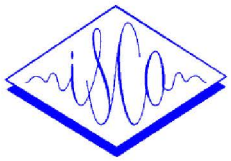

2nd European Conference on
Speech Communication and Technology
EUROSPEECH '91
Genova, Italy, September 24-26, 1991

RECENT WORK IN CONTINUOUS SPEECH RECOGNITION USING THE CONNECTIONIST VITERBI TRAINING PROCEDURE

Michael A. Franzini; Telefonica Investigacion y Desarollo; Emilio Vargas, 6;
28043 Madrid SPAIN

Alex H. Waibel; School of Computer Science; Carnegie Mellon University;
Pittsburgh, PA 15213 USA

Kai-Fu Lee; Apple Computer Corporation; Cupertino, CA USA

ABSTRACT

Hybrid methods which combine hidden Markov models (HMMs) and connectionist techniques take advantage of what are believed to be the strong points of each of the two approaches: the powerful discrimination-based learning of connectionist networks and the time-alignment capability of HMMs. Connectionist Viterbi Training (CVT) is a simple variation of Viterbi training which uses a back-propagation network to represent the output distributions associated with the transitions in the HMM. The work reported here represents the culmination of three years of investigation of various means by which HMMs and neural networks (NNs) can be combined for continuous speech recognition. This paper describes the CVT procedure, discusses the factors most important to its design and reports its recognition performance. Several changes made to the system over the past year are reported here, including: (1) the change from recurrent to non-recurrent NNs, (2) the change from SPHINX-style phone-based HMMs to word-based HMMS, (3) the addition of a corrective training procedure, and (3) the addition of an alternate model for every word. The CVT system, incorporating these changes, achieves 99.1% word accuracy and 98.0% string accuracy on the TI/NBS Connected Digits task ("TI Digits").

Keywords: *hybrid systems, neural networks, back propagation, TI Digits, viterbi training, Connectionist Viterbi Training, CVT*

1 Introduction: The Hybrid Strategy

Recent work in continuous speech recognition has focused on augmenting existing hidden Markov model (HMM) based techniques with other methods. One direction this research has taken is towards the use of powerful *discrimination* methods instead of the Maximum Likelihood Estimation (MLE) procedures typically used for training HMMs. Since speech recognition entails *discriminating* among speech units, learning procedures which are defined explicitly in terms of performing a discrimination task may be better suited to the task than MLE.

Another focus of recent work with HMM-based speech recognizers has been on modelling speech parameters directly, rather than using the drastically reduced representations of the speech signal produced by vector quantization (VQ). Systems which vector quantize have a distinct disadvantage, being deprived of information which may be of use in the recognition process. One approach to this problem has been to use continuous density HMMs. However, these systems incorporate assumptions about the distributions of speech parameters which may be inaccurate. (See [1].)

Connectionist learning procedures are designed to perform accurate *discrimination*, and they operate directly on real-valued parameters, without making any strong assumptions about the distributions of these parameters. Since the energy functions typically used in connectionist learning maximize the system's ability to discriminate among classes of input patterns, these procedures are well suited to speech recognition applications, in which the usual goal is to discriminate among words or phones. Most connectionist models include inputs defined over a continuous range of real numbers and exhibit no advantage with discrete inputs. Integrating these models into HMMs can relieve the need for VQ, while adding discrimination-based learning. Hence, such hybrid methods have been the subject of a great deal of recent investigation (e.g., [2, 3, 4]).

In building hybrid connectionist/HMM systems, speech recognition is viewed as a *static pattern classification* problem combined with a *time alignment* problem. These systems take advantage of the ability of connectionist networks to discriminate accurately among classes in static pattern classification problems. They use HMM technology to find the optimal time alignment based upon the output of the connectionist component of the system.

In this paper, we describe the Connectionist Viterbi Training (CVT) procedure, which is one such hybrid system. We present a general overview of the system, describe its components, and report a series of recent experiments in which we improved the performance of the system by more than 50% on the TI Digits task.

2 System Overview

The CVT system consists of a neural network (NN) and a hidden Markov model (HMM). These two components are not independent; the training of each depends on the other.

A fundamental idea underlying the architecture of the CVT system is that the connectionist section of the system performs a speech *classification* task and the HMM part of the system performs a *time alignment* task.

In the earliest version of this system, the NN looked at a wide window of speech and produced as its output a hypothesis about the identity of the word in its input window. These hypotheses were generated for input windows in every position on the input data. Then, a viterbi search was used to find the optimal path through these hypotheses. In this version of the system, the NN and HMM components were entirely independent; the outputs of the NN were simply passed for processing to the HMM.

In the most recent version of the CVT system, the two components of training are integrated. The outputs of the neural network no longer correspond to *linguistic* entities (as they did in a previous version of the system, which had output units corresponding to words and phones); they now are defined in terms of the HMM architecture. Each NN output unit maps to one transition in the HMM.

Figures 1a and 1b illustrate the the primary components of the system. In the first phase of processing, the system passes one frame of speech (along with several frames of context) to the NN, which outputs a vector of floating point numbers; this vector will serve as the output probabilities for the HMM. In the second phase of processing, once one vector of output probabilities has been generated for every frame of input speech, a Viterbi alignment is performed to determine the most likely path through the HMM. During training, this is a "forced alignment" (i.e. forced to pass through the correct word sequence), and the results of the alignment are used for re-training the NN. During recognition, the Viterbi alignment is free to pass through all words, and the sentence recognized is determined by observing the words entered.

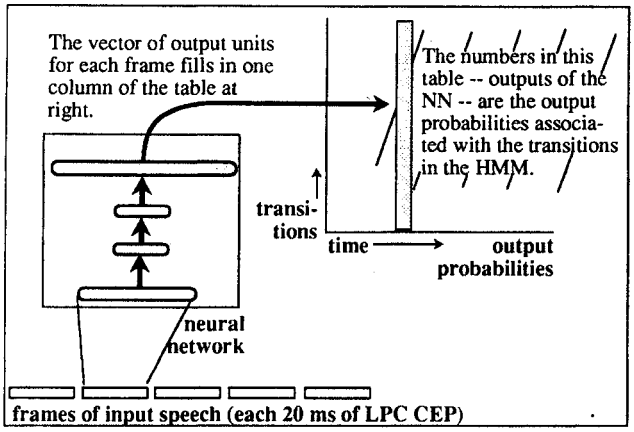

Figure 1a: NN Generation of Output Probabilities

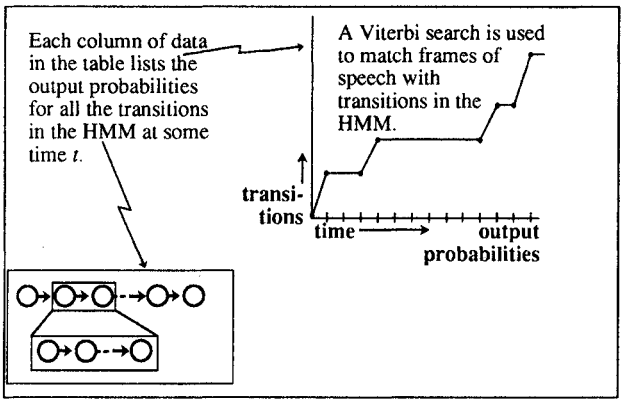

Figure 1b: HMM alignment of speech

3 The TI Digits Task

The Texas Instruments Connected Digits Recognition Task (commonly known as "TI Digits") has become one of the standard tasks on which recognition performance of systems is assessed. The database consists of studio-quality dialectically-balanced recordings of about 10,000 utterances of digit strings ranging in length from one to seven. The vocabulary includes the words "one" through "nine," "oh" and "zero."

The TI Digits task has several features which make is attractive for performance evaluation purposes:

- *TI Digits is a small but practical task.* Few speech recognition tasks are both useful applications and small, fairly easily manageable tasks. Speaker-independently recognizing continuous strings of numbers spoken in a natural voice is probably one of the most useful tasks in speech recognition. [1] Having a vocabulary of 12 words which are not highly confusable, this task is well suited to investigations of new algorithms which require many experimental trials. Larger, more complex tasks would not allow as many experimental paths to be tested, due to the greatly increased time per experiment.
- *TI Digits has ample training and testing data.* The limited size of training corpi is often one of the biggest problems that investigations in speech recognition must tackle. The need for large training corpi is especially poignant in neural-network based systems, which require a great deal of training data for good generalization.

The data, as provided by the NBS, was sampled at 20 KHZ. Before use for training or testing our system, the speech was downsampled to 16 KHz and pre-emphasized with a filter of $1 - 0.97z^{-1}$. Then, a Hamming window with a width of 20 ms was applied every 10 ms. Autocorrelation analysis with order 14 was followed by LPC analysis with order 14. Finally, 12 LPC-derived cepstral coefficients and one power value were computed for each frame.

4 Connectionist Architecture

4.1 Current CVT NN architecture

The current version of the CVT system uses a four-layer[2] network which accepts as input 91 speech coefficients and produces as output 120 floating point numbers between zero and one. The input consists of one 20 ms. frame of speech with three frames of context on each side (10 ms overlap between adjacent frames), and the output includes one value for every transition in the HMM. The two hidden layers each contain 34 units. Hence, the total number of connections in the network is 8,330. The network is illustrated in Figure 2.

The considerations that were made in designing the network included:

1. *Choosing the optimal number of layers* - We found that there was a significant performance benefit for using four layers instead of three. The classification error rate was reduced by nearly an order of magnitude when the number of layers in the network was increased from three to four.[3] The addition of a fourth layer resulted in a significantly longer training period, due to the slower convergence which is typically observed is deeper networks.[4] Adding a fifth layer did not produce any performance benefit and increased the training time by an order of magnitude over that of the four-layer network.
2. *Choosing the optimal number of units per hidden layer* - We found that the recognition performance of the network did not depend on the number of units in each hidden layer. However, when there were very few units in each hidden layer (fewer than ten), the convergence of the network was so slow that we were unable to complete training. [5]

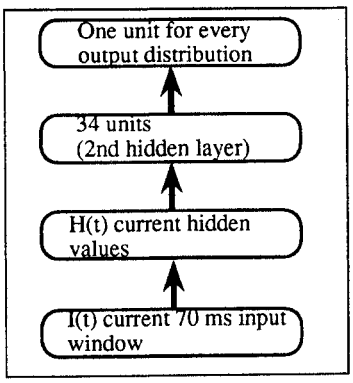

Figure 2:
The Network Used in the Current Version of the System

[1] The task of credit card purchase approval requires millions of person hours per year of spoken digit recognition - performed by human beings - which could be replaced by an automated system such as the one described in this report. One application for which speech recognition will be not just economical but essential is that of dialing cellular telephones in cars; many states in the US and several countries in Europe are enacting laws which prohibit manual dialing of car phones while in motion.

[2] The convention used here is that the term "layer" refers to a layer of *units* in the network; hence, a *four-layer* network has three layers of *weights*.

[3] This experiment was performed with a network trained to classify 500 ms blocks according to the digit to which they belonged. This network will not be described here.

[4] This is an empirical observation made by the authors and applies to networks trained for speech recognition. The same observation has been made by others not working on speech recognition - e.g., Hinton (personal communication).

3. *Choosing between recurrence and non-recurrence (and the structure of the recurrent mechanism when present)* – We have performed extensive investigations of the benefits of various types of recurrent networks, which we will mention only briefly here. We found that the best configuration for a recurrent network was that shown in Figure 3. This architecture is similar to that described by Elman [6]; however, in our network, there are ten groups of "history" or "representation" units, where Elman has only one. In our experiments, the network was unable to retain information across more than two time steps when only one set of history units was used. The system peformed 6% better with the recurrent version of the network than with the non-recurrent version, but, as discussed below, we felt that the computational cost was too high to justify this benefit.

4. *Determining the topology of connections between layers* – Although we have not examined different patterns of connection in the context of the complete CVT system, we did investigate the impact of using sparse connections between layers on training data classification performance. We found that accuracy was degraded by 10 - 50%, as the density of connections between layers varied between the maximum ("fully connected") and a pattern of local connections in which each unit had a fan-in of ten.[6]

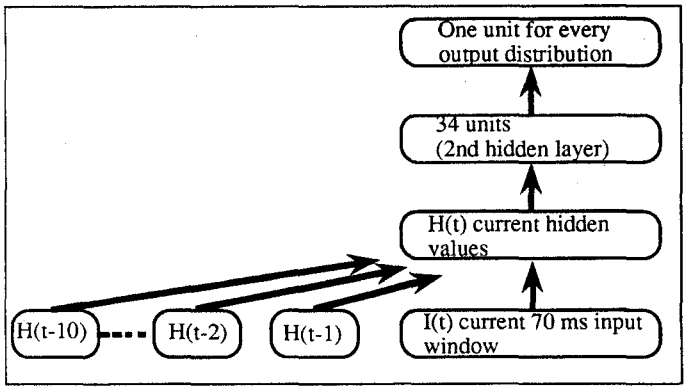

Figure 3:
The Recurrent Network Used in a Previous Version of the System

5 HMM Architecture

5.1 Current CVT HMM architecture

In the current version of the system, the HMM architecture is similar to that described by Bakis [7]. The system uses word models in which each transition corresponds on average to two frames of speech in a word. This is close to the optimal HMM topology reported by Picone [8] for this task; he found that the best configuration uses word models with one transition for every frame of speech in a prototypical utterance of the word modelled. We used half this number of transitions in order to reduce the computational cost of training and recognition.

Figure 4a illustrates the word models used in the current version of the CVT system, and Figure 4b shows the simpler Bakis-style models upon which our models are based. In our version of the word models, the duration controls are significantly tighter, since self-loops are not permitted.

5.2 Phone models vs word models

In a previous version of the system, we used phone-based HMMs with exactly the same topology as those used in the SPHINX system [9]. (See Figure 5.) When we switched from these to the word-based HMMs described above, the system performance improved by 40%.

[5] A past experiment [5] showed that NN performance on a speech classification task degraded slightly when the number of hidden units was reduced drastically (to about 2 or 3) and reached a plateau quickly (when the number of hidden units reached about 8).

[6] The term "fan-in" refers to the number of incoming connections to a unit.

The disadvantage of the phone models is that each transition in the HMM has to model a variety of speech frames, which are not highly localized within words. Specifically, we believe that the disadvantages of the phone models are owed to (1) parallel transitions, (2) self-loops, and (3) too few transitions per word; the best HMM architecture is that which models at the lowest level, with the most rigid correspondence between transitions and speech.

6 Hybrid Training

6.1 How the system works: a more detailed look

The training procedure proceeded as follows:

1. *Initialization.*
   - Use SPHINX to train a set of discrete HMMs on this task.
   - Perform a forced Viterbi alignment of all utterances in the training set using this set of HMMs. This alignment establishes a mapping from frames of speech in the input to transitions in the HMMs.
2. *First iteration.*
   - Train a connectionist network on the pairings (from frames of speech to HMM transitions) produced by the initial alignment.
   - Use the initial HMMs, including the SPHINX-trained transition probabilities, but replace the discrete-HMM output distributions with the distributions encoded by the network. That is, discard the VQ codebooks and lookup tables, and use the network to generate output probabilties from this point on.
3. *Subsequent iterations.*
   - Perform another forced Viterbi alignment of all the training data using the new HMMs (which now include the connectionist network).
   - Re-train the network on the new pairings from this alignment.
   - Re-estimate the transition probabilities. The probability of taking a transition from state i to state j is re-estimated as the ratio of the number of times transition ij was taken (in the state sequences generated by the Viterbi alignment) to the total number of times that transitions were taken from state i to any state.
   - Check the performance of the new model on a "validation set" of utterances. If improvement is observed, perform another iteration, beginning with a new forced Viterbi alignment.

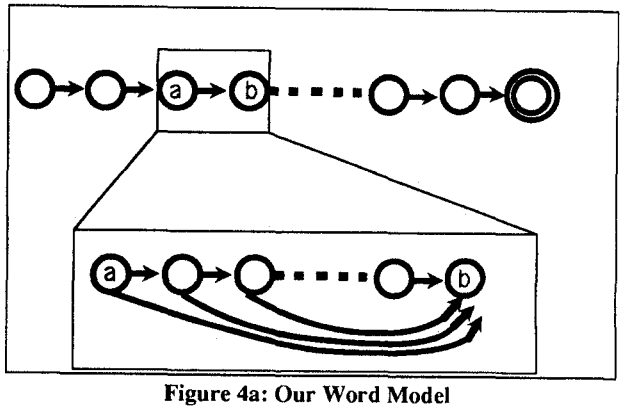

Figure 4a: Our Word Model

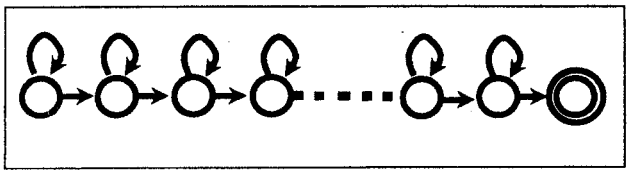

Figure 4b:
The Bakis-style Word Model upon which our Model is Based

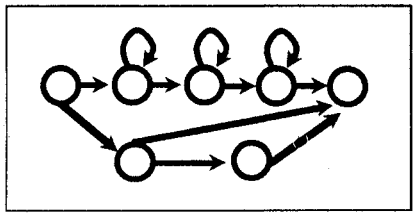

Figure 5:
The Phone Model Used in a Previous Version of the System

6.3 Corrective training

Using a form of corrective training, we have further reduced the error rate in the new non-recurrent word-based CVT system by about 6%. The general idea of the corrective training procedure is that emphasis in training should be placed on sentences in which the system is likely to committ recognition errors. This emphasis is achieved simply by performing extra training on misrecognized sentences. However, given the rate of recognition errors, there is not a large corpus of misrecognized sentences to use in this manner.

In order to generate more misrecognitions for the corrective training procedure, we suppress correct recognition of a random subset of training sentences; we prevent the Viterbi forced alignment from entering the correct word at certain randomly selected times. Not only does this augment the size of the corrective-training corpus, it also produces sentences which are likely to include realistic recognition errors – since the system is in effect making a "second choice" recognition, which we assume often corresponds to the sorts of errors made in actual recognition.

6.4 Multiple models

A second training strategy which has proved beneficial – yielding a 33% increase in performance – uses multiple models for each word. Once the single-model-per-word system was fully trained, an extra output unit for every transition in every word was added to the neural network. These weights were set equal to the corresponding previously trained weights, with the addition of a small (5%) random perturbation. Then, an additional HMM was created for every word, and these new models were associated with the new network output units.

CVT training proceeded as before; however, during the forced alignment phase, the system was permitted to enter *either* of the models for a word, based on the network scores. Hence, the system was able to develop models specialized for two primary pronunciations of each word. For example, we observed that the word 'eight' (phonetically represented as */ay/ /t/*) has two primary pronunciations: one in which the final stop is strongly pronounced and another in which it is hardly detectable. Using the new system configuration, the two pronunciations could be modelled seperately.

6.5 Results: summary of performance on TI Digits

Table 1 is a summary of the performance of the CVT system on the TI Digits task, showing the changes in performace which accompanied the recent changes to the system. The current version of the system achieves 99.1% word accuracy and 98.0% string accuracy on the TI Digits.

	Word Accuracy	String Accuracy	Incremental Improvement
Baseline CVT System 1990	98.5	95.0	
– recurrence	98.0	94.7	-6%
+ word models	98.7	96.8	+40%
+ corrective training	98.8	97.0	+6%
+ multiple models	**99.1**	**98.0**	+33%

Table 1: Improvements in Results on TI Digits

7 Conclusion

The goal of this work was to build a continuous-speech recognition system which combined the pattern-classification ability of connectionist networks with the time-alignment ability of hidden Markov models. We began with a system built of two distinct components: a NN frame classifier, and an HMM post-processor. Then, using the Connectionist Viterbi Training procedure, we integrated the training of the two parts of the system, such that the classification task being performed by the NN was in effect controlled by the HMM.

In making the most recent revisions to the system, we have reached several conclusions, which may be extensible to other approaches and other tasks as well: (1) that modelling speech at the lowest level possible appears to produce the best results and, when permitted by the task (i.e., when the vocabulary is sufficiently small and the size of the training corpus sufficiently large), word-based HMMs should be used in place of phone-based HMMs, (2) that the performance of non-recurrent NNs is only slightly worse than recurrent NNs, and the former allows a significant computational saving, (3) that a corrective training procedure can reduce the error rate by providing additional training on error-prone data, and (4) that using multiple models per word can result in a higher overall recognition rate, by allowing distinct representations of different pronunciations or different speaker characteristics.

The most general conclusion to be drawn from this work is that NN-HMM hybrid systems show great promise in the domain of continuous speech recognition. These systems, which have been under investigation for only about three years, have already achieved error rates within one order of magnitude of the best results on a task for which HMM-based recognizers have been under development for nearly a decade. This early success suggests that these hybrid systems may be one of the most viable means for performing high-accuracy continuous speech recognition.

References

[1] Brown, P., *The Acoustic Modeling Problem in Automatic Speech Recognition*, Ph.D. Thesis, Carnegie Mellon University, May, 1987.

[2] Franzini, M.A., Witbrock, M.J., Lee, K.F., "A Connectionist Approach to Continuous Speech Recognition," *Proc. ICASSP*, April, 1989.

[3] Huang, W., Lippmann, R. "HMM Speech Recognition with Neural Net Discrimination," *Proc. Neural Information Processing Systems (NIPS) Conference*, November, 1989.

[4] Bourlard, H. and Morgan, N. *Merging Multilayer Perceptrons and Hidden Markov Models: Some Experiments in Continuous Speech Recognition*, Tech. Report TR-89-033, July, 1989, International Computer Science Institute, Berkeley, CA.

[5] Franzini, M.A. "Speech Recognition with Back Propagation," *Proc. Ninth Conf. of the IEEE Engineering in Medicine and Biology Society*, November, 1987.

[7] Somewhat faster learning was observed when a desired value of 0.4 was assigned to output units which corresponded to different transitions within the correct phone. That is, the desired value for the unit corresponding to one transition in a phone model was set to 1.0, the desired value for units corresponding to other transitions in that phone model was set to 0.4, and all other desired values were set to 0.0.