Available online at www.sciencedirect.com
SCIENCE DIRECT®
NEUROCOMPUTING
ELSEVIER
Neurocomputing 61 (2004) 21–38
www.elsevier.com/locate/neucom

A self-growing probabilistic decision-based neural network with automatic data clustering☆

C.L. Tseng[a,*], Y.H. Chen[a], Y.Y. Xu[a], H.T. Pao[b], Hsin-Chia Fu[a]

[a] *Department of Computer Science and Information Engineering, National Chiao Tung University, Taiwan, Taiwan*

[b] *Department of Management Science, National Chiao Tung University, Hsin-Chu, Taiwan, ROC*

Available online 3 August 2004

Abstract

In this paper, we propose a new clustering algorithm for a mixture of Gaussian-based neural network and self-growing probabilistic decision-based neural networks (SPDNN). The proposed self-growing cluster learning (SGCL) algorithm is able to find the natural number of prototypes based on a self-growing validity measure, Bayesian information criterion (BIC). The learning process starts from a single prototype randomly initialized in the feature space and grows adaptively during the learning process until most appropriate number of prototypes are found. We have conducted numerical and real-world experiments to demonstrate the effectiveness of the SGCL algorithm. In the results of using SGCL to train the SPDNN for data clustering and speaker identification problems, we have observed a noticeable improvement among various model-based or vector quantization-based classification schemes.

Keywords: Self-growing probabilistic decision-based neural networks (SPDNN); Supervised learning; Automatic data clustering; Validity measure; Bayesian information criterion

1. Introduction

The last two decades have seen a growing number of researchers and practitioners applying neural networks (NNs) to a variety of problems in engineering applications and other scientific disciplines. In many of these neural network applications, data clustering techniques have been applied to discover and to extract hidden structure in a data set. Thus the structural relationships between individual data points can be

☆ This research was supported in part by the National Science Council under Grant NSC 90-2213-E009-047.
* Corresponding author.

0925-2312/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2004.03.002

detected. In neural network community, data clustering is commonly implemented by unsupervised competitive learning techniques [3,15,21,4]. The goal of unsupervised competitive learning is the minimization of clustering distortion or quantization error.

There are two major problems associated with competitive learning (CL), namely, sensitivity in selecting the initial location and difficulty in determining the number of prototypes. In general, selecting the appropriate number and location of prototypes is a difficult task, as we do not usually know the number of clusters in the input data a priori. It is therefore desirable to develop an algorithm that has no dependency on the initial prototype locations and is able to adaptively generate prototypes to fit the input data patterns.

A variety of clustering schemes have been developed, different in their approaches to competition and learning rules. The simplest and most prototypical CL algorithms are mainly based on the *winner-take-all* (WTA) [13] paradigm, where adaptive learning is restricted to the *winner* that is the single neuron prototype best matching the input pattern. Different algorithms in this paradigm such as LBG (or generalized Lloyd) [20,7,22] and k-means [23] have been well recognized. A major problem with the simple WTA learning is the possible existence of under-utilization or the so-called dead nodes problem [27]. In other words, some prototypes, due to inappropriate initialization can never become winners. Significant efforts have been made to deal with this problem. By relaxing the WTA criterion, soft competition scheme (SCS) [31], neural-gas network [24] and fuzzy competitive learning (FCL) [2], considering more than a single prototype as winners to a certain degree and updating them accordingly, results in the *winner-take-most* (WTM) paradigm (soft competitive learning). WTM decreases the dependency on the initialization of prototype locations; however, it has an undesirable side effect [21]: when all prototypes are attracted to each input pattern, some of them may be detracted from their corresponding clusters, and these prototypes may become biased toward the global mean of the clusters. Xu et al. [30] proposed a rival penalized competitive learning (RPCL) algorithm to tackle this problem. The basic idea in RPCL is that for each input pattern, not only the weight of the frequency-sensitive winner is learned to shift toward the input pattern, but also the weight of its rival (the 2nd winner) is delearned by a smaller learning rate. Another well-known problem with competitive learning is the difficulty in determining the number of clusters [8]. Determining the optimum number of clusters is a largely unsolved problem, due to lack of prior knowledge in the data set. The growing cell structure (GCS) [9] and growing neural gas (GNG) [10] algorithms are proposed to be different from the previously described models, by increasing the number of prototypes during the self-organization process. The insertion of a neuron is judged at each pre-specified number of iterations and the stop criterion is simply the network size or some ad hoc subjective criteria on the learning performance. In addition, it is also required that the initial number of prototypes be at least two, which is not always the choice since sometimes a single cluster may exist in the data set. To alleviate this problem, Zhang et al. [32] proposed a *one-prototype-take-one-cluster* learning paradigm and a self-splitting competitive learning (SSCL) algorithm, which starts the learning process from a randomly initialized single prototype in the feature space. During the learning period, one of the prototypes will be chosen to split into two prototypes according to a split validity criterion.

However, this method needs a threshold ε, to determine whether or not a prototype is suitable for splitting. Since usually no information about the threshold is available, it must be determined adaptively from the analysis of the feature space, e.g., the threshold ε may be defined as the average variance for Gaussian distributed clusters.

In this paper, we propose a new clustering algorithm for a mixture of Gaussian-based neural network, called self-growing probabilistic decision-based neural networks (SPDNN) [11]. Using the Bayesian information criterion (BIC) as a self-growing validity measure, the proposed self-growing cluster learning (SGCL) algorithm is able to find the natural number of prototypes in a class of input patterns. The learning process starts from randomly initializing a single prototype in the feature space and adaptively growing the prototypes until the most appropriate number of prototypes is reached. We have conducted numerical and real-world experiments to demonstrate the effectiveness of the SGCL algorithm. In the performance results of using SGCL to train the SPDNN, we observed a noticeable improvement among various model-based or vector quantization-based classification schemes.

The remainder of this paper is organized as follows. In Section 2, we describe in detail the architecture of SPDNN and its discriminant functions. Then, in Section 3 the learning rules of SPDNN and the SGCL Algorithm are presented. Section 4 presents the performance results on numerical clustering experiments and real-world applications. Finally, Section 5 draws the summary and concluding remarks.

2. Self-growing probabilistic decision-based neural network

As shown in Fig. 1, self-growing probabilistic decision-based neural network (SPDNN) is a multi-variate of Gaussian neural network [16,19]. The training scheme of SPDNN is based on the so-called locally unsupervised globally supervised (LUGS) learning. There are two phases in this scheme: during the locally-unsupervised (LU) phase, prototypes in each subnet are learned and grown according to the proposed self-growing cluster learning (SGCL) algorithm (see Section 3 A.2), and no mutual information across the classes may be utilized. After the LU phase is completed, the training enters the globally-supervised (GS) phase. In the GS phase, teacher information is introduced to reinforce or anti-reinforce the decision boundaries between classes. A detailed description of the SPDNN model and the proposed learning schemes will be given in the following sections.

2.1. Discriminant functions of SPDNN

One of the major differences between traditional multi-variate Gaussian neural networks (MGNN) [16,19] and SPDNN is that SPDNN adapts a flexible number of clusters instead of fixed number of clusters in a subnet, which models a class of data patterns. That is, the subnet discriminant functions of SPDNN are designed to model the log-likelihood functions of different complexed pixel distributions of a data pattern. Given a set of iid patterns $\mathbf{X} = \{x(t);\ t = 1, 2, \ldots, N\}$, we assume that the likelihood function $p(\mathbf{x}(t)|\omega_i)$ for class ω_i is a mixture of Gaussian distributions.

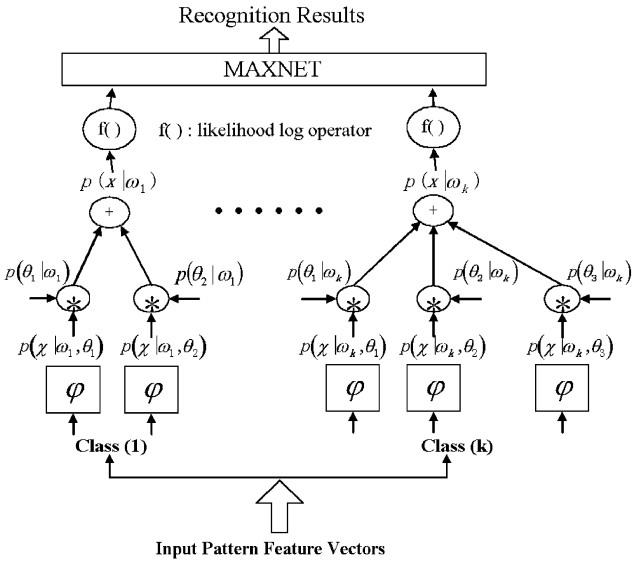

Fig. 1. The schematic diagram of a k-class SPDNN.

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 multi-class SPDNN models the log-likelihood function

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

where $\mathbf{w}_i = \{\mu_{r_i}, \mathbf{\Sigma}_{r_i}, P(\Theta_{r_i}|\omega_i), T_i\}$. T_i is the output threshold of the subnet i (cf. Section 3).

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., $\varphi(\mathbf{x}, \omega_i) = -\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{\Sigma}_{r_i}^{-1}\mathbf{x}$, 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:

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

where $\mathbf{x}(t)=[\mathbf{x}_1(t), \mathbf{x}_2(t), \ldots, \mathbf{x}_D(t)]^{\mathrm{T}}$ is the input, $\mu_{r_i}=[\mu_{r_i 1}, \mu_{r_i 2}, \ldots, \mu_{r_i D}]^{\mathrm{T}}$ is the mean vector, and diagonal matrix $\mathbf{\Sigma}_{r_i} = diag[\sigma_{r_i 1}^2, \sigma_{r_i 2}^2, \ldots, \sigma_{r_i D}^2]$ is the covariance matrix. As shown in Fig. 1, 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:

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

where $\theta_{r_i} = -D/2\ln 2\pi + 1/2\sum_{d=1}^{D}\ln\alpha_{r_i d}$. 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: $1/\alpha_{r_i d} = \sigma_{r_i d}^2$.

3. Learning rules and algorithms of SPDNN

Recall that the training scheme for SPDNN follows the LUGS principle. The locally unsupervised (LU) phase for the SPDNN learns proper number and location of clusters in a class of input patterns. Network learning enters the GS phase after the LU training is converged. As for the globally supervised (GS) learning, the decision-based learning rule is adopted. Both training phases need several epochs to converge.

3.1. Unsupervised training for LU learning

We have developed a new self-growing cluster learning (SGCL) algorithm that is able to find appropriate number and location of clusters based on a self-growing validity measure, Bayesian information criterion (BIC) [12].

(1) *Bayesian information criterion* (*BIC*): One advantage of the mixture-model approach to the clustering scheme is that it allows the use of approximate Bayes factors to compare models. This gives a means of selecting not only the parameterization of the model, the clustering method, but also the number of clusters. The Bayes factor is the posterior odds for one model against the other assuming neither is favored a priori. This paper proposes a Bayesian factor-based iteration method to determine the appropriate number of clusters in a class of input patterns. In the followings, we will describe an approximation method to derive the Bayes factor.

Given a set of patterns $\mathbf{X}^+ = \{\mathbf{x}(t); t = 1, 2, \ldots, N\}$ and a set of candidate models $\mathscr{M} = \{M_i | i = 1, \ldots, L\}$, each model associated with a parameter set $\mathbf{w}_i$. If we want to select a proper model M_i from $\mathscr{M}$ to represent the distribution of $\mathbf{X}^+$, the posterior probability can be used to make the decision. Assuming a prior distribution $P(\mathbf{w}_i|M_i)$ for the parameters of each model M_i, the posterior probability of a model M_i is

$$\begin{aligned} P(M_i|\mathbf{X}^+) &= \frac{P(M_i)P(\mathbf{X}^+|M_i)}{P(\mathbf{X}^+)} \\ &\propto P(M_i)P(\mathbf{X}^+|M_i) \\ &\propto P(M_i)\sum_i P(\mathbf{X}^+|\mathbf{w}_i, M_i)P(\mathbf{w}_i|M_i). \end{aligned}$$

To compare two models M_m and M_l, the posterior odds between these two models are:

$$\frac{P(M_m|\mathbf{X}^+)}{P(M_l|\mathbf{X}^+)} = \frac{P(M_m)}{P(M_l)} \frac{P(\mathbf{X}^+|M_m)}{P(\mathbf{X}^+|M_l)}. \tag{4}$$

Typically the prior over models is uniform, so that $P(M_m)$ is constant. If the odds are greater than one we choose model m, otherwise we choose model l. The rightmost quantity

$$BF(\mathbf{X}^+) = \frac{P(\mathbf{X}^+|M_m)}{P(\mathbf{X}^+|M_l)} \tag{5}$$

is called *Bayes factor*, the contribution of the data toward the posterior odds. However, for applications which deal with high-dimensional data but finite number of training patterns, the training performance and storage space discourage such modeling. A natural simplification is to use a so-called Laplace approximation to the integral followed by some simplifications [26] as follows:

$$\log P(\mathbf{X}^+|M_i) = \log P(\mathbf{X}^+|\hat{\mathbf{w}}_i, M_i) - \tfrac{1}{2}\, d(M_i \log N + O(1), \tag{6}$$

where $\hat{\mathbf{w}}_i$ is a maximum likelihood estimate of $\mathbf{w}_i$, $d(M_i)$ is the number of free parameters in model M_i, and N is the number of train data. The Bayesian information criterion (BIC) for model M_i and training data $\mathbf{X}^+$ is defined as

$$BIC(M_i, \mathbf{X}^+) \equiv -2\log P(\mathbf{X}^+|\hat{\mathbf{w}}_i, M_i) - d(M_i)\log N. \tag{7}$$

Therefore, choosing the model with minimum BIC is equivalent to choosing the model with the largest (approximate) posterior probability. For model selection purposes, BIC is asymptotically consistent as a selection criterion. In other words, given a family of models, including the true model, the probability that BIC will select the correct model approaches one as the sample size $N \to \infty$ [12]. Thus, BIC can be used to compare models with differing parameterizations, differing numbers of cluster components, or both.

Table 1
Grades of evidence corresponding to values of the Bayes factor for M_2 against M_1, the BIC difference, and the posterior probability of M_2

BIC difference	Bayes factor	$Pr(M_2\|\mathbf{X}^+)(\%)$	Evidence
0–2	1–3	50–70	Weak
2–6	3–20	75–95	Positive
6–10	2–150	95–99	Strong
>10	>150	>99	Decisive

Model selection by BIC: If there are two candidate models M_1 and M_2 for modeling a data set $\mathbf{X}^+$, Raftery et al., suggest that the BIC difference ΔBIC_{21} [17,25]:

$$\Delta\mathrm{BIC}_{21}(\mathbf{X}^+) = \mathrm{BIC}(M_2, \mathbf{X}^+) - \mathrm{BIC}(M_1, \mathbf{X}^+)$$

can be used to evaluate which model is a preferred one. Table 1 depicts the grade of evidence corresponding to the values of the Bayesian difference for favoring M_2 against M_1.

(2) *BIC-based self-growing cluster learning*: There are two aspects with respect to self-growing rules:

I1 *Which cluster should be split?*
I2 *How many clusters are enough?*

On Issue I1, suppose every cluster is split temporarily to calculate its value of $\Delta\mathrm{BIC}_{21}$. The cluster with the highest value of ΔBIC_{21} also larger than a predefined threshold of *growing_confidence*, will be selected as a candidate for splitting.

On Issue I2, the splitting process terminates when none of the ΔBIC_{21} is larger than a predefined threshold, *growing_confidence*, which is a lower bound for $\Delta BIC_{21}(\mathbf{X}_i^+)$ in favor of splitting.

Detailed computing processes are depicted in the following SGCL algorithm:

Self-growing cluster learning (SGCL) algorithm

Notations:

- *Input data set*: $\mathbf{X}^+ = \{\mathbf{x}(t) : t = 1, \ldots, N\}$.
- $BIC(\mathbf{X}_l^+, GMM_i)$: The BIC value of a mixture Gaussian model (GMM_i) with i components and a sub-data set $\mathbf{X}_l^+$ in $\mathbf{X}^+$.
- $\Delta BIC_{21}(\mathbf{X}_l^+) \equiv BIC(\mathbf{X}_l^+, GMM_2) - BIC(\mathbf{X}_l^+, GMM_1)$.
- $\Theta = \{\bar{\mathbf{w}}_j, \lambda_j | j = 1, 2, \ldots, G_c\}$: Θ represents the parameters of a mixture Gaussian model, where $\bar{\mathbf{w}}_j$ is the weight of the jth Gaussian component, $\lambda_j = \{\mu_j, \Sigma_j\}$ is the mean and covariance matrix of the jth Gaussian component, G_c is the number of components in the mixture Gaussian model.
- $\tilde{\Theta}_j = \{\tilde{\mathbf{w}}_j, \tilde{\lambda}_j\}$ represents the initial parameter setting of the jth Gaussian component.
- $EM\_cluster_i$ denotes the input data $\mathbf{x}(t)$ which belongs the ith Gaussian component after EM learning [4].

BEGIN
Initialization: Set the initial number of mixture Gaussian components:
$G_c = 1$, and the initial parameter values in Θ: $\bar{\mathbf{w}}_j = 0.1$, $\mu_j = 0.0$, and $\Sigma_j = I$,
for $j = 1, 2, \ldots, G_c$.
(1) If $\Delta BIC_{21}(\mathbf{X}^+) \leqslant growing\_confidence$
  Relearn Θ by applying EM algorithm on a uni-component mixture Gaussian on $\mathbf{X}^+$;
  GOTO END;
  else
  Increment G_c;
  Relearn Θ by applying EM algorithm on a 2-component mixture Gaussian on $\mathbf{X}^+$;
(2) Clustering:
  $EM\_cluster_j = \phi$, for $j = 1, 2, \ldots, G_c$;
  for each pattern $\mathbf{x}(t)$ in $\mathbf{X}^+$
  $c = \arg\max_j \{P(\lambda_j|\mathbf{x}(t))\}$; // where λ_j is for jth Gaussian component.
  Assign $\mathbf{x}(t)$ to $EM\_cluster_c$;
(3) Grows one component:
  Let $local\Delta BIC = \max\{\Delta BIC_{21}(EM\_cluster_i)\}$, for $i = 1, 2, \ldots, G_c$;
  $whichGrow = \text{argmax}_i\{\Delta BIC_{21}(EM\_cluster_i)\}$, for $i = 1, 2, \ldots, G_c$;
  If $local\Delta BIC \leqslant growing\_confidence$ // the algorithm terminates
  GOTO END;
  else
  Initialize $\tilde{\Theta}_1$, $\tilde{\Theta}_2$ of the newly split two components from $EM\_cluster_{whichGrow}$,
  $\tilde{\Theta}_1 = \{\tilde{\mathbf{w}}_1, \tilde{\lambda}_1\}, \tilde{\Theta}_2 = \{\tilde{\mathbf{w}}_2, \tilde{\lambda}_2\}$, $\tilde{\lambda}_i = \{\tilde{\mu}_i, \tilde{\Sigma}_i\}$, for $i = 1, 2$;
  Let $\tilde{\mathbf{w}}_1 = \tilde{\mathbf{w}}_2 = \frac{1}{2}\bar{\mathbf{w}}_{whichGrow}$;
  Remove the parameter $\tilde{\Theta}_{whichGrow}$ from Θ;
  Update Θ by putting $\tilde{\Theta}_1$ and $\tilde{\Theta}_2$ into Θ;
  Increment G_c;
(4) Global EM learning:
  Using current Θ as the initial values, perform EM learning on all the clusters.
  GOTO 2.
END

3.2. Global supervised learning

During the *supervised learning* phase, training data are then used to fine tune the decision boundaries of each class. Each class is modeled by a subnet with discriminant functions, $\varphi(\mathbf{x}(t), \mathbf{w}_i), i = 1, 2, \ldots, L$. At the beginning of each supervised learning phase, use the still-under-training SPDNN to classify all the training patterns $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \mathbf{x}_i(2), \ldots, \mathbf{x}_i(M_i)\}$ for i=1, …, L. $\mathbf{x}_i(m)$ is classified to 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 $\varphi(\mathbf{x}_i(m), \mathbf{w}_i) \geqslant T_i$, where T_i is the output threshold for subnet i. According to the classification results, the training patterns 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 to ω_i (correctly classified set)$\}$;
- $D_2^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \in \omega_i, \mathbf{x}_i(m)$ is misclassified to other class ω_j (false rejection set)$\}$;
- $D_3^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \notin \omega_i, \mathbf{x}_i(m)$ is misclassified to class ω_i (false acceptance set)$\}$.

The following reinforced and antireinforced learning rules [18] are applied to the corresponding misclassified subnets.

Reinforced learning:

$$\mathbf{w}_i^{(m+1)} = \mathbf{w}_i^{(m)} + \eta \nabla \varphi(\mathbf{x}_i(m), \mathbf{w}_i). \tag{8}$$

Antireinforced learning:

$$\mathbf{w}_j^{(m+1)} = \mathbf{w}_j^{(m)} - \eta \nabla \varphi(\mathbf{x}_i(m), \mathbf{w}_j). \tag{9}$$

In (8) and (9), η is a user-defined learning rate $0 < \eta \leqslant 1$. For the data set D_2^i, reinforced and antireinforced learning will be applied to class ω_i and ω_j, respectively. As for the false acceptance set D_3^i, antireinforced learning will be applied to class ω_i, and reinforced learning will be applied to the class ω_j, where $x_i(m)$ belongs to. The gradient vectors $\nabla\varphi$ in (8) and (9) can be computed in a similar manner, as proposed in [19].

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

4. Experimental results

In this section, experimental results are presented in three parts. In the first part we use a synthetic data set to demonstrate the capability of SGCL algorithm, the second part evaluates the proposed SGCL algorithm and the SPDNN for text-independent speaker identification (ID). And the third part explores the ability of SPDNN for real-world applications on anchor/speaker identification.

4.1. Experiment 1: synthetic data set drawn from a distribution of six Gaussian clusters

This set contains 600 synthetic data points, which are evenly divided into six Gaussian distributions. Fig. 2 depicts the self-growing and *EM* learning processes and the clustering results from the initially one up to the final six clusters. Six different initialization methods are used in EM learning to illustrate the sensitiveness of initial location to the clustering performance. The six different initialization methods are briefly explained below as follows:

- *Regular EM* method: initial locations (i.e., the clustering center) are randomly selected from training data.
- *K*-means method: initial locations are determined by *k*-means clustering method.
- *Single-link* method: initial locations are calculated by *single-link* hierarchical clustering method [14].
- *Average-link* method: initial locations are computed according to *average-link* hierarchical clustering [14].

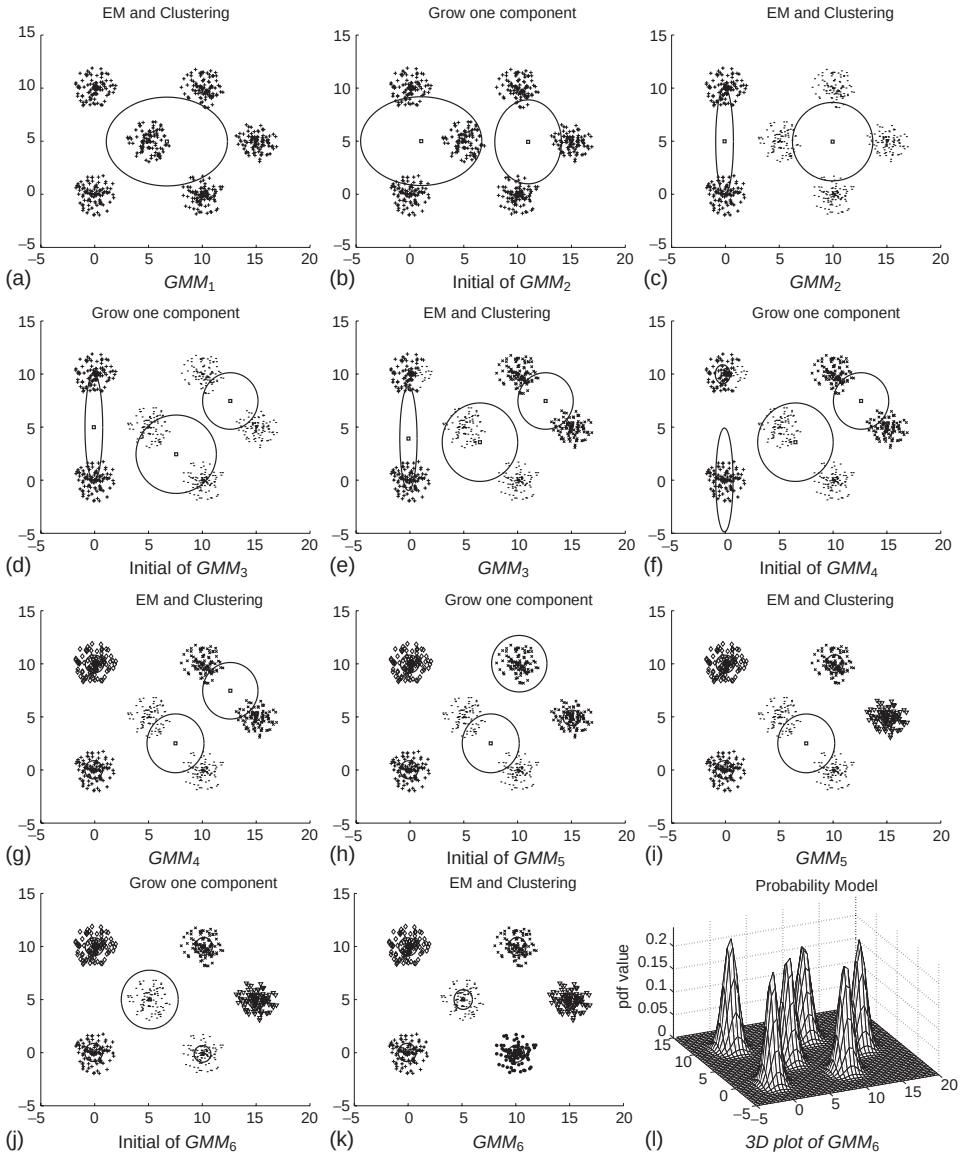

Fig. 2. The learning and splitting processes of automatic data clustering on the synthetic data set.

- *Complete-link* method: initial locations are determined by *complete-link* hierarchical clustering [14].
- *Self-growing* method: initial locations are determined according to the proposed BIC-based self-growing validity measure criterion.

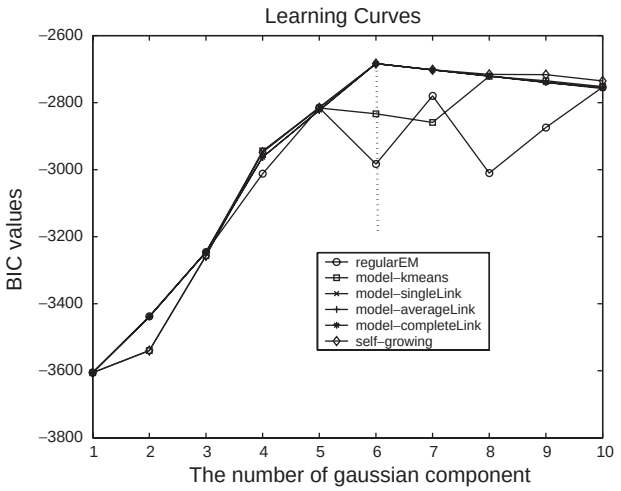

Fig. 3. Learning curves of the six different methods applied on the synthetic data. The learning curve of the Self-growing method peaks at GMM_6, it seems that the proposed SGCL method suggests a nature number of clusters for the synthetic data set.

In order to evaluate the capability of SGCL algorithm in determining the proper number of prototypes, experiments are designed to perform a self-growing process up to 10 prototypes. As shown in Fig. 3, the learning curve of the proposed *self-growing* method rises to its peak value when the number of clusters reaches 6, which is the number of prototypes in the data set.

4.2. Experiment 2: Text-independent speaker identification

This experiment demonstrates and compares the performance of SGCL on speaker identification problems with some well-known classification methods.

4.2.1. Database description

The speaker identification experiments were primarily conducted using a subset of the MAT TCC-300 speech database [28,29]. The MAT TCC-300 database is a collection of article reading speech spoken by 150 male and 150 female speakers and recorded from various microphones at high signal-to-noise ratio (SNR) environments. For each speaker, various lengths of Chinese article (approximately 475 characters in average) were read and recorded in a file.

4.2.2. Performance evaluation criterion

The evaluation of a speaker identification experiment was conducted in the following manner. The test speech was first processed to produce a sequence of feature vectors $\{x_1, x_2, \ldots, x_t\}$. To train and to evaluate different utterance lengths, the sequence of feature vectors was divided into overlapping segments of T feature vectors.

For instance, the first and second segments of a sequence would be: $\{Seg1 : x_1, x_2, \ldots, x_T\}$ and $\{Seg2 : x_2, x_3, \ldots, x_T, x_{T+1}\}$. A speech segment length of 5 s corresponds to $T = 500$ feature vectors at a 10 ms frame rate. Each feature vector is composed of a 20-dimensional mel-cepstral vector.

The performance evaluation was then computed as the percent of correctly identified segment overall test utterance segments.

$$\%\ correct\ identification = \frac{No.\ of\ correctly\ identified\ segments}{Total\ no.\ of\ segments} \times 100\%.$$

The evaluation was repeated for different values of T to evaluate performance with respect to test utterance length. Each speaker model had approximately equal amounts of testing speech, so the performance evaluation was not biased to any particular speaker.

4.2.3. Large population performance

One factor which defines the difficulty of the speaker identification task is the size of the speaker population. The following experiments examined the performance of the SPDNN speaker ID system as a function of population size consisting of 40 speakers (20 male and 20 female). Identification performance versus test utterance lengths for populations of 10, 20, 40 speakers (half male and half female) are shown in Fig. 4. It is clear that the SPDNN speaker ID system maintains high ID performance as the population size increases. The largest degradation for increasing population size is for 1-s test utterance length, but it rapidly reaches 76.9% of correctness for 5 s test utterance lengths.

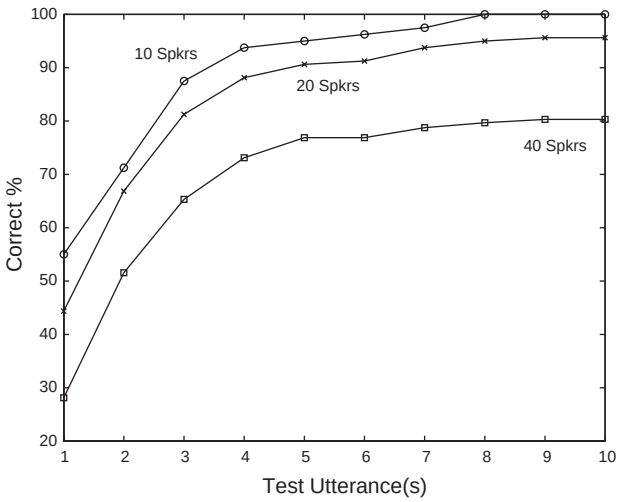

Fig. 4. Speaker identification performance versus test utterance length for population sizes of 10, 20, and 40 speakers.

Table 2
Performance results of different speaker identification methods

Classifier	10 speakers (%)	20 speakers (%)	40 speakers (%)
Feed-foward neural network	91	85	74
decision tree(CART)	87	82	76
GMM(32)	92	89	76
SGCL	93	88	76

The performance comparison of speaker identification experiments is shown in Table 2. We included experiments from some well-known classification methods, such as feed-forward neural networks [5,6] and decision trees [1]. We can see that under various speaker capacities, SGCL achieved an identification rate higher than feed-forward neural networks. We also used the decision tree method, CART, to implement speaker identifier. The performance is inferior to both types of neural network identifiers. Note that the identification result of this experiment does not imply that the SGCL speaker identifier has a performance superior to the traditional GMM approach. By properly choosing the clusters in each class, GMM may have better identification performance. However, this experimental result reveals several things. They are: (1) without any prior knowledge or experience, the SGCL method can automatically select the proper number of clusters in learning each class; (2) SGCL has a better identification performance than most well-known classification methods do.

4.3. Experiment 3: Anchor/speaker identification

The evaluation of the anchor/speaker identification experiments was conducted in the following manner. Speech data were collected from 19 female and 3 male anchor/speakers, of the evening TV news broadcasting programs in Taiwan. For each speaker, there are 180 TV news briefing of approximately 25 min sampling over 6 months. The speech data are partitioned into 5 s segments, which corresponds to 420 features vectors (*mfcc*). Each speaker was modeled by a subnet in an SPDNN. Each speaker model was trained by three different lengths of speech utterances (30, 60 and 90 s), and was tested by the rest of the speech data. Each segment of 5 second speech data was treated as a separate test utterance.

The experiments were primarily conducted to investigate the following issues:

(1) the capability of the proposed BIC-based self-growing cluster learning (SGCL) algorithm in determining the number of components in a mixture Gaussian model;
(2) the performance of the SPDNN for real-world problems, e.g., anchor/speaker identification.

Fig. 5 shows the learning curve of (BIC values) versus the number of Gaussian components used in building an SPDNN speaker model. There are several observations to be made from these results. First, the sharp increase in the BIC values from 1 to 8 mixture components, and leveling off above 16 components. The following experiments

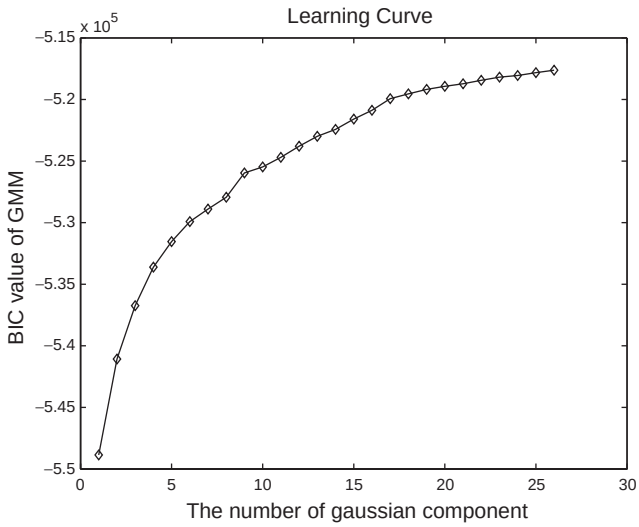

Fig. 5. The learning curve of the SPDNN for an anchor/speaker identification system. The training data of each speaker's model are a 90 s of speech. The BIC values shows a sharp increase from 1 to 8 mixture components, and leveling off above 16 components.

were performed to evaluate the identification performance according to:

(1) different lengths (30, 60, and 90 s) of training speech, and
(2) different dimensions (12, 16, 20 and 24) of *mfcc* (mel-frequency features) vectors.

As shown in Table 3, by using different lengths of training speech data and different dimensions of *mfcc* features, the identification performance of the SPDNNs with

Table 3
Anchor/speakeridentification performance for different lengths of training speech and dimension of *mfcc* feature vectors

Length of Training speech (s)	Dimension of *mfcc*			
	12	16	20	24
Identification performance by self-growing SPDNN				
30	12.32/1.45(89.51)	13.32/2.03(92.00)	14.84/2.54(94.24)	15.8/2.12(94.81)
60	17.31/2.16(90.28)	19/1.19(95.08)	20.84/2.41(96.81)	23.05/2.80(96.95)
90	20.32/2.67(93.44)	23.79/2.55(96.46)	25.94/2.97(97.78)	30.21/3.6(97.90)
Identification performance by fixed (32) component SPDNN				
90	32(92.57)	32(95.60)	32(97.39)	32(97.71)

In the body of the table, the first number is the mean value of the number of clusters, the second number after '/' is the standard deviation of the mean value, and the number in parentheses indicates the identification performance (%).

self-growing components are listed, and are compared with the performance of the SPDNNs with fixed number of components. It seems that the identification performance from self-growing SPDNN is slightly better than the fixed (32) component models.

5. Concluding remarks

In this paper, we present the SGCL algorithm for data clustering in SPDNN. The SGCL algorithm tries to tackle two long-standing critical problems in clustering, namely, (1) the difficulty in determining the number of clusters, and (2) the sensitivity to prototype initialization. The derivation of SGCL algorithm is based on a split validity criterion, Bayesian information criterion (BIC). Using SGCL for data clustering, we need to randomly initialize only one prototype in the feature space. During the learning process, according to the split validity criterion (BIC), one prototype is chosen to split into two prototypes. This splitting process terminates when the BIC values of each cluster reaches their highest points. We have conducted experiments on a variety of data types and demonstrated that the SGCL algorithm is indeed a powerful, effective, and flexible technique in finding a natural number of components for text-independent speaker identification problems. We also successfully applied SPDNN to TV news anchor/speaker identification. Since TV news speech data are highly dynamic, SGCL is able to adaptively split clusters according to the actual data sets presented. In addition, features in TV news speech are usually highly dimensional, SGCL has demonstrated its ability in dealing with such data.

Acknowledgements

The authors acknowledge Prof. S.Y. Kung, Dr. S.H. Lin for their helpful suggestions regarding the probabilistic DBNN, and statistical pattern recognition methods.

References

[1] L. Atlas, R. Cole, Y. Muthusamy, A. Lippman, J. Connor, D. Park, M. El-Sharkawai, R.J. Marks II, A performance comparison of trained multilayer perceptrons and trained classification trees, Proc. IEEE 78 (10) (1990) 1614–1619.

[2] F.L. Chung, T. Lee, Fuzzy competitive learning, Neural Networks 7 (3) (1994) 539–551.

[3] C. Dacaestecker, Competitive clustering, Proc. IEEE Int. Neural Networks Conf. 2 (1988) 833.

[4] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. 39 (1977) 1–38.

[5] K.R. Farrell, R.J. Mammone, K.T. Assaleh, Speaker recognition using neural networks and conventional classifiers, IEEE Trans. Speech Audio Process. 2 (1) (1994) 194–205.

[6] Hou Fenglei, Wang Bingxi, An integrated system for text-independent speaker recognition using binary neural network classifiers, in: Fifth International Conference on Signal Processing Proceedings, WCCC-ICSP, Vol. 2, 2000, pp. 710–713.

[7] E. Forgy, Cluster Analysis of multivariate data: efficiency vs. interpretability of classifications, Biometrics 21 (1965) 768.

[8] H. Frigui, R. Krishnapuram, A robust competitive clustering algorithm with applications in computer vision, IEEE Trans. Pattern Anal. Mach. Intell. 21 (1999) 450–465.

[9] B. Fritzke, Growing cell structures-A self-organizing network for unsupervised and supervised learning, Neural Networks 7 (1994) 1441–1460.

[10] B. Fritzke, Fast learning with incremental RNF networks, Neural Process. Lett. 1 (1) (1994) 2–5.

[11] H.C. Fu, H.Y. Chang, Y.Y. Xu, H.T. Pao, User adaptive handwriting recognition by self-growing probabilistic decision-based neural networks, IEEE Trans. Neural Networks 11 (6) (2000) 1373–1384.

[12] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Data Mining, Inference, and Prediction, Springer, New York, 2001, pp. 206–208.

[13] J. Hertz, A. Krogh, R.G. Palmer, Introduction to the Theory of Neural Computation, Addison-Wesley, New York, 1991.

[14] S.C. Johnson, Hierarchical clustering schemes, Psychometrika 2 (1967) 241–254.

[15] J.M. Jolion, P. Meer, S. Bataouche, Robust clustering with applications in computer vision, IEEE Trans. Pattern Anal. Mach. Intell. 13 (1991) 791–802.

[16] M.I. Jordan, R.A. Jacobs, Hierarchical mixture of experts and the EM Algorithm, Neural Comput. 6 (1994) 181–214.

[17] R.E. Kass, Bayes factors, J. Amer. Statist. Assoc. 90 (1995) 773–795.

[18] S.Y. Kung, J.S. Taur, Decision-based hierarchical neural networks with signal/image classification applications, IEEE Trans. Neural Networks 6 (1) (1995) 170–181.

[19] Shang-Hung Lin, S.Y. Kung, L.J. Lin, Face recognition/detection by probabilistic decision-based neural networks, IEEE Trans. Neural Networks, special issue on Artifi. Neural Network Pattern Recog. 8 (1) (1997) 114–132.

[20] Y. Linde, A. Buzo, R.M. Gray, An algorithm for vector quantizer design, IEEE Trans. Commun. 28 (1980) 84–85.

[21] Z.-Q. Liu, M. Glickman, Y.-J. Zhang, Soft-competitive learning paradigms, in: Z.-Q. Liu, S. Miyamoto (Eds.), Soft Computing and Human-Centered Machines, Springer, New York, 2000, pp. 131–161.

[22] S.P. Lloyd, Least squares quantization in PCM, IEEE Trans. Inform. Theory 28 (1982) 129–137.

[23] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the fifth Berkeley Symposium Mathematical and Statistical Probability, University of California Press, Berkeley, 1967, pp. 281–297.

[24] T.M. Martinetz, S.G. Berkovich, K.J. Schulten, Neural-gas network for vector quantization and its application to time-series prediction, IEEE Trans. Neural Networks 4 (1993) 558–568.

[25] A.E. Raftery, Bayesian model selection in social research in: Sociological Methodology, Blackwell, Oxford, 1995, pp. 111–196.

[26] B.D. Ripley, Pattern Recognition and Neural Networks, Cambridge University Press, Cambridge, 1996.

[27] D.E. Rumelhart, D. Zipser, Feature discovery by competitive learning, Cognitive Sci. 9 (1985) 75–112.

[28] TCC-300 speech database.. Association for Computational Linguistics and Chinese Language Processing, Institute of Information Science, Academia Sinica, Nangkang, Taipei, ROC. [Online]. Available: http://rocling.iis.sinica.edu.tw/ROCLING/MAT/TCC-300brief.htm.

[29] Hsiao-Chuan Wang, Speech corpora and ASR assessment in Taiwan, in: Proceedings of Oriental COCOSDA Workshop Beijing, China, 2000.

[30] L. Xu, A. Krzyzak, E. Oja, Rival penalized competitive learning for clustering analysis, RBF Net, and curve detection, IEEE Trans. Neural Networks 4 (1993) 636–649.

[31] E. Yair, K. Zeger, A. Gersho, Competitive learning and soft competition for vector quantizer design, IEEE Trans. Signal Processing 40 (1992) 294–309.

[32] Ya-Jun Zhang, Zhi-Qiang Liu, Self-splitting competitive learning: a new on-line clustering paradigm, IEEE Trans. Neural Networks 13 (2) (2002) 369–380.

Cheng-Lung Tseng received his B.S. degree in Computer Science and Information Engineering from National Chiao-Tung University, Taiwan, R.O.C., in 1999 and his M.S. degree in Computer Science from National Tsing-Hua University, Taiwan, R.O.C., in 2001. He is currently pursuing a Ph.D. degree with the Department of Computer Science and Information Engineering, National Chiao-Tung University. His main research interests include speech processing, neural networks, and statistical learning theory.

Yueh-Hong Chen received his B.S. degree and M.S. degree in science education from National Tainan Teachers' College, Taiwan, R.O.C., in 1998 and 2000, respectively. He is currently pursuing a Ph.D. degree with the Department of Computer Science and Information Engineering, National Chiao-Tung University. His main research interests include digital watermarking, cryptography, neural networks, and statistical learning theory.

Yeong-Yuh Xu received his B.S. degree in Electrical Engineering from National Sun Yat-sen University, Taiwan, R.O.C., in 1995 and his M.S. degree in Computer Science from National Chiao-Tung University, Taiwan, R.O.C., in 1997. He is currently pursuing a Ph.D. degree with the Department of Computer Science and Information Engineering, National Chiao-Tung University. His main research interests include OCR, Image processing, neural networks, and statistical learning theory.

H.T. Pao received her B.S. degree from National Cheng-kung University, Taiwan, R.O.C., in mathematics in 1976, and her 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 Telcommunication 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, ROC. Her research interests include Statistics and neural networks.

Hsin-Chia Fu (M'78) received his 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, Indiana. Since 1983, he has been on the faculty of the Department of Computer science 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 at National Chiao-Tung University. From September to December of 1994, he was a visiting scientist at Fraunhofer-Institute 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: Sun-Kung Book Co., 1986), and "Introduction to neural networks" (Taipei, Taiwan: Third Wave Publishing Co., 1994). Dr. Fu was the co-recipient of the 1992 and 1993 Long-Term Best Thesis Award with Dr. Koun-Tem Sun and Dr. Cheng-Chin Chiang, respectively, 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 International Symposium on Artificial Neural Networks, and served the Technical Committee on Neural Networks for Signal Processing of the IEEE Signal Processing Society from 1998 to 2000. He is a member of the IEEE Signal Processing and Computer Societies, Phi Tau Phi, and the Eta Kappa Nu Electrical Engineering Honor Society.