HMM Clustering for Connected Word Recognition

L. R. Rabiner, C. H. Lee, B. H. Juang, and J. G. Wilpon
AT&T Bell Laboratories
Murray Hill, New Jersey 07974

Abstract. It has been shown that when there are a sufficiently large number of training tokens of a given speech recognition unit (e.g., words, phones, syllables, etc.), it is generally worthwhile clustering the training tokens into two or more clusters and then creating either templates or statistical models from each individual cluster. Such techniques have been used successfully for isolated and connected word recognition for a number of years. Almost all of the clustering techniques, to date, have been based on conventional template-based techniques whereby a training set is broken into clusters on the basis of time aligned, pairwise distance scores between each pair of tokens in the training set. Because of the recent preponderance of recognizers based on statistical models (Hidden Markov Models), it was felt that an HMM clustering procedure would be more effective than a template-based approach — especially in the sense that it could be integrated directly into the HMM parameter estimation procedure. This paper describes such an HMM clustering procedure and discusses its application to connected word systems and to large vocabulary recognition based on phone-like units. It is shown that the conventional approach of maximizing likelihood is easily implemented but does not work well in practice as it tends to give improved models of tokens for which the initial model was generally quite good, but does not lead to improvements for tokens which are poorly represented by the initial model. As such we have developed a splitting procedure which initializes each new cluster (statistical model) by splitting off all tokens in the training set which were poorly represented (in the likelihood sense) by the current set of models. This procedure is shown to be highly efficient and gives excellent recognition performance in connected word tasks. In particular, for speaker independent connected digit recognition, using 2 HMM-clustered models, on the TI database, the recognition performance is as good as or better than previous results using from 4-6 models/digit obtained from template-based clustering.

I. Introduction

It has been shown that when a large amount of training data is available, for creating either templates or statistical models, performance of speech recognizers generally improves when more than one template or model is created for each of the recognition units [1, 2]. As such, a wide range of methods for clustering a set of training data into 2 or more clusters have been proposed and studied [3-5]. Most of the clustering methods, used for speech recognition, have been oriented towards the creation of multiple templates from a given set of word training data. For systems based on word templates, such procedures are both acceptable and have been shown to be capable of providing good representations of the variants in word pronunciation across talkers of the training set. Most recently, however, speech recognition systems based on statistical models have proliferated [6-8]. Although the clustering techniques used for template-based recognizers can be (and have been) successfully applied to the design of multiple statistical models for each vocabulary word, it seemed likely that a model-based clustering procedure would be more efficient than a template-based approach. The purpose of this paper is to describe such a clustering procedure, based on the use of hidden Markov models (HMM), and discuss its performance on the task of recognizing a string of digits in a speaker independent manner.

In the following sections we first briefly review the structure of an HMM-based recognizer which uses whole word models for recognition. Next we describe the training procedure, i.e. the way in which we first obtain representative tokens of whole words from continuous speech, and then the way in which we estimate sets of model parameters. Next we describe the template-based approach to creating multiple models of each vocabulary word. Finally we discuss the two methods we have investigated for direct HMM clustering of the data. We illustrate the potential gains in efficiency by comparing the recognition results on a speaker independent connected digit recognition task.

II. Overview of HMM Recognizer

A block diagram of the overall HMM recognizer is shown in Figure 1. There are essentially three steps in the recognition algorithm, namely:

1. Spectral analysis – the speech signal is converted to a set of LPC derived cepstral (weighted) and delta-cepstral (weighted) vectors.
2. HMM pattern matching – the sequence of spectral vectors of the unknown speech signal is matched against a set of whole word patterns (hidden Markov models) using a frame-synchronous Viterbi alignment procedure and a word grammar (generally realized as a finite state network) which specifies which word combinations can occur for valid sentences in the language. The output of the HMM pattern matcher is a set of candidate sentences (word strings), generally of different lengths (number of words in the string), ordered by likelihood score.
3. Postprocessor – The output candidate strings are subjected to further validity tests, e.g. word and/or state duration penalties are assessed, to eliminate (penalize) strings that are unreasonable with respect to duration constraints etc. The postprocessor chooses the most likely word string from the valid candidate strings.

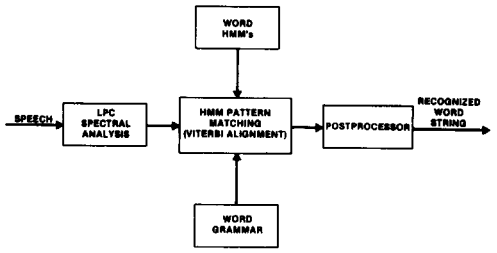

Figure 1 – Block Diagram of Continuous Speech Recognizer Based on Word HMM's.

II.1 HMM Characterization of Words

Figure 2 shows the form of the HMM used to characterize individual words. (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. This is equivalent to a null transition between word models.) The models are first order, left-to-right, Markov models. The features of the models include:

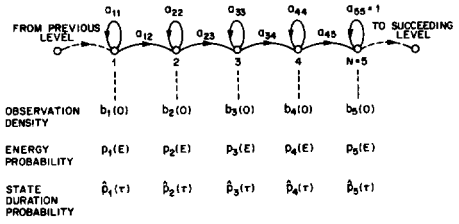

Figure 2 – Form of HMM Used to Characterize Individual Words.

– Variable number of models per word (designed via the clustering methods to be discussed later)

CH2673-2/89/0000-0405 $1.00 © 1989 IEEE

- Variable number of states per model (typically from 5 to 10 states per word model has proven adequate for most applications)
- Variable number of mixtures per state (typically from 2 to 9 mixtures per state have been used).

The spectral density in each state is characterized by a mixture of Gaussian densities. Log energy is characterized , in each state, by an empirically measured histogram; similarly state duration is characterized by an empirically measured histogram. Word duration is assumed to be Gaussian distributed.

II.2 Training the HMM's

In order to build (multiple) word models from a training set of labelled continuous speech, the first step is to optimally segment the continuous speech into individual words. For this task, a segmental k-means training procedure has been shown to be an effective way of converging at the optimum segmentation into words [9]. A block diagram of the segmentation procedure is given in Figure 3.

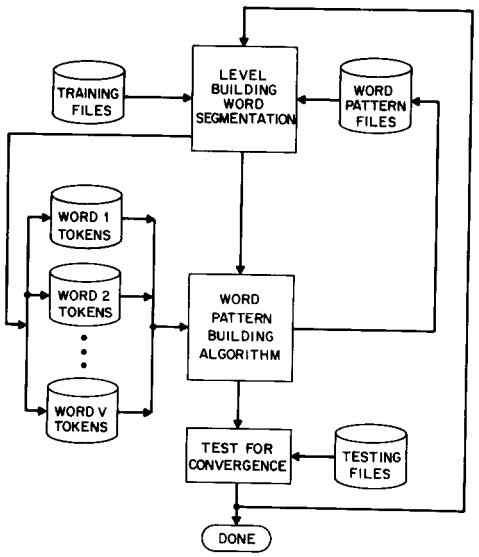

Figure 3 – Block Diagram of the Segmental K-Means Training Procedure.

We assume an initial set of word models is available. This initial set of models can be from a different talker or set of talkers, or can be from an iteration of the training loop. If no initial word models are available, the procedure can be bootstrapped from a set of isolated word occurrences of the vocabulary, or it can even be bootstrapped by assuming a uniform initial segmentation of the continuous speech into words.

Given the initial word model files, and the training files (which consist of continuous sentences of different lengths), an HMM pattern matching procedure is used to optimally segment the training strings into individual word tokens which are stored in word token files. A word model building algorithm (i.e. an estimation procedure for determining parameters of the word HMM's) is used to give an updated set of word models. (The word model building algorithm is itself a segmental k-means training procedure.) The above procedure is iterated until the difference in likelihood scores of the word models, in consecutive iterations, is sufficiently small.

III. Building Multiple Models Per Word

One by-product of the training procedure described in the previous section is an optimal segmentation of continuous speech into words. In cases where there is a large amount of training data (typically for speaker independent recognition), it has traditionally been found that the creation of multiple patterns (typically these have been templates) for each word improves recognition performance significantly. This has also been shown to be the case for statistical models for the task of connected digit recognition of a large, speaker independent, database [2].

The way in which multiple models have traditionally been designed is via a clustering procedure in which all the tokens for a single word are clustered into 2 or more sets, and a single model (or template) is designed for all the tokens within each cluster set. This procedure, although quite reasonable, is not consistent with the HMM methodology because the clustering procedures have all been based on some type of vector quantization or matrix quantization procedure, which in turn is based on measuring accumulated distance along a dynamic time alignment path. In an effort to increase the efficiency of the clustering we proposed and studied two HMM-based clustering procedures. We call the two procedures likelihood clustering and threshold clustering. We now describe the methods.

III.1 HMM Likelihood Clustering

The objective of maximum likelihood model training is to estimate the set of model parameters which gives the maximum likelihood based on the given set of training data. In creating multiple HMM's for the training set, the maximum likelihood objective can easily be extended in a straightforward manner. However the resulting set of models does not necessarily minimize the recognition error when tested on an independent set of test data. The standard assumption is that if a given model (or set of models) provides a better fit (in a maximum likelihood sense) to a training set of words than alternative models, it would then provide better fits to an independent test set of the words, and hence improve recognition performance.

The way in which we do likelihood clustering, based on HMM's, is as follows. Assume, for the v^{th} word in the vocabulary, we have designed the best (maximum likelihood) model, called λ_v. Further assume that for word v there is a set of Q_v word tokens in the training set. We denote this set as $T^v = \{T_1^v, T_2^v, ..., T_{Q_v}^v\}$ and associated with each token T_i^v is the likelihood score, $\mathcal{L}(T_i^v)$, based on the current model (or set of models).

The basic idea of likelihood clustering is to split the model, λ_v, into two models, namely λ_v^+ and λ_v^- by keeping all parameters of the models fixed except the spectral means for each component of each mixture in each state. The spectral means of each mixture are determined as

$$\mu_v^+(d) = \mu_v(d)(1 + (-1)^q \epsilon) \tag{1a}$$

$$\mu_v^-(d) = \mu_v(d)(1 - (-1)^q \epsilon) \tag{1b}$$

where $\mu_v(d)$ is the spectral mean of the d^{th} component of the original model, ϵ is a small number (typically 0.005 to 0.01) and q is a random variable whose value is either 0 or 1. Essentially the splitting of Eq. (1) is similar to the vector quantization splitting algorithm [10], except that it is applied to each dimension of each spectral vector independently. The result of the splitting procedure is a pair of models that are differentially different in the spectral component means.

The motivation behind the use of Eq. (1) is to avoid the influence of the observation vector length in the splitting (perturbation) process. Since the observation vector consists of LPC cepstral components whose length is directly related to the degree of spectral flatness rather than the spectrum itself, it is susceptible to many sources of distortions. A splitting procedure of the type used in Eq. (1) will minimize the effects of the various sources of distortions.

Once the model has been split, the segmental k-means procedure is reiterated on the pair of models until convergence is achieved. This procedure can be iterated to go from 2 to 3 models, for example, by determining the model with either the most tokens assigned to it, or by determining the model with the greatest total likelihood (sum of token likelihoods) and splitting that model.

III.2 HMM Threshold Clustering

A second way of clustering tokens, based on likelihoods obtained from HMM's, is to separate out from the set T^v, those tokens whose likelihood scores fall below some fixed or relative threshold, i.e. separate out all tokens with poor likelihood scores and try to create a new model out of the "outliers." The idea here is that improving

likelihood for tokens with high likelihood scores is unlikely to lead to significant performance improvements in recognition; however if some structure could be found from tokens whose likelihood scores, using the current set of models, is low, then potentially significant improvements in performance could result. The downside risk of threshold clustering is that some of the tokens assigned to the new model could be seriously flawed (e.g. errors in speaking etc.) and could adversely affect the resulting model. The other problem with threshold clustering is determination of a proper or suitable threshold for separating the training set into clusters. The choice of this threshold is generally a matter of guesswork and performance could be very sensitive to this parameter.

As in the case of likelihood clustering, once the tokens have been clustered, the segmental k-means training algorithm is used to give the optimal set of parameters for each of the models. Furthermore the procedure of threshold clustering is equally amenable to creating a new model from n initial models per word as it is from 1 model per word. However, experience indicates that the threshold parameter should get smaller as the number of models per word increases.

It should be clear that likelihood clustering leads to higher overall word likelihood scores than threshold clustering since the modelling of the vast majority of word tokens is improved by likelihood clustering whereas only a small number of word tokens get significantly improved likelihood scores with threshold clustering. However we will see in the next section that the performance improvement from threshold clustering is significantly higher than that obtained from likelihood clustering.

IV. Experimental Evaluation

To study the efforts on recognition performance of the proposed HMM clustering procedures, and to compare these results to those obtained from matrix quantization clustering, we used the TI connected digit database which consisted of 112 adult test talkers speaking a total of 8565 strings of digits (variable in length with from 1 to 7 digits per string), and an independent set of 113 adult talkers speaking 8578 digit strings [11]. Details of the database characteristics are given elsewhere [11].

Using the isolated digit occurrences as a bootstrap, the segmental k-means procedure was used to generate the best single model per digit for each of the 11 digits (the zero-nine and oh). We then used the likelihood and threshold clustering algorithms to generate a 2-model per digit set. In order to choose the threshold for the threshold clustering algorithm, several statistics of the training tokens based on a single model per digit were measured and shown in Table 1. Shown in this table, for each of the digits, is the count, maximum likelihood, minimum likelihood, 20% threshold and 10% threshold likelihood over the training tokens, where the α percentage threshold is defined as the likelihood score where α% of the tokens have likelihoods below this threshold.

Word	Count	$\mathscr{L}_{MAX}$	$\mathscr{L}_{MIN}$	$\mathscr{L}_{20\%}$	$\mathscr{L}_{10\%}$
Zero	2573	21.5	1.9	11.6	10.3
One	2572	22.0	−0.6	13.3	11.8
Two	2565	21.8	−1.1	12.9	11.2
Three	2563	22.7	−5.1	13.1	11.7
Four	2567	22.7	−0.2	14.2	12.7
Five	2522	24.2	3.9	15.6	14.1
Six	2570	23.6	3.0	14.4	13.0
Seven	2559	22.1	2.9	14.3	12.8
Eight	2552	23.3	−3.7	13.4	11.7
Nine	2529	22.3	4.2	14.2	12.8
Oh	2553	22.7	−7.3	11.7	10.0

Table 1
Likelihood Scores for Speaker Independent Digit Tokens Using 1 Model Per Digit

It can be seen that although there is very little variability in $\mathscr{L}_{MAX}$ for the digits, there is considerable variability in $\mathscr{L}_{MIN}$ ranging from −7.3 to 3.9. Based on some preliminary experimentation it was decided to set the threshold to $\mathscr{L}_{20\%}$ so as to initially include the tokens with the lowest 20% of the scores to define the new model. After iteration, the number of tokens assigned to the new model generally increased (typically to about 35% of the training tokens).

IV.1 Results

The recognition results on the independent testing set of connected digits are given in Table 2 and plotted in Figure 4. Table 2 shows string error rates on both the training and testing sets, for both unknown length (UL) and known length (KL) strings. The top part of the table is the results based on matrix quantization clustering; the middle part is the results based on likelihood clustering; the bottom part is the results based on threshold clustering.

Clustering Procedure	Number of Models Per Digit	Training Set		Testing Set	
		UL	KL	UL	KL
Matrix Quantization	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
Likelihood	1	2.45	1.09	4.02	2.38
	2	1.76	0.71	3.98	2.39
Threshold	1	2.04	1.06	3.59	2.31
	2	1.04	0.37	2.84	1.60
	3	1.07	0.40	2.93	1.61

Table 2
Digit String Error Rates (%) for 3 Clustering Procedures

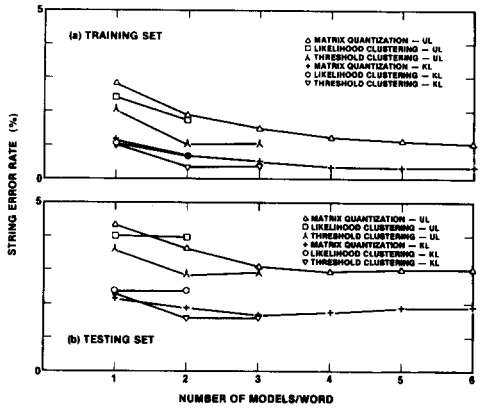

Figure 4 – Plots of String Error Rate Versus Number of Models Per Digit for the Different Clustering Procedures for the Training Set Data (Part a) and the Testing Set Data (Part b).

Unfortunately it is difficult to compare the results of the 3 clustering procedures against each other since several improvements to the HMM recognizer were incorporated between the time the matrix quantization results were obtained, and the time the threshold clustering techniques were obtained. However several observations can be made about the results, namely:

- the matrix quantization clustering requires 3-4 models per digit to give the best performance on the test set; however performance steadily gets better on the training set with up to 6 models per digit
- the likelihood clustering procedure does not improve performance at all on the testing set; however it does give substantial improvement on the training set when 2 models per digit are used. This again points out the obvious result that merely increasing likelihood of the training data is insufficient to guarantee improved performance on an independent test set.
- the threshold clustering procedure gives a significant improvement in performance, on both the test and training sets, when going from 1 to 2 models per digit; further increases to 3 models per digit do

not lead to any further performance gains.

It can also be seen from Figure 4 that both the matrix quantization clustering with 4 models/digit and threshold clustering with 2 models/digit yield almost the same performance scores on both the training and testing sets. It can also be seen in Figure 4 that increasing the number of models per digit has almost no effect on the performance of the recognizer. To determine whether this bottoming off was inherent to the digits vocabulary or was due to the amount of training data, a final experiment was performed in which all the connected digit data from the 225 talkers was used to train a single model per digit system. The resulting set of single digit models was then used to evaluate the performance on the original training set and on the original test set (which was now part of the new training set). The resulting string error rates are shown in Table 3.

Number Of Models Per Digit	Training Set		Testing Set	
	UL	KL	UL	KL
1	2.10	1.20	2.39	1.34

Table 3
Digit String Errors Rates (%) for Combined Training and Testing Sets

It can be seen that the UL string error rate is between 2.10 and 2.39% and the KL string error rate is between 1.20 and 1.34%. Hence the results using 2 models per digit from threshold clustering are within about 0.5% for UL strings and 0.3% for KL strings. These results indicate an inherent vocabulary error of about 1.3 to 2.3% for this recognizer, and show that the amount of training is probably sufficient for obtaining the best performance based on the current recognition algorithm.

V. Summary

We have shown that it is possible to create a set of multiple statistical whole word models for each word in the vocabulary of a continuous speech recognizer using HMM clustering techniques, in much the same way as conventional distance based clustering was used previously. We proposed two such clustering procedures, one based on globally increasing likelihood of the entire training set of word tokens, the other based on locally increasing likelihood for a small set of the tokens whose likelihood scores were the poorest in the training set. We showed that the threshold clustering led to significant performance improvements for the task of speaker independent, connected digit recognition whereas the likelihood clustering didn't improve recognition performance at all.

REFERENCES

1. L. R. Rabiner, S. E. Levinsion, A. E. Rosenberg and J. G. Wilpon, "Speaker Independent Recognition of Isolated Words Using Clustering Techniques," *IEEE Trans. Acoust., Speech, and Signal Processing*, Vol. ASSP-27, No. 4, pp. 336-349, Aug. 1979.
2. L. R. Rabiner, J. G. Wilpon and F. K. Soong, "High Performance Connected Digit Recognition Using Hidden Markov Models," *IEEE Trans. Acoust., Speech, and Signal Processing*, (to appear).
3. S. E. Levinson, L. R. Rabiner, A. E. Rosenberg and J. G. Wilpon, "Interactive Clustering Techniques for Selecting Speaker Independent Reference Templates for Isolated Word Recognition," *IEEE Trans. on Acoust., Speech, and Signal Processing*, Vol. ASSP-27, No. 2, pp. 134-141, Apr. 1979.
4. H. Spath, *Cluster Analysis Algorithms*, Ellis Harwood Ltd., Chichester, 1980.
5. J. G. Wilpon and L. R. Rabiner, "A Modified *k*-Means Clustering Algorithm for Use in Speaker Independent Isolated Word Recognition," *IEEE Trans. Acoust., Speech, and Signal Processing*, Vol. ASSP-33, No. 3, pp. 587-594, June 1985.
6. F. Jelinek, "Continuous Speech Recognition by Statistical Methods," *Proc. IEEE*, Vol. 64, No. 4, pp. 532-556, April 1976.
7. K. F. Lee, "Large Vocabulary Speaker Independent Continuous Speech Recognition: The SPHINX System," Ph.D. Thesis, Carnegie Mellon Univ., Pittsburgh, PA, 1988.
8. L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, (to appear).
9. L. R. Rabiner, J. G. Wilpon and B. H. Juang, "A Segmental *K*-Means Training Procedure for Connected Word Recognition," *AT&T Tech. J.*, Vol. 65, No. 3, pp. 21-31, May-June 1986.
10. Y. Linde, A. Buzo and R. M. Gray, "An Algorithm for Vector Quantization," *IEEE Trans. on Comm.*, Vol. COM-28, No. 1, pp. 84-95, Jan. 1980.
11. R. G. Leonard, "A Database for Speaker Independent Digit Recognition," *Proc. 1984 ICASSP*, pp. 42.11.1-4, March 1984.