EURASIP Journal on Applied Signal Processing 2004:17, 2626–2639
© 2004 Hindawi Publishing Corporation

# A Model-Selection-Based Self-Splitting Gaussian Mixture Learning with Application to Speaker Identification

**Shih-Sian Cheng**
*Institute of Information Science, Academia Sinica, Taipei 115, Taiwan*
*Email: sscheng@iis.sinica.edu.tw*
*Department of Computer Science and Information Engineering, National Chiao-Tung University, Hsinchu 300, Taiwan*

**Hsin-Min Wang**
*Institute of Information Science, Academia Sinica, Taipei 115, Taiwan*
*Email: whm@iis.sinica.edu.tw*

**Hsin-Chia Fu**
*Department of Computer Science and Information Engineering, National Chiao-Tung University, Hsinchu 300, Taiwan*
*Email: hcfu@csie.nctu.edu.tw*

*Received 3 December 2003; Revised 2 July 2004; Recommended for Publication by Kenneth Barner*

We propose a self-splitting Gaussian mixture learning (SGML) algorithm for Gaussian mixture modelling. The SGML algorithm is deterministic and is able to find an appropriate number of components of the Gaussian mixture model (GMM) based on a self-splitting validity measure, Bayesian information criterion (BIC). It starts with a single component in the feature space and splits adaptively during the learning process until the most appropriate number of components is found. The SGML algorithm also performs well in learning the GMM with a given component number. In our experiments on clustering of a synthetic data set and the text-independent speaker identification task, we have observed the ability of the SGML for model-based clustering and automatically determining the model complexity of the speaker GMMs for speaker identification.

**Keywords and phrases:** unsupervised learning, Gaussian mixture modelling, Bayesian information criterion, speaker identification.

## 1. INTRODUCTION

In many applications, data clustering techniques have been applied to discover and extract the hidden structure in a data set and, thus, the structural relationships between individual data points can be detected. Data clustering is also known as unsupervised learning [1]. A variety of competitive learning (CL) schemes has been developed, distinguishing in their approaches to competition and learning rules. The simplest and most prototypical CL algorithms are mainly based on the winner-take-all (WTA) [2] (or hard CL) paradigm, where adaptation is restricted to the single prototype that best matches the input patterns. Different algorithms in this paradigm such as LBG (or generalized Lloyd) [3, 4, 5] and $K$-means [6] have been well recognized.

In statistical pattern recognition, finite mixture models (usually the Gaussian mixture models (GMMs)) provide a formal approach to clustering [7, 8], namely the probabilistic model-based clustering. At the end of Gaussian mixture modelling process, all the training patterns could be grouped into clusters by distributing each pattern to the mixture component which is most likely to generate it. The most common approach for learning GMMs is the expectation maximization (EM) algorithm [9, 10, 11]. However, there are several issues related to EM. (1) EM is a local optimization algorithm that is highly sensitive to the initialization of parameters. (2) The full covariance matrices of the Gaussian mixture components could be singular. (3) It is still an open problem about how many components are enough to fit the probabilistic distribution of the training patterns. The component number is usually decided empirically.

There is no theoretical method for selecting the initial values of the parameters. A common way is to apply clustering methods (e.g., $K$-means clustering or hierarchical clustering [6, 7]) to locate the initial means of Gaussian components for the EM algorithm and, thus, the performance of

EM highly depends on the clustering result. Instead of finding better initial values for the parameters, Ueda et al. [12] proposed an SMEM (split and merge EM) algorithm that performs split and merge operations simultaneously (i.e., the number of components is kept, some components split while some components merge) after the regular EM training to escape the condition that there are too many components in some clusters and too few to well fit the other clusters. However, the SMEM algorithm only makes improvements to the estimation of the parameters with a given component number but does not contribute to the selection of the number of components.

EM breaks down when any of the covariance matrices of the Gaussian components becomes singular or nearly singular. In general, the singularity condition occurs especially when clusters contain a few observations or when too many components are used to fit the data set where there are actually fewer clusters. Ormoneit and Tresp [13] proposed some Bayesian regularization methods to deal with the singularity condition.

The selection of the number of components is an important issue in Gaussian mixture modelling. With too many components, the mixture model would overfit the data; on the other hand, with too few components, it would be not flexible enough to describe the structure of the data. Several model selection criteria have been proposed to select the number of components of a mixture model among a set of candidate models, such as Akaike's information criterion (AIC) [14], Bayesian information criterion (BIC) [15, 16, 17], minimum description length (MDL) [18] (which is formally identical to BIC), and minimum message length (MML) [19]. These criteria are also well known as penalized likelihood criteria. In the previous studies, the model-selection-based approaches first defined the upper bound $k_{\max}$ of the component number of the candidate models, and then the best model was the one that has the minimum value of the specific model selection criterion (MSC) among these candidate models; that is,

$$\widehat{k} = \arg\min_{k} \{ \operatorname{MSC}(\widehat{\mathbf{w}}_k) \}, \quad k = 1, 2, \dots, k_{\max}, \tag{1}$$

where $\widehat{\mathbf{w}}_k$ is the maximum likelihood estimate of parameters of a mixture model that has $k$ components. $\operatorname{MSC}(\widehat{\mathbf{w}}_k) = -\log p(\mathbf{X}|\widehat{\mathbf{w}}_k) + \operatorname{Penalty}(k)$, where $\log p(\mathbf{X}|\widehat{\mathbf{w}}_k)$ is the log-likelihood value of the training data set $\mathbf{X}$ for $\widehat{\mathbf{w}}_k$, and $\operatorname{Penalty}(k)$ is a monotonically increasing function of $k$ that penalizes more for more complicated models.

There are two major issues with these model-selection-based approaches. First, they need to define the upper bound of the component number $k_{\max}$ beforehand. If $k_{\max}$ is too large (much larger than the best component number determined by the MSC), the training process will require a high computation cost. On the other hand, if $k_{\max}$ is too small, the selected model may not be flexible enough to describe the structure of the training data. Second, $\operatorname{MSC}(\widehat{\mathbf{w}}_k)$ is calculated from $\log p(\mathbf{X}|\widehat{\mathbf{w}}_k)$, in which $\widehat{\mathbf{w}}_k$ is usually estimated by EM. In order to obtain a more robust calculation of $\operatorname{MSC}(\widehat{\mathbf{w}}_k)$, one may use the SMEM or the other learning algorithms to obtain a better estimation of $\widehat{\mathbf{w}}_k$, but the training process may require a higher computation cost and the problem of defining $k_{\max}$ still remains unsolved.

In this paper, we propose a self-splitting Gaussian mixture learning (SGML) algorithm that starts with a single component and successively splits the selected component into two new components until the most appropriate component number is found. Both the selection of component to be split and the determination of appropriate component number are based on BIC. The proposed approach has several advantages. (1) It provides a better initialization for EM. In the conventional Gaussian mixture learning (GML) process, the clustering phase often results in ill-initial mixture models with too many components in one part of the space and too few in another widely separated part of the space. The following EM phase often fails to escape from this configuration and falls into an ill-local maximum. In the splitting process of the proposed SGML, the BIC-MSC is used to determine which part of the space should be divided into two parts. In this way, the ill-initialization situation can be avoided to some extent and, thus, a better estimation of $\log p(\mathbf{X}|\widehat{\mathbf{w}}_k)$ can be obtained. (2) It automatically determines the appropriate component number without the need to define the upper bound of the component number beforehand. The SGML algorithm stops when a "significant maximum" in the learning curve (i.e., the BIC plot here) is found, and then outputs the model yielding the maximum BIC valve. The term "significant maximum" will be defined in Section 3.1. (3) It is deterministic. The output of the SGML algorithm is always the same in different runs on the same data set, since there is no randomization in the learning rules.

We have evaluated the proposed learning algorithm on two tasks. The first task involves applying the SGML algorithm in automatic clustering of a synthetic data set. The second task involves using the SGML algorithm to train speaker GMMs for the NIST 2001 speaker recognition evaluation. In both tasks, we have observed a noticeable improvement in Gaussian mixture modelling.

The rest of this paper is organized as follows. The EM algorithm, model selection, BIC, and the use of GMMs for speaker identification are reviewed in Section 2. Then, the proposed SGML algorithm is described in detail in Section 3, and the experimental results are discussed in Section 4. Finally, conclusions are drawn in Section 5.

## 2. REVIEWS

### *2.1. Gaussian mixture distribution and EM algorithm*

A random vector $\mathbf{x} \in R^d$ is said to follow an $R$-component Gaussian mixture distribution if its probability density function (pdf) is

$$p(\mathbf{x}|\mathbf{w}) = \sum_{r=1}^{R} p(\Theta_r)\, p(\mathbf{x}|\Theta_r), \tag{2}$$

where $p(\Theta_r) \geq 0$, $r = 1, 2, \dots, R$, and $\sum_{r=1}^{R} p(\Theta_r) = 1$. $p(\mathbf{x}|\Theta_r)$ is the $r$th Gaussian component of $p(\mathbf{x}|\mathbf{w})$, $\Theta_r = \{\mu_r, \Sigma_r\}$. $p(\Theta_r)$, $\mu_r$, and $\Sigma_r$ denote, respectively, the prior probability, mean vector, and covariance matrix of $p(\mathbf{x}|\Theta_r)$. $\mathbf{w} \equiv \{p(\Theta_r), \Theta_r \mid r = 1, 2, \dots, R\}$ is the complete set of parameters needed to specify the Gaussian mixture distribution.

Given a set of independent samples $\mathbf{X} = \{\mathbf{x}(t);\ t = 1, 2, \dots, N\}$ from a Gaussian mixture distribution, the estimation of the parameters is often carried out with the EM algorithm [9, 10, 11] in the maximum likelihood sense. The log-likelihood function of $\mathbf{X}$ under $\mathbf{w}$ can be computed by

$$\mathcal{L}(\mathbf{X}|\mathbf{w}) = \sum_{t=1}^{N} \log p(\mathbf{x}(t) \mid \mathbf{w}). \tag{3}$$

The EM algorithm iteratively reestimates the parameters of $\mathbf{w}$ through the following closed-form solutions and guarantees that the value of $\mathcal{L}(\mathbf{X}|\mathbf{w})$ will monotonically increase after each iteration:

$$\begin{aligned} p\left(\Theta_r^{(j+1)}\right) &= \frac{1}{N} \sum_{t=1}^{N} p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right), \\ \mu_r^{(j+1)} &= \frac{\sum_{t=1}^{N} p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right) \mathbf{x}(t)}{\sum_{t=1}^{N} p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right)}, \\ \Sigma_r^{(j+1)} &= \frac{\sum_{t=1}^{N} p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right) \left(\mathbf{x}(t) - \mu_r^{(j+1)}\right) \left(\mathbf{x}(t) - \mu_r^{(j+1)}\right)^T}{\sum_{t=1}^{N} p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right)}, \end{aligned} \tag{4}$$

where $p(\Theta_r^{(j)}|\mathbf{x}(t))$ is the posterior probability of $\Theta_r^{(j)}$ given $\mathbf{x}(t)$, which can be computed by

$$p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right) = \frac{p\left(\Theta_r^{(j)}\right) p\left(\mathbf{x}(t) \middle| \Theta_r^{(j)}\right)}{p(\mathbf{x}(t))}. \tag{5}$$

The following reestimation rule based on the Bayesian regularization can be used to avoid the singularity condition of a covariance matrix [13]

$$\Sigma_r^{(j+1)} = \frac{\sum_{t=1}^{N} p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right) \left(\mathbf{x}(t) - \mu_r^{(j+1)}\right) \left(\mathbf{x}(t) - \mu_r^{(j+1)}\right)^T + \lambda I_d}{\sum_{t=1}^{N} p\left(\Theta_r^{(j)} \middle| \mathbf{x}(t)\right) + 1}, \tag{6}$$

where $I_d$ is a $d$-dimensional identity matrix and $\lambda$ is a regularization constant determined by some validation data.

### 2.2. The Bayesian approach for model selection and BIC

Given a set of patterns $\mathbf{X} = \{\mathbf{x}(t);\ t = 1, 2, \dots, N\}$ and a set of candidate models $\mathcal{M} = \{M_k \mid k = 1, \dots, L\}$, each model associated with a parameter set $\mathbf{w}_k$ and the posterior probability $p(M_k|\mathbf{X})$ can be used to select a proper model from $\mathcal{M}$ to represent the distribution of $\mathbf{X}$. By applying Bayes' theorem, $p(M_k|\mathbf{X})$ can be expressed as follows:

$$p(M_k \mid \mathbf{X}) = \frac{p(M_k) p(\mathbf{X} \mid M_k)}{p(\mathbf{X})}, \tag{7}$$

where $p(M_k)$ is the prior probability of model $M_k$. $p(\mathbf{X})$ can be eliminated because it is identical for all models and will not affect the model selection. Furthermore, because the way to estimate the probability $p(M_k)$ is still unknown, we may simply assume each model is equally likely (i.e., $p(M_k) = 1/L$). As a result, $p(M_k|\mathbf{X})$ is proportional to the probability that the data conform to the model $M_k$, $p(\mathbf{X}|M_k)$, which can be computed by

$$p(\mathbf{X} \mid M_k) = \int p(\mathbf{X} \mid \mathbf{w}_k, M_k) p(\mathbf{w}_k \mid M_k) d\mathbf{w}_k. \tag{8}$$

The calculation of $\log p(\mathbf{X}|M_k)$ can be achieved by the Laplace approximation [16, 20], which gives

$$\log p(\mathbf{X} \mid M_k) \approx \log p(\mathbf{X} \mid \hat{\mathbf{w}}_k, M_k) - \frac{1}{2} \cdot d(M_k) \cdot \log N, \tag{9}$$

where $\hat{\mathbf{w}}_k$ is the maximum likelihood estimate of $\mathbf{w}_k$, $d(M_k)$ is the number of free parameters in model $M_k$, and $N$ is the number of training patterns. In [17], the BIC value of model $M_k$ over the data set $\mathbf{X}$, $\mathrm{BIC}(M_k, \mathbf{X})$, is defined as follows:[1]

$$\mathrm{BIC}\,(M_k, \mathbf{X}) \equiv 2 \log p(\mathbf{X} \mid \hat{\mathbf{w}}_k, M_k) - d(M_k) \cdot \log N. \tag{10}$$

Accordingly, the larger the value of BIC, the stronger the evidence for the model is. In other words, the model with the maximum BIC value will be selected. The BIC can be used to compare models with different parameterizations, different numbers of components, or both.

### 2.3. GMM-based text-independent speaker identification

In a GMM-based speaker identification system [22, 23], a group of speakers $\{S_1, S_2, \dots, S_M\}$ are represented by a set of GMMs $\{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_M\}$. Given a feature vector sequence $\mathbf{X} = \{\mathbf{x}(t);\ t = 1, 2, \dots, N\}$, the objective is to find the speaker model which has the maximum log-likelihood value with $\mathbf{X}$. The problem can be formally formulated as follows:

$$\hat{S} = \arg\max_k \log p(\mathbf{X} \mid \mathbf{w}_k). \tag{11}$$

If the individual observations are assumed to be independent, $p(\mathbf{X}|\mathbf{w}_k)$ can be decomposed as a product of$p(\mathbf{x}(t)|\mathbf{w}_k)$,

[1]Kass and Raftery [21] defined BIC as minus the value given in (10); Fraley and Raftery [17] used the definition of (10) to make it easier to interpret the plots of BIC values. We follow what Fraley and Raftery did and, thus, the model with the maximum BIC value in fact corresponds to the model with the minimum MSC in (1).

```
Input data set: $\mathbf{X} = \{\mathbf{x}(t);\ t = 1, \ldots, N\}$
Output parameter set: $\mathbf{w} = \{p(\Theta_r), \Theta_r \mid r = 1, 2, \ldots, \text{bestNum}\}$, where $p(\Theta_r)$ is the prior probability
of the $r$th Gaussian component, and $\Theta_r = \{\mu_r, \Sigma_r\}$ are the mean vector and covariance matrix of the
$r$th Gaussian component.
BEGIN
 (1) Initialization:
        $S$Range = 5;
        compoNum = 1;
        $\mathbf{w} = \{p(\Theta_1), \Theta_1 = \{\mu_1, \Sigma_1\}\}$, where $p(\Theta_1) = 1$, $\mu_1$,
        and $\Sigma_1$ are the sample mean vector and sample
        covariance matrix of $\mathbf{X}$;
        BIC_set(1) = BIC(GMM$_1$, $\mathbf{X}$); GMM_set(1) = $\mathbf{w}$;
        //BIC_set(compoNum): the BIC value of GMM$_{\text{compoNum}}$ over $\mathbf{X}$.
        //GMM_set(compoNum): the parameter set of GMM$_{\text{compoNum}}$ over $\mathbf{X}$.
 (2) Data clustering:
        EM_cluter$_i$ = $\phi$, for $i = 1, 2, \ldots$, compoNum;
        for each pattern $\mathbf{x}(t)$:
            $j = \arg\max_r \{p(\Theta_r|\mathbf{x}(t))\}$; EM_cluster$_j$ = EM_cluster$_j \bigcup \mathbf{x}(t)$; //add $\mathbf{x}(t)$ to
            EM_cluster$_j$.
 (3) Splitting (splitting one component into two new components):
        whichSplit = $\arg\max_i \{\Delta\text{BIC}_{21}(\text{EM\_cluster}_i)\}$;
        suppose the parameters of GMM$_2$ corresponding to EM_cluster$_{\text{whichSplit}}$ are
            $\bar{\lambda}_1 = \{p(\bar{\Theta}_1), \bar{\Theta}_1\}, \bar{\lambda}_2 = \{p(\bar{\Theta}_2), \bar{\Theta}_2\}$, where
            $\bar{\Theta}_r = \{\bar{\mu}_r, \bar{\Sigma}_r\}$, for $r = 1, 2$.
     Let
        $p(\bar{\Theta}_1) = p(\bar{\Theta}_2) = \frac{1}{2} p(\Theta_{\text{whichSplit}})$;
        $\mathbf{w} = \mathbf{w} \setminus \{p(\Theta_{\text{whichSplit}}), \Theta_{\text{whichSplit}}\}$;//remove $\{p(\Theta_{\text{whichSplit}}), \Theta_{\text{whichSplit}}\}$ from $\mathbf{w}$,
        $\mathbf{w} = \mathbf{w} \bigcup \{\bar{\lambda}_1, \bar{\lambda}_2\}$; //add $\{\bar{\lambda}_1, \bar{\lambda}_2\}$ to $\mathbf{w}$;
        compoNum = compoNum + 1;
 (4) Global EM learning:
        perform EM learning on all the clusters with $\mathbf{w}$ as the initial parameters,
        then $\mathbf{w}$ would be fine-tuned;
        GMM_set(compoNum) = $\mathbf{w}$;
        BIC_set(compoNum) = BIC($\mathbf{w}$, $\mathbf{X}$);
        if compoNum > $S$Range and BIC_set(compoNum-$S$Range) is the maximum value
        in the learning curve,
            bestNum = compoNum − $S$Range; $\mathbf{w}$ = GMM_set(bestNum); goto END;
        else
            goto 2;
END
```

Algorithm 1: The SGML algorithm.

$t = 1, 2, \ldots, N$, and (11) can be rewritten as follows:

$$\hat{S} = \arg\max_k \sum_{t=1}^{N} \log p\big(\mathbf{x}(t) \,|\, \mathbf{w}_k\big). \qquad (12)$$

The observation-independence assumption has been widely used in speaker recognition as well as speech recognition systems.

## 3. THE BIC-BASED SGML

### 3.1. The SGML

Given a set of training patterns $\mathbf{X} = \{\mathbf{x}(t);\ t = 1, 2, \ldots, N\}$, we can partition $\mathbf{X}$ into $k$ clusters after applying the EM algorithm to learn the $k$-component GMM$_k$. Here, we call the cluster defined by EM the EM_cluster. The self-splitting algorithm starts with a single component (i.e., single EM_cluster) in the feature space (i.e., $\mathbf{X}$) and splits the selected component adaptively during the learning process until the most appropriate number of components (EM_clusters) is found. The details of the proposed learning algorithm are illustrated in Algorithm 1. There are three main aspects with respect to our self-splitting learning rules.

(I1) How to group the training patterns into clusters?

(I2) Which component should be split into a pair of new components?

(I3) How many components are enough for representing the distribution of the training patterns?

*On issue (I1)*

After EM, each $\mathbf{x}(t) \in \mathbf{X}$ is distributed to the EM_cluster whose Gaussian component has the largest posterior probability given $\mathbf{x}(t)$; that is, let $\mathbf{x}(t)$ belong to EM_cluster$_j$ if $j = \arg\max_r p(\Theta_r | \mathbf{x}(t))$.

*On issue (I2)*

For each EM_cluster, we need to calculate the value of $\Delta\,\mathrm{BIC}_{21}$(EM_cluster) ($\Delta\,\mathrm{BIC}_{21}$(EM_cluster) = BIC(GMM$_2$, EM_cluster)−BIC(GMM$_1$, EM_cluster)). GMM$_1$ models the EM_cluster with a single Gaussian component while GMM$_2$ models the EM_cluster with a mixture of two Gaussian components. The mean vector and covariance matrix of GMM$_1$ can be optimally estimated in an analytic way (respectively, the sample mean vector and sample covariance matrix of EM_cluster) while those of GMM$_2$ must be estimated by the EM algorithm. We first use $K$-means clustering (here, $K = 2$) to find two initial mean vectors of GMM$_2$, and then apply EM learning to fine-tune the parameters. Suppose $\mu$ is the mean vector of GMM$_1$, we use $\mu - \varepsilon$ and $\mu + \varepsilon$ as the initial centroids of the $K$-means clustering, where $\varepsilon$ is a constant vector. The component which corresponds to the EM_cluster that has the largest value of $\Delta\,\mathrm{BIC}_{21}$ is split into two new components. According to the definition of BIC, the larger the $\Delta\,\mathrm{BIC}_{21}$ is, the higher the confidence that GMM$_2$ fits the corresponding EM_cluster better than GMM$_1$ does.

*On issue (I3)*

After the $j$th split, the number of components grows to $j+1$ and these Gaussian components are used as the initial values of EM to train the GMM$_{j+1}$ over $\mathbf{X}$. The value of BIC(GMM$_{j+1}$, $\mathbf{X}$) is estimated at the end of EM training. The BIC values of GMMs with an increasing number of mixture components form a learning curve (i.e., the BIC plot). The learning process stops when a "significant maximum" in the learning curve is found. Given that the SGML has generated $\{\mathrm{GMM}_1, \mathrm{GMM}_2, \ldots, \mathrm{GMM}_M\}$, if BIC(GMM$_{M-S\mathrm{Range}}$, $\mathbf{X}$) is the maximum among the BIC values from BIC(GMM$_1$, $\mathbf{X}$) to BIC(GMM$_M$, $\mathbf{X}$), the "significant maximum" in the learning curve is found and the SGML will stop and output the model corresponding to the "significant maximum," GMM$_{M-S\mathrm{Range}}$; otherwise, the SGML will continue to train GMM$_{M+1}$. Here, $S$Range is a positive integer, and around 5 is a reasonable choice according to our experience.

#### 3.1.1. Complexity analysis of SGML

To simplify the analysis, we assume that both $K$-means and EM stop when a predefined iteration number is reached. Let $T_{\mathrm{KM}}(k, D)$ denote the computation needed to apply the $K$-means clustering algorithm in the data set $D$ for initializing the parameters of a GMM$_k$ and let $T_{\mathrm{EM}}(\mathrm{GMM}_k, D)$ denote the computation needed to use EM for learning GMM$_k$ from $D$ given an initial model parameter set. The computation needed to train GMM$_k$ from $\mathbf{X}$ using the conventional learning process; that is, $K$-means followed by EM, is $T_{\mathrm{KM}}(k, \mathbf{X}) + T_{\mathrm{EM}}(\mathrm{GMM}_k, \mathbf{X})$ (random mean selection for initialization of $K$-means) or $\sum_{i=1}^{k} T_{\mathrm{KM}}(i, \mathbf{X}) + T_{\mathrm{EM}}(\mathrm{GMM}_k, \mathbf{X})$ (incremental mean splitting for initialization of $K$-means). Therefore, the total amount of computation required to select the best model by incrementing the Gaussian component number is roughly $\sum_{k=1}^{k_{\max}} [T_{\mathrm{KM}}(k, \mathbf{X}) + T_{\mathrm{EM}}(\mathrm{GMM}_k, \mathbf{X})]$. Young and Woodland [24] proposed to successively increment the mixture component from a single one. Their method split the mean vector of the Gaussian component with the largest weight into two new ones in each splitting step, and then performed EM to update all Gaussian components. Since there is no $K$-means involved in their method, the total computation is $\sum_{k=1}^{k_{\max}} T_{\mathrm{EM}}(\mathrm{GMM}_k, \mathbf{X})$.

For the proposed SGML, as shown in Algorithm 1, the computation needed in the *splitting* step for incrementing GMM$_R$ to GMM$_{R+1}$ is the computation needed to compute the sample mean vectors and sample covariance matrices for all clusters, $\sum_{j=1}^{R} T_{\mathrm{EM}}(\mathrm{GMM}_1, \mathrm{EM\_cluster}_j)$, plus the computation needed to apply $K$-means ($K = 2$) followed by EM to train GMM$_2$ for all clusters, $\sum_{j=1}^{R} [T_{\mathrm{KM}}(2, \mathrm{EM\_cluster}_j) + T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathrm{EM\_cluster}_j)]$, while the computation needed in the *global EM learning* step is $T_{\mathrm{EM}}(\mathrm{GMM}_{R+1}, \mathbf{X})$. $\sum_{j=1}^{R} T_{\mathrm{EM}}(\mathrm{GMM}_1, \mathrm{EM\_cluster}_j)$ is almost equal to $T_{\mathrm{EM}}(\mathrm{GMM}_1, \mathbf{X})$. From (4), it is obvious that $T_{\mathrm{EM}}(\mathrm{GMM}_k, \mathbf{X})$ is proportional to the component number $k$ and the amount of the training data $\mathbf{X}$. $T_{\mathrm{KM}}(k, \mathbf{X})$ has the same property too. Therefore, $\sum_{j=1}^{R} T_{\mathrm{KM}}(2, \mathrm{EM\_cluster}_j)$ can be approximated by $T_{\mathrm{KM}}(2, \mathbf{X})$, while $\sum_{j=1}^{R} T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathrm{EM\_cluster}_j)$ can be approximated by $T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathbf{X})$. In comparison with the method proposed by Young and Woodland [24], the SGML has an overhead of $T_{\mathrm{EM}}(\mathrm{GMM}_1, \mathbf{X}) + T_{\mathrm{KM}}(2, \mathbf{X}) + T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathbf{X})$ in each splitting step. $T_{\mathrm{EM}}(\mathrm{GMM}_1, \mathbf{X})$ is much smaller than $(1/2)T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathbf{X})$ because no likelihood calculation is involved in estimating GMM$_1$. Furthermore, $T_{\mathrm{KM}}(2, \mathbf{X})$ is much smaller than $T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathbf{X})$ too. Therefore, the overhead is just slightly higher than $T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathbf{X})$. In other words, when $R \gg 2$, the overhead for incrementing GMM$_R$ to GMM$_{R+1}$ is negligible since $T_{\mathrm{EM}}(\mathrm{GMM}_2, \mathbf{X})$ is much smaller than $T_{\mathrm{EM}}(\mathrm{GMM}_{R+1}, \mathbf{X})$.

### 3.2. The fast SGML

The computation requirement becomes an important issue when the training set is very large. A fast learning procedure is therefore desirable. Since we can validate whether GMM$_2$ fits a cluster better than GMM$_1$ does via the BIC, a straightforward idea is to split all the clusters whose $\Delta\mathrm{BIC}_{21}$ values are larger than a splitting confidence; that is, the confidence threshold for splitting, in each splitting step. The learning process stops when the $\Delta\mathrm{BIC}_{21}$ of all clusters are less than the splitting confidence. We call this approach the fast SGML. The fast SGML needs a much lower computation requirement than the SGML because it allows multiple splits in the splitting step. The fast SGML resembles the LBG algorithm [5], but a crucial difference is that the LBG algorithm successively splits each of the clusters into two clusters until a given cluster number is reached and no validation measure is applied while splitting. To avoid the

```
BEGIN
 (1) Initialization: same as the SGML.
 (2) Data clustering: same as the SGML.
 (3) Splitting components:
       (a) for each EM _cluster$_i$:
               if $\Delta$BIC$_{21}$(EM _cluster$_i$) > splitting confidence,
                   split the component corresponding to EM _cluster$_i$ into two components;
                   compoNum = compoNum + 1;
       (b) if no component is split in (a),
               return GMM _set(compoNum); goto END.
 (4) Global EM learning:
         perform EM learning to fine-tune the Gaussian components obtained at Step 3;
         if the learning curve starts to go down
             let bestNum be the component number which has the maximum value
             in the learning curve;
             return GMM _set(bestNum); goto END;
         else
             goto 2;
END
```

Algorithm 2: The fast SGML algorithm.

overfitting condition caused by a too small splitting confidence, the fast learning procedure stops not only when the $\Delta\text{BIC}_{21}$ of all clusters are less than the splitting confidence but also when the learning curve starts to go down. The details of the fast learning algorithm are illustrated in Algorithm 2.

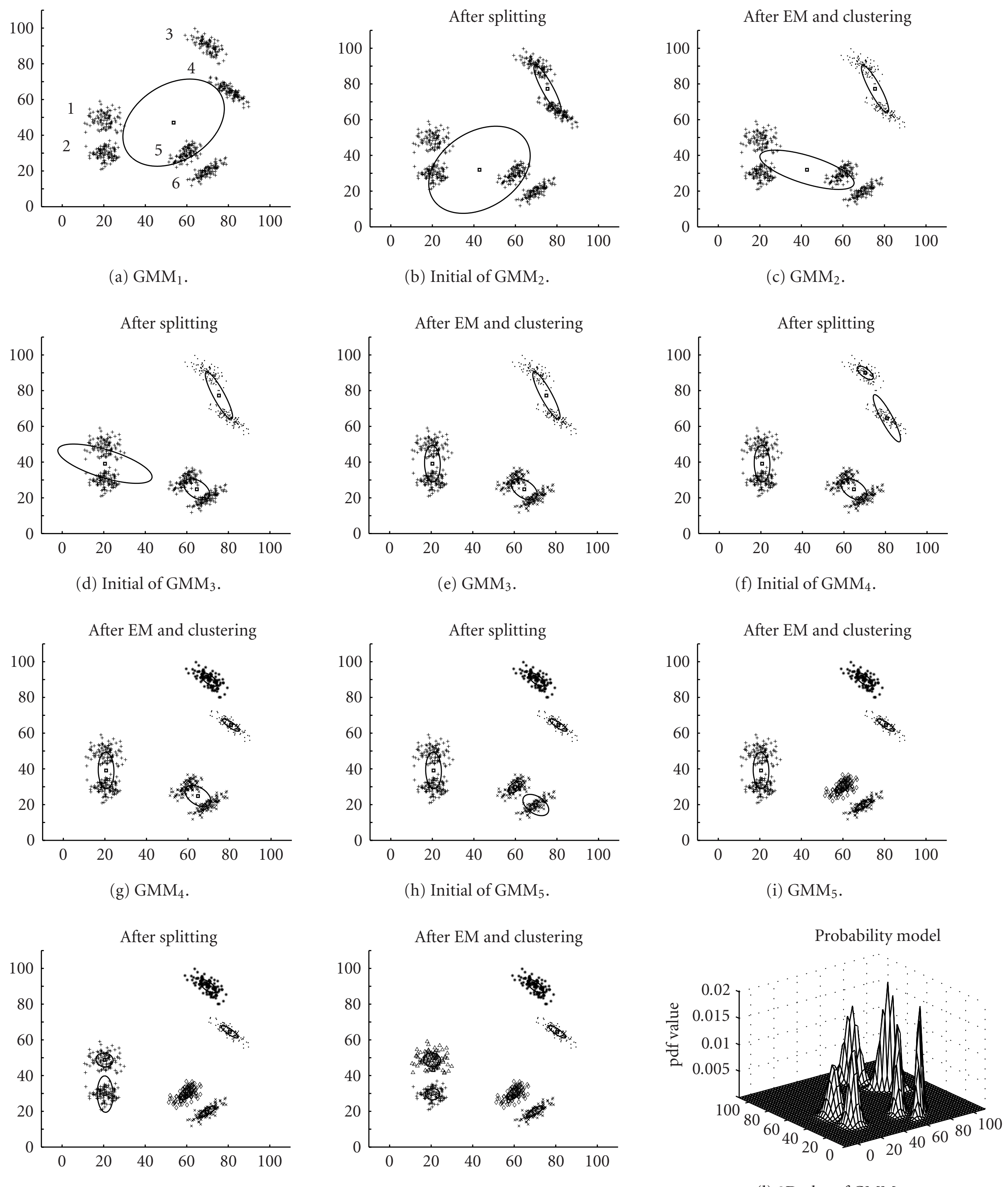
## 4. EXPERIMENTAL RESULTS

We have evaluated the proposed learning algorithms on two tasks. The first task involves applying the SGML algorithm in automatic clustering of a synthetic data set. The second task involves using the SGML algorithm to train speaker GMMs for the text-independent speaker identification application. In each task, the performance of the proposed learning algorithm was compared with that of the other EM-based learning approaches whose initial mean vectors of GMMs were located by hierarchical clustering-based or $K$-means clustering-based techniques, including the following.

(1) Hier-ComLink method. The initial Gaussian mean vectors in EM were determined by complete-link hierarchical clustering.
(2) Hier-SLink method. The initial Gaussian mean vectors in EM were determined by single-link hierarchical clustering.
(3) Hier-CenLink method. The initial Gaussian mean vectors in EM were determined by centroid-link hierarchical clustering.
(4) $K$-means-random method. The initial Gaussian mean vectors in EM were determined by $K$-means clustering, in which the initial centroids of the $K$-means clustering algorithm were randomly selected from the training patterns.
(5) $K$-means-BinSplitting method [5]. The initial Gaussian mean vectors in EM were determined by the LBG algorithm, in which each mean vector was split into two new ones in each splitting step until the desired number of clusters was reached.
(6) $K$-means-IncSplitting method [25]. The initial Gaussian mean vectors in EM were determined by the incremental splitting $K$-means algorithm, in which only the mean vector of the cluster with the largest total error was split into two new ones in each splitting step until the desired number of clusters was reached.
(7) EMSplitByMaxWeight method [24]. This method split the mean vector of the Gaussian component with the largest weight into two new ones in each splitting step, and then performed EM to update all Gaussian components. The number of mixture component was incremented from one to a predefined number.

### 4.1. The synthetic data clustering task

The synthetic data set, as shown in Figure 1a, was drawn from a distribution of six Gaussian clusters, and each cluster contains 100 samples. Figure 1 depicts how a full covariance GMM with six mixture components was learned in stages from the above synthetic data set by the SGML algorithm. The SGML algorithm stopped at $\text{GMM}_{11}$ and obtained a "significant maximum" at $\text{GMM}_6$. The result indicates that the SGML algorithm performed very well in automatic clustering of the synthetic data set. We have also tested the other methods on the same synthetic data set. In this experiment, all the Gaussian components are of full covariance and the number of mixture components of all methods is limited to an upper bound of 13. Figure 2 shows the learning curves obtained by various methods. We found that, except for the $K$-means random, all the methods could estimate the parameters of $\text{GMM}_k$ ($k = 1, 2, \dots, 13$) almost as well as the SGML on the synthetic data set.



(a) GMM$_1$. (b) Initial of GMM$_2$. (c) GMM$_2$.

(d) Initial of GMM$_3$. (e) GMM$_3$. (f) Initial of GMM$_4$.

(g) GMM$_4$. (h) Initial of GMM$_5$. (i) GMM$_5$.

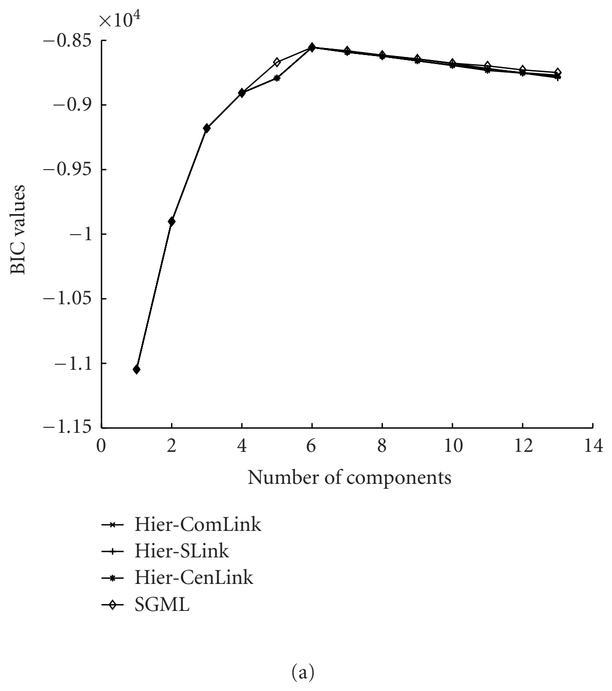
(j) Initial of GMM$_6$. (k) GMM$_6$. (l) 3D plot of GMM$_6$.

Figure 1: The learning process of applying the SGML in the synthetic data, in which full covariance GMMs were used.

When we used the diagonal covariance GMMs for the same task, as shown in Figure 3, the SGML grouped the synthetic data into ten clusters. From the perspective of "Gaussian mixture modelling" instead of the perspective of "data clustering," the learning process could be divided into three phases.

(1) The cluster-capturing phase (GMM$_1$–GMM$_6$). In this phase, the SGML captures the locations of all the clusters of the training data in the feature space roughly by the self-splitting rules.

(2) The shape-smoothing phase (GMM$_7$–GMM$_{10}$). A diagonal covariance Gaussian component is unable to



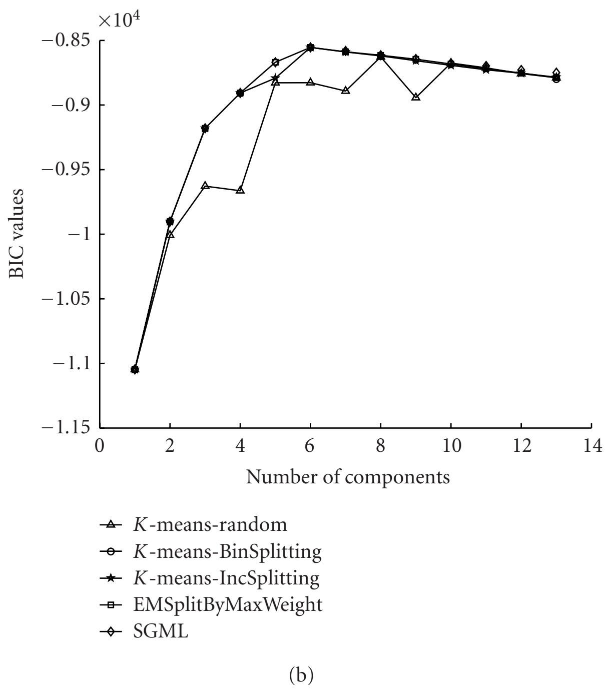


Figure 2: The learning curves of applying various methods in the synthetic data to learn full covariance GMMs.

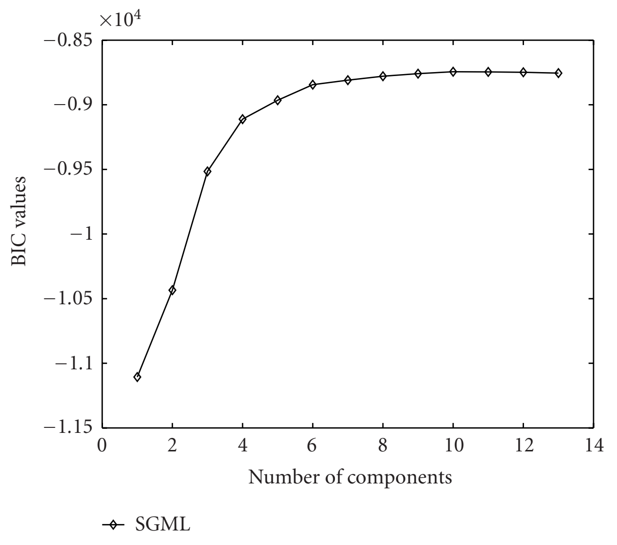


Figure 3: The learning curve of applying the SGML in the synthetic data to learn diagonal covariance GMMs.

model a cluster which has a complex shape, and this kind of cluster needs to be modelled by a mixture of Gaussians. For example, as shown in Figure 1a, clusters 3, 4, 5, and 6 are all oblique ellipses and each of them needs to be modelled by a mixture of two Gaussians. As a result, the learning curve in Figure 3 has the highest BIC value at $GMM_{10}$. It is obvious that the increase of BIC value in the shape smoothing phase is much smaller than that in the cluster-capturing phase.

(3) The overfitting phase ($GMM_{11}$–). After reaching the component number with the highest BIC value, the SGML tends to overfit the training data in each splitting step and makes the learning curve go down progressively.

### 4.2. The text-independent speaker identification task

In the past several years, Gaussian mixture modelling has been a predominant method used in speaker recognition applications [22, 23, 26, 27], out of its ability to provide smooth approximations to arbitrarily shaped densities of long-term spectrum that are considered related to the characteristics of the speaker's voice rather than the specific linguistic message. More recently, the universal background model (UBM) [27], which is a large GMM with many mixtures (e.g., 512), has been successfully applied in GMM-UBM speaker recognition systems.[2] Conventionally, the component number of the GMM is decided empirically. Both the issues of "How to train the GMM?" and "How many components should a GMM have?" have not yet been well addressed. The proposed SGML algorithm aims to handle both issues simultaneously. This section investigates if the SGML is capable of automatically determining model complexity for speaker GMMs according to the amount and characteristics of training data. Without losing generality and for ease of discussion, the speaker identification experiments were conducted

[2]In the GMM-UBM speaker recognition systems, the UBM is trained by a lot of speech data from a large population, while the speaker GMMs are adapted from the UBM by applying speaker adaptation techniques, such as maximum a posteriori (MAP), on the speaker specific training data.

using the conventional GMM method. From the perspective of data fitting, the SGML should be applicable to other speaker recognition tasks such as speaker verification using the GMM-UBM method.

### 4.2.1. Database description and feature selection

We have applied the proposed SGML to the text-independent speaker identification task. The NIST 2001 cellular speaker recognition evaluation database was used (see the benchmark tests at http://www.nist.gov/speech/tests/index.htm). There are 74 male speakers and 100 female speakers in the database, and each speaker has about 2 minutes of training data and 10 test segments on average. The duration of the test segments lies within the range of 2–54 seconds. In this paper, 50 male speakers and 50 female speakers were used for speaker identification experiments. Both the training data and test data were first processed by a voice activity detector (VAD) to discard silence noise frames (see the VIMAS speech codec at http://www.vimax.com and also [26, 27]). After passing the VAD, the amount of training data for each speaker lies within the range of 85–122 seconds, while the duration of the corresponding test segments ranges from 3 to 54 seconds. There are 615 test segments for the selected 50 male speakers, and 596 test segments for the selected 50 female speakers.

In the front-end of speech processing, spectral analysis was applied to a 32 millisecond frame of speech waveform every 10 millisecond. For each speech frame, 24 mel-frequency cepstral coefficients (MFCCs) were extracted as the observation feature vector.[3] Cepstral mean substraction (CMS) was applied in both training and test data for channel normalization.

### 4.2.2. Results in training of speaker GMMs

In this section, we compared the performance of the SGML and the other existing methods based on the training of GMMs for speaker identification. Figure 4 shows the learning curves obtained by various methods on 15 second speech data (corresponding to 1500 24-dimensional feature vectors) from one particular male speaker in the NIST 2001 cellular data. Figures 4a and 4b depict the full covariance case, while Figures 4c and 4d depict the diagonal covariance case. Their upper bounds of component number were limited to 15 and 40, respectively (in fact, the SGML stopped at $\text{GMM}_9$ and $\text{GMM}_{21}$, respectively). The corresponding "significant maximum" of the learning curves of SGML are, respectively, at $\text{GMM}_4$ and $\text{GMM}_{16}$, which are also, respectively, the global maximum in the respective learning curves of SGML. We can see that the learning curves of SGML are smoother than those of the other methods. The self-splitting learning process splits the cluster with the largest $\Delta\,\text{BIC}_{21}$ value in each splitting step, and makes the learning curve go up steadily before reaching the most appropriate component number. After reaching the best component number, the learning process tends to split a well-modelled cluster in each splitting step, and makes the learning curve go down progressively. As shown in Figure 4, the BIC values of the GMMs trained by the SGML are almost always higher than those of the GMMs trained by the other methods at any component number. As discussed in Section 2.2, with the same model complexity, the higher BIC value indicates the higher log-likelihood value. Therefore, the SGML also outperforms the other methods in learning the GMM with a given component number.

In many statistical pattern recognition tasks, the diagonal covariance GMM was used to model the probabilistic distribution of the training patterns [10, 22, 28, 29]. In the high dimension feature space case, the number of parameters of a diagonal covariance Gaussian component is much less than that of a full covariance one and, thus, a diagonal covariance GMM often needs more components to model the distribution of the training patterns than a full covariance GMM does. In the following, we further investigate into the learning of diagonal covariance GMMs from 60-second speech data of one particular male speaker in the NIST 2001 cellular data. Figure 5 shows the learning curves of SGML, fast SGML, and $K$-means-BinSplitting. The splitting confidence of the fast SGML is 150 (fast SGML-150), 100 (fast SGML-100), and 50 (fast SGML-50), respectively. The best component numbers determined by SGML, fast SGML-150, fast SGML-100, and fast SGML-50 were 40, 27, 37, and 43, respectively, and hence, $K$-means-BinSplitting was forced to stop at $\text{GMM}_{43}$. It seems that fast SGML-150 and fast SGML-100 underfit the training data while fast SGML-50 overfit them. From the learning curve of SGML, we can see that the GMMs with 30 to 50 components have similar BIC values to $\text{GMM}_{40}$ (the best model selected by the SGML). The fast SGML can obtain a GMM with a very close BIC value and a very close component number to the best GMM trained by the SGML when the splitting confidence is defined appropriately. Given a component number, the $K$-means-BinSplitting always results in smaller BIC values than the fast SGML.

### 4.2.3. Results of speaker identification

Extensive speaker identification experiments were performed in the gender dependent mode. All the GMMs used diagonal covariance matrices. In each gender, 30-, 60-, and 90-second speech data were used for training the GMMs. We run the experiments in both the variable-length test utterance case and the fixed-length test utterance case. In the variable-length case, there are 615 test segments from male speakers and 596 test segments from female speakers. These test segments range from 3 to 54 seconds as described in Section 4.2.1. In the fixed-length case, each test segment was divided into utterances of 3, 5, and 8 seconds, which yielded a total of 4714, 2706, and 1577 test utterances for male speakers, and 5583, 3234, and 1903 test utterances for female speakers. Notice that dividing utterances into short fixed-length pieces might result in nonindependent identification trials. However, the experimental results can still show the performance of the identification system being tested by short utterances.

[3]We did not use energy information and delta_MFCC in this work.

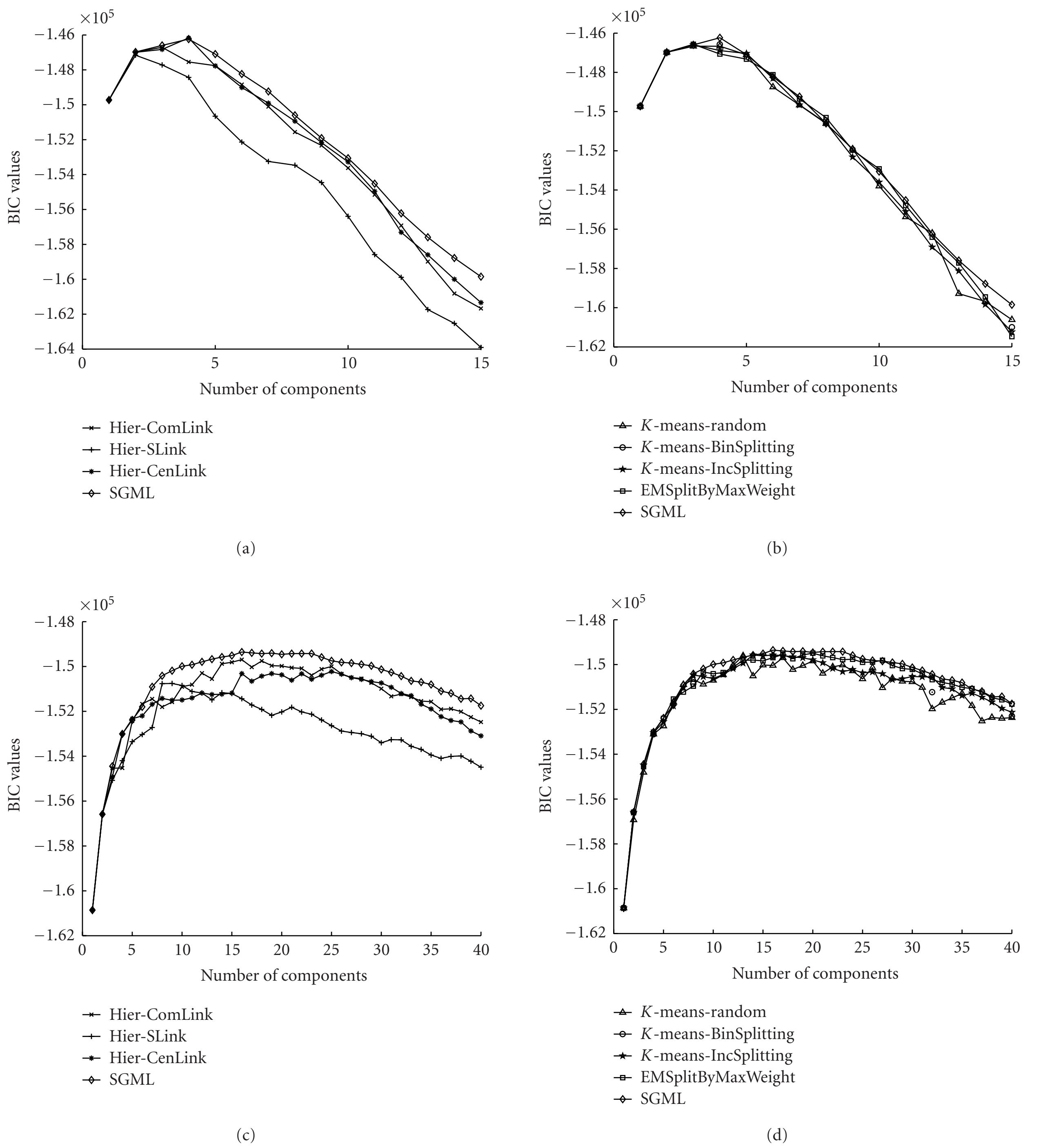


Figure 4: The learning curves of applying various GMM learning methods in the 15-second speech data of one particular male speaker in the NIST 2001 cellular data. (a) and (b) depicts the full covariance case, while (c) and (d) depict the diagonal covariance case.

Tables 1 and 2 summarize the mean and standard deviation of the component number of the male speaker GMMs and female speaker GMMs, respectively, obtained by the SGML and fast SGML on various amounts of training data. The splitting confidence within the range of 100 to 150 seems to be applicable for these two speaker identification tasks since, based on which, the fast SGML yielded similar model complexity of speaker GMMs compared to the SGML. Furthermore, it is interesting to find that, on average, a female speaker GMM needs more Gaussian components than a male speaker GMM, with the same amount of training data. It seems that the distribution of training feature vectors of a female speaker is more diverse than that of a male speaker.

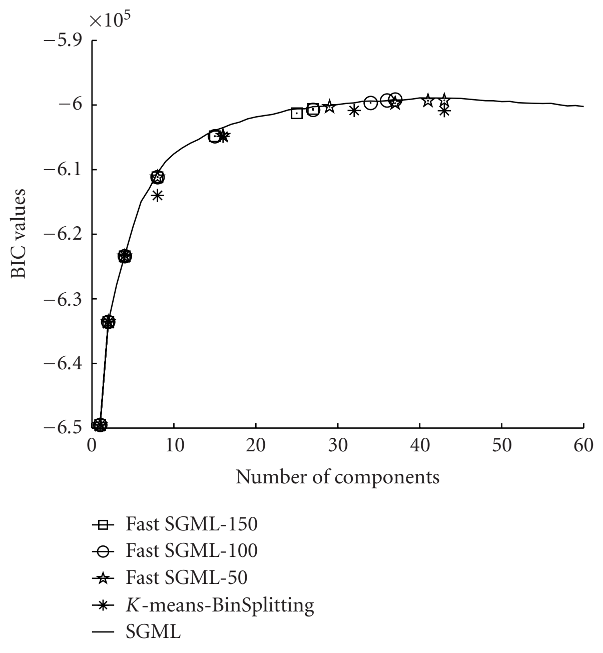


Figure 5: The learning curves of applying SGML, fast SGML, and $K$-means-BinSplitting in the 60-second speech data of one particular male speaker in the NIST 2001 cellular data to learn diagonal covariance GMMs. The splitting confidence of the fast SGML was 150, 100, and 50, respectively, and the $K$-means-BinSplitting was forced to stop at $\text{GMM}_{43}$.

Reynolds and Rose [22] concluded that there was no significant difference in speaker identification performance between $K$-means with randomly selected initial means and binary-splitting $K$-means clustering for initialization of GMM parameters. In this study, the $K$-means-BinSplitting was applied in the baseline GMM-based speaker identification system. Tables 3 and 4 show, respectively, the identification performance of the male and female speakers, in which the $K$-means-BinSplitting, SGML, and fast SGML methods were performed on various amounts of training speech data. In Table 3, in the male speaker case of 30 second training speech, the identification accuracy of $K$-means-BinSplitting was first improved by increasing the component number from 8 to 16, and then degraded by further increasing the component number to 32 and 64 due to the overfitting of the training process. In this case, the SGML yielded 23.92 Gaussian components on average for each male speaker, as shown in Table 1. In Table 4, for the female speaker case, it seems that the overfitting phenomenon is not as obvious as that in the male case. For example, in the female speaker case of 30 second training speech, a similar trend was observed, and the baseline system achieved the best identification accuracy with the component number 32. This conforms to the observation from Tables 1 and 2 that a female speaker GMM in general needs more Gaussian components than a male speaker GMM. In this case, the SGML yielded 29.44 components on average for each female speaker, as shown in Table 2. In the cases of 60 second and 90 second training speech in Tables 3 and 4, we can also observe that the SGML could automatically capture the adequate model complexity for speaker GMMs according to the amount and characteristics of training data, though no significant difference was found between the results of SGML and the best accuracies of $K$-means-BinSplitting under various training and testing conditions. We have also observed that there is a huge performance gap between the female case and the male case. This gap is obviously due to the diversity of feature vectors of a female speaker. For the female case, more training data are needed to cover the diverse feature space.

We have also conducted the same experiments using $K$-means-random and EMSplitByMaxWeight [24]. The experimental results have the same trend as that of the $K$-means-BinSplitting method and no significant difference in identification accuracy was found between these three methods. For example, as shown in Table 3, for the male speaker case of 90 second training speech and 64 Gaussian mixtures, the identification accuracies are 68.62, 61.99, 65.67, and 68.36, respectively, when the speaker identification system trained by the $K$-means-BinSplitting method was tested with variable-length, 3 second, 5 second, and 8 second test utterances. The accuracies are 69.92, 61.94, 65.34, and 68.67, respectively, when the EMSplitByMaxWeight method was evaluated under the same experimental condition, while they are 67.64, 60.54, 64.15, and 67.72, respectively, when the $K$-means-random method was applied.

From Tables 3 and 4, we can find that the fast SGML in general yielded as good performance as the SGML. Though the fast SGML with different splitting confidence might result in GMMs with different numbers of components as shown in Tables 1 and 2, there was no significant difference in identification accuracy between these models. The splitting confidence within the range of 100 to 150 seems to be applicable for these two speaker identification tasks since, based on which, the fast SGML yielded similar model complexity of speaker GMMs and identification accuracy compared to the SGML.

To further assess the statistical significance of the above experimental results, we applied a bootstrap resampling procedure [30] to the case of 90-second training speech and variable-length testing speech. The error bars obtained with 10 000 replications are shown in Figure 6, where the length of the error bars represents the standard deviation above and below the mean accuracy. It can be found that the error bars of the SGML and the fast SGML overlap with the error bars of the $K$-means-BinSplitting with 32 mixture components, either in the male case or the female case.

From the speaker identification experimental results, we can see that the proposed SGML and fast SGML could automatically find the appropriate component number for the GMMs, though the identification accuracy was not significantly improved as expected, compared to the best accuracies of the baseline system. The fast SGML is almost as effective as the SGML in the training of GMMs, but at a much lower computation cost.

Table 1: The mean and standard deviation of the component number of the diagonal covariance male speaker GMMs obtained by the SGML and fast SGML on various amounts of training data. The first number in parentheses is the mean value while the second number after "/" is the standard deviation.

| Amount of training speech | SGML | Splitting confidence (fast SGML) | | |
|---|---|---|---|---|
| | | 150 | 100 | 50 |
| 30 s | (23.92/5.07) | (20.58/5.31) | (25.32/6.63) | (27.16/5.99) |
| 60 s | (35.96/6.90) | (31.72/7.68) | (38.22/9.26) | (41.50/9.42) |
| 90 s | (46.70/9.23) | (41.30/9.48) | (48.58/11.13) | (53.00/10.65) |

Table 2: The mean and standard deviation of the component number of the diagonal covariance female speaker GMMs obtained by the SGML and fast SGML on various amounts of training data. The first number in parentheses is the mean value while the second number after "/" is the standard deviation.

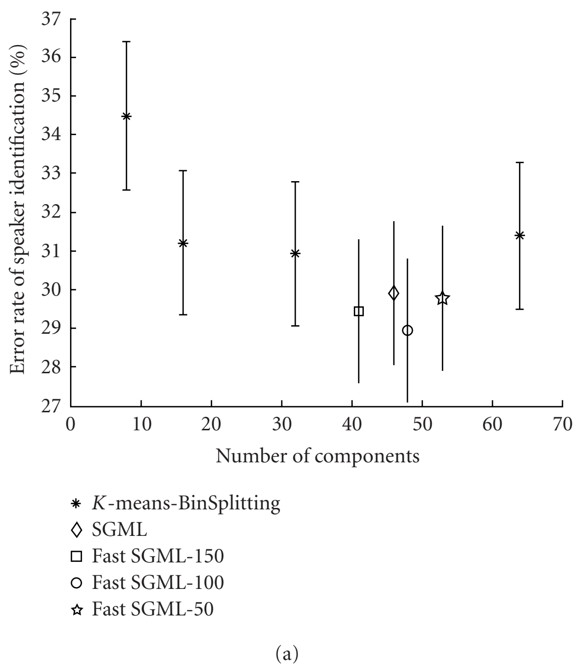
| Amount of training speech | SGML | Splitting confidence (fast SGML) | | |
|---|---|---|---|---|
| | | 150 | 100 | 50 |
| 30 s | (29.44/4.59) | (26.32/5.25) | (31.70/5.97) | (33.34/5.97) |
| 60 s | (45.06/7.18) | (42.22/6.93) | (50.26/7.85) | (52.60/10.40) |
| 90 s | (58.26/7.63) | (55.82/9.15) | (65.16/10.16) | (70.18/11.56) |

Table 3: Speaker identification accuracy (%) for the male speakers.

| Amount of training speech | Length of test utterance | Number of components ($K$-means-BinSplitting) | | | | SGML | Splitting confidence (fast SGML) | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 8 | 16 | 32 | 64 | | 150 | 100 | 50 |
| 30 s | Variable length | 61.46 | 62.28 | 61.46 | 59.35 | 61.95 | 61.46 | 62.76 | 61.79 |
| | 3 s | 51.68 | 54.41 | 54.41 | 52.38 | 54.09 | 53.67 | 53.86 | 54.07 |
| | 5 s | 56.91 | 58.39 | 57.02 | 55.99 | 57.98 | 57.58 | 57.80 | 57.91 |
| | 8 s | 59.80 | 60.68 | 60.24 | 58.47 | 61.88 | 60.11 | 61.19 | 60.49 |
| 60 s | Variable length | 63.25 | 66.02 | 66.34 | 65.85 | 66.67 | 66.67 | 67.15 | 66.83 |
| | 3 s | 54.29 | 58.38 | 58.91 | 59.29 | 59.10 | 59.65 | 59.80 | 59.16 |
| | 5 s | 58.94 | 61.90 | 62.82 | 62.12 | 63.45 | 62.75 | 63.19 | 62.75 |
| | 8 s | 61.95 | 65.25 | 66.14 | 65.88 | 66.39 | 66.65 | 67.22 | 66.77 |
| 90 s | Variable length | 65.53 | 68.78 | 69.11 | 68.62 | 70.08 | 70.57 | 71.06 | 70.24 |
| | 3 s | 55.28 | 59.10 | 61.03 | 61.99 | 61.69 | 61.96 | 61.79 | 61.75 |
| | 5 s | 60.01 | 63.34 | 65.59 | 65.67 | 65.78 | 64.56 | 65.37 | 65.78 |
| | 8 s | 63.86 | 66.01 | 68.55 | 68.36 | 69.93 | 69.75 | 69.37 | 68.29 |

Table 4: Speaker identification accuracy (%) for the female speakers.

| Amount of training speech | Length of test utterance | Number of components ($K$-means-BinSplitting) | | | | SGML | Splitting confidence (fast SGML) | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 8 | 16 | 32 | 64 | | 150 | 100 | 50 |
| 30 s | Variable length | 29.87 | 32.21 | 35.07 | 30.37 | 30.54 | 33.05 | 29.87 | 30.70 |
| | 3 s | 27.48 | 28.46 | 30.41 | 29.50 | 29.09 | 29.30 | 27.66 | 29.23 |
| | 5 s | 29.87 | 30.55 | 32.25 | 30.49 | 30.67 | 31.29 | 28.76 | 31.08 |
| | 8 s | 31.11 | 31.90 | 33.95 | 32.00 | 33.00 | 33.00 | 30.74 | 32.79 |
| 60 s | Variable length | 40.77 | 42.45 | 45.30 | 45.13 | 44.97 | 45.13 | 45.47 | 45.47 |
| | 3 s | 32.08 | 34.03 | 37.20 | 37.95 | 37.79 | 38.06 | 37.33 | 37.78 |
| | 5 s | 35.65 | 37.97 | 40.66 | 40.63 | 40.11 | 40.91 | 40.60 | 41.00 |
| | 8 s | 38.83 | 40.20 | 42.62 | 43.72 | 43.14 | 43.77 | 42.35 | 43.08 |
| 90 s | Variable length | 42.95 | 43.62 | 48.83 | 48.49 | 47.32 | 47.48 | 47.65 | 48.49 |
| | 3 s | 32.12 | 35.41 | 39.03 | 40.66 | 41.78 | 40.78 | 40.64 | 41.00 |
| | 5 s | 35.62 | 38.74 | 42.92 | 44.47 | 44.34 | 44.53 | 44.59 | 45.24 |
| | 8 s | 38.52 | 42.35 | 46.19 | 47.08 | 47.50 | 47.08 | 46.82 | 47.24 |



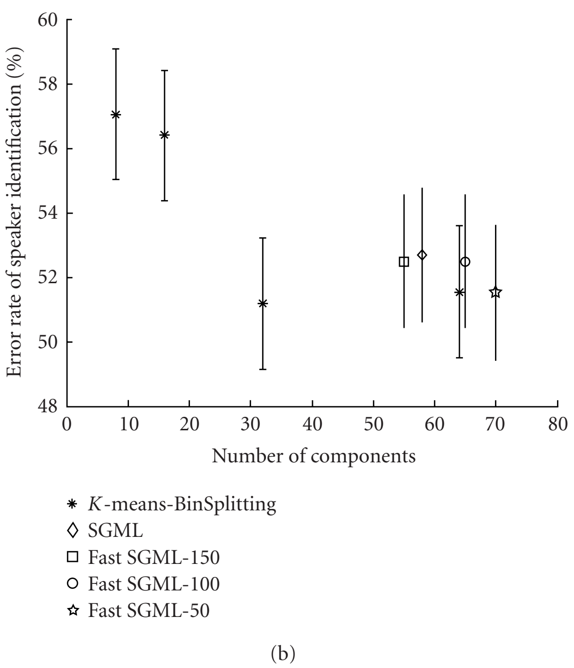


FIGURE 6: The error bars obtained by applying a bootstrap resampling procedure to the case of 90-second training speech and variable-length testing speech. (a) The male speaker case. (b) The female speaker case.

## 5. CONCLUSIONS

In this paper, we have presented the SGML algorithm for the learning of GMMs. The SGML algorithm tries to tackle two long standing critical problems in the EM-based Gaussian mixture modelling; namely, (1) the difficulty in determining the number of Gaussian components and (2) the sensitivity to prototype initialization. The SGML algorithm is based on a splitting validity criterion, BIC. During the learning process, according to the splitting validity criterion (BIC), the prototype with the largest $\Delta\,\text{BIC}_{21}$ value is split into two prototypes. The learning process automatically terminates when a significant maximum in the learning curve (i.e., the BIC plot) is observed.

A fast version of the SGML, named fast SGML, was also presented in this paper. The fast SGML splits multiple prototypes in each splitting step and, thus, needs a much lower computation requirement than the SGML. The fast SGML is somewhat like the LBG algorithm [5], but a crucial difference is that there is no validation measure to decide whether a cluster should be split into two clusters and how many clusters should be generated in the LBG algorithm.

We have conducted extensive experiments on clustering of a synthetic data set and text-independent speaker identification. Compared to the other EM-based Gaussian mixture modelling approaches, both the SGML and fast SGML achieved a noticeable improvement in the training of GMMs. In the speaker identification, the proposed algorithms could automatically find out the suitable model complexity for the speaker GMMs though no significant improvements in identification accuracy were obtained compared to the best performance of the baseline system.

## ACKNOWLEDGMENTS

This work was funded by Academia Sinica and the National Science Council of the Republic of China under Grant no. NSC 93-2213-E-001-017. We are grateful to Dr. Wei-Ho Tsai, Mr. Yi-Hsiang Chao, and Mr. Cheng-Lung Tseng for their helpful discussions and assistance in running the experiments. Useful comments and suggestions made by the anonymous referees are also greatly appreciated.

## REFERENCES

[1] J.-M. Jolion, P. Meer, and S. Bataouche, "Robust clustering with applications in computer vision," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 791–802, 1991.

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

[3] E. Forgy, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications," *Biometrics*, vol. 21, no. 3, pp. 768, 1965.

[4] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

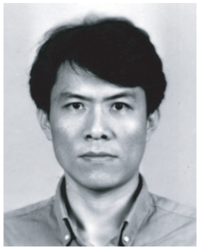
[5] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Communications*, vol. 28, no. 1, pp. 84–95, 1980.

[6] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, University of California Press, Berkeley, Calif, USA, 1967.

[7] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[8] G. McLachlan and D. Peel, *Finite Mixture Models*, John Wiley & Sons, New York, NY, USA, 2000.
[9] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B*, vol. 39, no. 1, pp. 1–38, 1977.
[10] X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*, Prentice-Hall, Upper Saddle River, NJ, USA, 2001.
[11] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review*, vol. 26, no. 2, pp. 195–239, 1984.
[12] N. Ueda, R. Nakano, Z. Ghahramani, and G. Hinton, "SMEM algorithm for mixture models," *Neural Computation*, vol. 12, no. 9, pp. 2109–2128, 2000.
[13] D. Ormoneit and V. Tresp, "Improved Gaussian mixture density estimates using Bayesian penalty terms and network averaging," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds., vol. 8, pp. 542–548, MIT Press, Cambridge, Mass, USA, 1996.
[14] K. P. Burnham and D. R. Anderson, *Model Selection and Inference: A Practical Information-Theoretic Approach*, Springer-Verlag, New York, NY, USA, 1998.
[15] G. Schwarz, "Estimation the dimension of a model," *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.
[16] A. E. Raftery, "Bayesian model selection in social research," *Sociological Methodology*, vol. 25, pp. 111–196, 1995.
[17] C. Fraley and A. E. Raftery, "How many clusters? which clustering method? Answers via model-based cluster analysis," *Computer Journal*, vol. 41, no. 8, pp. 578–588, 1998.
[18] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.
[19] J. J. Oliver, R. A. Baxter, and C. S. Wallace, "Unsupervised learning using MML," in *Proc. 13th International Conference on Machine Learning (ICML '96)*, pp. 364–372, San Francisco, Calif, USA, 1996.
[20] B. D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge, UK, 1996.
[21] R. E. Kass and A. E. Raftery, "Bayes factors," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 773–795, 1995.
[22] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. Speech, and Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995.
[23] L. Wang, K. Chen, and H. S. Chi, "Capture interspeaker information with a neural network for speaker identification," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 436–445, 2002.
[24] S. J. Young and P. C. Woodland, "State clustering in hidden Markov model-based continuous speech recognition," *Computer Speech & Language*, vol. 8, no. 4, pp. 369–384, 1994.
[25] S.-C. Chu and J. F. Roddick, "Pattern clustering using incremental splitting for non-uniformly distributed data," in *Proc. 5th International Conference on Knowledge-Based Intelligent Information Engineering Systems and Allied Technologies*, N. Baba, L. C. Jain, and R. J. Howlett, Eds., vol. 69 of *Frontiers in Artificial Intelligence and Applications*, pp. 1037–1041, IOS Press, Osaka, Japan, 2001.
[26] D. A. Reynolds, "Speaker identification and verification using Gaussian mixture speaker models," *Speech Communication*, vol. 17, no. 1, pp. 91–108, 1995.
[27] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1–3, pp. 19–41, 2000.
[28] H.-C. Fu, H.-Y. Chang, Y. Y. Xu, and H.-T. Pao, "User adaptive handwriting recognition by self-growing probabilistic decision-based neural networks," *IEEE Transactions on Neural Networks*, vol. 11, no. 6, pp. 1373–1384, 2000.
[29] S.-H. Lin, S.-Y. Kung, and L.-J. Lin, "Face recognition/detection by probabilistic decision-based neural network," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 114–132, 1997, Special issue on artificial neural network and pattern recognition.
[30] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*, Chapman & Hall, New York, NY, USA, 1993.

**Shih-Sian Cheng** received the B.S. degree in mathematics in 2000 from National Kaohsiung Normal University, Taiwan, and the M.S. degree in computer science and information engineering in 2002 from National Chiao Tung University, Taiwan. In 2002, he has joined the Spoken Language Group of the Chinese Information Processing Laboratory, Institute of Information Science, Academia Sinica, Taiwan, as a Research Assistant. He is currently pursuing the Ph.D. degree in the Department of Computer Science and Information Engineering, National Chiao Tung University. His research interests include pattern recognition, speech processing, and neural networks.



**Hsin-Min Wang** received the B.S. and Ph.D. degrees in electrical engineering from National Taiwan University in 1989 and 1995, respectively. Since then, he joined the Institute of Information Science, Academia Sinica, as a Postdoctoral Fellow. He was promoted to Assistant Research Fellow and then Associate Research Fellow in 1996 and 2002, respectively. He is an Adjunct Associate Professor at National Taipei University of Technology and National Chengchi University. His major research interests include speech processing, natural language processing, spoken dialogue processing, multimedia information retrieval, and pattern recognition. He is a Member of IEEE and ISCA.

**Hsin-Chia Fu** received the B.S. degree from the National Chiao-Tung University in electrical and communication engineering in 1972, and the M.S. and Ph.D. degrees from New Mexico State University, 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. Since 1983, he has been on the faculty of the Department of Computer science and Information engineering, National Chiao-Tung University, Taiwan. He is also the Taiwan Representative of TEI Consortium since 2003. His research interests include digital signal/image processing, multimedia information processing, and neural networks. He has been in the the Technical Committee on Neural Networks for Signal Processing, IEEE Signal Processing Society, from 1997 to 2000. He has authored more than 100 technical papers, and two textbooks *PC/XT BIOS Analysis*, and *Introduction to Neural Networks*, by Sun-Kung Book Co. and Third Wave Publishing Co., respectively. Dr. Fu is a Member of the IEEE Signal Processing and Computer Societies, Phi Tau Phi, and the Eta Kappa Nu Electrical Engineering Honor Society.