needs to be decided before training. An alternative is to use the self-growing (SG) algorithm, as discussed in Section III-B2. By applying the SG algorithm, a new class is splitted up from an existing class when an input character pattern is determined (by a vigilance test) to be sufficiently far away from the existing classes. Therefore, by using the K-means algorithm, we can initialize the number of classes in the proposed coarse classifier to 20, where each contains 3000 characters on average. Then, by applying the SG algorithm, larger classes can be split, such that the target number of characters in each coarse class is less than 500 $\pm$ 50. By applying the two public databases suggested in Section II-E, we can train the proposed coarse classifier in order to achieve our goal. The training and testing results are listed in Table II. At the end of the learning phase for the coarse classifier, the total number of classes was increased to 61, and the number of candidate characters within a class is reduced to 516 characters on average.

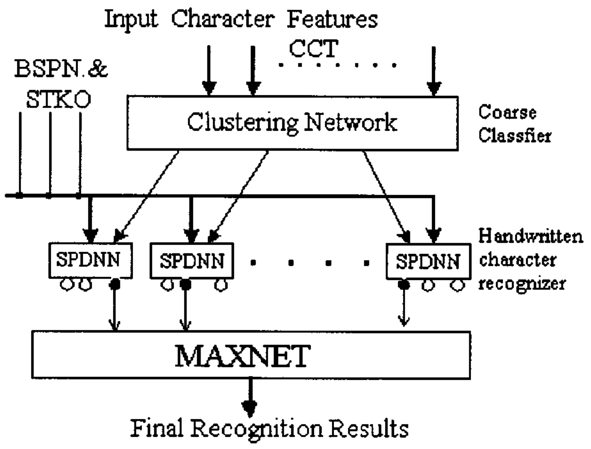


Fig. 6. The architecture of our SPDNN based two-stage handwritten character recognition system.

TABLE II
THE TRAINING AND TESTING RESULTS OF COARSE CLASSIFICATION ON THE CCL/HCCR1 AND THE CEDAR DATABASES. HALF OF THE CHARACTERS (EVEN NUMBERED) IN EACH OF DATABASES WERE USED FOR TRAINING AND THE OTHER HALF (ODD NUMBERED) OF THE CHARACTERS WERE USED FOR TESTING

Number of coarse class	Ave. No. of characters in a class	Training Accuracy	Testing Accuracy
61	516	99.9 %	99.8%

B. Phase Two: Batch Training in Character Recognition

The design of the character recognizer is also based on the SPDNN model. For a K-character recognition problem, an SPDNN character recognizer consists of K subnets. A subnet i in the SPDNN recognizer estimates the distribution of the patterns of character i only and treats those patterns which do not belong to character i as "non$i$" patterns. The combined features, such as CCT, BSPN, and STKO, are used in the SPDNN character recognizer. The parameters of each subnet i are initialized with one cluster according to (4) and (5). Then, the decision boundaries of the subnet are fine-tuned according to the reinforced and antireinforced learning rules, as shown in (6) and (7). Suppose that one cluster in a subnet is not sufficient to represent the complicated distribution of a character i; then, the self-growing and supervised learning rules are applied to create a new cluster. During the testing phase, each of the subnets corresponding to the candidate characters (which are included in the cluster that is selected by the coarse classifier) produces a score according to its discrimination function $\varphi(\mathbf{x}(t), \mathbf{w}_i)$. The subnet which produces the highest score is the winner, and its corresponding reference character is taken as the recognition result.

Two experimental results from the SPDNN character recognizer will be discussed. The first type of recognition experiment were performed on the CCL/HCCR1 [15] handwriting database, which has been used by several handwriting recognition research groups [2], [4], [10]. The second type of experiment explored the ability of SPDNN to deal with the multilinguistic handwriting recognition problem, which has seldom been discussed in the character recognition literature.

TABLE III
PERFORMANCE OF DIFFERENT HANDWRITING RECOGNIZERS ON THE CCL/HCCR1 DATABASE. A PORTION OF THIS TABLE IS ADAPTED FROM LI *et al.* [10] AND TSENG *et al.* [14]

Various Systems	Recognition Accuracy	Features Used	Train & testing data used	Classification time
SPDNN	86.12%	92	50-50	0.24 sec/char
Li *et al.*	88.65%	400	50-1	NA
Tseng *et al.*	88.55%	256	100-100	0.6 sec/char

1) Experiment 1—Handwritten Chinese Recognition: According to the most recent survey on handwriting recognition [1], [12], [13], most of the handwritten Chinese OCR studies are designed for small databases, i.e., training and testing on very small character sets, e.g., a few hundred characters. As for studies conducted on recognition using a complete set of commonly used Chinese characters, Xia [16] developed an experimental system with a 3755 character set and achieved an 80% recognition rate. In [10], Li and Yu reported 93.43% recognition accuracy on the CCL/HCCR1 database. Recently, Tseng *et al.* [14] used the $M\_distance$ method in their recognition system to achieve 88.55% accuracy on the CCL/HCCR1 database. The SPDNN character recognizer achieved 86.12% recognition accuracy. Table III summarizes a performance comparison of these systems evaluated using the CCL/HCCR1 database. We would like to comment on the overall performance of these systems as follows. First, compared to the huge number of character features used by other researchers, e.g., 400 features used in [10] or 256 features used in [14], the SPDNN recognizer uses only 92 features. A more relevant comparison could be made if a comparable number of training and testing features for these two systems were available. In fact, the SPDNN character recognizer is designed to use no more than 100 sets of features since more feature sets would require more memory storage and longer recognition time. Two reasons explain why an SPDNN-based system can have fewer features yet achieve comparable performance. 1) The mixed Gaussian-based discrimination function permits SPDNN to learn the character decision boundary precisely. 2) The self-growing rules allow a small number of Gaussian clusters to be sufficient to represent the character image distribution.

2) Experiment 2—Multilinguistic Handwriting Recognition: We have conducted two types of experiments with or without rejection criteria. Rejection criteria were implemented using the threshold value T_i, which can be learned by means of the reinforced and antireinforced learning rules. In general, when an input character is correctly recognized with a certain degree of confidence, its output of the discriminate function should maintain a certain gap ($\geq T_i$) with respect to the second largest output obtained by other discrimination functions.

The experimental results are discussed as follows: The recognition accuracy with 0% and 6.7% false rejection rates in the testing phase are shown in Table IV. None of the systems developed by Li *et al.* [10] and Tseng *et al.* [14] includes rejection criteria. We think that rejection criteria can be beneficial in reducing the false rejection and false acceptance rates.

TABLE IV
PERFORMANCE OF SPDNN HANDWRITTEN CHARACTER RECOGNIZERS WITH AND WITHOUT REJECTION ON THE CCL/HCCR1 AND CEDAR DATABASES

Systems	Top 1 Accu.	Top 2 Accu.	Top 3 Accu.	Rej. %
SPDNN	90.12%	93.49%	94.75 %	0 %
SPDNN	94.11%	97.01%	97.67 %	6.7 %

C. Phase Three: Personal Adaptation in Handwriting Recognition

As indicated in Section II-D, recognition performance for personal handwriting of the batch-trained handwriting recognizers can be as poor as 40–45%. Thus, we propose that user adaptive training be employed to further enhance the recognition accuracy of the SPDNN handwriting recognition system. According to the proposed incremental EM algorithm, the training is designed to adapt user input samples. In order to minimize inconvenience for the user, adaptive training is performed only when recognition errors are detected and corrected by the user.

Statistical theory of pattern recognition shows that the decision boundaries generated by the SPDNN posterior probabilities produce near minimum classification error. Since the SPDNN model is derived based on the assumption that **all** the data "are known," the SPDNN decision boundaries may not be suitable for recognition problems with unknown data, in this case, handwriting of the current user. One simple example is shown in Fig. 7(a). For l-class recognition problems, the proposed SPDNN decision boundaries are determined by the batch-learning process, which divides the feature space into l different regions. The asteroid points represent that misclassified characters, caused by the handwritten variation of the current user. Suppose x_i is a character that belongs to class K_1 but is misclassified into class K_4. According to the incremental learning rules given in Section III-B3, reinforced and antireinforced learning will be applied to K_1 and K_4, respectively. As shown in Fig. 7(b), the learning process moves the boundary of K_1 toward x_i, and inversely moves the boundary of K_4 away from x_i. Hopefully, x_i would be correctly recognized due to retraining of SPDNN. However, let us look at Fig. 7(b) more closely; there are some areas of intersection between the new region of K_1, the regions of classes K_5 and K_3,etc. It is possible that character points located in these intersection regions may not be correctly recognized by the retrained SPDNN. Thus, the job of adaptation is not only to adjust the decision boundaries of the data class that is directly related to misclassified characters, but also to modify the decision boundaries of the classes that may be affected by the retraining process. *What are the affected classes and how can they be found?* According to the decision boundary distribution, these affected regions due to

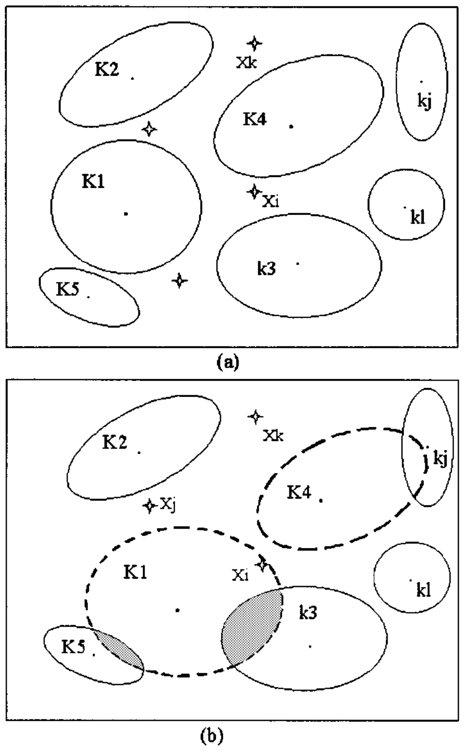


Fig. 7. An example of an l-class character distribution diagram. "*" represents misclassified characters. (a) Character x_i is supposed to be in class K_1 but is misclassified into K_4. (b) The distribution boundaries of K_1 and K_4 are retrained based on the misclassified character x_i and the proposed incremental EM algorithm.

the retraining of class K_1 could be decision boundaries near the boundary of class K_1. From the recognition point of view, these nearby classes are the classes in the top candidate list of each recognition of a character assumed to be in class K_1. Thus, we propose a two-step incremental training method.For each misclassified character x_i:

1) retrain the decision boundary of K_i if the character $x_i \in K_i$ is misclassified into class $K_j, j \neq i$;
2) retrain the decision boundaries of character classes near class K_i.

In the first step, misclassified characters are used as negative training samples, and the assumed characters are used as positive training samples. When the error correction processes are finished, the system enters the second step of retraining by issuing a list of characters which are collected from the top ten candidates for each misclassified character. The user is asked to prepare free-handwritten samples of the issued characters. These handwritten samples again are used to train the system to enhance its recognition capability. Gradually, the SPDNN character recognizer will learn the user's own writing style with minimum training effort.

1) Experimental Results and Performance Evaluation: In order to evaluate the recognition performance of the character recognizer before and after the adaptation processes, we prepared our own in-house database (NCTU/NNL) in the following manner. We first selected the most commonly used 300 characters from the Chinese textbooks used in elementary schools in Taiwan. Then, these 300 Chinese characters and 62 alphanumerics were written without any restrictions on the writing style by several students in our university as the original training sets. We intended to prepare natural and general freehand-written training and testing data in this manner. After the original training characters were recognized and any errors corrected, the SPDNN system generated a set of retraining characters. These characters were then written by the current user as a set of new training samples. These samples were tested and corrected like the original samples. Again, a new retraining character list was generated for the next training process. Repeatedly, retraining was performed nine more times. The testing results for five user's adaptation processes are listed in Table V. The recognition rate increased from 44.2% to 82.4% at the end of the fifth learning cycle. The performance finally increased up to 90.2% in ten learning cycles. In addition, the total number of training characters generated by each user in these ten training cycles was about 600 characters on average.

TABLE V
APPLIED TO 300 COMMONLY USED CHARACTERS WRITTEN WITHOUT ANY CONSTRAINTS BY FIVE STUDENTS, THE PROPOSED ADAPTIVE PROCESS PRODUCED SIGNIFICANT IMPROVEMENT IN RECOGNITION ACCURACY DURING TEN LEARNING CYCLES

Trial	user#1	user#2	user#3	user#4	user#5	avg.
1st	50.6%	33.7%	38.6%	52.5%	45.9%	44.2%
2nd	67.8%	69.1%	55.9%	56.8%	61.3%	62.2%
3rd	78.6%	80.1%	70.4%	71.8%	72.8%	74.7%
4th	84.4%	78.8%	69.8%	76.0%	86.2%	79.7%
5th	84.6%	87.7%	74.4%	80.1%	85.3%	82.4%
6th	82.1%	89.1%	76.5%	80.4%	84.2%	82.4%
7th	86.5%	89.7%	80.2%	79.0%	84.7%	84.0%
8th	89.6%	90.3%	80.0%	87.1%	89.5%	87.3%
9th	90.5%	90.7%	81.7%	87.9%	89.5%	88.1%
10th	93.6%	91.5%	85.1%	90.6%	90.4%	90.2%

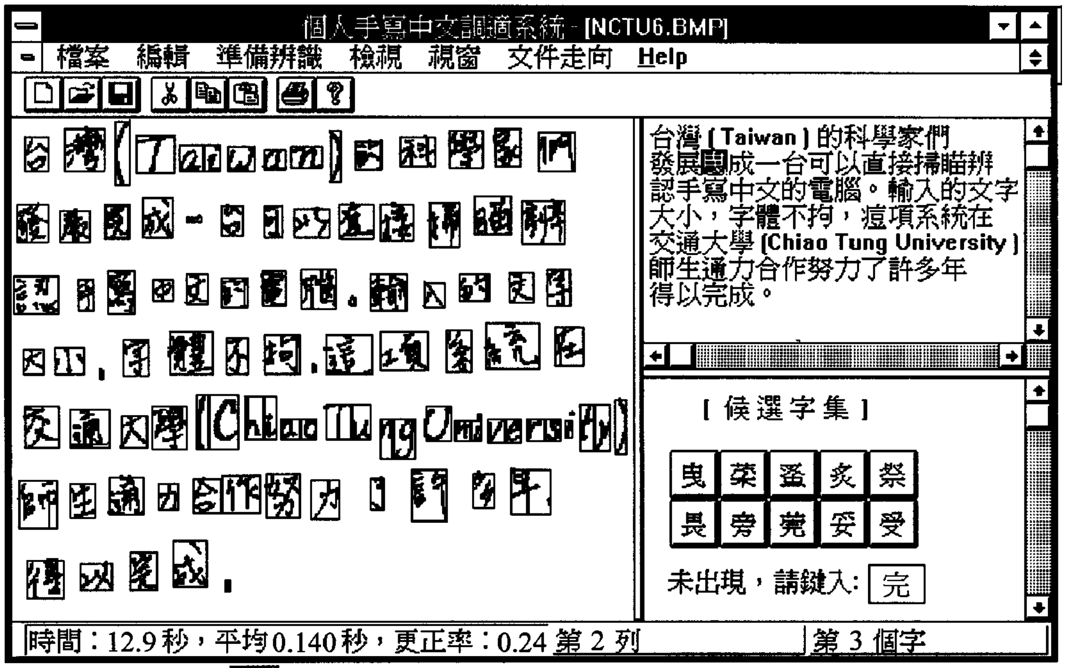
D. Prototyping and Graphical User Interface

The proposed two-stage recognition system has been implemented on a personal computer. Since the system may require a user to correct recognition errors, a friendly user interface is necessary. Fig. 8 depicts the graphical user interface of the prototype system. The score bar at the bottom of the window depicts the performance of the this system with respect to both recognition accuracy and processing time.

The user interface window contains three partitions: the left partition displays a binary image of a handwritten document, the upper right partition displays the recognition results and the lower right partition provides an interactive error correcting interface. When a user wants to correct a recognition error, (s)he can move the cursor over the misclassified character first; then, its corresponding top ten recognition candidates will be displayed at the buttons. If the correct character is among these ten candidates, then the user can move the cursor and click on the corresponding button to make the correction. Otherwise, the user can manually input a character to correct a recognition error.

V. CONCLUDING REMARKS

In this paper, a neural-network-based user adaptive handwriting recognition system has been proposed and implemented on a Pentium-II-based personal computer. This recognition


個人手寫中文調適系統 - [NCTU6.BMP]
檔案 編輯 準備辨識 檢視 視窗 文件走向 Help
[候選字集]
臾 棄 蚤 炙 祭
畏 旁 莧 妥 受
未出現，請鍵入: 完

台灣 (Taiwan) 的科學家們
發展圓成一台可以直接掃瞄辨
認手寫中文的電腦。輸入的文字
大小，字體不拘，這項系統在
交通大學 (Chiao Tung University)
師生通力合作努力了許多年
得以完成。

時間：12.9秒，平均0.140秒，更正率：0.24 第 2 列 第 3 個字

Fig. 8. The graphical user interface of the prototype system. There are three windows in this interface: 1) the input character image window, 2) the recognition results window, and 3) the error correction window. An input character image is acquired from a scanner, and then it is preprocessed into isolated character images. Each isolated character image is recognized and displayed in the upper-right window. The third character in the second row, shown in reverse video, is picked up by the user due to misclassification. In the mean time, ten buttons in the lower-right window display the top ten candidates of the misclassified characters. Personal adaptation is performed incrementally when the user corrects a misclassified character. The message bar at the bottom of the interface window shows that the total processing time for the sample character image is 12.9 s, which is about 0.140 s per character, and that the error rate is 0.24.

system performs preclassification, character recognition, and personal adaptation. The SPDNN has been applied to implement the major recognition modules of this system. This modular neural network deploys one subnet for one object (character); therefore, it is able to approximate the decision region of each class locally and precisely. This locality property is attractive, especially for personal handwriting recognition or signature identification. Moreover, because its discrimination function obeys a probability constraint, SPDNN has some nice properties, such as low false acceptance/false rejection rates. An incremental EM algorithm-based adaptation module has been proposed and implemented to further improve the recognition performance for personal handwriting. On the other hand, due to the enormous number of variations involved, handwriting recognition still requires more research work before it will be able to achieve a level of performance comparable to that of a human. Therefore, document analysis and recognition have become interesting research topics in the field of intelligent information processing.

Appendix

Lemma: If $\eta(N) = (\sum_{n=1}^{N}(\prod_{s=n+1}^{N}\lambda(s)))^{-1}$ and $\eta(1) = 1$, then $\eta(N) = (1 + \lambda(N)/\eta(N-1))^{-1}$.

Proof:

$$\eta(N) = \left(\sum_{n=1}^{N}\left(\prod_{s=n+1}^{N}\lambda(s)\right)\right)^{-1}$$

$$= \frac{1}{\sum_{n=1}^{N}\left(\prod_{s=n+1}^{N}\lambda(s)\right)}$$

$$= \frac{1}{\lambda(N)(\eta(N-1))^{-1} + 1}$$

$$= \left(1 + \frac{\lambda(N)}{\eta(N-1)}\right)^{-1}.$$

Theorem: If $f(N) = \eta(N)\sum_{n=1}^{N}(\prod_{s=n+1}^{N}\lambda(s))M(n)$, then $f(N) = (1-\eta(N))f(N-1) + \eta(N)M(N)$.

Proof:

$$\begin{aligned} f(N) &= \eta(N)\sum_{n=1}^{N}\left(\prod_{s=n+1}^{N}\lambda(s)\right)M(n) \\ &= \eta(N)\lambda(N)\sum_{n=1}^{N-1}\left(\prod_{s=n+1}^{N-1}\lambda(s)\right)M(n) + \eta(N)M(N) \\ &= \frac{\eta(N)}{\eta(N-1)}\lambda(N)f(N-1) + \eta(N)M(N). \end{aligned}$$

In Lemma, we know that $\lambda(N)/\eta(N-1) = 1/\eta(N) - 1$. Therefore,

$$\begin{aligned} f(N) &= \left(\frac{1}{\eta(N)} - 1\right)\eta(N)f(N-1) + \eta(N)M(N) \\ &= (1-\eta(N))f(N-1) + \eta(N)M(N). \end{aligned}$$

ACKNOWLEDGMENT

The authors thank Prof. S. Y. Kung and Dr. S. H. Lin for their helpful suggestions regarding the probabilistic DBNN, and D. Ludwig for his editorial effort in improving the English writing in this paper.

REFERENCES

[1] F. H. Cheng and W. H. Hsu, "Research on Chinese OCR in Taiwan," *Int. J. Pattern Recognition Artificial Intell.*, vol. 5, no. 1/2, pp. 139–164, 1991.

[2] C.-C. Chiang, T. Cheng, and S.-S. Yu, "An iterative rule-based character segmentation method for Chinese documents," in *Proc. Int. Conf. Chinese Comput.'96*, Singapore, June 4–7, 1996.

[3] P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*. Englewood Cliffs, NJ: Prentice-Hall, 1982.

[4] H. C. Fu and K. P. Chiang, "Recognition of handwritten Chinese characters by multistage neural network classifiers," in *Proc. 1995 IEEE Int. Conf. Neural Networks*, Perth, Australia.

[5] H. C. Fu, S. C. Chuang, Y. Y. Xu, W. H. Su, and K. T. Sun, "A personal adaptive module for unconstrained handwritten Chinese characters recognition," in *Proc. Int. Symp. Multitechnol. Inform. Processing*, Hsinchu, Taiwan, R.O.C., Dec. 1996.

[6] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 550–554, June 1994.

[7] M. I. Jordan and R. A. Jacobs, "Hierarchical mixture of experts and the EM Algorithm," *Neural Comput.*, vol. 6, pp. 181–214, 1994.

[8] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, "Modified quadratic discriminant functions and application to Chinese character recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-9, pp. 149–153, 1987.

[9] S. Y. Kung and J. S. Taur, "Decision-based hierarchical neural networks with signal/image classification applications," *IEEE Trans. Neural Networks*, vol. 6, pp. 170–181, Jan. 1995.

[10] T.-F. Li and S.-S. Yu, "Hand-printed Chinese character recognition using the probability distribution feature," *Int. J. Pattern Recognition Artificial Intell.*, vol. 8, no. 5, pp. 1241–1258, 1994.

[11] S.-H. Lin, S. Y. Kung, and L. J. Lin, "Face recognition/detection by probabilistic decision-based neural networks," *IEEE Trans. Neural Networks (special issue on Artificial Neural Network and Pattern Recognition)*, vol. 8, pp. 114–132, Jan. 1997.

[12] S. Mori, C. Y. Suen, and K. Yamamoto, "Historical review of OCR research and development," *Proc. IEEE*, vol. 80, pp. 1029–1058, July 1992.

[13] J.-W. Tai, "Some research achievements on Chinese character recognition in China," *Int. J. Pattern Recognition Artificial Intell.*, vol. 5, no. 1/2, pp. 199–206, 1991.

[14] Y.-H. Tseng, C.-C. Kuo, and H.-J. Lee, "Speeding up Chinese character recognition in an automatic document reading system," *Pattern Recognition*, vol. 31, no. 11, pp. 1601–1612, 1998.

[15] L. T. Tu, Y. S. Lin, C. P. Yeh, I. S. Shyu, J. L. Wang, K. H. Joe, and W.-W. Lin, "Recognition of handprinted Chinese characters by feature matching," in *Proc. 1991 First Nat. Workshop Character Recognition*, Taipei, R.O.C., 1991, pp. 166–175.

[16] Y. Xia, "Research report on interactive self-learning system of handwritten Chinese characters," Department of Computer Science, QingHua University, Nov. 1989.

[17] L. Xu, M. I. Jordan, and G. E. Hinton, "A modified gating network for the mixture of experts architecture," in *Proc. World Congr. Neural Networks*, San Diego, CA, 1994, pp. II405–II410.

Hsin-Chia Fu (M'78) received the B.S. degree from National Chiao-Tung University, Taiwan, R.O.C., in electrical and communication engineering in 1972, and the M.S. and Ph.D. degrees from New Mexico State University, Las Cruces, both in electrical and computer engineering in 1975 and 1981, respectively.

From 1981 to 1983, he was a Member of the Technical Staff at Bell Laboratories, Indianapolis, IN. Since 1983, he has been on the faculty of the Department of Computer Sience and Information Engineering at National Chiao-Tung University. From 1987 to 1988, he served as the Director of the Department of Information Management, Research Development and Evaluation Commission, Executive Yuan, R.O.C. From 1988 to 1989, he was a Visiting Scholar at Princeton University, Princeton, NJ. From 1989 to 1991, he served as the Chairman of the Department of Computer Science and Information Engineering, National Chiao-Tung University. From September to December of 1994, he was a Visiting Scientist at Fraunhofer-Institut for Production Systems and Design Technology (IPK), Berlin, Germany. His research interests include digital signal/image processing, VLSI array processors, and neural networks. He has authored more than 100 technical papers and two textbooks: *PC/XT BIOS Analysis* (Taipei, Taiwan, R.O.C.: Sun-Kung, 1986) and *Introduction to Neural Networks,* (Taipei, Taiwan, R.O.C.: Third Wave, 1994).

Dr. Fu was the corecipient of the 1992 and 1993 Long-Term Best Thesis Award with Koun Tem Sun and Cheng Chin Chiang, and the recipient of the 1996 Xerox OA Paper Award. He has served as a founding member, Program Cochair in 1993, and General Cochair in 1995 of the International Symposium on Artificial Neural Networks. He is presently serving the Technical Committee on Neural Networks for Signal Processing of the IEEE Signal Processing Society. He is a member of the IEEE Signal Processing and Computer Societies, Phi Tau Phi, and the Eta Kappa Nu Electrical Engineering Honor Society.

Hung-Yuan Chang was born in Miaoli, Taiwan, R.O.C., in 1966. He received the B.S. and M.S. degrees from the National Chiao-Tung University, Taiwan, R.O.C., both in computer science and information engineering in 1989 and 1993, respectively. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Information Engineering at the National Chiao-Tung University.

His research interests include handwriting recognition, speech recognition, and neural networks.

Yeong Yuh Xu was born in Hsinchu, Taiwan, R.O.C., in 1973. He received the B.S. degree in electrical engineering in 1995 from National Sun Yat-Sen University, Kaohsiung, Taiwan, R.O.C., and the M.S. degree in Computer Science and Information Engineering in 1997 from National Chiao-Tung University, Hsinchu, Taiwan, R.O.C. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Information Engineering at the National Chiao-Tung University.

His research interests include pattern recognition, neural networks, and content-based image/video retrieval.

H.-T. Pao received the B.S. degree from National Cheng-Kung University, Taiwan, R.O.C., in mathematics in 1976, and the M.S. and Ph.D. degrees from National Chiao-Tung University, Taiwan, R.O.C., both in applied mathematics in 1981 and 1994, respectively.

From 1983 to 1985, she was a Member of the Assistant Technical Staff at Tel-communication Laboratories, Chung-Li, Taiwan, R.O.C. Since 1985, she has been on the faculty of the Department of management science at National Chiao-Tung University, in Taiwan, R.O.C. Her research interests include statistics and neural networks.