A SEGMENTAL *k*-MEANS TRAINING PROCEDURE FOR CONNECTED WORD RECOGNITION

Lawrence R. Rabiner, Jay G. Wilpon, and Biing-Hwang Juang

Lawrence R. Rabiner *is head of the Speech Research Department at AT&T Bell Laboratories in Murray Hill, New Jersey, and* ***Jay G. Wilpon*** *and* ***Biing-Hwang Juang*** *are members of technical staff in that department. Mr. Rabiner, who joined AT&T in 1962, is responsible for speech and speaker recognition research, and speech coding. He has a B.S., M.S., and Ph.D. in electrical engineering from Massachusetts Institute of Technology. Mr. Wilpon, who joined AT&T in 1977, is engaged in speech communications research, concentrating on problems in recognizing isolated- and connected-word speech. He has a B.S. in mathematics and an A.B. in economics from Lafayette College, and an M.S. in electrical engineering and computer science from Stevens Institute of Technology. Mr. Juang, who joined*
(continued on page 31)

Algorithms for recognizing strings of connected words from whole-word patterns have become highly efficient and accurate, although computation rates remain high. Even the most ambitious connected-word recognition task is practical with today's integrated circuit technology, but extracting reliable, robust whole-word reference patterns still is difficult. In the past, connected-word recognizers relied on isolated-word reference patterns or patterns derived from a limited context (e.g., the middle digit from strings of three digits). These whole-word patterns were adequate for slow rates of articulated speech, but not for strings of words spoken at high rates (e.g., about 200 to 300 words per minute). To alleviate this difficulty, a segmental *k*-means training procedure was used to extract whole-word patterns from naturally spoken word strings. The segmented words are then used to create a set of word reference patterns for recognition. Recognition string accuracies were 98 to 99 percent for digits in variable length strings and 90 to 98 percent for sentences from an airline reservation task. These performance scores represent significant improvements over previous connected-word recognizers.

Background

One of the most interesting and promising areas of speech recognition is connected word recognition, where we optimally match a continuous string of words to every (syntactically valid) possible concatenation of word reference patterns. Connected-word recognition ostensibly allows the user to speak normally. There is no need to pause between words, or to overarticulate individual words for clarity. Yet the recognition problem can be solved with a large, but reasonable amount of computation using individual-word reference patterns for the matching procedure.

In a sense, connected word recognizers represent the frontier of speech recognition technology. They can be used for a wide variety of tasks (telephone number dialing, order entry, spelling, voice control, etc.) and can be built with inexpensive hardware.

However, connected-word recognition technology presents some strong limitations:

- The number of vocabulary words for which word reference patterns are suitable is severely limited (generally about 100 words).
- The recognition algorithm (no matter what procedure it uses) is guaranteed to "break" as the talker's articulation rate gets higher and higher.
- Current procedures for training the word-reference patterns are clearly inadequate for fluent strings.

Because training for large vocabulary, connected-word recognition is essentially impractical, the vocabulary size limitation has no obvious remedy. Systems that need large vocabularies (e.g., a voice typewriter) require continuous-recognition algorithms. To "solve" the vocabulary explosion problem, such systems represent words in terms of smaller units (dyads, phonemes, syllables, etc.). Hence, they need to derive reference training patterns only for a finite set of speech units to represent, at least theoretically, any vocabulary of words.

The second and third limitations of current connected-word recognizers are the topic here. How do we improve the word training procedure to increase pattern reliability and robustness, and make it work for fluent strings of words?

Earlier Training Methods

To understand the training philosophy that we adapted, it is worthwhile to review earlier training methods.

The earliest connected-word recognizers used isolated-word training.[1-6] It was the easiest way to derive whole-word patterns that could then be used for checking the connected-word recognition algorithm. Such isolated-word training worked well for slowly articulated word strings, and even for moderate-speed word strings with little coarticulation between adjacent words. However, for highly coarticulated word sequences (such as the digit sequences /38/ and /66/), isolated-word training patterns were often inadequate.

To alleviate some problems associated with using isolated-word training patterns, an embedded-word training algorithm was devised. Isolated-word reference patterns were used to extract word tokens embedded in known three-word strings.[7] Then the so-called embedded patterns were combined with isolated-word patterns to produce an improved set of word reference patterns.

This training procedure was successfully used to improve reference patterns for connected digit recognition[7,8] and for spelled letter recognition of names.[9]

The limitations of even the embedded training procedure should be clear. If each word can occur *only* within a word triplet, the articulation rate of training strings can be, and generally is, vastly different from the articulation rate of longer strings. Thus, the digit two in the string /123/ has significantly different characteristics from each two in the string /3228628/.

Clearly, the best training procedure is to use completely fluent, naturally spoken word strings (whose word identities are known) and develop word models strictly from the word tokens extracted from these strings. This is precisely the procedure described here.

Segmental *k*-Means Training Procedure

Figure 1 is a block diagram of the segmental *k*-means training procedure. As an example, assume that the vocabulary of interest is the set of ten digits (i.e., $V = 10$ words). Also assume that the training strings are a large set of random size, random digit strings (one to seven digits).

The initial set of word reference patterns (stored in the word pattern files) is any convenient set of word pat-

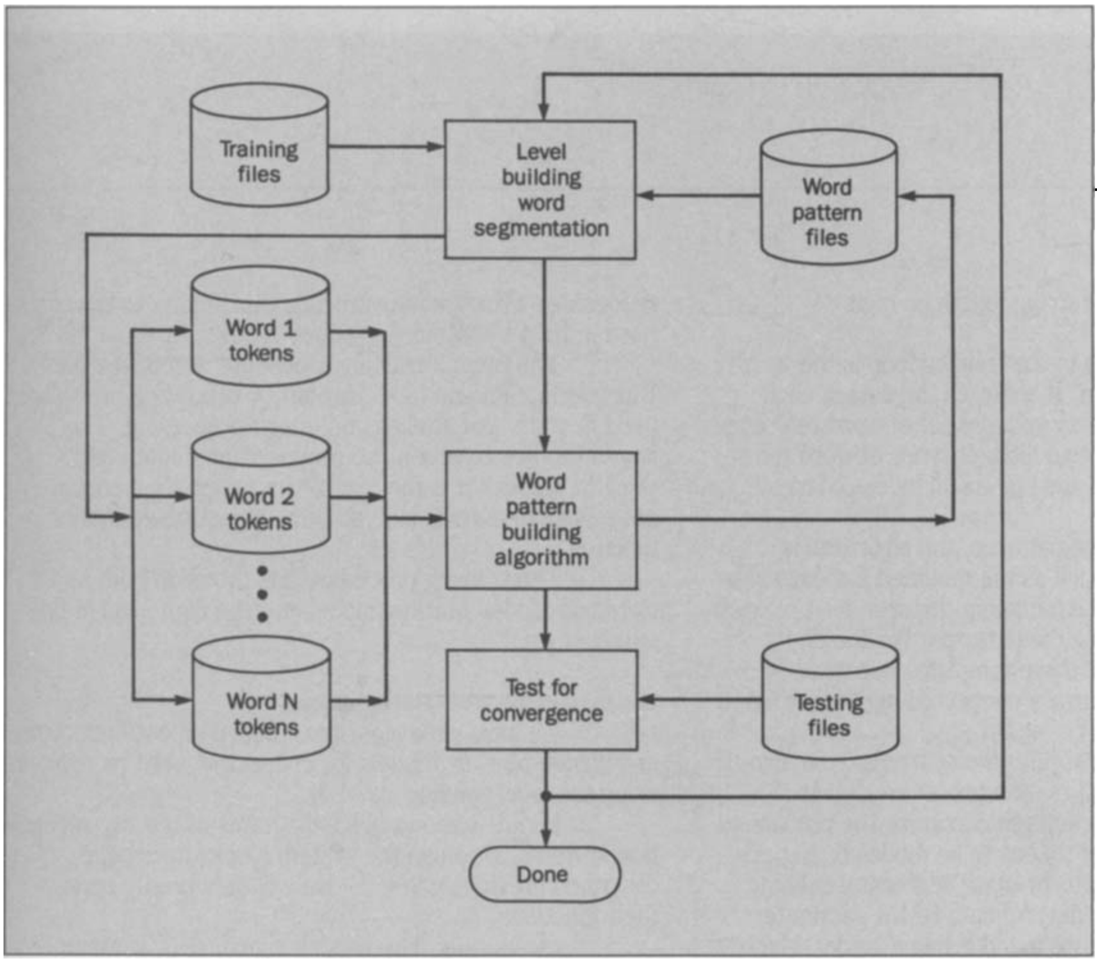

Figure 1. Flow diagram of the continuous-word training procedure.

terns; typically, a set of isolated word patterns. They could be obtained from the talker being trained or from a different talker, or they could be a speaker independent set of patterns derived from an earlier analysis. Alternatively, the initial pattern set could be the continuous-word training patterns from a previously trained talker.

Because the initial set of patterns serves only as a way to start the training procedure, the algorithm is insensitive to the exact choice. For the digits vocabulary, we initially used a speaker independent set of patterns (24 patterns per word) that consisted of both isolated word patterns (12 per word) and embedded word patterns (12 per word).[7]

The first step in the training loop is to segment the strings in the training set into individual word patterns, using a level-building word segmentation algorithm. We used this procedure because the word reference patterns consist of multiple patterns for each word. In a single forward pass, the level building algorithm can efficiently determine the optimum time alignment path between the words in each training string and the entire set of word reference patterns.

Using the level building procedure, each training string is segmented into constituent words that are then sorted into individual word files for further processing. Thus, the digit string /4646934/ would be segmented into seven constituent digits, resulting in three occurrences of the digit /4/, two of the digit /6/, and one of the digits /3/ and /9/. For each constituent digit, the unnormalized autocorrelation vectors were stored in the individual word files.

A reasonably sized training set is required to ensure the training set has enough occurrences of each word. For digit recognition, each digit should occur equally often in strings of different lengths, to give a reasonable sampling of the effects of string length on digit articulation.

The second step in the training loop is the word pattern building algorithm. It analyzes the data in each word token file and gives an updated set of word reference patterns. We considered two distinct types of word reference patterns: templates and statistical hidden Markov models (HMMs).

For template-type patterns, the algorithm is a clustering procedure,[10] such as the modified k-means algorithm, that determines the minimum-distortion set of word templates for representing the patterns. For the digits vocabulary, we found that three templates per word were adequate to give high accuracy connected-digit recognition scores.

For Markov model reference patterns, the algorithm is a parameter reestimation procedure.[11-12] It chooses the HMM coefficients to maximize the probability of occurrence of the word tokens to be modeled. Experience has shown that a state-by-state segmental training procedure works well for determining HMM parameters.[12] For the digits vocabulary, we found a single hidden Markov model per word was adequate.

The final step in the training loop is the test for convergence, the stopping criterion for the training procedure. In theory, one could examine the difference (in some well-defined sense) between the current set of word reference patterns and those of the previous training loop iteration. If this difference is small enough, then the procedure stops.

What is actually done is more pragmatic. An independent set of connected word strings, stored in a testing file, is used to evaluate the recognition performance of the updated set of word reference patterns. If recognition accuracy increases, the training loop is reiterated. Otherwise, convergence is assumed and the training loop ends.

The training procedure forms a closed loop, where each iteration improves the objective function. It reduces the average distance in the template-based recognizer, or increases the likelihood in the model-based recognizer. (The training procedure is similar to the one used in IBM's continuous speech recognizer.[13])

The overall training procedure resembles the EM (estimation and modification) or relaxation procedures used in statistical studies and image processing. The key difference between the proposed procedure and the EM algorithm is the use of the pragmatic performance evaluation test, instead of an objective evaluation function.

The training procedure has provided both template and hidden Markov models for the digits and airline vocabularies.

The Connected Word Recognizers

We have used algorithms based on both template and hidden Markov models for connected word recognition using the level building strategy.

Figure 2 shows block diagrams of the two recognition systems. Because the system blocks have been described previously,[4,11,12,14,15] we will only briefly review their functions.

LPC Analysis. The speech signal, $s(n)$, is assumed to be recorded off a local, dialed-up, telephone line and is sampled at a 6.67-kHz rate.

The speech is preemphasized by a first-order digital network ($H(z) = 1 - 0.95z^{-1}$), and then blocked into 45-ms (300 sample) frames. Adjacent frames are spaced 15-ms (100 samples) apart. A Hamming window is used on each frame, and an eighth-order autocorrelation analysis is done, followed by an eighth-order LPC (linear predictive coding) analysis. Thus, for each frame of speech (as determined by a speech endpoint detector), a set of eight LPC coefficients is generated.

A speech pattern for a string is the sequence (in time) of LPC frame vectors.

Cepstral Transformation. For statistical modeling techniques, it is desirable to convert the LPC frame to a cepstral representation. The set of LPC-derived cepstral coefficients is known to have better statistical properties than the raw LPC coefficients.[16]

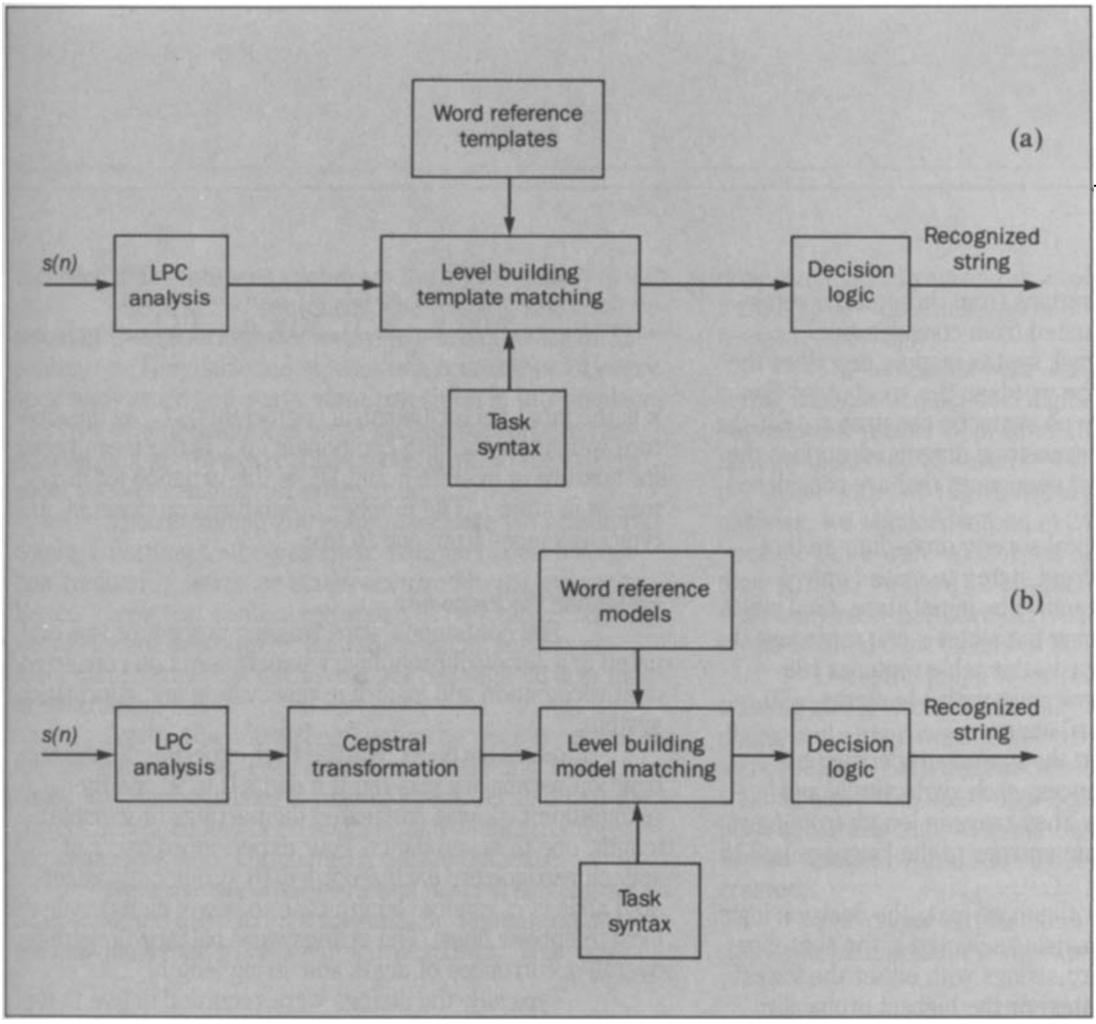

Figure 2. Two recognition systems: (a) template-based recognizer that uses level building and task syntax; (b) recognizer based on a hidden Markov model (HMM).

Although one could extend the set of cepstral coefficients beyond eighth order (based on a Laurent series expansion of the filter polynomial), we chose not to do so. Thus, the cepstral representation used is the set of eight cepstral coefficients per frame (the zero-order cepstral coefficient is not used).

Level-Building Pattern Matching. The level building algorithm determines the optimum sequence of word reference patterns that matches a given connected word string.

If the word reference patterns are templates, then a dynamic time warping procedure is used, within each level, to time align the reference and test patterns.[4] Inherent durational constraints are readily applied by restricting the time alignment path to fixed limits; e.g., a maximum expansion of two to one, and a minimum expansion of half to one.

If the word reference patterns are statistical models, then a viterbi procedure is used, within each level, to align reference models to the test observation sequence.[17] Because our statistical models were restricted to only five states, subword (state) durational constraints are not readily applied. Even word durational constraints are applied only as a level post-processor.[17]

The level building procedure can keep track of multiple word candidates at each ending frame at each level. For tasks such as connected digit recognition, this capability leads to multiple choices of digit strings that can be post-processed (when hidden Markov models are used) using state duration probabilities. Thus, digit strings with grossly different state durations from those of the reference models can be eliminated from consideration.

Task Syntax. The task syntax module describes the syntactic constraints on the words in the vocabulary. For digit sequences, there are no syntactic constraints. For the airline vocabulary, a state transition diagram describes the syntax and defines all word sequences that are considered valid sentences.[17]

We use a topological sorting procedure so that there can be a transition from state j to state l only if $j < l$. Thus, a table—organized by initial state, final state, and vocabulary words joining the states—can represent the grammar. For the airline task, this table requires 198 entries to represent the grammar with 144 states, 450 transitions, and 21 terminal states.

The language that the grammar specified has more than 6×10^9 sentences, each syntactically and semantically well formed. They range in length from four to 22 words. The maximum entropy of the language is 2.15 bits/word.[18]

Decision Logic. For the most part, the decision logic is simple. The recognized string selected is the sentence in the set of allowable word strings with either the lowest distance score (for templates) or the highest probability (for statistical models).

For connected digit sequences recognized from hidden Markov models, word models with state durational distributions process all reasonable candidate strings. Then the string with highest probability is picked.

Hidden Markov Models. We used a left-right, five-state hidden Markov model with a transition matrix $A = a_{ij}$ that satisfies the constraint

$$a_{ij} = 0 \quad j < i, j > i + 2$$

and a continuous mixture density matrix $B = b_j(\mathbf{x})$ of the form

$$b_j(\mathbf{x}) = \sum_{m=1}^{M} C_m N[\mathbf{x}, \mu_{mj}, \mathbf{U}^2_{mj}]$$

$\mathbf{x}$ is the input vector (cepstral coefficients), C_m is the mixture weight for the mth component, μ_{mj} is the mean vector for mixture m in state j, and $\mathbf{U}^2_{mj}$ is the variance for mixture m in state j. The number of mixture components, M, typically ranges from one to five.

Evaluating the Procedure

The continuous word training procedure was evaluated in a series of preliminary experiments on connected digit recognition and an airline reservation and information system.[15]

Connected Digits. For the first experiment, the recognition vocabulary was the ten digits 0 to 9, and the recognition task was connected digit strings of variable length, one to seven digits. Four experienced users of speech recognizers each recorded 1050 randomly generated strings of random length (one to seven digits) over local telephone lines. The strings were balanced regarding overall occurrences of digits and string length.

Typically, the strings were recorded in five to ten sessions, each occurring on a different day and with a new dialed-up telephone line. Each string was manually checked to verify that the talker spoke the correct string. All other processing—including endpoint detection, training, and recognition—was done automatically.

The 1050 strings were divided in half. The first 525—which consisted of 75 one-digit strings, 75 two-digit strings, and up to 75 seven-digit strings—were used as the training set. The second 525 strings (again balanced over digits and string lengths) were used as an independent test set. We will give the results of recognition tests on these connected digit strings later.

Airline Sentences. The second experiment used a vocabulary of 129 airline terms, and the task was the airline reservation and information system.[15] For this experiment, two experienced users of speech recognizers recorded 460 sentences each from the airline task.

The first 260 sentences, the training set, consisted of five repetitions of a carefully selected set of 52 sentences. They included at least one occurrence of every vocabulary word and every state transition in the language. The five sets of sentences were spoken naturally in random sequence. No instructions were given to the talkers about speed or manner of articulation.

The remaining 200 sentences were generated randomly, according to the grammar, with no concern about how frequently words or states occurred in the grammar. Hence, these test sentences tended to be longer (more words) and more complex than the training sentences. Also, more sentences in the test set had digit strings than in the training set.

Again, the 460 airline sentences were recorded in four or five sessions, each on different days and with a new dialed-up telephone line. A manual check verified that the correct sentence had been spoken, and all incorrectly spoken sentences were eliminated from the database.

Connected Digits Results. Each talker's training set of 525 connected digit strings was initially segmented into the individual digit tokens within the strings. This was done automatically using the level building procedure, with a training set that consisted of 36 speaker independent templates per digit.

The templates were derived partially from a clustering analysis of individual digits extracted (also using a segmental *k*-means loop) from 1520 connected digit strings spoken by 19 talkers.[7] (There were 80 digit strings per talker with two to five digits per string.) From this analysis, we obtained a total of 24 templates per digit. But because there were no isolated digits in the set of 1520 digit strings, we used 12 speaker independent, isolated digit templates that were derived from a clustering analysis of the isolated digit tokens of 100 talkers.[19]

Using the initial word-reference set, each talker's training string was automatically segmented into individual digits, and either word templates or hidden Markov models were created. Each digit occurred about 210 times in the training set. From these digit occurrences, either a template set (three templates per digit) or a statistical hidden Markov model (with three or five mixtures per state) was created.

The choice of three templates per digit (TPW) was somewhat arbitrary. We experimented with one to 24

Table I. Average String Accuracy for Variable-Length, Connected-Digit Strings for Both Templates and HMMs

	Template set							
	Speaker independent 36 TPW		Embedded training 3 TPW		Segmental *k*-means 3 TPW		Segmental *k*-means HMM	
Talker	UL	KL	UL	KL	UL	KL	UL	KL
1	80.2	82.5	83.6	84.6	99.1	99.1	97.9	99.8
2	98.5	99.2	92.0	94.3	99.1	99.6	97.9	99.6
3	69.9	76.6	72.8	74.7	98.1	98.1	97.0	98.3
4	87.6	88.6	76.6	85.0	98.5	99.1	98.3	100.0
Average	84.1	86.7	81.3	84.7	98.7	99.0	97.8	99.4

NOTE: HMM = hidden Markov model; TPW = template per word (digit); UL = unknown-length string; KL = known-length string.

Table II. Average String Accuracy of Template-Based Recognizer as a Function of Number of Training Strings

	Number of training strings			
	70		525	
Talker	UL	KL	UL	KL
1	98.5	99.2	99.1	99.1
2	97.7	98.3	99.1	99.6
3	96.2	96.6	98.1	98.1
4	97.3	98.1	98.5	99.1
Average	97.4	98.1	98.7	99.0

NOTE: UL = unknown-length string; KL = known-length string.

TPW, and the results with three TPW were almost identical to those with 24.

At each iteration of the training loop, recognizer performance was evaluated by scoring the performance for the current set of word reference patterns against the test set. We found that, in general, the performance on the first iteration was almost as good as the second iteration. However, all iterations beyond that point showed no real performance improvement. We attributed this result to the excellent initial set of word reference patterns used to start the training procedure and to the consistency of the talkers used in the experiment.

The connected digit recognizer's performance was measured using two criteria. First, we assumed that the string length was unknown (UL); thus, the recognizer would sometimes make deletion or insertion errors. We also measured performance for a known string length (KL). Clearly, KL performance length was always better than UL performance (Table I).

As a performance check, the template-based recognizer was also scored two ways. One used the original, speaker independent, startup set of 36 templates per word, and the other, a set of speaker dependent templates that were derived from an earlier study on combining isolated and embedded patterns.

Table I gives the UL and KL results of the connected digit recognition tests. It shows average string accuracy for the 525-string test set for template- and HMM-based recognizers. For all four talkers, average string accuracies were 98 to 100 percent for templates and HMMs derived from the segmental *k*-means training procedure.

The performance differences among talkers and between the two types of recognizer are small (about 1 percent) and are probably not statistically significant.

For the template-based recognizer, we see that the templates derived from the segmental *k*-means procedure performed significantly better than either the speaker-independent template set or the speaker-dependent embedded training set. In particular, we see that string accuracy increased by 12.3 to 17.4 percent across the different conditions. These results show clearly the degree of improvement in word reference pattern that is obtained from the continuous training procedure.

We can also obtain results on average digit accuracy, because the average string length was four digits (equal occurrences from one to seven digits). For the template-based recognizer, the average digit accuracy was 99.7 percent for unknown length strings and 99.8 percent for known lengths.

We also ran the training procedure for the template-based system, with only 70 training strings (instead of 525). Thus, only ten strings of each length (one to seven digits) were used. Table II compares the recognizer's performance with the two training sets.

We can see that reducing the training set size by a factor of 7.5 degrades performance about one percent. Whether this performance degradation is significant or not depends on the intended application. However, Table II shows that overall performance is only weakly sensitive to the amount of training data.

Airline Vocabulary Results. The training set of 260 airline sentences was segmented into individual words (via

Table III. Average Sentence and Word Accuracy for Test Set of Airline Sentences

	Template-based				HMM-based			
	1 robust TPW		Segmental *k*-means 3 TPW		SI model		Segmental *k*-means 1 mixture/state	
Talker	Sentence accuracy	Word accuracy	Sentence accuracy	Word accuracy	Sentence accuracy	Word accuracy	Sentence accuracy	Word accuracy
1	60.4	90.4	89.4	98.9	39.4	85.9	87.0	99.0
2	62.0	91.3	97.5	99.8	33.0	87.7	89.9	98.1
Average	61.2	90.9	93.5	99.4	36.2	86.8	88.5	98.6

NOTE: HMM = hidden Markov model; SI = speaker independent; TPW = template per word; UL = unknown-length string; KL = known-length string.

the level-building procedure) using an initial word reference set that consisted of 12 speaker independent templates per word. These templates were derived from a clustering analysis of 100 isolated tokens of each word;[20] 100 talkers spoke each word once. Both template sets and hidden Markov models were derived from the initial segmentation.

The training process was iterated several times but, like the digits, the best recognition performance generally occurred at the second iteration. However, unlike the digits, hidden Markov models with one mixture per state performed slightly better than those with three mixtures per state. This was a result of the sparse training data for many words in the vocabulary. Several words had only five occurrences, which was grossly inadequate for designing a hidden Markov model with three mixtures per state. We still used three templates per word, even though a single template per word would have been adequate for most words.

Table III gives the results of the airline vocabulary test. It shows both average sentence and word accuracy for the template- and HMM-based systems. Also included are results for one robust TPW, template-based system and for a set of speaker-independent models for the HMM-based system.

For word patterns derived from continuous training, the template-based system outperformed the HMM-based system by about 5-percent average string accuracy. This is a result of the HMM-based recognizer's inability to apply fine temporal constraints (such as state durations) on the matching procedure. Unlike the digits where a postprocessor could apply state duration constraints, the lack of fine temporal structure led to word errors between different word models, which led to sentence errors.

The most obvious way to correct this limitation is to use hidden Markov models that have built-in durational constraints. An alternative would be to use considerably larger models where the number of states per word matches the average word duration. We avoided this alternative because we want our models to rely on the sound structure of the word; hence, maintaining a small number of states is desirable.

Let us compare the template-based recognizer's performance for the simple, robust, single TPW reference set versus the segmental *k*-means, three TPW reference set. We see a 32.3-percent average improvement with the continuous training procedure. Similarly, for the HMM-based recognizer and the speaker-independent set of models versus the segmental *k*-means set, we see an average increase in sentence accuracy of 60.7 percent.

To verify that sparse training data caused the degraded performance of the continuously trained hidden Markov models, we used the same models to score the training set sentences.

For the two talkers, the string accuracies on the training set were 98.1 percent and 100 percent, using hidden Markov models with three mixtures per state. This result merely punctuates our contention that the training set did not adequately represent the variability of words in the true airline environment. Furthermore, the $M = 3$ mixture model outperformed the $M = 1$ mixture model on the training set (by 1.7 percent on average). This emphasizes that a better model on a training set is only better if the training set's statistics are the same as the test set's, which was clearly not the case for these sentences.

One final point; sentences in the training set averaged 9.67 words while testing sentences averaged 11.85 (about two more words per test sentence).

Discussion

Our results show the power of the continuous training technique. The digit string accuracies of 98 to 99 percent represent strong advances over earlier recognition systems. Even the string accuracies for the airline vocabulary are significantly higher than those for earlier word training procedures.

We intend to test the segmental *k*-means training procedure for a large class of talkers, and have recorded both digit strings and airline sentences from 50 talkers who had no previous experience with speech recognizers. The continuous training procedure will be tested on each talker individually (speaker dependent training), and the talkers (or some subset of them) will be combined to give a speaker independent set of reference patterns. We hope that this procedure will lead to a speaker independent set of digit patterns that are suitable for connected digit recognition.

For the airline vocabulary, both training set inadequacy and variability of words in continuous speech may prove major obstacles in developing a speaker independent set of connected words.

References

1. L. R. Rabiner and M. R. Sambur, "Experiments in the Recognition of Connected Digits," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-24, No. 2, April 1976, pp. 170-182.
2. L. R. Rabiner and C. E. Schmidt, "Application of Dynamic Time Warping to Connected Digit Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-28, No. 4, August 1980, pp. 377-388.
3. H. Sakoe, "Two Level DP-Matching — A Dynamic Programming Based Pattern Matching Algorithm for Connected Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-27, No. 6, December 1979, pp. 588-595.
4. C. S. Myers and L. R. Rabiner, "Connected Digit Recognition Using a Level Building DTW Algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-2, No. 3, June 1981, pp. 351-363.
5. J. S. Bridle, M. D. Brown, and R. M. Chamberlain, "An Algorithm for Connected Word Recognition," *Automatic Speech Analysis and Recognition,* edited by J. P. Haton, D. Reidel Publishing Co., Dordrecht, Holland, 1982, pp. 191-204.
6. H. Ney, "The Use of a One-Stage Dynamic Programming Algorithm for Connected Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-32, No. 2, April 1984, pp. 263-271.
7. L. R. Rabiner, A. Bergh, and J. G. Wilpon, "An Improved Training Procedure for Connected Digit Recognition," *The Bell System Technical Journal,* Vol. 61, No. 6, July-August 1982, pp. 981-1001.
8. 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 Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-32, No. 2, April 1984, pp. 272-280.
9. L. R. Rabiner, J. G. Wilpon, and S. G. Terrace, "On the Application of Embedded Training to Connected Letter Recognition for

Directory Listing Retrieval," *AT&T Bell Laboratories Technical Journal,* Vol. 63, No. 3, March 1984, pp. 459-477.

10. J. G. Wilpon and L. R. Rabiner, "A Modified *k*-Means Clustering Algorithm for Use in Isolated Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-33, June 1985, pp. 587-594.
11. 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," *The Bell System Technical Journal,* Vol. 12, No. 4, April 1983, pp. 1035-1074.
12. 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 Technical Journal,* Vol. 64, No. 6, July-August 1985, pp. 1211-1234.
13. F. Jelinek, "Continuous Speech Recognition by Statistical Methods," *Proceeding of the IEEE,* Vol. 64, No. 4, April 1976, pp. 532-556.
14. C. S. Myers and L. R. Rabiner, "A Level Building Dynamic Time Warping Algorithm for Connected Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-29, No. 2, April 1981, pp. 284-297.
15. S. E. Levinson and L. R. Rabiner, "A Task-Oriented Conversational Mode Speech Understanding System," *Speech and Speaker Recognition,* edited by M. R. Schroeder, S. Karger AG., Basil, Switzerland, 1985, pp. 149-196.
16. J. D. Markel and A. H. Gray Jr., *Linear Prediction of Speech,* Springer-Verlag, Berlin, Heidelberg, New York, 1976.
17. L. R. Rabiner and S. E. Levinson, "A Speaker-Independent, Syntax-Directed, Connected Word Recognition System based on Hidden Markov Models and Level Building," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-33, No. 3, June 1985, pp. 561-573.
18. M. M. Sondhi and S. E. Levinson, "Computing Relative Redundancy to Measure Grammatical Constraint in Speech Recognition," *Proceedings of the ICASSP,* Tulsa, Oklahoma, April 1978, pp. 409-412.
19. L. R. Rabiner, S. E. Levinson, A. E. Rosenberg, and J. G. Wilpon, "Speaker Independent Recognition of Isolated Words Using Clustering Techniques," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-27, No. 4, August 1979, pp. 336-349.
20. J. G. Wilpon, L. R. Rabiner, and A. Bergh, "Speaker Independent Isolated Word Recognition Using a 129 Word Airline Vocabulary," *Journal of the Acoustical Society of America,* Vol. 72, No. 2, August 1982, pp. 390-396.

Biographies (continued)

AT&T in 1982, does research on speech recognition, speech coding and enhancement, and stochastic processes. He has a B.S. in electrical engineering from National Taiwan University, and an M.S. and Ph.D. in electrical and computer engineering from the University of California at Santa Barbara.

(Manuscript received February 19, 1986)