A Speaker-Independent Continuous Speech Recognition System Using Continuous Mixture Gaussian Density HMM of Phoneme-Sized Units

Yunxin Zhao, *Member, IEEE*

Abstract— **This paper describes a large vocabulary, speaker-independent, continuous speech recognition system which is based on hidden Markov modeling (HMM) of phoneme-sized acoustic units using continuous mixture Gaussian densities. A bottom-up merging algorithm is developed for estimating the parameters of the mixture Gaussian densities, where the resultant number of mixture components is proportional to both the sample size and dispersion of training data. A compression procedure is developed to construct a word transcription dictionary from the acoustic–phonetic labels of sentence utterances. A modified word-pair grammar using context-sensitive grammatical parts is incorporated to constrain task difficulty. The Viterbi beam search is used for decoding. The segmental K-means algorithm is implemented as a baseline for evaluating the bottom-up merging technique. The system has been evaluated on the TIMIT database (1990) for a vocabulary size of 853. For test set perplexities of 24, 104, and 853, the decoding word accuracies are 90.9%, 86.0%, and 62.9%, respectively. For the perplexity of 104, the decoding accuracy achieved by using the merging algorithm is 4.1% higher than that using the segmental K-means (22.8% error reduction), and the decoding accuracy using the compressed dictionary is 3.0% higher than that using a standard dictionary (18.1% error reduction).**

I. Introduction

In recent years, a number of successful hidden Markov model (HMM) based speaker-independent continuous speech recognition systems have been demonstrated [1], [2]. In these systems, the acoustic signals are modeled by phoneme-sized subword units using either VQ-based discrete density HMMs, or continuous mixture Gaussian density HMMs, where the parameters of Gaussian densities are estimated using the segmental K-means algorithm [3]. The word transcription dictionaries used in these systems are prepared from standard references which generally yield a single transcription per lexicon entry. These systems have been primarily evaluated on the DARPA Resource Management database among DARPA related research groups.

This paper describes a newly developed speaker-independent continuous speech recognition system for a large vocabulary [4], [5]. On the acoustic signal level, the system also uses hidden Markov models of phoneme-sized acoustic units since they are flexible and trainable for large vocabulary continuous speech recognition tasks. Continuous mixture Gaussian densities are used to model the feature parameters of auditory spectra, where correlation between the feature components are captured by the structure of the covariance matrices. A new algorithm is developed for estimating the parameters of the mixture Gaussian densities. The objective of the new algorithm is to find the multi-modal structure of the training data clusters, and fit the data distribution by a mixture Gaussian density. The search of the data clusters is accomplished through a bottom-up merging procedure, and each cluster is modeled by a Gaussian density. The resultant number of mixture components is proportional to both the dispersion and the sample size of the training data. In contrast, the segmental K-means algorithm clusters the training data through iterations of nearest-neighbor assignments and centroid updates, which requires *a priori* choices of the number of clusters and their centroids.

The TIMIT database [6], [7] was released to the public by the National Institute of Standards and Technology (NIST) at the time the system was started. Since the database has a rich collection of sentences spoken by speakers from the dialect regions of the United States and supplies labels of acoustic–phonetic units for all the sentences, it provides us with an excellent source for the study of continuous speech. For these reasons, this database was chosen for system training and evaluation. Upon considering pronunciation variations in different dialects, speaking contexts, etc., it seems that using the labels of actual speech utterances and multiple transcriptions for certain lexicon entries might improve the decoding accuracy. Since using the word labels directly for a dictionary leads to too many variations of word transcriptions, an automatic procedure is developed to build more compact dictionaries. Since the TIMIT database is not designed as a task-oriented database, e.g., the DARPA Resource Management database, modified word-pair grammars are generated to evaluate the decoding performance of the system under controlled task difficulties. The Viterbi algorithm is used to decode the sentences into word strings. To further evaluate the quality of the merging algorithm for model training, a baseline training algorithm of the segmental K-means is implemented. A comparison of decoding rates from using the two different training algorithms is provided.

The paper is organized into seven sections. In Section II, an overview of the system structure is given, and the functions of its major modules are summarized. Section III focuses on the method of generating mixture Gaussian density models of

Manuscript received April 21, 1991; revised October 15, 1992. The associate editor coordinating the review of this paper and approving it for publication was Dr. David Nahamoo.

The author is with the Speech Technology Laboratory, Panasonic Technologies Inc., Santa Barbara, CA 93105.

IEEE Log Number 9208584.

phone units. Section IV develops the method of dictionary preparation. Section V defines the use of context-sensitive grammatical parts in word-pair grammars. The experimental results on the TIMIT database are provided in Section VI, and a conclusion is made in Section VII.

II. System Overview

The system consists of five modules: feature extraction, HMM phone model training, dictionary preparation, grammar estimation, and sentence decoding. A block diagram of the system is shown in Fig. 1. The functions of these modules are outlined as follows.

A. Feature extraction

The front-end uses perceptually based linear prediction (PLP) analysis [8]. The PLP analysis transforms the spectra of speech signals into auditory spectra, and then fits each spectrum with an all-pole model. The cepstrum coefficients of each auditory spectrum are further weighted by a linear scale, where the weighting has been shown to improve the robustness of the feature representation [9]. These weighted cepstrum coefficients and log energy are used as instantaneous features, and the temporal regression coefficients of the instantaneous features are then taken as dynamic features [10]. The use of weighted cepstrum coefficients leads to a spectral-slope based distance measure for comparing the mean vectors of Gaussian density pairs in the bottom-up model merging.

B. HMM phone model training

The phoneme-sized acoustic–phonetic segments used for labeling the TIMIT sentences are chosen as basic phone units of the hidden Markov models. An HMM phone model, as shown in Fig. 2, has the well-known structure of three tied-states, where the states B and E represent the beginning and ending transition portions of the phone, and the state M represents the center portion of the phone. This structure limits the duration of a phone unit to be at least two frames, which is the lower limit of the phone duration observed from the labels in the TIMIT database.

Each state of an HMM phone model corresponds to a sub-segment of a phonetic segment, and is modeled by a mixture Gaussian density. The traditional structure of covariance matrices of the mixture Gaussian densities have been either diagonal or full. The use of a diagonal covariance matrix assumes that the feature components are independent, which is not a correct assumption in general; and the use of a full covariance matrix allows the feature components to be correlated and is more accurate. On the other hand, the diagonal structure requires fewer parameters than the full structure and is therefore computationally more efficient. A different covariance structure is used in this work. Since there exists a natural separation of the features into the instantaneous and dynamic sets, an assumption is made that the correlations within each set are much stronger than the correlations between the two sets of features. Therefore the between-set correlations are not modeled. This assumption leads to a block-diagonal structure of covariance matrices. This block-diagonal structure has the advantages of being more accurate than the diagonal structure, and yet it requires half the number of parameters of the full structure.

Let $x = (c_1, \cdots c_L, p)'$ be an instantaneous feature vector, where the c_i's are weighted cepstrum coefficients, p is the log energy which is normalized by the maximum framed-based energy within the corresponding sentence, and the symbol $'$ denotes the transpose for a vector or a matrix throughout the paper. Let the corresponding dynamic feature vector be $\Delta x = (\Delta c_1, \cdots, \Delta c_L, \Delta p)'$, where each dynamic feature is a first-order temporal regression coefficient of the corresponding instantaneous feature. A mixture Gaussian density with M mixture components is defined as

$$f(x, \Delta x) = \sum_{i=1}^{M} \alpha_i f_{s,i}(x) f_{d,i}(\Delta x) \tag{1}$$

where $f_{s,i}$ and $f_{d,i}$ are both Gaussian densities with $f_{s,i} \sim \mathcal{N}(\mu_{s,i}, C_{s,i})$, $f_{d,i} \sim \mathcal{N}(\mu_{d,i}, C_{d,i})$, and the subscripts s and d stand for the instantaneous and dynamic features, respectively. The α_i's are the weights of the mixture components with $\alpha_i \geq 0$ and $\sum_{i=1}^{M} \alpha_i = 1$. The product of $f_{s,i}$ and $f_{d,i}$ is equivalent to a single Gaussian density $\mathcal{N}(\mu_i, C_i)$, where

$$\mu_i = (\mu'_{s,i}, \mu'_{d,i})'$$

and

$$C_i = \begin{pmatrix} C_{s,i} & 0 \\ 0 & C_{d,i} \end{pmatrix}.$$

From an autoregressive modeling point of view, each stationary portion of a phone can be considered to be coming from a single Gaussian source. This segmental structure of a speech signal is utilized such that a phone segment is divided into sub-segments, and each sub-segment is modeled by a Gaussian density. Since the sentences are labeled in the TIMIT database, it is straightforward to extract acoustic–phonetic segments from the sentences. Each segment is further divided into three sub-segments corresponding to the states of the unit model. Let a phone segment have a length of L frames. The initial segmentation rule is that the beginning sub-segment has $L/4$ frames, the center one $L/2$ frames, and the ending one $L/4$ frames. The sample mean and covariance matrix are estimated for each sub-segment. In case that a unit is only two frames long, the center state of the phone unit is skipped. Assume that in the training data there are N Gaussian densities corresponding to N sub-segments tied to a state of an HMM phone unit. To obtain parameter estimates of a mixture density from the N densities, a merging algorithm is developed based on the criterion of minimizing the increment of the average trace of the mixture density (see Section III for more details). The mixture Gaussian densities thus generated become the initial models of the phone units. These initialized models are then used to compress a full dictionary which is built from the acoustic–phonetic labels of the training sentences. Using the compressed dictionary and the initial models, each sentence is automatically resegmented into acoustic–phonetic units, as well as their states, using the Viterbi algorithm. The newly obtained sub-segments are then used to reestimate the

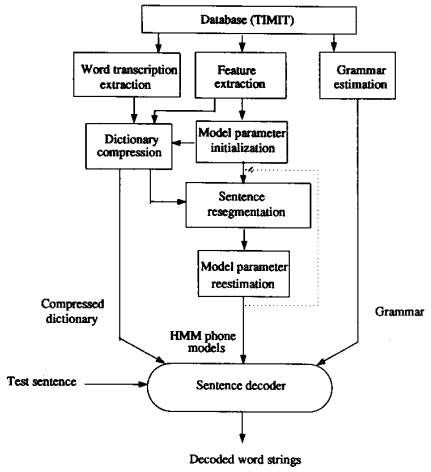

Fig. 1. The block diagram of the recognition system.

individual Gaussian densities, which are once again merged into mixture Gaussian densities using the merging algorithm. The resegmentation of the training sentences and the model merging can be implemented as an iterative loop, shown as the dotted path in Fig. 1. The assumption is that the newly merged models will lead to more accurate segmentation of the training sentences, and the accurate resegmentation of the training sentences will in turn lead to a better set of models through merging. The loop terminates when there is either no more change in segmentation or in models generated from merging. In practice, when the initial models are good, the sentences are already well segmented after one pass of the resegmentation. Since the initial segmentations are based on the acoustic–phonetic units labeled in the TIMIT database, the initial models are well trained, and experiments showed that there is no need to go through the iterative procedure of resegmentation and merging.

The transitions between states are constrained by the topology of the HMM phone models, shown in Fig. 2. Since both the segmentation and decoding are performed using the Viterbi algorithm, the probabilistic weighting of state transitions is not necessary and the estimation of the transition probabilities is avoided. If the Baum–Welch forward–backward algorithm is used, it will be necessary to use the transition probabilities. The topology of the HMM can be represented by the 0–1 transition constraints. Specifically, let the function $\phi(i,j)$ define the transition constraint between the node i and node j, then $\phi(i,j) = 1$ if there is a link from the node i to node j, and $\phi(i,j) = 0$ otherwise. This treatment of the transition probabilities is based on the informal evaluation results where higher decoding accuracy was obtained when the transition probabilities were not used.

C. Dictionary Preparation

Since the TIMIT database has labels of acoustic–phonetic units for all the sentences, the word transcription dictionary is generated by first extracting the word transcriptions directly from the database, and then is compacted through a compression procedure to retain only representative transcriptions. The compression is based on a cost measure of joint word likelihood. Each step of compression finds a transcription to eliminate which minimizes the decrease of the joint likelihood, and the compression terminates when the cost exceeds a threshold. The compressed dictionary is used for resegmenting the training sentences as well as for decoding the test sentences.

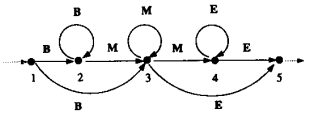

Fig. 2. The topology of an HMM phone model, where B, M, and E represent the beginning, center, and ending states, respectively.

D. Grammar

A word-pair grammar has been used in continuous speech recognition systems for evaluating the system performance at a controlled task difficulty level [11]. To achieve flexible control of task difficulty on the TIMIT database, the word-pair grammar is modified such that context-sensitive grammatical parts are defined to smooth the transitions between words instead of using the grammatical parts directly. The modified grammar allows the control of different levels of perplexity by varying the neighborhood size of the context dependency, which is convenient for evaluating a recognition system on progressively more difficult tasks.

E. Sentence Decoding

The Viterbi algorithm is used for sentence decoding. The node expansion is controlled via a threshold in the beam search. The likelihood score of a mixture density is evaluated by taking the maximum of the logarithmic score among its components, i.e.,

$$\log \hat{f}(x,\Delta x) = \max_{i=1,\cdots,M} \{\log \alpha_i + \log f_{s,i}(x) + \log f_{d,i}(\Delta x)\}. \tag{2}$$

This search of the maximum value is more efficient in computation than the average calculation of $f(x,\Delta x)$ in (1) (especially when M is large), since multiplication and exponential evaluation of Gaussian densities are avoided. This method of computing the score of a mixture density is similar to the technique of PGAM in [12]. The difference is that in [12] the mixture weights are set to be uniform, whereas the weights in (2) are nonuniform and are trained to be context sensitive. The word durations are modeled by Gaussian densities, and a word penalty is also used for reducing insertion errors. The path that has the highest likelihood score is back tracked to produce the decoded word sequence of the sentence.

III. HMM Model Training

Each state of an HMM phone model is modeled by a mixture Gaussian density. For each mixture density, the mixture size and the Gaussian density parameters are determined by the bottom-up merging algorithm; the mixture weights are separately estimated via an iterative procedure. To increase the

model robustness, smoothings are performed on the estimated covariance matrices and the mixture weights.

A. The Bottom-Up Merging Algorithm

The training sentences are segmented into sub-segments of phone units. The initial segmentation can be performed by using the acoustic–phonetic labels provided by the TIMIT database and some simple segmentation rules; or it can be done by using HMM phone models trained from other readily available sources. The initial HMM phone models are used to evaluate the joint word likelihoods for dictionary compression (see Section IV for more details). After the word transcription dictionary is compressed, a refined segmentation is performed using the Viterbi algorithm, which is constrained by the sentence texts, word transcriptions, and the HMM phone model topology, to reflect the possible changes of segment boundaries. After each pass of segmentation, the merging algorithm is used to generate mixture Gaussian density models for each state of each phone unit.

Let there be N sub-segments tied to a state of a given HMM phone unit. Since the features within a sub-segment represent the spectral frames of a relatively stationary portion of an allophone from a single speaker, they form a small cluster in the multidimensional feature space. Each small cluster is initially modeled by a Gaussian density $\mathcal{N}(\mu_i, C_i)$. Because the sizes of the sub-segments are usually small, the Gaussian densities are crude parametric characterizations of the small clusters. The mean vector μ_i is the centroid of the cluster, and the covariance matrix C_i measures the dispersion of features about the mean. As the merging proceeds, adjacent small clusters are merged, and the distributions of feature points in the merged clusters become better characterized by Gaussian densities. At the end of the merging, the multimodal structure of the training data is represented by the remaining clusters. The distribution of features can then be modeled by a mixture Gaussian density. The task of the merging algorithm is to estimate a mixture Gaussian density from the initial N Gaussian densities, i.e., to merge the initial N clusters into larger, yet still compact, clusters which approximate the multimodal structure of the training data.

Objective Function

To merge N small clusters into M larger ones, an ideal situation is to partition the N clusters into M subsets, resulting in the minimum dispersion over the M subsets. Let the sample set of the ith cluster be $\Omega_i = \{x_n^{(i)}, n = 1, 2, \cdots, L_i\}$, where $x_n^{(i)}$ is the nth feature sample in the ith subset, and let the entire sample set be $\Omega = \cup_{i=1}^{N} \Omega_i$. The sample sizes of the subsets are $L_i, i = 1, 2, \cdots, N$, and the size of the entire set is $L = \sum_{i=1}^{N} L_i$. Denote the sample mean vector and covariance matrix of Ω_i as μ_i and C_i, respectively, and those of Ω and μ and C, respectively. The histogram probability of the ith cluster is defined as $p_i = L_i/L$. The following relationship holds:

$$C = \sum_{i=1}^{N} p_i C_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} p_i p_j (\mu_i - \mu_j)(\mu_i - \mu_j)'. \quad (3)$$

See Appendix for a derivation of (3).

The first term on the right hand side of (3) is called the within-cluster covariance, and the second term is called the between-cluster covariance. From matrix mathematics, the dispersion of data in an eigenvector direction is measured by the associated eigenvalue. The total dispersion is measured by the trace of the covariance matrix, which is the sum of the eigenvalues, i.e., $\text{trace}(C_i) = \sum_{j=1}^{J} \lambda_j$, where J is the order of the matrix and λ_j's are the eigenvalues. Taking the trace of the matrices in (3), we get

$$\text{trace}(C) = \sum_{i=1}^{N} p_i \,\text{trace}(C_i) + \sum_{i=1}^{N} \sum_{j=i+1}^{N} p_i p_j \|\mu_i - \mu_j\|^2. \quad (4)$$

Note the similarity of the relationship of the traces in (4) to that of the matrices in (3). Equation (4) shows that given a training data set, the total trace is a constant, regardless of how the sample set is partitioned. By minimizing the within-cluster trace, the between-cluster distances are also maximized. The trace of a matrix is also the sum of the diagonal elements of the matrix, where for the covariance matrix C_i:

$$\text{trace}(C_i) = \frac{1}{L_i} \sum_{n=1}^{L_i} \|x_n^{(i)} - \mu_i\|^2.$$

Therefore, the trace of a covariance matrix is equivalent to the average distance of the samples within the cluster from the cluster centroid. The within-cluster trace is called the average trace of the mixture density. Since the average trace measures the compactness of the data clusters, it is used as the objective function in the merging algorithm. For N clusters, the average trace T_N is denoted as

$$T_N = \sum_{i=1}^{N} p_i \,\text{trace}(C_i). \quad (5)$$

Since merging clusters eliminates the related between-cluster distance terms, it increases the average trace. For merging N clusters into M larger, compact clusters, the objective is to minimize the increment of the average trace $\Delta T = T_M - T_N$.

Distance Measure

If a pair of Gaussian densities with nonidentical mean vectors is merged, the trace of the resultant covariance matrix will be strictly greater than the average of the original pair of covariance matrices. Suppose that in (3) the jth and the kth Gaussian densities are merged into a Gaussian density of $\mathcal{N}(\hat{\mu}, \hat{C})$. Let $\tilde{p}_j = L_j/(L_j + L_k)$ and $\tilde{p}_k = L_k/(L_j + L_k)$. Then the mean vector of the merged density is given by

$$\hat{\mu} = \tilde{p}_j \mu_j + \tilde{p}_k \mu_k \quad (6)$$

and its covariance matrix is

$$\hat{C} = \tilde{p}_j C_j + \tilde{p}_k C_k + \tilde{p}_j \tilde{p}_k (\mu_j - \mu_k)(\mu_j - \mu_k)'. \quad (7)$$

The average trace of the resultant $N-1$ covariance matrices is

$$T_{N-1} = \sum_{\substack{i=1 \\ i\neq j,k}}^{N} p_i \operatorname{trace}(C_i) + (p_j + p_k)\operatorname{trace}(\hat{C})$$
$$= \sum_{i=1}^{N} p_i \operatorname{trace}(C_i) + (p_j + p_k)\tilde{p}_j\tilde{p}_k\|\mu_j - \mu_k\|^2.$$

The difference of the average trace after and before merging is, therefore,

$$\Delta T(j,k) = T_{N-1} - T_N$$
$$= (p_j + p_k)\tilde{p}_j\tilde{p}_k\|\mu_j - \mu_k\|^2$$
$$\geq 0. \tag{8}$$

The trace increment of (8) is the squared Euclidean distance of the mean vectors weighted by their respective sample size proportions, as well as the sample size proportion of the pair to the entire set. Since merging the model pair with the smallest weighted distance in (8) minimizes the increment of the average trace, the equation is used as the distance measure for selecting the density pairs in the merging algorithm.

Merging Procedures

As was discussed above, the objective of the merging algorithm is to minimize the increment of the average trace $\Delta T = T_M - T_N$ when N small clusters are merged into M larger, compact clusters. A global minimization of the trace increment, however, requires a combinatorial search and is not practical when N is large. The merging algorithms developed herein are therefore locally optimal. The merging proceeds iteratively, and during each iteration one or more pairs of densities are merged. The merging continues until a termination criterion is met.

When only one pair of densities is merged during each iteration, the pair with the minimum distance is selected. Let (i^*, j^*) be the indices of the Gaussian density pair selected for merging, then

$$(i^*, j^*) = \underset{(i,j)}{\operatorname{argmin}} (p_i + p_j)\tilde{p}_i\tilde{p}_j\|\mu_i - \mu_j\|^2. \tag{9}$$

Since the pair (i^*, j^*) has the minimum distance, the merging results in the minimum trace increment. When multiple pairs of densities are merged during each iteration, where each pair merges into one density, they are selected from the pair with the smallest distance to those with progressively higher distances until an ending condition is met. For example, if the restriction is that only K pairs of densities are merged during an iteration, the set of the indices of the K pairs $\{(i_k^*, j_k^*) : i_k^* \neq j_k^*; k = 1, 2, \cdots, K\}$ is selected so that the sum of K distance terms d^* from this set

$$d^* = \sum_{k=1}^{K} (p_{i_k^*} + p_{j_k^*})\tilde{p}_{i_k^*}\tilde{p}_{j_k^*}\|\mu_{i_k^*} - \mu_{j_k^*}\|^2 \tag{10}$$

is minimum among the sum of K distance terms from all the other possible set of K pairs. A constraint used in the merging of multiple pairs is not to merge any model twice during the same iteration. Based on merging a single pair within an iteration or merging multiple pairs within an iteration, three merging procedures are developed and are discussed below.

1) One-Pair-at-a-Time Merging: The flowchart of this merging procedure is shown in Fig. 3(a). For each iteration, one pair of densities is selected for merging according to (9), and its distance (using (8)) is compared against a termination threshold. The merging terminates when the distance equals or exceeds the threshold. If the distance is below the threshold, the selected pair is merged into one density, and the parameters of the new density are calculated according to (6) and (7). The distance between the new density and each of the existing densities is calculated and the merging proceeds to the next iteration. Since the minimum distance needs to be determined at each iteration, the merging procedure becomes very slow when N is large.

2) K-Pairs-at-a-Time Merging: This merging procedure has two phases, called phase I and phase II. In phase I, the number of densities is above a threshold, and multiple pairs of densities are merged during each iteration. In phase II, the number of remaining densities is below the threshold, and only one pair of densities is merged at an iteration. In this way, the merging proceeds efficiently during phase I, and during phase II the merging is "fine-tuned" and the number of mixture components is determined by the termination distance threshold.

To select the K density pairs to merge within an iteration, the distance terms are sorted into an increasing order, i.e.,

$$d(i_1, j_1) \leq d(i_2, j_2) \leq \cdots \leq d(i_{N'}, j_{N'})$$

where for N densities the list has $N' = N(N-1)/2$ distance terms. Starting from the beginning of the list, the density pairs are merged until K pairs are done. Again, a model is never merged twice during an iteration. Since sorting a distance array can be done very efficiently using library routines such as Quick sort [13] which is $O(N' \log N')$ on the average case, the K-pairs-at-a-time method is significantly faster than the one-pair-at-a-time method when N is large.

The flowchart of the merging procedure is shown in Fig. 3(b). The dotted box contains the flowchart for phase I, where the procedure goes through iterations of K-pair merging. At each iteration, the distances between the existing densities are calculated, sorted, and K pairs of densities are merged. The parameters of the new densities are updated using (6) and (7). Since in merging K pairs of densities, K new densities are generated to replace the $2K$ densities that were merged, each iteration reduces the number of densities by K. The procedure proceeds to phase II when the number of densities is reduced to the threshold number.

Let the threshold number of densities for entering phase II be L. If the initial number of densities is $N = RK + L$, the algorithm will go through R iterations of K-pair merging in phase I, and then proceeds to phase II with L densities remaining. If $R = 0$, the algorithm goes into the second phase directly. In most cases the initial number of models is $N = RK + L + Q$, where $0 < Q < K$. This initial number of N can be handled by either adding Q to L so that $Q + L$ densities are to be merged in phase II and K pairs

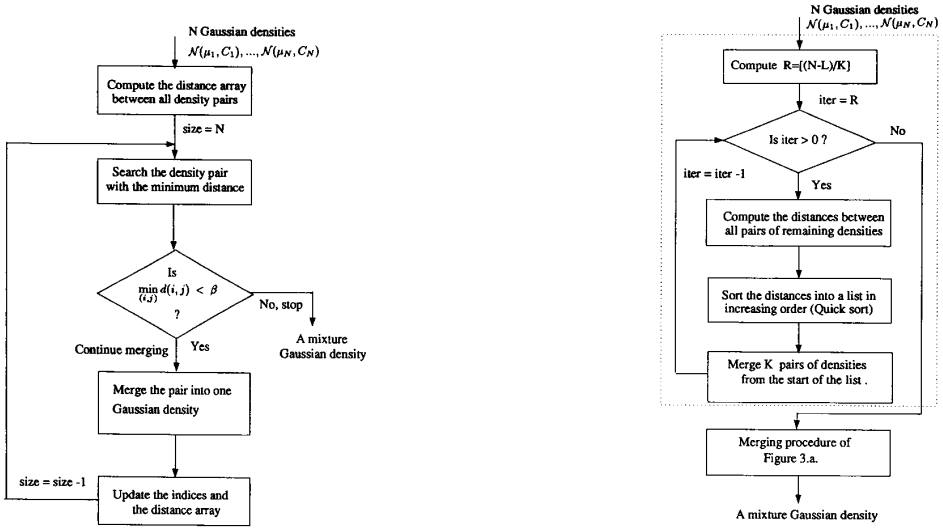

Fig. 3. (a) The merging procedure of one-pair-at-a-time, where β is the termination threshold. (b) The merging procedure of K-pairs-at-a-time, where the dotted block contains the flowchart of phase I, and the bottom block is for phase II with its flowchart in (a). The input densities to phase II are from the output densities of phase I.

of densities are merged in each iteration of phase I; or Q pairs can be merged in the first iteration of phase I. Herein, the latter method has been used.

As is shown in (10), each iteration of merging K density pairs minimizes the increment of the average trace as the sum of K distance terms. To avoid forced merging of density pairs of large distances, the parameter K should not be too large. Furthermore, since merging K pairs of densities requires the existence of $2K$ densities, it is convenient to have $K + L \geq 2K$, or $L > K$. The parameters L, K, and R are heuristic parameters used to balance the multiple-pair and the single-pair merging, and they can be adjusted in variety of ways.

3) Variable-Pairs-at-a-Time Merging: Fixing the number K of multiple-pair merging poses the problem of how to choose K for different sample sizes, which is often the case for training models of different phones where their sample sizes vary widely. For large N, a large K is preferable for fast merging; whereas for small N, the merging speed is not a problem and a small K might be desirable to avoid the forced merging of density pairs with large distances. To circumvent the problem of pre-choosing K, a merging procedure is developed which is similar to the K-pairs-at-a-time merging but uses a distance threshold to control the multiple-pair merging. In phase I, the distances of density pairs are compared against a threshold: those with distances below the threshold are merged, and those with distances above the threshold remain intact. Again, using a high threshold allows many density pairs to be merged during each iteration, and using a low threshold requires more iterations but pairs with large distances are avoided in each iteration. For practical implementation, it is feasible to choose the threshold of multiple-pair merging as a fraction $\alpha(0 < \alpha < 1)$ of the termination threshold of phase II. Appropriate choices of α are given in experiments of Section VI.

Termination Threshold

The termination of the merging process is controlled by the distance threshold. Since the multimodal structure of the training data is to be found through the bottom-up merging, the termination threshold controls the resolution of the data clusters. A large threshold allows clusters with large distances to be merged, and a small threshold prohibits the merging of clusters with large distances. Since each cluster is modeled by a Gaussian density, a high threshold leads to a mixture density with small number of components, and a low threshold leads to a mixture density with large number of components.

When merging the Gaussian densities of each state of each phone model, a termination threshold of β/L is experimentally shown feasible. The parameter β is the same for all states of all phone models, and L is the sample size of a specific state of a specific phone model. The comparison of the distance against the threshold

$$(p_i + p_j)\tilde{p}_i\tilde{p}_j\|\mu_i - \mu_j\|^2 < \beta/L$$

is equivalent to a comparison of

$$(L_i + L_j)\tilde{p}_i\tilde{p}_j\|\mu_i - \mu_j\|^2 < \beta$$

where L_i and L_j are the sample sizes of the ith and the jth densities, respectively. Since the distance term on the left-hand side is proportional to both the sample size and the distance between the mean vectors of the density pair, the resulting mixture size is also proportional to both factors. A quantitative analysis of the relationship among the mixture size, the data sample size, as well as data dispersion is given in Section VI.

B. Distance Weighting

Since a feature vector consists of components of different scales, the distance between two mean vectors needs to be properly weighted for each component. For this purpose, the average trace is modified as follows

$$T_N = \sum_{i=1}^{N} p_i \text{trace}(AC_i)$$

where A is a diagonal matrix,

$$A = \text{diag}(a_{s,1}, \cdots, a_{s,L+1}, a_{d,1}, \cdots, a_{d,L+1})$$

and each diagonal element of the matrix is a weight for the corresponding feature component.

The ratio of feature scales varies among phone units. For example, the absolute value of log energy $|p|$ is generally much larger for stops than for vowels. The weight matrix A is therefore chosen as phone unit specific. Let the total number of frames for training a phone unit be K and the analysis order be L. The scales of the weighted cepstrum coefficients $(c_1, \cdots, c_L)$, log energy p, regression of the weighted cepstrum coefficients $(\Delta c_1, \cdots, \Delta c_L)$, and regression of the log energy Δp, are calculated as r_1, r_2, r_3, and r_4, respectively. The scales are defined as the norm-squared of the features averaged over the sample size, i.e.,

$$r_1 = \frac{1}{KL} \sum_{i=1}^{L} \sum_{k=1}^{K} c_{i,k}^2;$$

$$r_2 = \frac{1}{K} \sum_{i=1}^{K} p_k^2,$$

$$r_3 = \frac{1}{KL} \sum_{i=1}^{L} \sum_{k=1}^{K} (\Delta c_{i,k})^2;$$

$$r_4 = \frac{1}{K} \sum_{i=1}^{K} (\Delta p_k)^2.$$

Note that r_1 and r_3 are the component-wise average scales.

The diagonal elements of A are inversely proportional to the scales of the features. A degree of freedom is given to the relative weighting between the instantaneous and dynamic features by using a weighting factor w_d. Define S as the common divisor,

$$S = \frac{1}{r_1} + \frac{1}{r_2} + \frac{1}{w_d}\left(\frac{1}{r_3} + \frac{1}{r_4}\right).$$

The elements of A are then defined as

$$a_{s,\cdot} = a_{s,i} = \frac{1}{r_1} S^{-1}, \qquad i = 1, 2, \cdots, L$$

$$a_{s,L+1} = \frac{1}{r_2} S^{-1}$$

$$a_{d,\cdot} = a_{d,i} = \frac{1}{w_d r_3} S^{-1}, \qquad i = 1, 2, \cdots, L$$

$$a_{d,L+1} = \frac{1}{w_d r_4} S^{-1}.$$

The weight $w_d = 2$ was experimentally determined as appropriate for good decoding performance. The weighted distance is a convex combination of the four distance terms of $(c_1, \cdots, c_L), p, (\Delta c_1, \cdots, \Delta c_L)$, and Δp. Note, however, that the weighted distance as a whole still depends on the data scale. Because the scales of data vary with phones, the distance threshold in the merging procedure should also be made phone dependent. This can be achieved by multiplying the threshold β by S^{-1} for each phone. In this case, each distance term will be normalized by its scale, and, therefore, the distances become scale independent.

C. Mixture Weight Estimation

The weighting coefficients of the mixture densities, the a_i's of (1), are estimated based on a simple procedure which combines maximum likelihood estimation with the sentence segmentation using the Viterbi algorithm. The sentences are segmented into phonetic segments, which are further divided into sub-segments corresponding to states of phone units. The likelihood scores of feature data with respect to the mixture components in the corresponding sub-segments are computed. Let the frame sample size of a mixture density be K and the number of mixture components be M. The objective function for weight estimation is defined as a sum of the log likelihood for each frame:

$$J = \sum_{k=1}^{K} \log \left(\sum_{i=1}^{M} \alpha_i f_i(x_k, \Delta x_k) \right) + \lambda \left(\sum_{i=1}^{M} \alpha_i \right)$$

where λ is the Lagrangian multiplier. Taking the partial derivative of J with respect to α_i and setting it to zero, i.e., $\partial J / \partial \alpha_i = 0$, gives

$$\sum_{k=1}^{K} \frac{f_i(x_k, \Delta x_k)}{\sum_{i=1}^{M} \alpha_i f_i(x_k, \Delta x_k)} = -\lambda. \tag{11}$$

Multiplying (11) by α_i, summing over i, and solving for λ yields $\lambda = -K$. The maximum likelihood estimate α_i^* satisfies

$$1 = \frac{1}{K} \sum_{k=1}^{K} \frac{f_i(x_k, \Delta x_k)}{\sum_{i=1}^{M} \alpha_i^* f_i(x_k, \Delta x_k)}.$$

Since the equation is nonlinear for α_i^*, an iterative reestimation is taken as

$$\alpha_i^{(n+1)} = \frac{1}{K} \sum_{k=1}^{K} \frac{\alpha_i^{(n)} f_i(x_k, \Delta x_k)}{\sum_{i=1}^{M} \alpha_i^{(n)} f_i(x_k, \Delta x_k)}. \tag{12}$$

Equation (12) has a simple interpretation: the weight of a mixture component is simply the average of the frame-based ratio of the likelihood score evaluated using the component density to the likelihood score evaluated using the mixture density. The initial values of the $\alpha_i^{(0)}$'s can be simply set as uniform, and in experiments one iteration has proven sufficient to give good estimates of the α_i's.

The weights of mixture Gaussian densities have been shown effective in context-dependent modeling of HMM phone units [2]. This approach is very useful when the amount of training

data is small. Generally speaking, estimating the weights of mixture components requires less training data than estimating the mean vectors and covariance matrices. Therefore the mixture weights are estimated for each tri-phone context. Equation (12) is used to estimate the context-dependent mixture weights, where the average is taken over the frames belonging to a state of a phone unit within a specific context environment.

D. Model Smoothing

In estimating the HMM phone model parameters from a limited amount of training data, there exists a tradeoff between the amount of detail and the robustness. Detail is important for the models to distinguish the fine differences in acoustic signals which are of linguistic significance; while robustness is important for the models to cope with mere statistical variations in the signals. Model smoothing has proven useful for compromising the two demands, where the parameters from the robust, non-detailed models are used to smooth the parameters from the detailed, non-robust models. The smoothed detailed models, in general, give better decoding performance on test sets than the detailed models alone.

For the models trained with the merging procedures in Section III-A, the tradeoff between the amount of detail and robustness is controlled by the termination threshold. Since the training sample size is limited, a high resolution in data clusters results in small sample sizes in some clusters. Because a high-order covariance matrix requires a large amount of data to be estimated reliably, a small sample size tends to yield an unreliable estimate. Furthermore, the mixture weights are estimated with respect to each tri-phone context. Since there are usually only a few training samples for each context, the mixture weights also tend to be undertrained. To improve the robustness of the HMM phone models while also maintaining a good level of detail, smoothings are performed on both the covariance matrices and the mixture weights, and the mixture weights are estimated after the covariance matrices are smoothed.

Covariance smoothing

An unreliable estimate of the covariance matrix typically has some eigenvalues close to zero. The diminishing eigenvalues lead to a Gaussian density with a very sharp peak. The reason is that the magnitude of the peak of a Gaussian density is inversely proportional to the square root magnitude of the determinant of the covariance matrix; and the determinant is simply the product of the eigenvalues. A Gaussian density with a very sharp peak gives very low likelihood scores to feature points which are only slightly deviated from the mean, and hence the model robustness is low. In the degenerate case where a covariance matrix is singular, the Gaussian density becomes a delta function which is identically zero except at the mean point. In this case, only the feature samples which coincide with the mean (if any) have a nonzero-likelihood value, and any deviation from the mean will lead to a zero likelihood.

To quantify the sharpness of a mixture component, the ratio of the peak value of the component density to the average peak value of the mixture density is used. A ratio measure is desirable since the peak value of a Gaussian density is scale dependent, due to the scale-dependency of the determinant. Let there be M components in a mixture density with the covariance matrices $C_i, i = 1, 2, \cdots, M$. The relative sharpness of the ith component R_i is calculated as

$$R_i = \frac{1/|C_i|^{1/2}}{1/\left(\prod_{k=1}^{M} |C_k|^{1/2}\right)^{1/M}}.$$

If R_i is above a threshold, the density is identified as being too sharp.

On the robustness extreme, a unimodal Gaussian density is estimated for each state of a phone, which can be obtained by merging all the mixture components into one Gaussian density. Let the covariance matrix of the unimodal density be C. Then a smoothing is performed on C_i if R_i is above the threshold. The smoothing is defined as a linear interpolation of the detailed estimate C_i and the robust estimate C. Let the smoothed covariance matrix be $\hat{C}_i$, then

$$\hat{C}_i = \lambda C_i + (1 - \lambda)C$$

where $0 \leq \lambda \leq 1$ is the interpolation parameter. In general, if C_i is estimated from many samples, a large λ can be used so that the smoothed estimate will contain more information from the detailed estimate; if the sample size is small, a small λ is desirable so that the smoothed estimate will contain more information from the robust estimate. Since the structure of the covariance matrices is separated into two blocks, one for instantaneous features and one for dynamic features, each block can be smoothed separately. The interpolation parameter λ is empirically determined from experiments. Results from smoothing the covariance matrices are given in Section VI.

Mixture-weight smoothing

Context-independent mixture weights are at the robust extreme of the mixture weight estimates. The context-dependent mixture weights are therefore smoothed by the context-independent mixture weights. Let $\alpha_{m,i}$ be the weight of the ith mixture component for a context m, and let α_i be the corresponding context-independent weight. Then the smoothed weight $\hat{\alpha}_{m,i}$ is defined as

$$\hat{\alpha}_{m,i} = \lambda \alpha_{m,i} + (1 - \lambda)\alpha_i.$$

The interpolation parameter λ is also determined empirically from experiments. The interpolation can also use a hierarchy of weights at varying levels of robustness and detail, such as the mixture weights corresponding to different degrees of context dependency. Specific interpolation values for weight smoothing related to decoding experiments are given in Section VI.

To further improve the robustness of the HMM phone models, the unimodal density is also added to the mixture density as an extra component. Informal evaluations of decoding accuracy showed improved performance when including the unimodal density, compared to using the original mixture components alone.

IV. Dictionary Preparation

It is straightforward to prepare a word transcription dictionary from some standard references (standard dictionaries), and such a dictionary generally gives a single transcription for a lexicon entry. Speaker-independent continuous speech recognition systems have shown satisfactory results from using dictionaries prepared in this way, e.g., see [1], [2].

Since the TIMIT database has a complete acoustic–phonetic labeling of sentences, a dictionary can be constructed by extracting the word transcriptions from these labels. However, using the labels directly from the database generally yields many transcriptions for a single lexicon entry. The different transcriptions of words could reflect differences in dialect, context, as well as some possible errors in labeling [14]. For sentence decoding, it is desirable to compress the dictionary into a more compact one through an automatic procedure. An iterative procedure based on a measure of log-likelihood ratio was developed to compress the dictionary, where in each iteration the transcription which causes the least decrease of the joint word likelihood is eliminated.

Let there be N word samples of a lexicon entry, and let there be K distinct transcriptions labeled for the N words. The likelihood of each word is evaluated under each transcription using the HMM phone models concatenated according to the transcription. For example, for the word w_n, the likelihood scores are evaluated as $p(w_n|T_k)$, $k = 1, 2, \cdots, K$, where T_k denotes the kth transcription. The two-best transcriptions $b(n)$ and $s(n)$ for a word w_n are defined as those that yield the highest and the next highest likelihood scores, respectively, i.e.,

$$b(n) = \underset{k=1,\cdots,K}{\operatorname{argmax}}\, p(w_n|T_k)$$

and

$$s(n) = \underset{\substack{k=1,\cdots,K \\ k\neq b(n)}}{\operatorname{argmax}}\, p(w_n|T_k).$$

Any transcription which is not the best for any word is immediately eliminated. The word samples are partitioned into groups according to their best transcriptions.

Let the kth group labeled with the transcription T_k have N_k word samples. By definition any word w_n in the group has its best transcription index as $b(n) = k$. The cost of eliminating a transcription is defined as the difference of the maximum joint word likelihood before and after the elimination of the transcription. Let the maximum joint word likelihood be denoted as ML, then it is calculated as

$$ML = \sum_{n=1}^{N} \frac{1}{L_n} \log p(w_n|T_{b(n)})$$

where L_n is the number of frames for the word w_n, and it is used for normalizing the effect of the number of frames on likelihood scores of word samples. Let ML_k denote the maximum joint word likelihood after the elimination of the transcription T_k, which is

$$ML_k = \sum_{n:b(n)\neq k} \frac{1}{L_n} \log p(w_n|T_{b(n)}) + \sum_{n:b(n)=k} \frac{1}{L_n} \log p(w_n|T_{s(n)}).$$

The cost $C(k)$ of eliminating the transcription T_k is defined as the difference of the maximum joint word likelihood before and after the elimination, i.e.,

$$C(k) = ML - ML_k = \sum_{n:b(n)=T_k} \frac{1}{L_n} \left\{ \log \frac{p(w_n|T_{b(n)})}{p(w_n|T_{s(n)})} \right\}. \quad (13)$$

The cost is, therefore, the sum of log-likelihood ratio of each word under the best transcription to that under the second best one in the group. Equation (13) has the property that the cost of eliminating a transcription representing a large group has more cost terms than that representing a small group. The index of the transcription to be eliminated, k^*, is the one incurring the minimum cost, i.e.,

$$k^* = \underset{k}{\operatorname{argmin}}\, C(k). \quad (14)$$

After a transcription is eliminated, each word in that group is assigned to a remaining group according to its second best transcription. The elimination continues until the cost exceeds a threshold and the number of transcriptions falls below a limit.

To make the number of transcriptions of a lexicon entry relatively independent of the number of training word samples, the termination threshold is chosen as γN, where γ is a common factor for the whole lexicon, and N is the word sample size of a specific lexicon entry. Define the histogram probability of a transcription T_k by $p_k = N_k/N$, and define the normalized cost of $C(k)$ by $\tilde{C}(k) = (1/N_k)C(k)$. Then (14) is equivalent to

$$k^* = \underset{k}{\operatorname{argmin}}\, p_k \tilde{C}(k)$$

and the comparison of the cost with the termination threshold becomes $p_{k^*}\tilde{C}(k^*) < \gamma$.

To maintain a compact dictionary, the termination threshold is chosen such that the majority of lexicon entries have a single transcription, and the rest have multiple transcriptions with an upper limit of three. It was found experimentally that most transcriptions thus generated are satisfactory, yet some still need to be corrected manually.

V. Grammar

A word-pair grammar has been used in continuous speech recognition systems for evaluating the system performance at a controlled level of task difficulty [11]. For a task-oriented continuous speech database such as the DARPA Resource Management Database, the perplexity of a word-pair grammar is not very high (about 60). The TIMIT database is not designed as a task-oriented database, and the perplexity of a word-pair grammar is substantially high.

To obtain flexible control over the grammar perplexity, the word-pair grammar is modified such that context-sensitive grammatical parts are defined to smooth the transitions between words instead of using the grammatical parts directly. Specifically, consider a two-sided first-order context dependency. Let the center grammatical part be a, its left neighbor b, and its right neighbor c, then a context-sensitive grammatical part is defined as $g_{(b,a,c)}$. A word w_j is labeled with the grammatical part $g_{(b,a,c)}$ if w_j has occurred in a text where a is its grammatical part, b is the grammatical part of the preceding word, and c is the grammatical part of the following word. Similarly, a word w_k is labeled with the context-sensitive grammatical part $g_{(b',a',c')}$, if w_k has occurred in a text where a' is its grammatical part, b' is the grammatical part of the preceding word, and c' is the grammatical part of the following word. The transition from the word w_j to the word w_k is bridged by the transition of their corresponding context-sensitive grammatical parts. The probability of transition from w_j to w_k, denoted as $p(w_j \rightarrow w_k)$, is calculated as

$$p(w_j \rightarrow w_k) = \sum_{g(b,a,c)} \sum_{g(b',a',c')} p(g_{(b,a,c)}|w_j)p(g_{(b,a,c)} \rightarrow g_{(b',a',c')}) \cdot p(w_k|g_{(b',a',c')}).$$

If this probability term is nonzero, the word w_j can be followed by w_k. After the list of valid followers is determined, the transition probabilities from w_j to all of its followers are assigned to an equal value. Grammars at different levels of perplexity can be obtained by varying the degree of context dependency of the grammatical parts. The higher the degree of context dependency is, the lower the perplexity will be. Although the modified word-pair grammars thus generated cannot function as the grammars generated from a natural language modeling point of view (e.g., bigram and trigram grammar [15], context free grammar [16], etc.), the flexibility of generating the grammars at various levels of perplexity from a limited training text is useful for evaluating the decoding accuracy of a speech recognition system at controlled levels of task difficulties.

VI. Experiments

A. Training and Test Sets

The system was initially both trained and evaluated on the prototype version (1988) of the TIMIT database. Experiments were performed on the SX3-SX200 sentences. Since the 1988 release has only the training set, the speakers were partitioned into a training set and a test set (test set I). The HMM models trained from this split training set were later used in evaluating the system performance on the standard test set (test set II) of the TIMIT database (1990) [7]. Some relevant information on the training set and the two test sets are summarized in Table I.

TABLE I
Training and Test Sets†

	Training set	Test set I	Test set II
No. of speakers	325	95	75
dialect coverage	8	8	8
No. of male	224	66	50
No. of female	101	29	25

†Subsets of the TIMIT database, vocabulary size of 853.

B. Feature Analysis Front-end

The speech signals were down-sampled from 16 to 10.67 kHz. Cepstrum coefficients were computed using PLP analysis of an order 8, 200 samples per frame weighted by a Hamming window, and 100 samples per step. The regression coefficients were computed over an interval of approximately 50 ms. Including log-energy and its regression coefficients as components, the feature dimension is 18.

C. Dictionary Preparation

The word transcriptions were first extracted from the training sentences, where the word boundaries were manually marked according to the acoustic–phonetic labels in the database (the prototype TIMIT database does not contain word boundary files). A total of 2519 transcriptions were extracted for a total of 4865 words. The phone models trained in the early experiments [4] were used as the initial phone models in the dictionary compression. Using a compression threshold of $\gamma = 0.2$, a compacted dictionary is obtained which has 983 transcriptions for the 853 words. This dictionary is used as a default one in the following evaluations of decoding performance.

The 1990 release of the TIMIT database also comes with a phonemic dictionary for the whole vocabulary. A standard phonetic dictionary was generated from the phonemic dictionary by mapping the phonemes to their respective acoustic–phonetic units using some of the rules suggested by NIST. Specifically, the stops /b/, /d/, /g/, /p/, /t/, /k/ are mapped to their respective two unit representation consisting of a closure and a release, e.g., /b/ → /bcl/ /b/; the affricate /jh/ and /ch/ are also mapped to their two unit forms, i.e., /jh/ → /dcl/ /jh/, and /ch/ → /tcl/ /ch/. Using the standard dictionary and the same initial phone models, the sentences were resegmented using the Viterbi algorithm, and the phone models are then reestimated through model merging. The standard dictionary was used as a baseline for evaluating the quality of the dictionaries prepared from the compression procedure.

D. Model Training

Analysis of Phone Units. Using the default dictionary and the initial phone models, the Viterbi algorithm was used to resegment the training sentences. The number of acoustic-phonetic units in the training sentences after the resegmentation are listed in Table II. The symbol /sil/ denotes the union of /epi/ and /pau/, and it was trained as an HMM phone model for modeling the sentence initial and end silence, as well as the between-word pauses. The /h#/ was discarded and counted as 0. The frequency of occurrence of different phones varies significantly from the highest of /ix/ to the lowest of /em/.

Analysis of Sub-Segments. As was discussed in Section III-A on model training, the sub-segments of each phone unit

TABLE II
NUMBER OF EACH PHONETIC SEGMENT IN THE TRAINING SET

aa	406	epi	134	ow	296
ae	493	er	299	oy	39
ah	361	ey	399	pau	84
ao	377	f	341	p	545
aw	93	g	231	pcl	562
ax	690	gcl	230	q	690
ax-h	41	h#	0	r	822
axr	538	hh	211	s	1184
ay	330	hv	37	sh	213
b	310	ih	587	t	699
bcl	283	ix	1467	tcl	1005
ch	108	iy	843	th	94
d	419	j	0	uh	42
dcl	640	jh	183	uw	98
dh	330	k	825	ux	231
dx	325	kcl	908	v	320
eh	486	l	844	w	442
el	206	m	648	y	245
em	12	n	1102	z	738
en	100	ng	188	zh	17
eng	0	nx	75	sil	218

form the initial small clusters from which and through iterative merging a mixture density is generated. Since the mean vector and covariance matrix are initially estimated for each sub-segment, it is relevant to investigate the sizes of the sub-segments. The sizes of the sub-segments vary with the phone classes. For vowel-type phones, the phone durations are long and so are the sub-segment sizes; and vice versa for the burst-type phones. The sizes for the vowel /aa/ and the burst /d/ are shown in the histograms in Fig. 4(a) and 4(b), respectively. As seen from the histograms, there is also a variation in sizes of the beginning and center sub-segments (the sizes of ending sub-segments are similar to those of the beginning sub-segments). For /aa/, the center sub-segments have large sizes and the beginning sub-segments have smaller sizes. For /d/, the sizes are small for both the beginning and center sub-segments. In Fig. 4(b), the zero count on frames represents the case when the center state was skipped. The difference of sizes between /aa/ and /d/ is reasonable, since in general, the vowels are stationary and the bursts are transient.

Since the sample sizes of many sub-segments are small, the estimates of Gaussian density parameters are only crude and convenient characterizations of the small clusters formed by the features in each sub-segment. However, since only the mean vectors of the Gaussian densities are used in distance calculation as in (8), the small sample sizes in parameter estimates do not cause problems in model merging.

Analysis of Mixture Size. As was discussed in model training, the number of mixture components generated through merging is proportional to both the sample size and the dispersion of the training data. A study is made on the relation among the three variables by a mean-squared error (MSE) fitting on the merging results. Let there be a total of M mixture densities for the phone models. Let the vectors of the mixture size be m, data sample size be l, and data dispersion be d, where the ith components of the vectors are for the ith mixture density. Let the initial number of densities be N_i for the ith mixture density. The dispersion of training data of each mixture density is defined as an average of the between-cluster mean distances,

$$d_i = \frac{1}{N_i(N_i-1)} \sum_{j=0}^{N_i} \sum_{\substack{k=0 \\ k\neq j}}^{N_i} \|\mu_j - \mu_k\|^2.$$

An MSE fitting of the relation $m = f(l,d)$ is in the nonlinear form

$$m = \alpha l + \beta d + \lambda q$$

where

$$m = [m_1, m_2, \cdots, m_M]'$$
$$l = [l_1, l_2, \cdots, l_M]'$$
$$d = [d_1, d_2, \cdots, d_M]'$$
$$q = [l_1 d_1, l_2 d_2, \cdots, l_M d_M]'.$$

Define $P = [\alpha, \beta, \gamma]'$ and $A = [l, d, q]$, the parameter estimate $\hat{P}$, from MSE fitting is $\hat{P} = (A'A)^{-1}A'm$ and the total fitting squared error is $e^2 = (m - A\hat{P})'(m - A\hat{P})$.

A fitting is made for the K-pairs-at-a-time merging with $K = L = 100$, and a termination threshold of $\beta = 4.3$. The parameter estimate is $\hat{P} = [0.0051, 23.2061, 0.0963]'$. A contour plot is shown in Fig. 5 to illustrate the estimated relationship. The figure shows that the mixture size is strongly proportional to both the sample size and the dispersion of the training data.

For this special case of merging, the average number of mixture components per mixture density is 19.2 and the average estimation error per mixture density is 2.9. The mixture density with the maximum mixture size (=60) occurred in /s/, and those with the minimum size (=1) occurred in /ax-h/, /em/, and /zh/.

Model Smoothing. As was discussed in Section III-D, the covariance matrices of the instantaneous and dynamic features are separately smoothed. A few smoothing methods are investigated. The default method is one where the covariance matrices of the dynamic features are selectively smoothed, and those of the instantaneous features are uniformly smoothed. The sharpness ratio threshold is $R_i = 100$, and the interpolation parameter is $\lambda = 0.5$. As will be shown in the decoding evaluation, this smoothing method gives a better decoding result than the other experimented methods. The mixture weights are estimated using Eqn. (12). The two-sided context-dependent weights are interpolated by the left-context-dependent weights, the right-context-dependent weights, and the uniform weights with the respective interpolation parameters $\lambda_{l,r}, \lambda_l, \lambda_r, \lambda_u$, where $\lambda_{l,r} + \lambda_l + \lambda_r + \lambda_u = 1$. The default parameters are $\lambda_{l,r} = 0.5, \lambda_l = 0.2, \lambda_r = 0.2, \lambda_u = 0.1$.

E. Word Duration Model

The durations of words are modeled by Gaussian densities, where the sample mean and variance are calculated for each

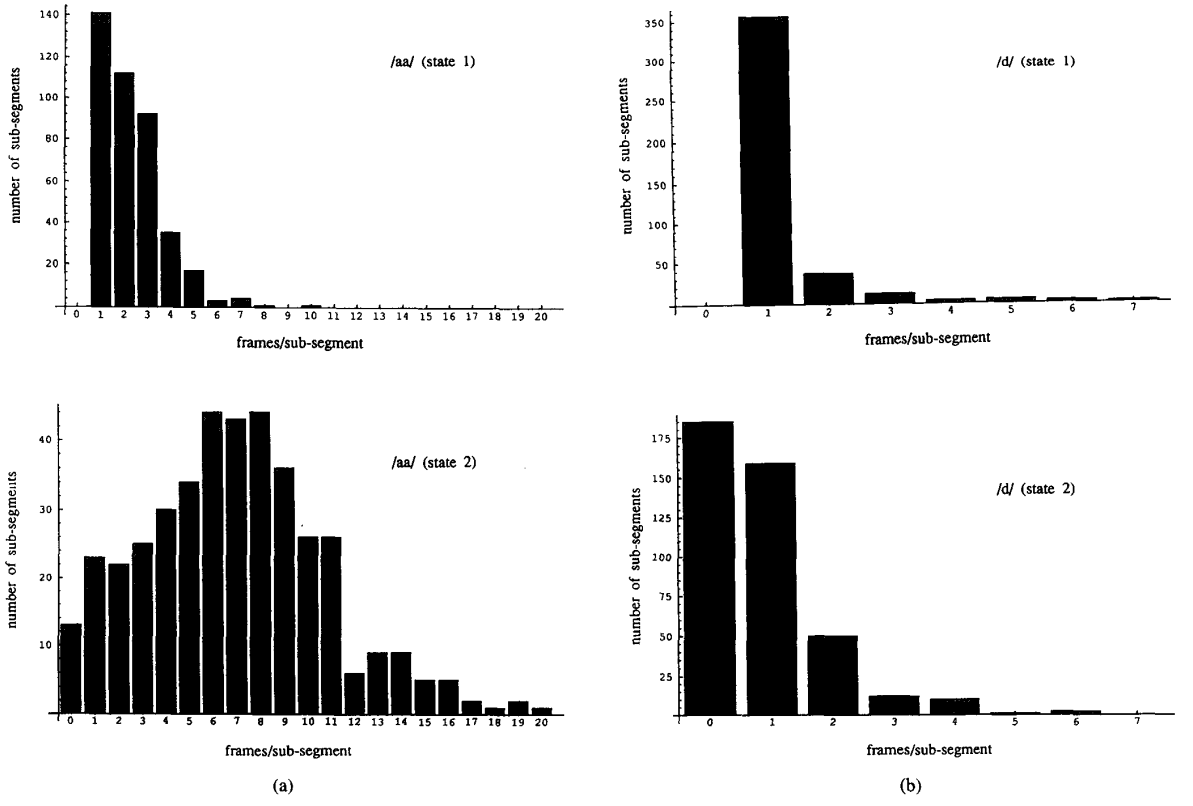

Fig. 4. (a) The histogram of the sub-segment sizes of /aa/. (b) The histogram of the sub-segment sizes of /d/.

duration model. Since very few samples are available for estimating parameters of each duration model, for each word $w(i)$, the quantities $2 \times \min(\mathrm{dur}(w(i)))$ and $1/2 \times \max(\mathrm{dur}(w(i)))$ were used to supplement the duration data. Word penalties were also used for reducing insertion errors.

F. Grammar

The sentences were labeled by 19 grammatical parts. Two grammars, grammar-I and grammar-II, are generated. Grammar-I uses grammatical parts with a first-order dependency on left-right neighbors, and grammar-II with the first-order dependency on the left neighbor only. Grammar-II is used as the default grammar in the decoding evaluations. For the first and second grammars, the test set perplexities on test set I are 25 and 106, respectively, and on test set II are 24 and 104, respectively. Experiments were also performed with a null grammar which has a perplexity of 853.

G. Baseline Comparison

To enable better evaluations of the quality of the model-merging algorithm developed in this work, the segmental K-means algorithm is implemented as a baseline for comparison. The segmental K-means algorithm has been used in training models of words and phones for isolated word and continuous speech recognition. The algorithm uses the Viterbi algorithm to segment words or sentences into the desired units. Units corresponding to the same state of the same HMM model are tied together, and a VQ-based clustering is performed on the frame-based features within each state of HMM models [17]. The K-means clustering requires the number of clusters to be specified *a priori*. Starting from the given initial centroids, the algorithm goes through iterations of nearest neighbor assignments and centroid updates. The number of clusters has been determined as either fixed for all models [17], or chosen to vary in proportion to the sample size of training data [2]. The former is adopted mostly in isolated word model training where the numbers of training word samples are often the same for all words, and the latter is used in training phone models for continuous speech recognition where the numbers of training samples usually vary significantly for different phones.

In the baseline implementation of the segmental K-means algorithm, the number of clusters is proportional to the training sample size. Let the total number of feature samples in the training set be L, and the total number of mixture components from all the mixture densities be M. Then the average number of feature samples per mixture component is determined as $\tau = L/M$. Let the number of feature samples in a state of a phone model be L_i, then the number of clusters M_i for the state is determined as $M_i = L_i/\tau$. The initial centroids are taken as every other τ feature samples in the sequence of training data. A further investigation of the performance difference between the two algorithms is made by fixing the number of clusters in the segmental K-means to be the same

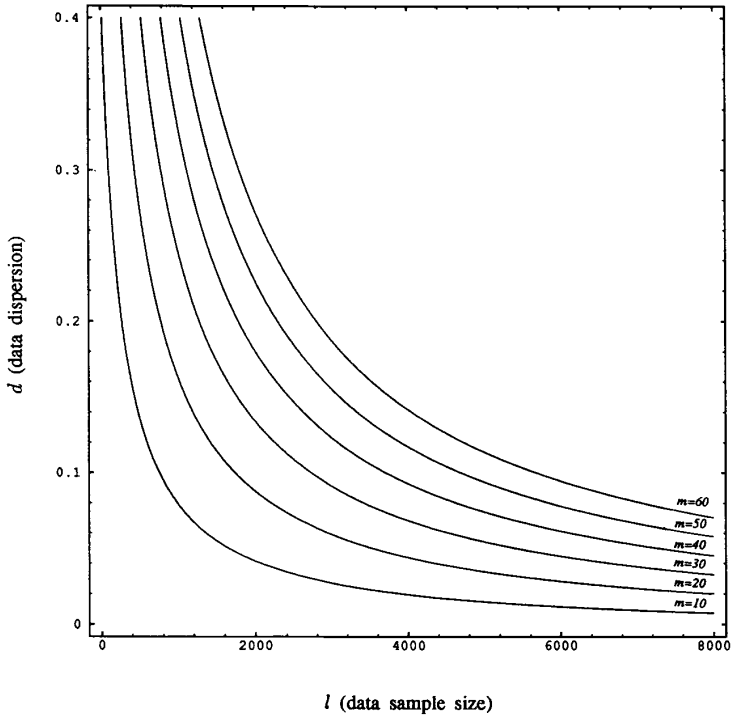

Fig. 5. The contour plot of the MSE fitting $m = f(l, d)$, where m, l, and d are mixture size, data sample size, and data dispersion, respectively.

as the merged mixture density for each state of each phone model. Performance Comparisons on the two model-training algorithms are provided in the next subsection of decoding results.

It is to be noted that the segmental K-means algorithm as is implemented here is based on its general principle. Improvements on the basic technique might be achieved if certain clusters are split after the K-means clustering [2]. Consequently, the comparison of the two training methods serves only as a guideline to evaluate the quality of the merging algorithm, and no attempt is made to compare different recognition systems.

H. Decoding Evaluation

The decoding performances of the recognition system are evaluated under various conditions. Specifically, the variations of performance with the model merging schemes, the termination threshold of merging, smoothing, dictionary compression, and grammar perplexities are investigated. Performance comparison is also made between the merging algorithm and the baseline of the segmental K-means algorithm. The test set II is used as the default test set for all the evaluation experiments. The default dictionary, grammar, and smoothing methods are as discussed in the above subsections unless otherwise specified. The results shown in Tables III–IX are for a task with a vocabulary size of 853 and a test set perplexity of 104.

The decoding rates are scored using the standard routine provided by NIST. The words that share common transcriptions from the dictionary merging are counted as homophones. There are seven standard homophones from 14 words, additionally there are six nonstandard homophones from 13 words which become homophones due to reduced pronunciations in the continuous speech environment.

1) Merging Techniques: Comparisons are first made to the three merging methods of one-pair-at-a-time, K-pairs-at-a-time, and variable-pairs-at-a-time, with a fixed termination threshold of $\beta = 4.3$. The decoding rates are listed in Table III, where the total mixture size of each case is also listed. As seen from the table, the K-pairs-at-a-time method with $K = L = 100$ and variable-pairs-at-a-time method with $\alpha = 0.4, 0.5, 0.6$ and $L = 200$ all give similar results better than the other choices.

As is shown in Table III, the mixture sizes vary slightly with the merging schemes. To exclude the effects of mixture size variation on the decoding performance, another experiment is performed where the mixture size of each mixture density is fixed to be the same as a reference for all the merging schemes. The one-pair-at-a-time merging is used as a reference, with the termination threshold of $\beta = 4.3$. The decoding rates from using various merging schemes are shown in Table IV. The performance from the segmental K-means is also included in the table, where the number of clusters of each mixture density is initialized to be the same as the reference. Table IV indicates the variation of decoding performances among

TABLE III
COMPARISON OF DECODING RATES FROM VARIOUS MERGING SCHEMES

Merging method	One-pair	K-pair $K = L = 200$	K-pair $K = L = 100$	K-pair $K = 50$ $L = 100$	Var-pair $L = 200$ $\alpha = 0.4$	Var-pair $L = 200$ $\alpha = 0.5$	Var-pair $L = 200$ $\alpha = 0.6$
Word correct	87.0%	87.5%	88.6%	87.3%	88.5%	88.7%	88.7%
Word accuracy	84.2%	84.9%	86.0%	85.5%	86.1%	86.1%	85.9%
Total mixtures	3494	3464	3456	3357	3489	3473	3475

The termination threshold of merging is fixed at $\beta = 4.3$.

TABLE IV
COMPARISON OF DECODING RATES FROM VARIOUS MERGING SCHEMES

Merging method	One-pair	K-pair $K = L = 200$	K-pair $K = L = 100$	K-pair $K = 50$ $L = 100$	Var-pair $L = 200$ $\alpha = 0.5$	K-means
Word correct	87.0%	87.8%	88.4%	88.1%	88.3%	84.2%
Word accuracy	84.2%	85.0%	86.1%	86.0%	85.5%	82.3%

The mixture size of each mixture density is fixed to be the same as the mixture size of the one-pair-at-a-time merging (the termination threshold is $\beta = 4.3$). The result of the segmental K-means is also included.

the merging methods. As shown in the table, the multiple-pair merging methods give higher decoding accuracy than the one-pair-at-a-time merging method. The K-pairs-at-a-time with $K = L = 100$ gives the best decoding rate, and the larger value of $K = 200$ decreases the decoding accuracy by about 1%. Note that the segmental K-means gives much lower decoding rate than any of the merging methods.

The variation of decoding accuracies among the merging schemes with the chosen parameters is smaller than the difference of decoding accuracies between the mergings and the K-means. In the merging methods, the models trained from the most compact data clusters (given a number of clusters) could give the best decoding accuracy on the training set, but they do not guarantee the best results on the test sets. Furthermore, both the distribution assumption of mixture Gaussian density and the smoothings applied to the estimated model parameters constitute complicated factors in the performance variations. In general, the multiple-pair merging methods are successful; they not only make the training on large sample set fast, but they also maintain or improve the decoding accuracy.

2) Termination Threshold: The termination threshold of merging is varied to investigate the variation of decoding rates with the mixture sizes of the HMM phone models. The merging method of the K-pairs-at-a-time with $K = L = 100$ is used in the evaluation. The thresholds are varied from $\beta = 3.7$ to 5.2, with an incremental step of 0.3. The maximum mixture size is limited to 60. As is shown in Table V, the total mixture size varies from 2832 to 4014, and the best decoding rate is achieved with the threshold of $\beta = 4.3$. The further decrease of the termination threshold (increase of mixture size) did not lead to the further improvement of the decoding accuracy, where the slightly decreased performance seems to indicate the reduced reliability of parameter estimates. The increase of the termination threshold (decrease of mixture size) leads to the decrease of decoding performance, which indicates the need of maintaining resolution of data clusters for good decoding accuracy.

TABLE V
DECODING RATES FROM USING THE K-PAIRS-AT-A-TIME MERGING AT THE VARIOUS TERMINATION THRESHOLDS

	Termination threshold β values					
	5.2	4.9	4.6	4.3	4.0	3.7
Word correct	87.2%	87.5%	87.7%	88.6%	88.3%	88.3%
Word accuracy	84.2%	84.6%	85.0%	86.0%	85.7%	85.8%
Total mixtures	2832	3026	3242	3456	3717	4014

3) Baseline Comparison: For the segmental K-means algorithm, the number of frames per mixture component is varied from $\tau = 50$ to 62. Similar to the merging methods, the maximum mixture size is limited to 60. The decoding rates are shown in Table VI. Comparing Table VI with Tables III–V we observe that the decoding rates from the segmental K-means are all lower than those from the merging schemes. Furthermore, as is shown by the last entry in Table VI, the decoding performance is slightly improved by fixing the mixture size of each mixture density to be the same as those in Table IV (the elimination of a few very small clusters caused a slight difference of mixture size).

The different performances between the K-means and the bottom-up merging could be attributed to the different ways the clusterings are accomplished in the two algorithms. The K-means clustering requires assumptions on the structure of the training data, i.e., the number of clusters and the cluster centroids. Although the centroids are improved through the iterative clustering, the resultant estimates would be only

TABLE VI
DECODING RATES FROM USING THE K-MEANS AT VARIOUS MIXTURE SIZES

Frames/ mixture	$\tau = 62$	$\tau = 58$	$\tau = 54$	$\tau = 50$	fixed
Word correct	83.4%	83.5%	84.2%	84.5%	84.2%
Word accuracy	80.9%	81.2%	82.0%	81.1%	82.3%
Total mixtures	2889	3073	3277	3520	3474

TABLE VII
DECODING RATES FROM USING DIFFERENT COVARIANCE SMOOTHING

Smoothing method	case 1	case 2	case 3	case 4	case 5
Word correct	84.2%	83.5%	86.4%	86.8%	88.6%
Word accuracy	79.7%	80.5%	82.7%	83.1%	86.0%

optimal in the local neighborhood of the initial centroids. In contrast, the bottom-up merging algorithm does not impose these assumptions on the structure of the training data. The algorithm is data driven, and the data clusters are found from the closeness among the small clusters. The use of the subsegments as the initial clusters and their centroids in distance computation also reduces the effect of possible outliers in data clustering.

4) Smoothing Effect: The effects of smoothing on the covariance matrices are investigated for the following cases and the results are shown in Table VII.

Case 1. Covariance matrices are not smoothed unless they are singular. The singular covariance matrices are substituted by the unimodal covariance matrix. This corresponds to an interpolation parameter of $\lambda = 1.0$.

Case 2. The covariance matrices are all replaced by that of the unimodal density, corresponding to an interpolation parameter of $\lambda = 0.0$.

Case 3. The covariance matrices are all smoothed by an interpolation parameter of $\lambda = 0.5$.

Case 4. The covariance matrices are all smoothed by comparing the sharpness ratio R_i with a threshold of 100, and then those above the threshold are smoothed with the interpolating parameter of $\lambda = 0.5$.

Case 5. The default smoothing.

As seen from the table, case 1 of no smoothing and case 2 of too much smoothing both led to the poor performances. In the former case the models are unreliable, and in the latter case some of the useful details of the models are lost. The compromised smoothings of cases 3 and 4 both led to the improved performances, but the best performance was achieved in case 5. The requirement of heavier smoothing on the instantaneous features than the dynamic features might be attributed to the former having more variations across speakers and contexts (thus needs robustness), and the latter being more consistent in the different environments (thus prefers details). An alternative point of view is that the covariance estimates of the dynamic features might carry more information than those of the instantaneous features, and the useful information, when preserved, might help to increase the decoding accuracy.

TABLE VIII
DECODING RATES FROM USING DIFFERENT MIXTURE WEIGHT SMOOTHING

Smoothing method	case 1	case 2	case 3	case 4
Word correct	82.7%	87.2%	88.3%	88.6%
Word accuracy	80.3%	83.0%	86.1%	86.0%

The smoothing effects on the mixture weights are investigated for the following cases, and the results are shown in Table VIII.

Case 1. $\lambda_{l,r} = 0.0, \lambda_l = 0.0, \lambda_r = 0.0, \lambda_u = 1.0$.
Case 2. $\lambda_{l,r} = 1.0, \lambda_l = 0.0, \lambda_r = 0.0, \lambda_u = 0.0$.
Case 3. $\lambda_{l,r} = 0.5, \lambda_l = 0.0, \lambda_r = 0.0, \lambda_u = 0.5$.
Case 4. $\lambda_{l,r} = 0.5, \lambda_l = 0.2, \lambda_r = 0.2, \lambda_u = 0.1$.

In the first case, only context-independent weights are used; in the second case, context-dependent weights are used but no smoothing was applied; in the third case, the weights of the tri-phone contexts are smoothed by the context-independent weights; and in the fourth case, the weights of the tri-phone contexts are smoothed by the weights of the left contexts, weights of the right contexts, and weights without context. As seen from Table VIII, compared to using the context-independent mixture weights, using the raw context-dependent mixture weights improved decoding accuracy by approximately 3%. The decoding rates are further improved after the mixture weights are properly smoothed, and the smoothings in cases 3 and 4 give similar decoding accuracies.

The interpolation parameters used in the smoothings are empirically chosen through experiments. Interpolation parameters for smoothing HMM phone models can also be estimated from training data using the statistical technique of cross validation [1]. However, when the amount of training data is small, the variance of the estimate from cross validation is usually high [18]. Since the sample size of each phone is not large, as shown in Table I, the use of the fixed smoothing interpolation parameters is reasonable.

5) Dictionary Compression: The dictionaries are compressed using the thresholds of $\gamma = 0.1, 0.2, 0.3$. The results from using the compressed dictionaries are shown in Table IX, and the results from using the standard dictionary is also shown in the table for comparison. The total number of transcriptions for each dictionary are also listed as a reference. We observe that the compressed dictionaries consistently outperform the standard dictionary, with an error rate reduction of up to 18%. The highest decoding rate is achieved by using the threshold $\gamma = 0.1$. The results indicate that using multiple transcriptions extracted from various speaking contexts and dialects improves decoding rates. The compression procedure seems to be capable of capturing the representative pronunciations from among many variations. It is to be noted here that although the compressed dictionary has more homophones than the standard dictionary, essentially equal decoding rates were obtained when using the two sets of homophones in the alignment of the decoded sentences with the reference.

6) Task Perplexity: The decoding rates for various grammar perplexities on the test set I, test set II, and the training set are summarized in Table X. To compensate for the more

TABLE IX
DECODING RATE FROM USING VARIOUS WORD TRANSCRIPTION DICTIONARIES, INCLUDING A STANDARD DICTIONARY

Compression threshold	standard dictionary	$\gamma = 0.3$	$\gamma = 0.2$	$\gamma = 0.1$
Word correct	85.5%	88.6%	88.6%	88.7%
Word accuracy	83.4%	86.2%	86.0%	86.4%
Total number of transcriptions	857	952	982	1108

TABLE X
DECODING RATES OF TEST SET I, TEST SET II, AND THE TRAINING SET ON VARIOUS TASK PERPLEXITIES

Grammar	grammar-I		grammar-II		null grammar	
Decoding rate	word correct	word accuracy	word correct	word accuracy	word correct	word accuracy
Test set II	93.1%	90.9%	88.6%	86.0%	66.3%	62.9%
Test set I	93.7%	92.6%	86.5%	84.8%	65.8%	61.6%
Training set	98.2%	97.7%	97.8%	97.5%	91.8%	91.7%

frequent insertion errors as the grammar perplexity increases, the word penalty value varies from light to heavy as the perplexity goes up. For the test set I using grammar-II (test set perplexity = 106), the confusion between the words "the" and "a" counts toward about 16% of the total sentence errors and 15% of the total word errors, and the two articles count toward about 9% of the words in the test set. From Table 10 we observe that the training set performance far exceeds that of the test set in the case of a null grammar. This significant difference seems to indicate that the current training data (as indicated in Table II) is still not large enough to train HMM phone models with sufficient power of generalization.

VII. CONCLUSION

In this paper, the major features of a speaker-independent continuous speech recognition system are summarized. A bottom-up merging algorithm for generating mixture Gaussian densities from models of sub-segments of phone units is developed. The mixture size of a generated mixture density is proportional to both the sample size and the dispersion of the training data. The bottom-up merging algorithm has been proven successful in enhancing the recognition performance when comparing with the baseline of the segmental K-means. An automatic procedure of preparing word transcription dictionaries from the acoustic-phonetic labels of training speech sentences is also described. The developed procedure is capable of capturing the representative pronunciations, and also has led to improved decoding performance when comparing using a standard dictionary. The system has been trained and evaluated on the TIMIT database and has achieved above 86% word accuracy on a task vocabulary size of 853 and grammar perplexity of 104. The decoding accuracy achieved from the merging algorithm is 4.1% higher than that from the basic segmental K-means; and the decoding accuracy achieved from using the compressed dictionary is 3.0% higher than that from using the standard dictionary. The error rate reductions from the two improvements are 22.8% and 18.1%, respectively. There is still a significant difference in decoding accuracy on the training set and the test sets, which implies that the performance might be further improved if more data is used in training the HMM phone models. So far, the system has been evaluated on a relatively conservative basis, i.e., the training and test sets have a complete overlap in vocabulary and sentence syntax. It remains interesting to further evaluate the system performance on more challenging tasks with less vocabulary overlap and more syntax difference between the two sets.

APPENDIX

This appendix derives (3). Let a sample data set be $\Omega = \{x_i, i = 1, 2, \cdots, L\}$. The set Ω is partitioned into N subsets with $\Omega = \cup_{n=1}^{N} \Omega_n$, where $\Omega_n = \{x_i^{(n)}, i = 1, 2, \cdots, L_n\}$, for $n = 1, 2, \cdots, N$. The grand mean μ and covariance matrix C of the sample set are

$$\mu = \frac{1}{L} \sum_{i=1}^{L} x_i$$

and

$$C = \frac{1}{L} \sum_{i=1}^{L} (x_i - \mu)(x_i - \mu)'.$$

The mean μ_n and covariance matrix C_n of each subset are defined similarly for $n = 1, 2, \cdots, N$,

$$\mu_n = \frac{1}{L_n} \sum_{i=1}^{L_n} x_i^{(n)}$$

and

$$C_n = \frac{1}{L_n} \sum_{i=1}^{L_n} (x_i^{(n)} - \mu_n)(x_i^{(n)} - \mu_n)'.$$

Let the histogram probabilities of the subsets be $p_n = L_n/L$, $n = 1, 2, \cdots, N$. The relationship between the grand mean and the subset means is

$$\mu = \sum_{n=1}^{N} p_n \mu_n.$$

The grand covariance matrix C can be expanded as follows:

$$\begin{aligned} C &= \frac{1}{L} \sum_{i=1}^{L} (x_i - \mu)(x_i - \mu)' \\ &= \frac{1}{L} \sum_{n=1}^{N} \sum_{i=1}^{L_n} (x_i^{(n)} - \mu)(x_i^{(n)} - \mu)' \\ &= \sum_{n=1}^{N} p_n C_n + \sum_{n=1}^{N} p_n (\mu - \mu_n)(\mu - \mu_n)'. \end{aligned}$$

The above expansion of C is in agreement with the scatter matrix expansion in [19].

Let the second summation term be C_B. Then C_B can be further expanded as

$$\begin{aligned} C_B &= \sum_{n=1}^{N} p_n(\mu-\mu_n)(\mu-\mu_n)' \\ &= \sum_{n=1}^{N} p_n\mu_n\mu_n' - \mu\mu' \\ &= \sum_{n=1}^{N} p_n\left(\sum_{j=1}^{N} p_j\right)\mu_n\mu_n' - \sum_{k=1}^{N}\sum_{j=1}^{N} p_kp_j\mu_k\mu_j' \\ &= \sum_{n=1}^{N}\sum_{j=n+1}^{N} p_np_j(\mu_n\mu_n' + \mu_j\mu_j') \\ &\quad - 2\sum_{k=1}^{N}\sum_{j=k+1}^{N} p_kp_j\mu_k\mu_j' \\ &= \sum_{n=1}^{N}\sum_{j=n+1}^{N} p_np_j(\mu_n-\mu_j)(\mu_n-\mu_j)'. \end{aligned}$$

Therefore,

$$C = \sum_{n=1}^{N} p_nC_n + \sum_{n=1}^{N}\sum_{j=n+1}^{N} p_np_j(\mu_n-\mu_j)(\mu_n-\mu_j)'.$$

ACKNOWLEDGMENT

The author would like to sincerely acknowledge Dr. H. Wakita for his constant support during the course of this work. The author would also like to acknowledge the valuable comments made by the anonymous reviewers and G. De Haan and Dr. B. Hanson of Speech Technology Laboratory. Other help by Lisa King, Sheu-jen Ting, and Brian Mak is also acknowledged.

REFERENCES

[1] K. F. Lee, "Speaker-independent continuous speech recognition using hidden Markov models," Ph.D. dissertation, Carnegie-Mellon Univ., Pittsburgh, PA, 1988.
[2] C. H. Lee, "Acoustic modeling of subword units for speech recognition," in *Proc. ICASSP*, Albuquerque, NM, Apr. 1990, pp. 721–724.
[3] R. L. Rabiner, J. G. Wilpon, and B. H. Juang, "A segmental K-means training procedure for connected word recognition," *AT&T Tech. J.*, vol. 65, no. 3, pp. 21–31, 1986.
[4] Y. Zhao and H. Wakita, "Experiments with a speaker-independent continuous speech recognition system on the TIMIT database," in *Proc. ICSLP*, Kobe, Japan, Sept. 1990, pp. 697–700.
[5] Y. Zhao, H. Wakita, and X. Zhuang, "An HMM based speaker-independent continuous speech recognition system with experiments on the TIMIT database," in *Proc. ICASSP*, Toronto, Canada, May, 1991, pp. 333–336.
[6] L. F. Lamel, R. H. Kassel, and S. Seneff, "Speech database development: Design and analysis of the acoustic-phonetic corpus," in *Proc. Speech Recognition Workshop (DARPA)*, 1986.
[7] D. S. Pallet, "Speech corpora and performance assessment in the DARPA SLS program," Proc. ICSLP, pp. 24.3.1–24.3.4, Kobe, Japan, Sept. 1990.
[8] H. Hermansky, B. A. Hanson, H. J. Wakita, "Perceptually based linear predictive analysis of speech," Proc. ICASSP, pp. 509–512, Tampa, Florida, 1985.
[9] B. Hanson and H. Wakita, "Spectral slope distance measures with linear prediction analysis for word recognition in noise," IEEE Trans. ASSP, ASSP-35, pp. 968–973, 1987.
[10] S. Furui, "Speaker-independent isolated word recognition using dynamic features of speech spectrum," IEEE Trans. ASSP, ASSP-34, pp. 52–59, 1986.
[11] F. Kubala, Y. Chow, A. Derr, M. Feng, O. Kimball, J. Makhoul, P. Price, J. Rohlicek, S. Roucos, R. Schwartz, and J. Vandegrift, "Continuous speech recognition results of the BYBLOS system on the DARPA 1000-word resource management database," Proc. ICASSP, pp. 291–294, New York, Apr. 1988.
[12] B. H. Juang and L. R. Rabiner, "Mixture autoregressive hidden Markov models for speech signals," IEEE Trans. ASSP, ASSP-33, pp. 1404–1413, 1985.
[13] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C*. Cambridge, 1988.
[14] Y. Zhao, H. Wakita, and X. Zhuang, "Generate word transcription dictionary from sentence utterances and evaluate its effects on speaker-independent continuous speech recognition," Proc. of the 2nd EuroSpeech, Genova, Italy, Sept. 1991, pp. 679–682.
[15] F. Jelinek, "Markov modeling of text generation," in *The impact of Processing Techniques on Communications*, J. K. Skiwirzynski, Ed., Nato ASI, Series E, no. 91, pp. 569–598, 1985.
[16] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Language, and Computation*. Reading, MA: Addison-Wesley, 1979.
[17] R. L. Rabiner, B. H. Juang, S. E. Levinson, and M. M. Sondhi, "Recognition of isolated digits using hidden Markov models with continuous mixture densities," *AT&T Tech. J.*, vol. 64, no. 6, pp. 1211–1233, 1985.
[18] B. Efron, "Estimating the error rate of a prediction rule: Improvement on cross-validation," *J. Amer. Stat. Asso.*, vol. 78, no. 382, pp. 316–331, June 1983.
[19] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

Yunxin Zhao (S'86–M'88) received the B.S. degree in 1982 from Beijing Institute of Posts and Telecommunications, Beijing, China, and the M.S.E.E. and Ph.D. degrees in 1985 and 1988, respectively, from the University of Washington, Seattle.

She has done research on computer network performance analysis, time–frequency signal analysis, speech and image processing, and recognition. She is currently with Speech Technology Laboratory, Panasonic Technologies Inc., where her primary research emphasis is on speech recognition.