A Continuous Training Procedure for Connected Digit Recognition

L. R. Rabiner
J. G. Wilpon
B. H. Juang

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

Abstract. Algorithms for recognizing strings of connected words from whole word patterns (either templates or statistical models) have advanced to the point of high efficiency and accuracy. Although the computation rate of these connected word recognition algorithms remains high, advances in VLSI hardware make even the most ambitious connected word recognition tasks practical with todays technology. The greatest impediment to the successful utilization of connected word recognizers is the difficulty in extracting reliable, robust whole word reference patterns. In the past, connected word recognizers have relied on either isolated word reference patterns (which are trivially obtained), or reference patterns derived from limited context strings of words (e.g. the middle digit from strings of 3 digits). The resulting whole word reference patterns were adequate for slow rates of speech articulation, but proved inadequate when users spoke strings of words at high rates (e.g. on the order of 200-300 words per minute). To alleviate this difficulty, a training procedure for extracting whole word patterns from naturally spoken word strings has been implemented and is described here. The training procedure is essentially a k-means loop in which a set of known word strings is segmented into individual words based on matching an initial set of word reference patterns (typically a speaker independent set of isolated word reference patterns is used). The segmented words are then used to create an updated set of word reference patterns (either via clustering methods, for templates or via statistical techniques, for word models), which are then used in the segmental loop to give an updated set of word tokens from the labelled training set. This procedure is iterated until a stable set of whole word reference patterns is obtained.

The training procedure was implemented and tested in a connected digits recognition task. For this task, *string* accuracies (on variable length strings with from 1-7 digits) on the order of 98-99% were obtained.

I. Introduction

One of the most interesting and promising areas of speech recognition is connected word recognition, in which a continuous string of words is recognized by optimally matching it to every (syntactically valid) possible concatenation of word reference patterns. This problem is interesting because it obstensibly allows the user of such a recognizer to speak the string to be recognized in a continuous manner (i.e. there is no need to pause between words, or to overarticulate individual words for clarity), and yet the recognition problem can be solved with a large but not unreasonable amount of computation using individual word reference patterns for the matching procedure. Hence connected word recognizers represent, in a sense, the frontier of speech recognition technology since they can be utilized in a wide variety of tasks (e.g. telephone number dialing order entry, spelling, voice control etc.), and they can be built with relatively inexpensive hardware.

There are some strong limitations to connected word recognition technology; the number of vocabulary words for which word reference patterns are suitable is severely limited (generally on the order of 100 words); the recognition algorithm (no matter which specific procedure is used) is guaranteed to "break" as the rate of articulation of the talker gets higher and higher; and the current procedures for training the word reference patterns are clearly inadequate for fluent strings.

The first limitation, namely the vocabulary size limitation, has no obvious remedy for connected word technology. This is because the training for connected word recognition is totally impractical. For systems which have a requirement for large vocabularies (e.g. the voice typewriter), the class of continuous recognition algorithms is required. Such systems "solve" the vocabulary explosion problem by representing words in terms of smaller units (e.g. dyads, phonemes, syllables etc), and hence need only derive reference training patterns for the finite set of speech units to be able to represent, at least theoretically, any vocabulary of words.

It is the second and third limitations of current connected word recognizers that is the topic of discussion of this paper, namely how to improve the word training procedure so as to increase the resulting pattern reliability and robustness, and to make it work for fluent strings of words. To understand the philosophy of training adapted here, it is worthwhile reviewing earlier methods used to train connected word recognizers. The earliest connected word recognizers essentially used isolated word training [1-3] since this was the easiest way to derive whole word patterns which could then be used to check out the connected word recognition algorithm. Such isolated word training worked very well for slowly articulated word strings, and even worked well for moderate speed word strings when there was very little coarticulation between adjacent words. However for coarticulated word sequences (e.g. the digit sequences /38/, /66/ etc), isolated word training patterns were often inadequate.

To alleviate, partially, the problems associated with using isolated word training patterns, an embedded word training algorithm was devised in which the isolated word reference patterns were used to extract word tokens that were embedded in known 3-word strings [4]. The so-called embedded patterns were combined with the isolated word patterns to give an improved set of word reference patterns. This training procedure was successfully utilized for giving improved reference patterns for connected digit recognition [4], and for spelled letter recognition of names.

The limitations of even the embedded training procedure should be clear. By restricting each word to occur only within a word triplet, the rate of articulation of the training strings can be, and generally is, vastly different from the rate of articulation of longer strings. Thus the digit two in the string /123/ has significantly different characteristics from each of the digits two in the string /3228628/. Clearly, the "ultimate" training procedure is to use completely fluent, naturally spoken word strings (whose word identities are known) and to develop word models strictly from the word tokens extracted from such naturally spoken strings. This is precisely the procedure to be described in this paper.

The improved word training procedure works as follows. We assume an initial set of word reference patterns exists. (In all cases we have used a previously obtained, speaker independent set of word reference patterns as the initial set.) For each labelled string in the training set, a dynamic time warping (DTW) alignment between the labelled string and the concatenated word reference patterns corresponding to the string is obtained via the level building DTW algorithm [2]. In this manner, each labelled string (in the training set) is then segmented into individual words, and a new set of word reference patterns is created from the segmented training data. This new set of word reference patterns could be either templates (obtained via a clustering procedure [5]) or they could be statistical models (e.g. a hidden Markov model, HMM, obtained via a reestimation procedure [6]). (In fact we have actually used both types of patterns and will present results on both methods later in this paper). Using the updated set of word reference patterns, the segmental loop is repeated until no further

ICASSP 86, TOKYO CH2243-4/86/0000-1065 $1.00 © 1986 IEEE

improvements in reference patterns are obtained (as evidenced by scoring the resulting reference patterns using an independent set of word strings). The training procedure is similar to the training used in the IBM continuous speech recognizer [7].

We have tested the new word training procedure on a connected digit recognition task. To date only 4 talkers have been tested. However the results on this small set of talkers indicate the high degree of performance improvement over previous implementations. In particular, digit string accuracies (in a speaker trained mode) of between 98 and 99% were obtained for variable length strings of from 1 to 7 digits.

II. The Segmental k-means Training Procedure

Figure 1 shows a block diagram of the segmental k-means training procedure. By way of example we assume that the vocabulary of interest is the set of ten digits (i.e. $V=10$ words), and that the training strings are a large set of random size, random digit strings (of length from 1 to 7 digits).

The initial set of word reference patterns (as stored in the word pattern files) is any convenient set of word patterns. Typically the initial set is a set of isolated word patterns. The isolated word patterns could be obtained from the talker being trained, from a different talker, or they could be a speaker independent set of patterns derived from an earlier analysis. Alternatively, the initial word reference pattern set could be the continuous word training patterns from a previously trained talker. Since the initial set of word reference patterns only serves as a means of bootstrapping the training procedure, the algorithm is relatively insensitive to the exact choice. For the digits vocabulary, we used an initial set of word reference patterns that was a speaker independent set of patterns (24 patterns per word), consisting of both isolated word patterns (12 per word), and embedded word patterns (12 per word) [4].

The first step in the training loop is the segmentation of the strings in the training set into individual word patterns. This is accomplished using a level building word segmentation algorithm. The level building procedure is used since the word reference patterns consist of multiple patterns for each word. 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 in a single forward pass. Using the level building procedure, each training string is segmented into the constituent words which are then sorted into individual word files for further processing. Thus the digit string /4646934/ would be segmented into the 7 constituent digits, resulting in 3 occurrences of the digit /4/, 2 occurrences of the digit /6/, and 1 occurrence of the digits /3/ and /9/. For each constituent digit, the unnormalized autocorrelation vectors were stored in the individual word files.

A reasonable sized training set is required to insure that a sufficient number of occurrences of each word occur within the training set. For digits it is also desirable that each digit occur equally often in strings of different lengths so as 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 which analyzes the data in each individual word token file and gives an updated set of word reference patterns. There are two distinct types of word reference patterns that we have considered, namely templates and statistical hidden Markov models. For template-type word reference patterns, the word pattern building algorithm is a clustering procedure, such as the modified k-means algorithm [5], which determines the minimum distortion set of word templates for representing the word patterns. For the digits vocabulary, 3 templates per word has been found adequate to give high accuracy connected digit recognition scores. For Markov model reference patterns, the word pattern building algorithm is a model

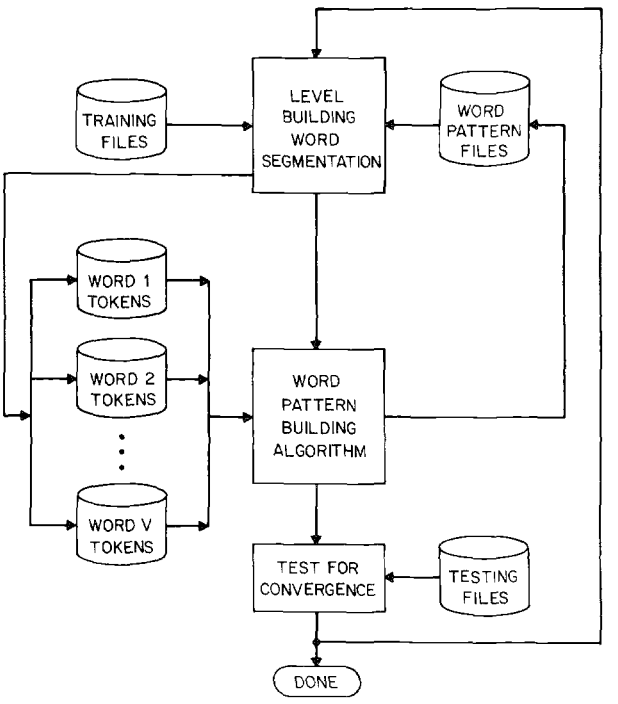

Fig. 1 Flow diagram of the continuous word training procedure.

parameter reestimation procedure [6,8], which chooses the HMM coefficients to maximize the probability of occurrence of the word tokens to be modelled. Experience has shown that a state-by-state segmental training procedure works well for determining HMM parameters [8]. For the digits vocabulary, a single HMM per word has been found to be adequate.

The final step in the training loop is the test for convergence, or the stopping criterion on the entire 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, and if this difference is sufficiently small, the procedure is terminated. 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 is terminated.

The training procedure forms a closed loop in that each iteration improves the objective function, i.e. reduces the average distance in the template-based recognizer, or increases the likelihood in the model-based recognizer. The training procedure has been used to provide both template and HMM models for the digits vocabulary.

III. The Connected Word Recognizers

We have used both template-based and HMM-based algorithms for connected word recognition using the level building (LB) strategy. Figure 2 shows block diagrams of the two recognition systems. Since the details of each of the system blocks have been previously described [2,6,8], we will only briefly review the functions of some of the blocks here.

3.1 LPC Analysis

The speech signal, $s(n)$, is assumed to be recorded off of a local, dialed-up, telephone line, and is sampled at a 6.67 kHz rate. The speech is first preemphasized by a first order digital network

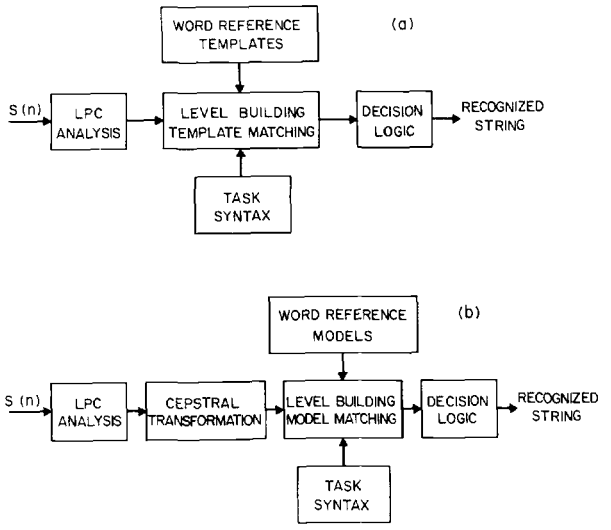

Fig. 2 (a) Block diagram of template-based recognizer using level building and task syntax; (b) block diagram of HMM-based recognizer.

($H(z)=1-0.95z^{-1}$), and is then blocked into 45 msec (300 sample) frames, with adjacent frames spaced 15 msec (100 samples) apart. Each frame is Hamming windowed and a $p=8^{th}$ order autorcorrelation analysis is performed, followed by an 8^{th} order LPC analysis. Thus for each frame of speech (as determined by a speech endpoint detector) a set of 8 LPC coefficients is generated. A speech pattern for a string is the sequence (in time) of LPC frame vectors.

3.2 Cepstral Transformation

For statistical modelling 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. Although one could extend the set of cepstral coefficients beyond 8^{th} order we have not chosen to do so. Thus the cepstral representation used is the set of 8 cepstral coefficients per frame (the zeroth order cepstral coefficient is not used).

3.3 Level Building Pattern Matching

The level building algorithm is used to determine the optimum sequence of word reference patterns which 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 [2]. Inherent durational constraints are readily applied by restricting the time alignment path to fixed limits, e.g. a maximum expansion of 2 to 1, and a minimum expansion of 1/2 to 1. 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 [8]. In this case, sub-word (i.e. state) durational constraints are not readily applied, and even word durational constraints are only applied as a level post-processor [8]. This is because the statistical models we use are restricted to only a small number of states (i.e. 5 states).

The level building procedure has the capability of keeping 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, which can be post-processed, in the case of HMM's, using state duration probabilities. In this manner digit strings with grossly different state durations from those of the reference models, can be eliminated from consideration.

3.4 Decision Logic

For the most part the decision logic is simple. The recognized string is chosen as the sentence in the set of allowable word strings, with either the lowest distance score (for templates) or with the highest probability (for statistical models). In the case of connected digit sequences recognized from HMM's, all reasonable candidate strings are first post-processed by word models with state durational distributions, and then the post-processed string with highest probability is chosen.

3.5 HMM Models

The type of HMM we have used is a left-right 5 state model with a transition matrix $A = a_{ij}$, which satisfies the constraint

$$a_{ij} = 0 \quad j < i\,, \quad j > i + 2 \tag{1}$$

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

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

where $\mathbf{x}$ is the input vector (cepstral coefficients), C_m is the mixture weight for the m^{th} component, μ_{mj} is the mean vector for mixture m in state j, and $\mathbf{U}_{mj}^2$ is the variance for mixture m in state j. The number of mixture components, M, is typically in the range 1 to 5.

IV. Experimental Evaluation of the Training Procedure

The continuous word training procedure was evaluated as follows. The recognition vocabulary was the ten digits (0-9), and the recognition task was connected digit strings of variable length, 1 to 7 digits. For this experiment 4 experienced users of speech recognizers each recorded 1050 randomly generated strings of random length (from 1 to 7 digits) over local, dialed-up, telephone lines. The strings were balanced with regard to overall occurrences of digits, and with regard to string length. Typically the strings were recorded over 5 to 10 recording sessions, each recording session occurring on a different day and with a new dialed-up telephone line. Each string was manually checked to verify that the talker did actually speak the correct string. All other processing, including endpoint detection, training, and recognition, was done fully automatically.

The 1050 strings were divided in half. The first 525 strings, consisting of 75 strings each of 1 digit, 75 strings each of 2 digits, up to 75 strings each of 7 digits, were used as the training set. The second 525 strings (again balanced over digits and string lengths) were used as an independent test set. The results of the recognition tests on these connected digit strings are given later in this section.

4.1 Results for Connected Digits

The training set of 525 connected digit strings, for each talker, was initially segmented (automatically) into the individual digit tokens within the strings using the LB procedure, with a training set consisting of 36 speaker independent templates per digit. The set of speaker independent templates was derived partially from a clustering analysis of the individual digits extracted (also using a segmental k-means loop) from 1520 connected digit strings spoken by 19 talkers (i.e. 80 digit strings per talker with from 2-5 digits per string) [4]. From the analysis of these connected strings, a total of 24 templates per digit were obtained. In addition, because there were no isolated digits in the set of 1520 digit strings, a set of 12 speaker independent isolated digit templates was used. These speaker independent isolated digit templates were derived from a clustering analysis of isolated digit tokens of 100 talkers [9].

Using the initial word reference set described above, the 525 training strings, for each talker, were automatically segmented into individual digits, and either word templates or HMM's were created. For each digit there were approximately 210 occurrences within the training set.

From these 210 digit occurrences either a template set with 3 templates per digit, or a statistical HMM with either 3 or 5 mixtures per state, was created. (The choice of 3 templates per digit was somewhat arbitrary; we experimented with from 1 to 24 templates per digit, and the results with 3 templates per digit (TPW) were almost indistinguishable from those with 24 templates per digit).

At each iteration of the training loop the recognizer performance was evaluated by scoring the performance of 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 that of the second iteration; however all iterations beyond that point tended to produce no real improvement in performance. This result is attributed to the excellent initial set of word reference patterns which was used to bootstrap the training procedure.

The performance of the connected digit recognizer was measured using two criteria. First we assumed that the string length was unknown (UL), in which case the recognizer would sometimes make deletion or insertion errors. We also measured performance for the case when the string length was known (KL). Clearly performance for the KL case was always better than that for the UL case. As a check against performance, the template-based recognizer was also scored using the original, speaker-independent, bootstrapping set of 36 templates per word, and with a set of speaker dependent templates derived from an earlier study on combining isolated and embedded patterns.

The results of the connected digit recognition tests are given in Table I. This table shows average string accuracy for the 525 string test set for both template-based and HMM-based recognizers. Results are given for both the UL and KL cases. The results show average string accuracies in the range of 98 to 100% for both templates and HMM's derived from the segmental k-means training procedure for all 4 talkers. The differences in performance among talkers and between the two types of recognizer are small (on the order of 1%) and are probably not statistically significant.

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

Although we have been presenting results on average string accuracy, we can also obtain results on average digit accuracy by using the fact that the average string length was 4 digits (i.e. equal occurrences from 1 to 7 digits). Thus for the template-based recognizer, the average digit accuracy was 99.7% for unknown length strings, and 99.8% accuracy for known length strings.

Talker Independent	Template Set						Segmental k-Means HMM	
	Speaker Training −36 TPW		Embedded k-Means −3 TPW		Segmental Model −3 TPW			
	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

TABLE I

Average String Accuracy for Variable Length Connected Digit Strings for Both Templates and HMM's.

For the template-based system the training procedure was also run with only 70 training strings (instead of the 525 strings used for Table I). Thus only 10 strings of each length (from 1 to 7 digits) were used in the training. A comparison of the performances of the template-based recognizer with the different size training sets is given in Table II.

It can be seen that by reducing the training set size by a factor of 7.5, the performance degrades by about 1%. Whether this performance degradation is significant or not depends on the intended application. However the importance of the results of Table II is that they show that overall performance is only weakly sensitive to the amount of training data.

Talker	Number of Training Strings			
	70		525	
	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

TABLE II

Average String Accuracy of Template-Based Recognizer as a Function of the Number of Training Strings

REFERENCES

[1] H. Sakoe, "Two Level DP-Matching — A Dynamic Programming Based Pattern Matching Algorithm for Connected Word Recognition," *IEEE Trans. on Acoustics, Speech, and Signal Processing,* Vol. ASSP-27, No. 6, pp. 588-595, Dec. 1979.

[2] C. S. Myers and L. R. Rabiner, "Connected Digit Recognition Using a Level Building DTW Algorithm," *IEEE Trans. on Acoustics, Speech, and Signal Processing,* Vol. ASSP-29, No. 3, pp. 351-363, June 1981.

[3] J. S. Bridle, M. D. Brown, and R. M. Chamberlain, "An Algorithm for Connected Word Recognition," *Automatic Speech Analysis and Recognition,* J. P. Haton, Ed., pp. 191-204, 1982.

[4] L. R. Rabiner, A. Bergh, and J. G. Wilpon, "An Improved Training Procedure for Connected Digit Recognition," *Bell System Tech. Journ.,* Vol. 61, No. 6, pp. 981-1001, July-Aug. 1982.

[5] J. G. Wilpon and L. R. Rabiner, "A Modified k-Means Clustering Algorithm for Use in Isolated Word Recognition," *IEEE Trans. on Acoustics, Speech, and Signal Processing,* Vol. ASSP-33, pp. 587-594, June 1985.

[6] 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 System Tech. Journ.,* Vol. 12, No. 4, pp. 1035-1074, April 1983.

[7] F. Jelinek, "Continuous Speech Recognition by Statistical Methods," *Proc. IEEE,* Vol. 64, No. 4, pp. 532-556, April 1976.

[8] 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. Journ.,* Vol. 64, No. 6, pp. 1211-1234, July-August 1985.

[9] L. R. Rabiner, S. E. Levinson, A. E. Rosenberg, and J. G. Wilpon, "Speaker Independent Recognition of Isolated Words Using Clustering Techniques," *IEEE Trans. on Acoustics, Speech, and Signal Processing,* Vol. ASSP-27, No. 4, pp. 336-349, Aug. 1979.