High Performance Connected Digit Recognition Using Hidden Markov Models

LAWRENCE R. RABINER, FELLOW, IEEE, JAY G. WILPON, SENIOR MEMBER, IEEE, AND FRANK K. SOONG, MEMBER, IEEE

Abstract—**Algorithms for connected word recognition based on whole word reference patterns have become increasingly sophisticated and have been shown capable of achieving high recognition performance for small or syntax-constrained, moderate size vocabularies in a speaker trained mode. In particular, it has been demonstrated that for a vocabulary of digits, in a speaker trained mode, very high string accuracy is achievable using either hidden Markov models (HMM) or templates as the digit reference patterns. In this paper we use an enhanced analysis feature set consisting of both instantaneous and transitional spectral information and test the HMM-based connected digit recognizer in speaker trained, multispeaker, and speaker independent modes. For the evaluation, we used both a 50 talker connected digit database recorded over local, dialed-up telephone lines, and the Texas Instruments, 225 adult talker, connected digits database which has been widely distributed through the National Bureau of Standards. Using these databases, the performance that was achieved was 0.35, 1.65, and 1.75 percent string error rates for known length strings, for speaker trained, multispeaker, and speaker independent modes, respectively, and 0.78, 2.85, and 2.94 percent string error rate for unknown length strings of up to 7 digits in length for the 3 modes. Several experiments were carried out to determine the best set of conditions (e.g., training, recognition parameters, etc.) for recognition of digits. The results, and the interpretation, of these experiments will be described in this paper.**

I. INTRODUCTION

THE problem of recognizing strings of connected digits is crucial to a number of applications such as voice dialing of telephone numbers, automatic data entry, credit card entry, PIN (personal identification number) entry, entry of access codes for transactions, etc. In the last several years, several highly successful algorithms for recognizing spoken connected word strings from word prototypes have evolved [1]–[5]. These algorithms, all based on statistical pattern recognition methods, have achieved great success when applied to the problem of connected digit recognition [5]–[7]. The reasons for this success are twofold: namely, the fact that the recognition algorithms are optimal in the sense that they find the string of digit reference patterns that best (in some objective sense) matches the spoken digit string, and the development of highly successful training procedures which derive the digit reference patterns from a training set of fluent, connected, digit strings [5]–[9].

Manuscript received February 9, 1988; revised November 5, 1988.
The authors are with AT&T Bell Laboratories, Murray Hill, NJ 07974.
IEEE Log Number 8928735.

Earlier investigations showed that when a reasonable size training set was available for deriving the digit reference patterns, a fairly good recognizer could be implemented using either hidden Markov model (HMM) or template characterizations of the digits, with the HMM-based system achieving somewhat higher performance than the template based approach [7]. For such systems, the highest performance scores were achieved in a speaker trained mode (typical string accuracies of from 98 to 99 percent); however, performance was found to degrade seriously in either a multispeaker or a speaker independent mode. Bush and Kopec found that by incorporating acoustic-phonetic knowledge into the recognizer, improved performance on speaker independent, connected digit recognition resulted [5]. Their results, which used some manual segmentation in order to bootstrap the training, showed speaker independent, connected digit string accuracies of from 96 to 97 percent.

In an effort to improve performance of the fully automatic connected digit recognition algorithms, a major change was made in the front end spectral analysis. The analysis feature vector used for recognition, nominally an extended cepstral vector derived from LPC analysis, was augmented by the so-called delta cepstrum information [10], [11]. (The delta cepstrum vector is the least squares fit of the time derivative of each of the cepstral parameters, defined over a finite time window.) The resulting augmented analysis vector characterizes both the *short-time spectrum* (via the cepstrum) and the *short-time spectral derivative* (via the delta cepstrum). The motivation behind this change was the observation that, by including information about the time derivative of the cepstral vector, a more complete 2-dimensional (time and frequency) spectral representation of the time-varying speech signal is obtained, and the performance of a vector quantization (VQ) based talker verification system improved dramatically (error rate decreased by a factor of 2) [11].

The new analysis feature set was tested in the HMM-based connected digit recognizer in speaker trained, multispeaker, and speaker independent modes, and was found to effectively reduce the string error rates by factors of 2 or more, often with considerably less computation than used previously [7]. In particular, digit string error rates of 0.78, 2.85, and 2.94 percent were obtained for *unknown length* (UL) strings of from 1 to 7 digits for speaker

0096-3518/89/0800-1214$01.00 © 1989 IEEE

trained, multispeaker, and speaker independent tests, respectively. Comparable rates for *known length* (KL) strings were 0.35, 1.65, and 1.75 percent, respectively.

The organization of this paper is as follows. In Section II we review the fundamentals of the HMM connected digit recognizer. In this section we define precisely the way in which the new analysis feature vector is computed. In Section III we describe the experimental evaluation of the improved system in each of the three modes in which it was tested, namely, speaker trained, multispeaker, and speaker independent. Finally, in Section IV, we summarize the results and point out the relevance to general problems in speech recognition.

II. Review of HMM Connected Digit Recognizer

A fairly comprehensive description of the complete connected digit recognizer is given in [7]. Thus, in this section, we will give only an overview of the recognition system, and then will focus on the improved front end spectral analysis.

A block diagram of the overall level building, connected-digit recognizer is shown in Fig. 1. There are essentially three steps in the recognition algorithm, as follows.

1) Spectral analysis—The speech signal, $s(n)$, is converted to a set of LPC derived cepstral (weighted) and delta-cepstral (weighted) vectors.

2) Level building pattern matching—The sequence of spectral vectors of the unknown speech signal is matched against a set of stored single-digit patterns (hidden Markov models) using the level building algorithm with Viterbi matching within levels. The output of this process is a set of candidate digit strings, generally of different lengths (i.e., different number of digits per string).

3) Postprocessor—The output candidate strings from level building are subjected to further validity tests, e.g., state duration, to eliminate unreasonable candidates. The postprocessor chooses the most likely digit string from the remaining (valid) candidate strings.

In the remainder of this section we expand further on the LPC spectral analysis (since this is fundamentally different from the one used in previous studies), and on the form of the HMM's. All other signal processing in the recognizer is essentially identical to that described in [7].

A. LPC Spectral Analysis

The LPC front end processing for recognition is shown in Fig. 2. The overall system is a block processing model in which a frame of N samples is processed and a vector of features is computed. (Strictly speaking, as we will see below, this is not correct since the system uses a 5 frame window to compute the delta cepstrum vector.) The steps in the processing are as follows.

1) Preemphasis—The digitized (at a 6.67 kHz rate) speech signal is processed by a first-order digital network in order to spectrally flatten the signal.

2) Blocking into frames—Sections of N consecutive speech samples (we use $N = 300$ corresponding to 45 ms

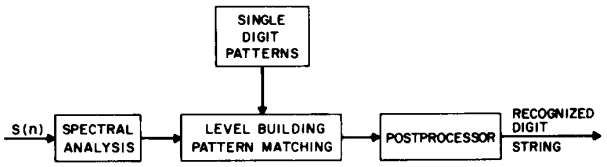

Fig. 1. Block diagram of connected digit recognizer.

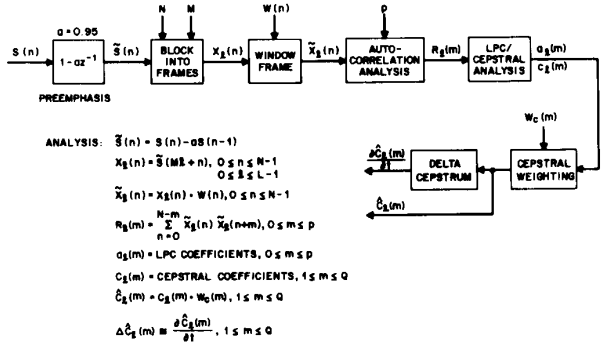

Fig. 2. Block diagram of improved front end LPC analysis incorporating instantaneous and transitional cepstral information.

of signal) are used as a single frame. Consecutive frames are spaced M samples apart (we use $M = 100$ corresponding to 15 ms frame spacing, or 30 ms frame overlap).

3) Frame windowing—Each frame is multiplied by an N-sample window (we use a Hamming window) so as to minimize the adverse effects of chopping an N-sample section out of the speech signal.

4) Autocorrelation analysis—Each windowed set of speech samples is autocorrelated to give a set of $(p + 1)$ coefficients, where p is the order of the desired LPC analysis (we use $p = 8$).

5) LPC/cepstral analysis—For each frame, vectors of LPC coefficients are computed from the autocorrelation vector using a Levinson or a Durbin recursion method. The LPC derived cepstral vector is then computed up to the Qth component, where $Q > p$, and $Q = 12$ in our implementation.

6) Cepstral weighting—The Q-coefficient cepstral vector, $c_l(m)$, at time frame l, is weighted by the window, $W_c(m)$, of the form [12], [13]

$$W_c(m) = \left[1 + \frac{Q}{2}\sin\left(\frac{\pi m}{Q}\right)\right], \qquad 1 \le m \le Q \quad (1)$$

to give

$$\hat{c}_l(m) = c_l(m) \cdot W_c(m). \quad (2)$$

(In theory, the use of cepstral weighting is irrelevant for diagonal covariance HMM's. However, since we used the weighted cepstal coefficients in the design of a codebook for choosing mixture parameters, based on a Euclidean distance, the weighting is relatively important.)

7) Delta cepstrum—The time derivative of the sequence of weighted cepstral vectors is approximated by a first-order orthogonal polynomial over a finite length win-

dow of $(2K + 1)$ frames, centered around the current vector. ($K = 2$ in our implementation; hence, the derivative is computed from a 5 frame window.) The cepstral derivative (i.e., the delta cepstrum vector) is computed as

$$\Delta\hat{c}_l(m) = \left[\sum_{k=-K}^{K} k\hat{c}_{l-k}(m)\right] \cdot G, \qquad 1 \le m \le Q \quad (3)$$

where G is a gain term so that the variances of $\hat{c}_l(m)$ and $\Delta\hat{c}_l(m)$ are about the same. (For our system, the value of G was 0.375.)

The overall observation vector, $\boldsymbol{O}_l$, used for scoring the HMM's is the concatenation of the weighted cepstral vector, and the corresponding weighted delta cepstrum vector, i.e.,

$$\boldsymbol{O}_l = \left\{\hat{c}_l(m), \Delta\hat{c}_l(m)\right\}, \quad (4)$$

and consists of 24 coefficients per vector.

B. Hidden Markov Model Characterization of Words

Fig. 3 shows the form of the HMM used to characterize individual digits [14]–[16]. (Transitions between words are handled by a switch mode from the last state of one word model, to the first state of another word model, in the level building implementation.) The models are first-order left-to-right Markov models with N states.[1] (We have used values of N from 5 to 10.)[2] Each state, j, is characterized by the following.

1) A state transition vector, $\boldsymbol{a}_j$, with components a_{ji} = probability of making a transition to state i (at the next transition instant), given that the system is currently in state j. For the ergodic models of Fig. 3, a_{ji} satisfies the constraints

$$a_{ji} = 0, \qquad i < j, \qquad \text{and for } i > j + 1 \quad (5)$$

since we allow transitions from state j only to itself or to state $j + 1$.

2) A state observation density, $b_j(\boldsymbol{O})$, of the form

$$b_j(\boldsymbol{O}) = \sum_{m=1}^{M} c_{mj}\boldsymbol{N}[\boldsymbol{O}, \mu_{mj}, U_{mj}] \quad (6)$$

i.e., a continuous mixture density where $\boldsymbol{O}$ is the observation vector (e.g., cepstral coefficient vector resulting from the LPC analysis), c_{mj} is the mixture weight for the mth component in state j, $\boldsymbol{N}$ represents a multivariate normal density, μ_{mj} is the mean vector for mixture m in state j, and U_{mj} is the covariance matrix for mixture m in state j. Typically, we use anywhere from $M = 1$ to $M = 9$ mixture components. In practice, we have observed that components of $\boldsymbol{O}$ are essentially uncorrelated. Hence, we

[1]We use the notation N to represent the number of states in an HMM, and use M to represent the number of mixtures per state. Previously we used N and M as the number of samples per frame in the LPC analysis and the shift per frame. This should cause no confusion as we will not refer to the LPC-analysis parameters again in the math.

[2]The choice of values of number of states, N, and number of mixtures open state, M, for the word vocabulary is somewhat of an art, and is highly dependent on both the vocabulary words and the amount of training data. No simple rules for such choices are known.

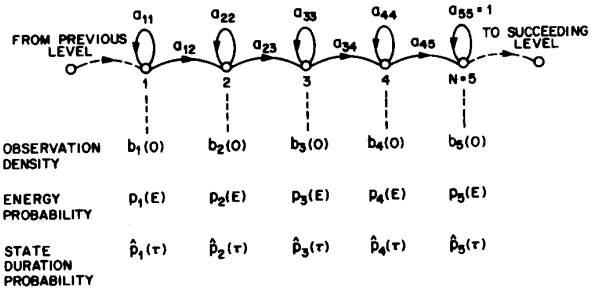

Fig. 3. Form of word HMM used to characterize individual digits.

assume that all components of $\boldsymbol{O}$ are statistically uncorrelated. Thus, U_{mj} becomes a diagonal covariance matrix, and (6) can be expressed simply as

$$b_j(\boldsymbol{O}) = \sum_{m=1}^{M} c_{mj} \frac{\prod_{d=1}^{D} \exp\left[-\left(O(d) - \mu_{mjd}\right)^2/2\sigma_{mjd}^2\right]}{(2\pi)^{D/2}\left(\prod_{d=1}^{D}\sigma_{mjd}^2\right)^{1/2}} \quad (7)$$

where $\boldsymbol{O}^{(d)}$ is the dth component of the observation vector, D is the number of components in $\boldsymbol{O}$, μ_{mjd} is the dth component of μ_{mj}, and σ_{mjd}^2 is the dth covariance of U_{mj}.

3) Energy probability, $p_j(\epsilon)$, where ϵ is the dynamically normalized frame energy, and p_j is a nonparametric discrete density of energy values in state j obtained empirically from training data. The energy values are computed on a log scale (i.e., in decibels), and the energy probabilities are quantized into 25 3-dB regions from 0 dB (absolute peak over a syllable length window) down to -75 dB and suitably normalized (based on a training set) so that

$$\sum_{i=1}^{25} p_j(\epsilon_i) = 1. \quad (8)$$

The choice of a 75 dB range on energy values reflects the range of variability of talkers, channels, and speech levels within a given channel.

4) State duration probability, $\hat{p}_j(\tau)$, where τ is the number of frames spent in state j, and $\hat{p}_j$ is an empirically measured, discrete density of duration values in state j. State durations are inherently quantized and are limited to values of 25 (frames) or less. Again, the state duration probabilities are normalized so that

$$\sum_{\tau_i=1}^{25} \hat{p}_j(\tau_i) = 1. \quad (9)$$

In addition to the observation density, energy probability, and state-duration probability, each HMM (for each word, v) is also characterized by an overall word-duration density, $p_v(D)$, of the form

$$p_v(D) = \boldsymbol{N}[\bar{D}_v, \sigma_v^2] \quad (10)$$

where $\bar{D}_v$ is the average duration for word v, σ_v^2 is the variance in duration for word v, and N is the normal density.

Based on the above, the process of building an HMM, of the type shown in Fig. 3, to characterize a word, requires estimation of

1) $2N - 2$ values of a_{ij}, the state transition coefficients;
2) NM values of c_{mj}, the mixture gains;
3) NMD values of μ_{mjd}, the mean values of the observations;
4) NMD values of $[U_{mj}]_d$, the diagonal covariances of the observations;
5) $25N$ values of $p_j(\epsilon)$, the energy probability;
6) $25N$ values of $\hat{p}_j(\tau)$, the state-duration density; and
7) the value of $\bar{D}_v$ and of σ_v^2, the average and variance of overall word duration.

All these parameters are estimated (or measured) from a training set as discussed in the next section.

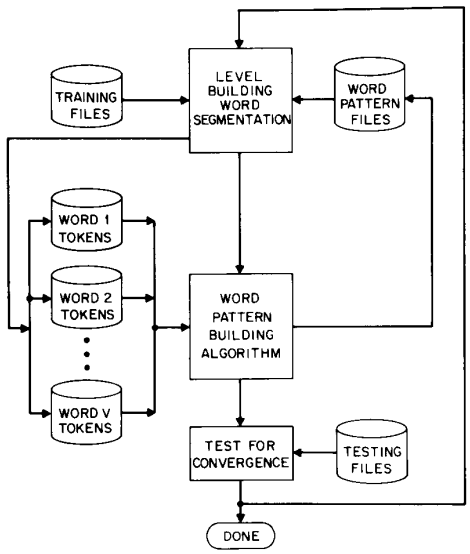

Fig. 4. Block diagram of the segmental k-means training procedure.

C. Training the Hidden Markov Models

In the case of building digit models from connected-digit training strings, the first step in building digit models is to segment optimally the digit strings into individual digits. For this task, a segmental k-means training procedure has been shown to be an effective way of converging at the optimum string segmentation [6]. A block diagram of the segmentation procedure is given in Fig. 4.

We assume that an initial set of word-pattern files is available. These initial files can equally well be a set of templates or HMM's. The templates or models can be a speaker independent set, a speaker trained set, or a designated training speaker set.

Give the initial word-pattern files and the training files (which consist of digit strings of various lengths), a level-building word-segmentation algorithm (of the type discussed in Section II-C) is used to optimally segment the training strings into individual word tokens which are stored in word-token files. A word-pattern-building algorithm (i.e., a model-estimation procedure for HMM's) is used to give an updated set of word patterns. The above procedure is iterated until the difference between the word patterns in consecutive iterations is sufficiently small [6].

1) Building HMM's for Each Word: Given a training set of W tokens of a word, the training problem for the HMM-based recognizer is to get optimal estimates of the complete set of HMM parameters for one or more models. The case of multiple models is essentially identical to that of single models. The only difference is that the training set is first clustered (using standard clustering procedures) and then a model is built for each cluster. Hence, we will concentrate here on parameter estimation for a single HMM from a finite training set of word tokens.

We have considered, and used, two methods for obtaining estimates of the HMM parameters, namely, the Baum–Welch reestimation procedure, and a segmental k-means loop based on segmentation of the words into individual states.

The Baum–Welch reestimation procedure is the standard method for estimation of the HMM parameters [14], [15]. Consider a single observation sequence, $\boldsymbol{O}$, corresponding to a single word token. Let $\boldsymbol{O} = \boldsymbol{O}_1\boldsymbol{O}_2 \cdots \boldsymbol{O}_T$. We define the forward calculation, $\alpha_t(i)$, as

$$\alpha_t(i) = \text{Prob}\,(\boldsymbol{O}_1\boldsymbol{O}_2 \cdots \boldsymbol{O}_t \quad \text{and state } i \text{ at } t \,|\, \lambda) \tag{11}$$

i.e., the probability of being in state i, at time t given the model λ. We compute $\alpha_t(i)$ recursively, i.e.,

$$\alpha_t(i) = \left[\sum_{j=1}^{N} \alpha_{t-1}(j)a_{ji}\right]\left(b_i(\boldsymbol{O}_t) \cdot p_i(\epsilon_t)^{\gamma_\epsilon}\right) \tag{12}$$

where $p_i(\epsilon_t)^{\gamma_\epsilon}$ is a weighted probability that energy ϵ_t occurs in state i. Similarly, we define the backward function $\beta_t(j)$ as

$$\beta_t(j) = \text{Prob}\,(\boldsymbol{O}_{t+1}\boldsymbol{O}_{t+2} \cdots \boldsymbol{O}_t \,|\, \text{state } j \text{ at } t \text{ and } \lambda). \tag{13}$$

We compute $\beta_t(j)$ recursively as

$$\beta_t(j) = \left[\sum_{i=1}^{N} \alpha_{ji}\left[b_i(\boldsymbol{O}_{t+1})\, p_i(\epsilon_{t+1})^{\gamma_\epsilon}\right] \beta_{t+1}(i)\right]. \tag{14}$$

To complete the reestimation picture, we need a third function, $\rho_t(j, m)$, defined as

$$\rho_t(j, m) = \text{Prob}\,(\boldsymbol{O}_1\boldsymbol{O}_2 \cdots \boldsymbol{O}_t, \text{ state } j, \text{ mixture } m \text{ at } t \,|\, \lambda). \tag{15}$$

The ρ function can be computed recursively as

$$\rho_t(j, m) = \sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij}\left[c_{mj} b_j^{(m)}(\boldsymbol{O}_t)\, p_j(\epsilon_t)^{\gamma_\epsilon}\right]. \quad (16)$$

Using α, β, and ρ, the reestimation equations for a, c, μ, U, and $p(\epsilon)$ are

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i)\, a_{ij} b_j(\boldsymbol{O}_{t+1})\, p_j(\epsilon_{t+1})^{\gamma_\epsilon}\, \beta_{t+1}(j)}{\sum_{t=1}^{T} \alpha_t(i)\, \beta_t(i)}, \quad 1 \le i, j \le N; \quad (17)$$

$$\bar{c}_{mj} = \frac{\sum_{t=1}^{T} \rho_t(j, m)\, \beta_t(j)}{\sum_{t=1}^{T} \alpha_t(j)\, \beta_t(j)}, \quad \begin{matrix} 1 \le m \le M \\ 1 \le j \le N; \end{matrix} \quad (18)$$

$$\bar{\mu}_{mjd} = \frac{\sum_{t=1}^{T} \rho_t(j, m)\, \beta_t(j)\, \boldsymbol{O}_t^{(d)}}{\sum_{t=1}^{T} \rho_t(j, m)\, \beta_t(j)}, \quad \begin{matrix} 1 \le m \le M \\ 1 \le j \le N \\ 1 \le d \le D; \end{matrix} \quad (19)$$

$$\bar{U}_{mjrs} = \frac{\sum_{t=1}^{T} \rho_t(j, m)\, \beta_t(j)(\boldsymbol{O}_t^{(r)} - \mu_{mjr})(\boldsymbol{O}_t^{(s)} - \mu_{mjs})}{\sum_{t=1}^{T} \rho_t(j, m)\, \beta_t(j)}, \quad \begin{matrix} 1 \le m \le M \\ 1 \le j \le N \\ 1 \le r, s \le D; \end{matrix} \quad (20)$$

$$\bar{p}_j(\epsilon(k)) = \frac{\sum_{t \text{ s.t. } \epsilon_t = \epsilon(k)} \alpha_t(j)\, \beta_t(j)}{\sum_{t=1}^{T} \alpha_t(j)\, \beta_t(j)}. \quad (21)$$

Equations (11)–(21) are readily extended to the case of multiple training sequences [14].

The basic problem with the reestimation equations is the amount of computation associated with implementing equations (11)–(21) with large scale problems, e.g., 2500 training tokens and eight-state five-mixture models. To alleviate this difficulty, a segmental k-means algorithm was used to provide estimates of c, μ, U, and $p(\epsilon)$. This method segments each training token into states by determining the optimal (Viterbi) alignment of the current model with each training token. All frames for a given word, in a given state, are used as input to a clustering algorithm (i.e., a vector quantizer design procedure) which determines the best M cluster solution (codebook). The reestimate of the c's is just the number of vectors in a given cluster, divided by the total number of vectors in the state. The μ and U for each cluster are determined from the vectors within each cluster. The transition coefficients are reestimated based on the average duration in a state, and the energy probability histogram is measured from the vectors occurring in the state. Similarly, the state duration probability is measured from the training-set segmentation into states, and the overall word duration and variance is measured directly from the initial k-means segmentation into words.

The segmental k-means reestimation of the HMM parameters is faster than the Baum–Welch reestimation procedure, and all our experimentation indicates that the resulting parameter estimates are essentially identical in that the resulting HMM's had essentially the same likelihood values. As such, we have extensively used the segmental k-means procedure, and all results to be presented here are based on this algorithm.

D. Level Building on HMM's

The way in which level building is used on HMM's is illustrated in Fig. 5. If we denote the set of V word HMM as λ_v, $1 \le v \le V$, then to find the optimum sequence of HMM's that match $\boldsymbol{O}$ (i.e., that maximize the likelihood), a sequence of Viterbi matches is performed. For each HMM, λ_v, we do a Viterbi match against $\boldsymbol{O}$, starting at frame 1, level 1, and retain for each possible frame, i, the following:

1) $P_l^v(i)$, the accumulated probability to frame i, at level l, for reference model λ_v, along the best path; and
2) $F_l^v(i)$, a backpointer indicating where the path started at the beginning of the level.

To compute $P_l^v(i)$, we need a local measure for the probability that observation $\boldsymbol{O}_i$ occurred in state j of model λ_v. The one we use is of the form

$$p_j^v(\boldsymbol{O}) = b_j^v(\boldsymbol{O}_i) \cdot \left[p_j^v(\epsilon_i)\right]^{\gamma_\epsilon} \cdot K_1 \quad (22)$$

where γ_ϵ is an energy-scaling coefficient, and K_1 is a normalization constant (which depends on γ_ϵ) such that (22) is a true probability. The value used for γ_ϵ in our experiments was 0.375. The state transition coefficients enter the calculation of $P_l^v(i)$ via the dynamic programming optimization in determining the Viterbi path.

At the end of each level, l, a maximization over v is performed to get the best model, at each frame, i, as follows:

$$P_l^B(i) = \max_{1 \le v \le V} P_l^v(i) \quad (23a)$$

$$W_l^B(i) = \operatorname*{argmax}_{1 \le v \le V} P_l^v(i) \quad (23b)$$

$$F_l^B(i) = F^{W_l^B(i)}(i). \quad (23c)$$

A best string of size l words ($1 \le l \le L$) with probability $P_l^B(I)$ is obtained by backtracking using the backpointer array $F_l^B(i)$ to give the words in the string. The overall best string is the maximum of $P_l^B(i)$ over all possible levels, l.

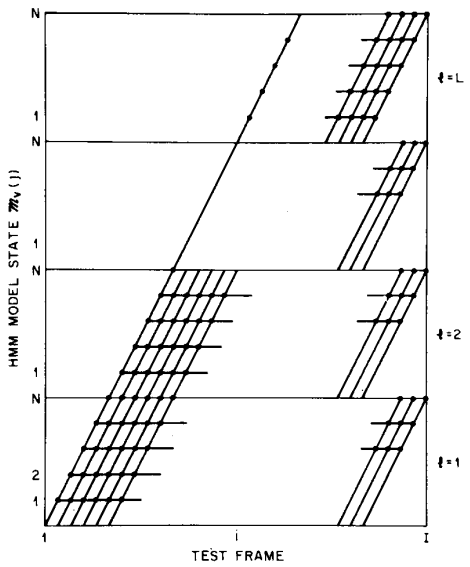

Fig. 5. Illustration of how level building is used with hidden Markov models.

E. Use of Duration (State, Word) in HMM Scoring

In the process of determining the best set of models that match a given input string, for each frame, i, of the test sequence, we use a local measure of the probability that the observation, $\boldsymbol{O}_i$, occurred in state j of model v. If we denote this probability as p_i, then we have

$$p_i = b_j^v(\boldsymbol{O}_i) \cdot \left[p_j^v(\epsilon_i)\right]^{\gamma_\epsilon} \cdot K_1 \tag{24}$$

as the local probability measure. For convenience, a "local distance" can be obtained by taking the negative log of p_i, giving

$$\begin{aligned} d_i &= -\log(p_i) \\ &= -\log\left(b_j^v(\boldsymbol{O}_i)\right) - \gamma_\epsilon \log\left(p_j^v(\epsilon_i)\right) - \log K_1. \end{aligned} \tag{25}$$

The major problem associated with scoring HMM models using (24) or (25) is that there is no durational information incorporated, either explicitly or implicitly, into the local distance measure. The implicit state duration density is exponential because of the Markov property of the model, i.e., the probability of duration τ in state j is proportional to $(a_{jj})^\tau$. Clearly, such a duration density is incorrect in almost all cases. There are several ways in which other duration densities could be incorporated into the HMM scoring including the "Ferguson" internal model in which an explicit duration density is used (this work by Ferguson is unpublished), the Levinson model in which a parametric form of duration density is used [17], and a postprocessor model in which duration is accounted for in a postprocessor [16]. (In this case, the exponential duration density is still explicitly present in the model.) Since both the Ferguson and Levinson models significantly increase the computation associated with scoring, we have opted for the simpler postprocessor duration model.

The way in which word and state durations are incorporated into the model scoring is as follows. At the end of each level, for each frame, the accumulated probability, $P_l^B(i)$, is modified by determining the word duration, $\tau_w(i)$, as

$$\tau_w(i) = i - F_l^B(i) + 1 \tag{26}$$

and then multiplying the accumulated probability by the word duration probability, i.e.,

$$\hat{P}_l^B(i) = P_l^B(i) \cdot \left[N(\tau_w(i), \overline{D}, \sigma^2)\right]^{\gamma_{wD}} \cdot K_2 \tag{27}$$

where N is a normal density with mean $\overline{D}$ and variance σ^2, and where γ_{wD} is a weighting factor on word durations, and K_2 is a normalization constant which depends on γ_{wD} and which makes (27) a true probability. The value used for γ_{wD} in our experiments was 3.

State duration probabilities are incorporated in a postprocessor. The level-building recognizer provides multiple candidates at each level. Hence, overall probability scores are provided for R^L strings of length L, where R is the number of candidates per level (typically $R = 2$). Each of the R^L strings is backtracked to give individual words and individual states within the words. For an L-word string, if we denote the duration of state j at level l as $\Delta_l(j)$, then, for each possible string, the postprocessor multiplies the overall accumulated probability, $\hat{P}_L^B(I)$, by the state-duration probabilities, giving

$$\hat{P}_L^B(I) = P_L^B(I) \cdot \prod_{l=1}^{L} \prod_{j=1}^{N} \left[\hat{p}_j^{v(l)}(\Delta_l(j))\right]^{\gamma_{SD}} \cdot K_3 \tag{28}$$

where γ_{SD} is a weighting factor on state durations, and K_3 is a normalization constant which depends on γ_{SD}. The value used for γ_{SD} in our experiments was 0.75. The computation of (28) is performed for all $(R)^L$ strings, and a reordered list of best strings is obtained. The incremental cost of the postprocessor computation is negligible compared to the computation to give $\hat{P}_L^B(I)$, and its performance has been shown to be comparable to the performance of the internal duration models [16].

F. Comments

It is worth noting that a frame synchronous implementation (rather than the level synchronous used here), suitable for real-time processing, has been developed and shown to yield results identical to those of the current system [18]. Also, a real-time implementation of the system has been built using a processor board, called the ASPEN (AT&T Systolic Processor Ensemble) board, with 8 DSP-32 chips (AT&T), an AT&T PC 6300+ personal computer, and a special purpose LPC spectral analysis board (again based on a DSP-32 chip) [19].

III. Experimental Evaluation and Results

To evaluate the performance of the connected-digit recognizer, in speaker trained, multispeaker, and speaker independent modes, two databases were used.

The first database consisted of 50 talkers (25 male, 25 female) drawn from the local, nontechnical, population (i.e., all talkers were native New Jersey residents). Each talker recorded 1200 connected-digit strings in about five sessions, during a 1-week period, over local dialed-up telephone lines. A new line was used for each recording session. The digits vocabulary consisted of the 10 digits (zero to nine); the word "oh" was excluded. Each talker recorded an equal number of strings with from 1 to 7 digits. Within each string, the digits were selected at random; however, during the test there was a constraint that there be an equal number of occurrences of each digit. All recordings were made in a reasonably quiet environment; however, because of line variations and talker loudness variations, some recordings had very bad signal-to-noise ratios (i.e., on the order of 10–20 dB). A check was made on each recorded string to guarantee that the correct string was spoken. Subsequent checking of a part of the database showed that the checking process was itself error prone and in fact at least 155 of the recorded strings which passed the first check had some type of recording problem. Because of the inexperience of the 50 talkers, a rather large number of the spoken strings were unusable (generally because of gross speaking errors in which only partial or incomplete strings were spoken), and about 21 percent of the 60 000 recorded strings (i.e., 12 600 strings) were eliminated. The talker with the most difficulty had about 50 percent of his strings (604 of 1200) eliminated; the talker with the least difficulty had only 47 of 1200 strings eliminated. Overall there remained 47 336 strings in the database. We denote the 50-talker database as DB50 in tables and in the text. This database was used in the speaker trained, and multispeaker evaluations.

The second database, which was used to evaluate the connected digit recognizer in a speaker independent mode, was the TI connected digits database [20], as distributed by the National Bureau of Standards. This database contained connected digit strings from 225 adult talkers[1] (equally distributed among male and female talkers), and was conveniently divided into training and testing sets, for consistency of comparison of results among the different researchers using this database. This database was dialectically balanced with an equal mix of talkers from 22 dialectical regions. At least 10 talkers (5 male, 5 female) from each dialectical region were included in the database. The vocabulary consisted of 11 words, namely, the 10 digits and "oh." Each talker spoke 77 sequences of these digits, consisting of 2 tokens of each of the 11 digits in isolation, and 11 sequences of each of 2, 3, 4, 5, and 7 digits (i.e., no 6-digit sequences were spoken). Digits were selected at random without replacement with one exception, namely, the digits zero and "oh" never occurred in the same string. The digit strings were recorded in an acoustically treated sound room using a high quality microphone (Electro Voice RE-16 Dynamic Cardiod). All recorded strings were verified by a team of listeners at TI [20]. We refer to this database as DBTI in figures and in tables.

As provided by the National Bureau of Standards, the digitized strings were sampled at a 20 kHz rate. For consistency with the telephone bandwidth of the strings of DB50, all strings were digitally filtered to a 3.2 kHz bandwidth, and downsampled to a 6.67 kHz rate. A total of 8568 training strings and 8578 testing strings were used (a small number of the strings on the digital tapes were unreadable). It should be noted that many of the strings had distinct silence gaps between groups of digits. Although it would have been possible to account for these silence gaps either by explicit methods (i.e., reendpoint the recorded strings) or by creating a silence model, neither of these procedures was actually used.

Database DB50 was split (at random) into a training set and a testing set, each consisting of roughly half the utterances for each talker and for each string length in the database. The training and testing sets for DBTI were specified by TI as an integral part of the database. The training sets were used to derive additional word HMM's; the independent test sets were used to measure system performance. The segmental k-means training procedure was always bootstrapped from word models derived from the isolated digits within the database [6].

A. Speaker Trained Mode Results

For the speaker trained case, two sizes of HMM's were studied, namely, 1 with 5 states and 3 mixtures per state (the same size model as was used in [7]), and 1 with 8 states and 5 mixtures per state. (The choice of these operating points is somewhat arbitrary and was based on previous experience with representing digits by HMM's.) The results of the recognition runs for the speaker trained case are given in Table I. Table I(a) gives string error rates (in percent) for unknown length (UL) and known length (KL) strings, for both the training set and the independent testing set, for the two HMM's that were studied. (All results for speaker trained runs were obtained using DB50.) Table I(b) gives a breakdown of the string error rates for unknown length strings as a function of the number of digits in the string.

The results given in Table I show the following.

1) Recognition performance is uniformly better for the larger model (8 states) than for the smaller model (5 states).

2) String error rates on the testing set are about twice as large as on the training set, although the absolute differences in error rates are still small.

3) String error rates for KL strings are about half those of UL strings for both the training and testing sets.

4) UL string error rates increase uniformly with the number of digits in the string, up to about 4 digits per string; for longer strings the error rates are much larger (around 1.4 percent), and are relatively insensitive to the number of digits in the string. (The reason for such behavior is unclear; the only explanation is that the average

[1]Included in the TI database were connected digit strings from children. These strings were not used in our evaluations.

TABLE I

(a) STRING ERROR RATES (PERCENT), FOR SPEAKER TRAINED MODE, FOR UNKNOWN LENGTH (UL), AND KNOWN LENGTH (KL) STRINGS ON DB50, FOR TWO SIZES OF HMM. (b) STRING ERROR RATES (PERCENT), FOR SPEAKER TRAINED MODE, FOR UNKNOWN LENGTH STRINGS, AS A FUNCTION OF THE NUMBER OF DIGITS IN THE STRING, ON DB50, FOR TWO SIZES OF HMM

HMM	Training Set		Testing Set	
	UL	KL	UL	KL
8 states, 5 mixtures/state	0.39	0.16	0.78	0.35
5 states, 3 mixtures/state	0.62	0.29	1.02	0.47

(a)

HMM	Number of Digits in String						
	1	2	3	4	5	6	7
8 states, 5 mixtures/state	0.11	0.28	0.50	0.59	1.51	1.43	1.21
5 states, 3 mixtures/state	0.38	0.40	0.94	0.89	1.78	1.52	1.36

(b)

rate of articulation for strings of length 4–7 digits, for this database, is approximately constant [7].)

The results given in Table I are based on the 23 750 strings spoken by the 50 talkers in the experiment. One interesting statistic is the individual speaker performance. This performance is illustrated in Fig. 6 which shows a cumulative plot of the percentage of talkers with testing string error rates above some threshold, E, for both UL [part (a)] and KL [part (b)] strings. It can be seen that for UL strings, the median string error rate is 0.6 percent (slightly lower than the average string error rate of 0.78 percent) and the talker with the highest error rate had 4.6 percent string errors. Similarly, for KL strings, the median string error rate is 0.23 percent (again slightly smaller than the average rate of 0.35 percent) and the talker with the highest error rate had 1.8 percent string errors.

B. Multispeaker Mode Results

For the multispeaker mode, using the training set of DB50, recognition systems were studied with from 1 to 6 models for each digit. The way in which multiple models were created was as follows. First, all the training strings were used to create a set of digit HMM's. (Two things should be noted here; first, only one-fourth of the set of training strings were used, i.e., about 6000 strings, because of computational constraints in the clustering algorithms; second, based on experience and intuition, we only considered models with $N = 10$ states, $M = 9$ mixtures per state.) Using a single model per digit set (designed using standard methods), the 6000 training strings were optimally segmented into individual digits, and these digit tokens were clustered into from 1 to 6 clusters for each of the 10 digits. An individual HMM was designed for each of the clusterings, thereby leading to sets of HMM's with from 1 to 6 models per digit.

The results of the recognition tests in the multispeaker mode are given in Table II which shows string error rate breakdowns for training and testing sets, and as a function of the number of digits per string for cases with from 1 to

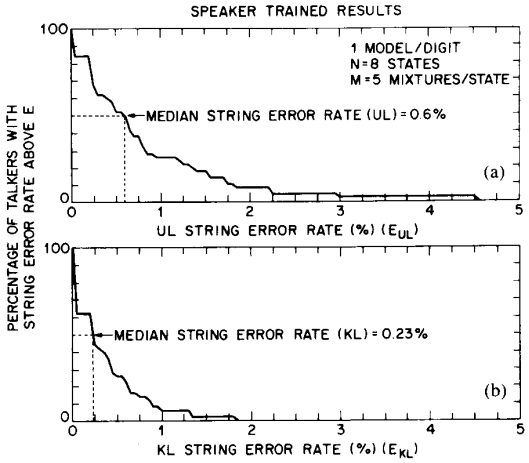

Fig. 6. Cumulative plots of percentage of talkers with testing set string error rates above a threshold for the speaker trained case; (a) for UL strings; (b) for KL strings.

TABLE II

(a) STRING ERROR RATES (PERCENT), FOR MULTISPEAKER MODE, FOR UNKNOWN LENGTH (UL), AND KNOWN LENGTH (KL) STRINGS ON DB50, AS A FUNCTION OF THE NUMBER OF MODELS PER DIGIT, FOR AN HMM WITH 10 STATES, 9 MIXTURES/STATE. (b) STRING ERROR RATES (PERCENTS), FOR MULTISPEAKER MODE, FOR UNKNOWN LENGTH STRINGS, AS A FUNCTION OF THE NUMBER OF DIGITS IN THE STRING AND THE NUMBER OF MODELS PER DIGIT, ON DB50, FOR AN HMM WITH 10 STATES, 9 MIXTURES/STATE

Number of Models Per Digit	Training Set		Testing Set	
	UL	KL	UL	KL
1	4.89	2.30	5.61	2.53
2	3.43	1.81	4.14	2.17
3	2.84	1.45	3.59	1.86
4	2.54	1.42	3.23	1.77
5	1.98	1.12	3.12	1.79
6	1.74	0.98	2.85	1.65

(a)

Number of Models Per Digit	Number of Digits in String						
	1	2	3	4	5	6	7
1	0.30	2.65	4.15	6.58	8.60	8.59	9.37
2	0.19	1.99	3.06	4.52	6.30	6.61	7.07
3	0.22	1.57	2.62	3.84	5.31	5.83	6.42
4	0.22	1.59	2.15	3.40	5.13	5.40	5.27
5	0.14	1.59	2.27	3.51	4.52	5.15	5.18
6	0.22	1.57	1.97	3.10	4.59	4.34	4.68

(b)

6 models per digit. The results in Table II show the following.

1) String error rates are significantly reduced by using more than 1 model per digit. For the training set, string error rates are reduced by a factor of about 2.5 as the number of models per digit is increased from 1 model/digit to 6 models/digit; for the testing set the comparable reduction in error rate is about 1.7 to 1.

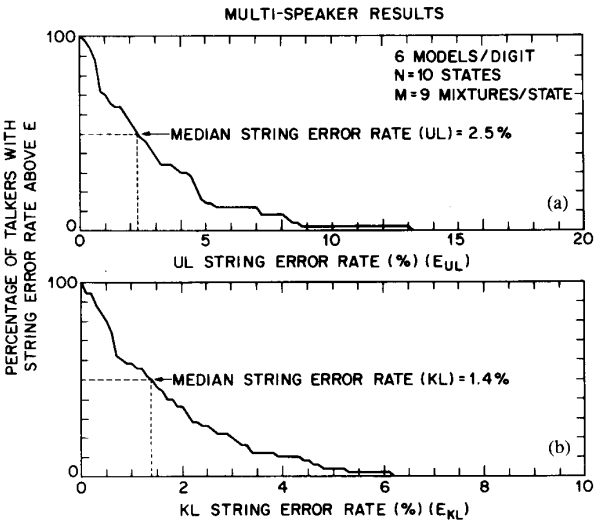

Fig. 7. Cumulative plots of percentage of talkers with testing set string error rates above a threshold for the multispeaker case; (a) for UL strings; (b) for KL strings.

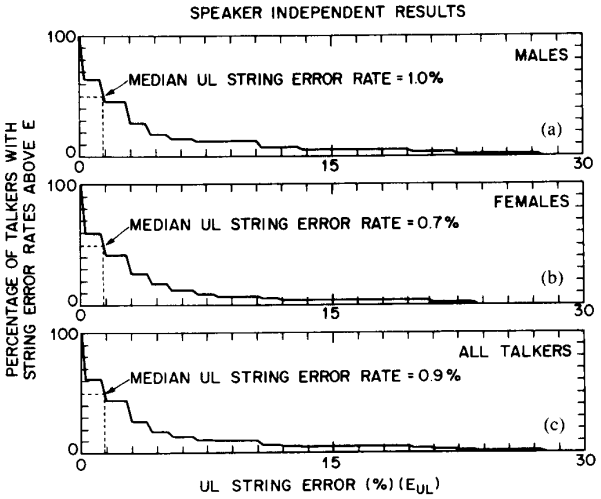

Fig. 8. Cumulative plots of percentage of talkers with testing set UL string error rates above a threshold for the speaker independent case; (a) for male talkers; (b) for female talkers; (c) for the combined talker set.

2) String error rates for training and testing sets are considerably closer than they were for the speaker trained case of Table I.

3) For the case of 6 models per digit, the resulting string error rates on the independent test set were 2.85 percent for unknown length strings and 1.65 percent for known length strings.

4) The error rates for isolated digits are very low (0.22 percent for 6 models per digit); the string error rates rise uniformly for 2–5 digit strings, then flatten off at a rate of about 4.5 percent.

Fig. 7 shows a cumulative plot, for the testing set, of the percentage of talkers with string error rate above a rate, E, for UL [part (a)], and KL [part (b)] strings, respectively. The median error rates of 2.5 percent (UL strings) and 1.4 percent (KL strings) are slightly lower than the average string error rates of 2.85 and 1.65 percent. The talker with the highest string error rate had 13 percent string errors (UL) and 6 percent string errors (KL). (Note that the scales in parts (a) and (b) of Fig. 8 are different.)

C. Speaker Independent Mode Results

For the speaker independent tests of the recognizer, database DBTI was used. The specified training set was used to create from 1 to 6 models per digit, in a manner similar to the one used in the multispeaker case. All 8565 training strings were used to create each set of models. The complete set of 8578 testing strings was used to evaluate the recognizer performance on the testing set. (The reader is reminded that, for this database, the vocabulary included the digits "oh" and zero, as well as one to nine.)

The results of the speaker independent recognition tests are given in Table III. The form of the table is the same as was used for Table II in the previous section. The results show the following.

TABLE III

(a) String Error Rates (Percent), for Speaker Independent Mode, for Unknown Length (UL), and Known Length (KL) Strings, on DBTI, as a Function of the Number of Models per Digit, for an HMM with 10 States, 9 Mixtures/State. (b) String Error Rates (Percent), for Speaker Independent Mode, for Unknown Length Strings, as a Function of the Number of Digits in the String, and the Number of Models per Digit, on DBTI, for an HMM with 10 States, 9 Mixtures/State

Number of Models Per Digit	Training Set		Testing Set	
	UL	KL	UL	KL
1	2.84	1.19	4.35	2.15
2	1.90	0.71	3.64	1.88
3	1.52	0.53	3.10	1.67
4	1.24	0.36	2.94	1.75
5	1.13	0.34	3.01	1.89
6	1.05	0.35	3.01	1.90

(a)

Number of Models Per Digit	Number of Digits in String					
	1	2	3	4	5	7
1	0.69	2.20	5.13	5.90	7.13	8.73
2	0.61	1.96	3.99	5.33	6.64	6.37
3	0.49	1.79	3.42	4.43	5.74	5.39
4	0.69	1.79	3.10	3.77	5.08	5.47
5	0.61	1.79	3.59	4.18	5.16	5.14
6	0.73	1.55	3.42	4.02	5.41	5.22

(b)

1) For the training set there is a reduction in string error rate of about 3 to 1 as the number of models per digit increases from 1 to 6; for the independent testing set the reduction in string error rate is only a factor of 1.5 for UL strings and 1.2 for KL strings.

2) A very large difference in performance exists between the training and testing sets, both for UL and KL strings. For example, for 6 models per digit, the string

error rate for UL strings is a factor of 3 smaller; for KL strings the error rates differ by a factor of 5.5.

3) The string error rates on the testing set level off at about 3–4 models per digit; for 4 models per digit the UL string error rate is 2.94 percent, the KL string error rate is 1.75 percent.

4) The isolated digit error rate for 6 models per digit is 0.73 percent; string error rates for UL strings increase uniformly from 2 to 5 digits per string. For 7 digit strings, the string error rate is essentially equal to that of 5 digit strings since no possibility of digit insertions existed, i.e., the maximum string length allowed was 7 digits.

Fig. 8 shows cumulative plots, for the testing set based on UL strings, of the percentage of talkers with string error rate above a threshold, E_{UL}, for the male talkers [part (a)], the female talkers [part (b)], and the combined talker set [part (c)]. Table IV gives a breakdown according to the number of string errors (out of the 77 spoken strings per talker), for males and females. The median UL string error rates are 1 percent for males, 0.7 percent for females, and 0.9 percent for the combined population. The UL string error rates are lower, by a factor of about 3, than the average error rates reported in Table III using 3 models per digit, showing that a large percentage of the string errors were generated by a small fraction of the talkers. This result was noted by Bush and Kopec [5].

TABLE IV
DISTRIBUTION OF STRING ERRORS FOR UL STRINGS IN THE TESTING SET BASED ON USING THREE MODELS PER DIGIT

	Number of String Errors											
Gender	0	1	2	3	4	5	6	7	8	9	10	>10	Total
Men	20	10	10	5	2	1	0	0	3	0	1	3(15,18,22)	55
Women	23	10	9	7	3	2	1	0	1	1	0	2(16,18)	57

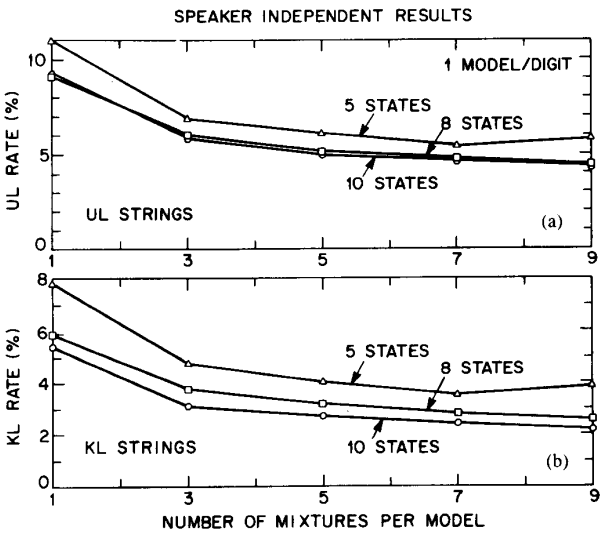

Fig. 9. Plots of testing set string error rates as a function of the number of mixtures per model for the speaker independent case; (a) UL strings, (b) KL strings.

D. Effects of Number of States, Number of Mixtures Per State

To demonstrate the effects of using fewer than 9 mixtures per state, or fewer than 10 states in each word HMM, an experiment was run using the DBTI database, in which a set of HMM's were designed with $N = 5$, 8, and 10 states, and with $M = 1$, 3, 5, 7, and 9 mixtures per state. In each case, for reasons related to computation, only a single HMM was designed for each digit; hence, the string error rates will be no better than the results in the first row of Table III(a). The results of this experiment, in the form of string error rates for UL strings [part (a)] and KL strings [part (b)], as a function of M, for different values of N, are given in Fig. 9. It can be seen that as M goes from 1 to 3, a significant reduction in string error rate occurs. Further increases in M result in small reductions in string error rate. Eventually, for large enough M, a statistical fluctuation in string error rate results. The effects of N are also clear. As N goes from 5 to 8, a significant reduction in string error rate results; however, as N goes from 8 to 10, there is no real performance difference for UL strings, but for KL strings there is a significant performance improvement.

E. Effects of Using Delta Cepstrum or Cepstrum Alone

Two brief experiments were carried out, using database DBTI, in which the feature set for recognition (the observation vectors) was changed from the combination of weighted cepstrum and weighted delta cepstrum to just the weighted delta cepstrum alone, or just weighted bandpass cepstrum alone (as was used in [7]). For these tests, a one-model-per-digit system was used with 10 states and 9 mixtures per state for each model. The results for these tests showed the following.

1) The error rates on the testing set, using just the weighted delta cepstrum coefficients, were 10.2 percent for UL strings, and 4.3 percent for KL strings, as contrasted to 4.35 percent for UL strings, and 2.15 percent for KL strings using the combined cepstrum.

2) The error rates on the testing set, using just the weighted cepstrum coefficients, were 12.7 percent for UL strings and 6.4 percent for KL strings.

These results show that the information in the "instantaneous" and "transitional" feature is somewhat complementary and that the combination of the feature sets gave much better performance than either of the individual feature sets. Whether or not further gains could be achieved using higher order transitional features is an open question of interest.

One last experiment was tried, based on work by Furui [10], in which the observation vector was created by adding the weighted delta cepstrum to the weighted cepstrum to create a feature set with only 12 components. The results using this additive set of features were the following (again with 1 model per digit, 10 states, and 9 mixtures per state).

1) The error rates on the testing set were 8.1 percent for UL strings and 2.8 percent on KL strings. These re-

sults, although better than either of the individual feature vectors, are still inferior to the results obtained from the combined feature vectors, again pointing out the importance of retaining both instantaneous and transitional spectral information in the observation vector.

F. Error Analysis for SI Runs

An analysis of the errors for the speaker independent testing run (using DBTI data) showed the following.

1) 66 digit insertions—exactly half (33) of these involved inserting an "oh" after the digit zero. This type of error leads to a semantically incorrect string and could easily be eliminated in the digit grammar. The digits 6 and 2 were inserted 8 and 7 times, respectively. All other digits were inserted 5 times or less.

2) 61 digit deletions—almost half of these (29) involved a deletion of the word "oh" in the context of 2 or more repetitions of "oh," e.g., "oh-oh" was recognized as "oh" or "oh-oh-oh" was recognized as "oh-oh." The digit "oh" was deleted 8 other times (i.e., not in the context of multiple "oh's"), and the digit 8 was deleted 19 times. No other digit was deleted more than 2 times.

3) 221 digit substitutions—about 40 percent of the digit substitutions (89) involved the word "oh." Most of the time an "oh" was substituted for the correct digit. The only other consistent digit substitution was a 9 for a 5, which occurred 23 times.

The above analysis shows that in about half the digit errors, the word "oh" was involved. This result is to be expected since "oh" can be spoken rather rapidly and therefore is a prime candidate for digit insertion, deletion, or substitution.

An analysis was also made of errors in the recognition of the training set, and trends very similar to those discussed above were found.

IV. Discussion

In this paper we have presented results that demonstrate major improvements in our ability to recognize relatively unconstrained strings of connected digits (i.e., strings up to 7 digits in length). We have shown that by incorporating information about the time derivatives of the cepstral coefficients, along with instantaneous cepstral coefficients, we can significantly enhance recognizer performance. A summary of the recognizer performance, in each of the 3 modes in which it was tested, is given in Table V. Overall string error rates of less than 3 percent for unknown string lengths and less than 2 percent for known string lengths were obtained on independent testing sets of data for both speaker independent and multispeaker modes. String error rates of less than 1 percent for unknown string lengths and less than 0.5 percent for known string lengths were obtained in the speaker trained case.

These results show that the transitional cepstral information made the recognizer relatively robust to talkers. In another paper, we have shown that the addition of the delta cepstrum analysis significantly improves performance with other vocabularies (e.g., the alphabet) in isolated word recognition tasks [21].

TABLE V
SUMMARY OF STRING ERROR RATES FOR THE THREE RECOGNITION MODES

Recognition Mode	Database	Training Set		Testing Set	
		UL	KL	UL	KL
Speaker Trained	DB50	0.39	0.16	0.78	0.35
Multi-speaker (6 Models Per Digit)	DB50	1.74	0.98	2.85	1.65
Speaker Independent (4 Models Per Digit)	DBTI	1.24	0.36	2.94	1.75

To see how much progress has been made, it is worthwhile contrasting the results presented here with those of earlier studies. In earlier work, using the same databases and recognizer, but with a standard instantaneous cepstral analysis (i.e., without the transitional cepstral information), Rabiner *et al.* reported testing set string error rates of 1.83 percent (UL), and 0.81 percent (KL) in the speaker trained mode, and 6 percent (UL) and 3.4 percent (KL) in the multispeaker mode (using 10 models per digit as opposed to 6 models per digit here). The string error rates reported here are lower by a factor of 2 or more! Furthermore, in the speaker independent mode, the results [22] obtained for the testing set were string error rates of 7.9 percent (UL) and 5.2 percent (KL), again using 10 models per digit. Here the string error rates are lower by a factor of about 3 to 1, based on 4 models per digit. These comparisons strongly point out the advantages of the transitional cepstral information for recognition.

The only other comparison worth making is with the work of Bush and Kopec [5] who also used the TI database for their recognition tests. The best performance results on the testing set, obtained by Bush and Kopec, were 4 percent (UL) and 3 percent (KL) string error rates. The Bush and Kopec results were based on digit models derived from acoustic-phonetic knowledge, using wider bandwidth spectral analysis, with a network representation that handled some difficult cases (e.g., prepausal "oh" or eight), and with an explicit background silence model. The results given here were obtained *fully automatically*, using telephone bandwidth data, with no explicit silence model, and with no rules or corrections for difficult digit sequences.

V. Summary

In this paper we have shown that a very high performance connected digit recognition system can be implemented automatically based on our current understanding. The key to the improvement in performance over earlier implementations was the use of an analysis that included both instantaneous and transitional (time derivative) spectral information. The system was tested in three modes, namely, speaker trained, multispeaker, and speaker independent, and shown to be capable of recognizing digit strings with greater than 97 percent accuracy in all cases.

REFERENCES

[1] H. Sakoe, "Two level DP-matching—A dynamic programming based pattern matching algorithm for connected word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 588–596, Dec. 1979.

[2] C. S. Myers and L. R. Rabiner, "Connected digit recognition using a level building DTW algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 351–363, June 1981.

[3] J. S. Bridle, M. D. Brown, and R. M. Chamberlain, "An algorithm for connected word recognition," *Automat. Speech Anal. Recognition*, J. P. Haton, Ed., pp. 191–204, 1982.

[4] H. Ney, "The use of a one-stage dynamic programming algorithm for connected word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 263–271, Apr. 1984.

[5] M. A. Bush and G. E. Kopec, "Network-based connected digit recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 1401–1413, Oct. 1987.

[6] L. R. Rabiner, J. G. Wilpon, and B. H. Juang, "A segmental *k*-means training procedure for connected word recognition based on whole word reference patterns," *AT&T Tech. J.*, vol. 65, no. 3, pp. 21–31, May/June 1986.

[7] ——, "A model-based connected-digit recognition system using either hidden Markov models or templates," *Comput. Speech Language*, vol. 1, no. 2, pp. 167–197, Dec. 1986.

[8] L. R. Rabiner, A. Bergh, and J. G. Wilpon, "An improved training procedure for connected digit recognition," *Bell Syst. Tech. J.*, vol. 61, no. 6, pp. 981–1001, July–Aug. 1982.

[9] L. R. Rabiner, J. G. Wilpon, A. M. Quinn, and S. G. Terrace, "On the application of embedded digit training to speaker independent, connected digit recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 272–280, Apr. 1984.

[10] S. Furui, "Speaker independent isolated word recognition based on dynamics emphasized cepstrum," *Trans. IECE Japan*, vol. 69, no. 12, pp. 1310–1317, Dec. 1986.

[11] F. K. Soong and A. E. Rosenberg, "On the use of instantaneous and transitional spectral information in speaker recognition," in *Proc. ICASSP 1986*, Tokyo, Japan, Apr. 1986, pp. 877–880, Paper 17.5.1.

[12] Y. Tohkura, "A weighted cepstral distance measure for speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 1414–1422, Oct. 1987.

[13] B. H. Juang, L. R. Rabiner, and J. G. Wilpon, "On the use of bandpass liftering in speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 947–954, July 1987.

[14] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, "An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition," *Bell Syst. Tech. J.*, vol. 12, no. 4, pp. 1035–1074, Apr. 1983.

[15] F. Jelinek, "Continuous speech recognition by statistical methods," *Proc. IEEE*, vol. 64, pp. 532–556, Apr. 1976.

[16] L. R. 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, pp. 1211–1234, July–Aug. 1985.

[17] S. E. Levinson, "Continuously variable duration hidden Markov models for automatic speech recognition," *Comput. Speech Language*, vol. 1, no. 1, pp. 29–45, Mar. 1986.

[18] C. H. Lee and L. R. Rabiner, "A frame synchronous level building algorithm for connected word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, to appear.

[19] A. L. Gorin and D. B. Roe, "Parallel level building on a tree machine," in *Proc. ICASSP 88*, New York, Apr. 1988, pp. 295–298.

[20] R. G. Leonard, "A database for speaker-independent digit recognition," in *Proc. 1984 ICASSP*, Mar. 1984, pp. 42.11.1–4.

[21] L. R. Rabiner and J. G. Wilpon, "Some performance benchmarks for isolated word, speech recognition systems," *Comput. Speech Language*, vol. 2, pp. 343–357, Sept./Dec. 1987.

[22] L. R. Rabiner, J. G. Wilpon, and B. H. Juang, "A performance evaluation of a connected digit recognizer," in *Proc. ICASSP-87*, Dallas, TX, 1987, pp. 101–104, Paper 3.10.1.

Lawrence R. Rabiner (S'62–M'67–SM'75–F'76) was born in Brooklyn, NY, on September 28, 1943. He received the S.B. and S.M. degrees simultaneously in June 1964, and the Ph.D. degree in electrical engineering in June 1967, all from the Massachusetts Institute of Technology, Cambridge.

From 1962 through 1964 he participated in the cooperative plan in electrical engineering at Bell Laboratories, Whippany and Murray Hill, NJ. He worked on digital circuitry, military communications problems, and problems in binaural hearing. Presently he is engaged in research on speech recognition and digital signal processing techniques at Bell Laboratories, Murray Hill. He is coauthor of the books *Theory and Application of Digital Signal Processing* (Englewood Cliffs, NJ: Prentice-Hall, 1975), *Digital Processing of Speech Signals* (Englewood Cliffs, NJ: Prentice-Hall, 1978), and *Multirate Digital Signal Processing* (Englewood Cliffs, NJ: Prentice-Hall, 1983).

Dr. Rabiner is a member of Eta Kappa Nu, Sigma Xi, Tau Beta Pi, The National Academy of Engineering, and is a Fellow of the Acoustical Society of America.

Jay G. Wilpon (M'84–SM'87) was born in Newark, NJ, on February 28, 1955. He received the B.S. and A.B. degrees (cum laude) in mathematics and economics, respectively, from Lafayette College, Easton, PA, in 1977. He received the M.S. degree in electrical engineering/computer science from Stevens Institute of Technology, Hoboken, NJ, in 1982.

Since June 1977 he has been with the Speech Research Department at AT&T Bell Laboratories, Murray Hill, NJ, where he is a member of the Technical Staff. He has been engaged in speech communications research and is presently concentrating on problems in isolated and connected word speech recognition. He has published extensively in this field and has been awarded several patents. His current interests lie in training procedures for both speaker dependent and speaker independent recognition systems, speech detection algorithms, and determining the viability of implementing speech recognition systems for general usage over the telephone network.

Mr. Wilpon received the IEEE Acoustics, Speech, and Signal Processing Society's Paper Award in 1987 for his work on clustering algorithms for use in training automatic speech recognition systems.

Frank K. Soong (S'76–M'82) received the B.S., M.S., and Ph.D. degrees from the National Taiwan University, the University of Rhode Island, and Stanford University, respectively, all in electrical engineering.

Since 1982 he has been a member of the Technical Staff at AT&T Bell Labs, Murray Hill, NJ, first with the Acoustics Research Department and later with the Speech Research Department. His research work includes developing new coding algorithms at medium to low bit rates, investigating spectral distortion measures for speech processing, applying vector quantization to speaker recognition and normalization, and studying instantaneous and transitional spectral information of speech signals for speech and speaker recognition. He was an invited researcher at NTT-ECL, Japan, from 1987 to 1988. At NTT he has developed a phonetically labeled acoustic segment approach to speech processing and applied it to speech recognition, very low bit rate speech coding, and speech synthesis.