A HYBRID NEURAL NETWORK, DYNAMIC PROGRAMMING WORD SPOTTER

Torsten Zeppenfeld *Alex H. Waibel*

School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3890 USA

ABSTRACT

We present a novel keyword-spotting system that combines both neural network and dynamic programming techniques. Our system makes use of the strengths of TDNN neural networks, which include strong generalization ability, potential for parallel implementations, robustness to noise, and time shift invariant learning.- Dynamic programming models are used by our system because they have the useful capability of time warping input speech patterns. We have trained and tested our system on the Stonehenge Road Rally database, which is a 20 keyword vocabulary, speaker independent, continuous speech corpus. Currently, our system performs at a Figure of Merit (FOM) rate of 82.5%. FOM is the detection rate averaged from 0 to 10 False alarms per keyword hour. This measure is explained in detail below.

1. INTRODUCTION

Recently, there has been a surge of interest in the development of keyword spotting systems for continuous, speaker independent speech recognition[1,2,4]. This paper presents one such system. It is an approach based on a hybrid neural network - dynamic programming scheme.

Standard speech recognition systems usually require that all words presented to the system be part of a known dictionary. Thus for many practical uses, the recognition task quickly becomes unmanageable given the number of different words that are present in unconstrained speech. keyword spotting systems bypass this bottleneck by attempting to recognize only a few selected 'important' keywords; they are designed to ignore the extraneous speech.

Although many word spotting systems today use a hidden Markov model approach to recognition [1,2], neural networks, with their discriminatory ability, have gained a strong following. Other important strengths for speech recognition include the ability to generalize, the potential for parallel implementations, and robustness to speaker variation and noise. Neural networks have already been shown to be good phoneme recognizers [3]. For the above reasons, we feel that they will also play a useful role in keyword spotting systems [4]. Indeed, our system, while still under constant improvement, already shows recognition results on par with those of other word spotting systems [1,2,4].

2. THE DATABASE

With the hope of creating a standard word-spotting database, ITT and NIST have distributed a database called the 'Stonehenge' Road Rally task (and an additional extension called 'Waterloo'). The database consists of approximately 140 speakers (both male and female) recording conversations, read paragraphs, and/or read keyword sentences. This speech contains several (20) keywords, embedded in extraneous speech. Keywords can have variable suffixes, such as -s, -ed,-ing. The task is to spot the occurrences of these twenty different keywords, while minimizing the number of false detections. The Stonehenge portion of the database was recorded at 10KHz using a high quality microphone, while the Waterloo extension was recorded over telephone lines (also at 10KHz). The database labelling consists of markers at the beginning and ending of all keywords present. The speech is not labelled phonetically, nor is the extraneous speech labelled.

3. SYSTEM ARCHITECTURE

The present work is based upon the TDNN [3] and more recently the MS-TDNN [5]. A diagram of the basic network architecture is shown in figure 1. The keyword spotting network consists of an input layer and a hidden layer, connected to a state layer and an output layer for each word to be spotted.

Conceptually, the system includes the following features:

3.1. Weights

The system uses TDNN style linked weights between the layers of the network. This lets the system learn shift invariant speech

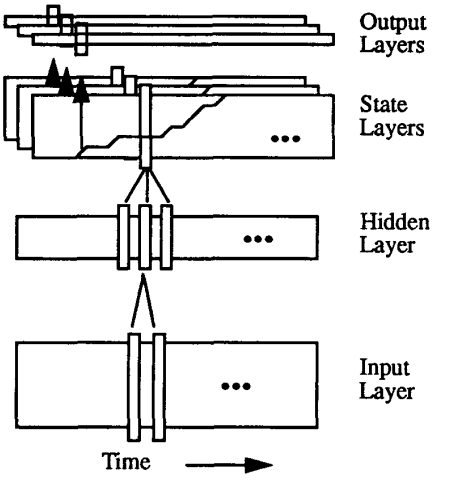

Fig. 1. System Architecture

0-7803-0532-9/92 $3.00 © 1992 IEEE

patterns (i.e. phonemes) as the network is shifted through each part of the speech input signal.

3.2. State Unit modelling

State units are used to model keywords, instead of phonemes. This is done for several reasons.

- The database does not contain phonetically labelled speech. Since this is the case, ad hoc assumptions about phonetic boundaries would have to be made in order to train the system.
- A dictionary would be needed that gives the phonetic transcriptions of each keyword. Since speakers' pronunciations for words differ greatly, system performance would depend on how well this dictionary matched the speaker variability.
- State units are believed to be more reliable in noisy speech, and when using small vocabularies. Since there are no shared states or phonemes, each state unit can model the important features for a specific keyword.

3.3. Dynamic Programming

A Dynamic Time Warping (DTW) algorithm is used at the state layer level. This allows the system to map the different variable length occurrences of the same keyword to a single output unit. This is accomplished by assuming that the activation of each state in the state layer represents the likelihood of the corresponding feature being present in the input speech. By finding the score of the best path through the states, we are effectively finding the probability of a keyword occurrence. Since we allow a state to be active for several frames, we are normalizing the length of every possible keyword to a constant number of states.

3.4. Learning Algorithm

Backpropagation is done at the word output level. This allows the network to optimize itself for word recognition, instead of lower level (phoneme or state) recognition. During training, we know when a keyword should occur. We do not know a priori the duration of each state in this word (and cannot know, since the state representations evolve over time). For these reasons, the system must integrate the information of each state into a single quantity, i.e. the word output activation. Backpropagation can thus be used to minimize word errors.

3.5. Sigmoidal Output units

MS-TDNN style sigmoidal output units are used. The equations for this unit are:

$$\text{net\_in} = CW * (\text{best DTW Score} / \text{Length of path}) \quad (1)$$

$$\text{Output Activation} = \text{Sigmoid}(\text{net\_in}) \quad (2)$$

where CW represents a constant weight multiplier. With a properly chosen CW, the output unit will aid recognition for the following reasons:

- It can prevent small deviations at the state level from affecting the overall classification performance. Since the sigmoid will begin to saturate before all states do, some variation in input speech will not affect the word level activations significantly.
- Because of this saturation, a sigmoidal unit will limit the amount of error that is back-propagated from the word level when enough frames generate positive state layer activity to allow the network to spot a particular word. Thus the network will be able to concentrate on the cases that are still being categorized incorrectly, while not ignoring the cases that it gets correct.

4. IMPLEMENTATION

4.1. Speech Input

The 10KHz sampled speech signal is presented to a preprocessing module, which converts the input speech into melscale frequency filterbank coefficients, using a fast fourier transform. The current system uses as input 15 melscale coefficients and 1 average power coefficient every 10 milliseconds.

4.2. Weights

The system incorporates TDNN style shift linked weights to model the time varying speech input shift invariantly. Each frame in the hidden layer is fully connected through two frames of input units spanning delays from -1 frame to +1 frame. The connections between the hidden layer and each word state layer is similarly made to be fully connected through three frames, spanning delays from -2 frames to 2 frames of hidden units. As the input speech is presented to the system, the weights are conceptually shifted one frame of speech at a time, giving new unit activations for each layer, at each frame.

4.3. State Layers

The state layers are used to represent the differing parts of the keywords. There is a state layer for each keyword, as well as a corresponding output layer. Note that these layers are not shared between keywords. This design decision was made for the several reasons discussed in section 3.2

Each word has several states associated with it (typically eight states for the longer words). After several frames of word state unit activations have been computed, a simple DTW algorithm is used to find the best path through these states (see figure 2). The DTW path has the following properties:

- Each keyword has its own DTW path, independent of other possible keywords.
- At each frame, the best path either stays in the same state, or moves to the next state. Thus no states are skipped.
- When moving between states, we subtract a state transition penalty from the accumulated score. This penalty is proportional to the activation of the next state. This is done so that a state will not transition to the next state immediately. It will wait until the next state has a positive activation.
- For each frame that the path stays in a particular state, we subtract a length penalty. This penalty is proportional to how long the path has been in this state. This is done so that a path will not be active for too long a duration.

Figure 3 shows how this is accomplished. Starting at frame A1, we accumulate DTW scores for each frame, until we have calculated scores for at least 'max' frames. 'Max' will be the maximum duration of the keyword that we will allow our system to spot, and

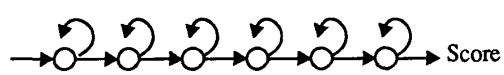

Fig. 2. DTW Word State Model

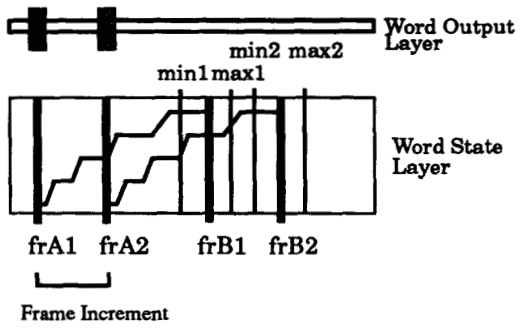

Fig. 3. Calculating the Output Layer with DTW

'min' is the minimum possible duration of the keyword. Checking the scores at the last state from frame 'min1' to 'max1', one finds the best path ending at frame B1. This score will be used to find the activation of the word output layer unit for frame A1.

As in the MS-TDNN, this best path's score first normalized by the path's duration. This value is then multiplied by a constant weight and passed through a sigmoidal function and used as the activation of the word output layer for that frame. We then increment the starting frame from A1 to A2. The above described DTW procedure for each word is repeated as new activations become available for each frame of speech.

4.4. Word Output Layers

The presence of a possible hit at the word output layer is decided by the activation of each output layer. If an output layer activation is above threshold for several succeeding (typically eight) frames, a possible hit occurs. Some simple post-processing is done to insure that only one keyword is ever detected at a single time, and that the most likely (most active) one is chosen if there is a possible overlapping of hits.

5. TRAINING

The system is trained with an ordinary backpropagation algorithm, in two major phases: A preliminary bootstrapping phase, and a best path phase. Each phase has positive as well as discriminatory learning.

5.1. Bootstrap Training

Before the DTW algorithm can be used to train the network, the network weights must be initialized to give plausible word state layer activations. This is accomplished by the bootstrap training phase. In this phase:

- Backpropagation is done from the state level. Since a best path through states with randomly initialized weights is virtually meaningless, word output activation calculations based on such a path would also be fruitless.
- Training is done on sections of the speech which contain a keyword and approximately one second of speech before and after this occurrence. This is done to help balance negative and discriminatory training.
- During keyword occurrences, a linear DTW path is assumed. This means that state targets are taken to be 'ON' for equal times through all states in the word sequentially.
- Discriminatory training is accomplished by setting the state targets to be 'OFF' at all times not in the path.
- Do to the amount of negative training available as compared to positive training, discriminatory training in the bootstrap phase is watered down. This means that it is given a smaller learning rate (approx 1/6th) than the corresponding positive learning cases. If this were not done, the network would be overwhelmed by the amount of negative training.

5.2. Best Path Training

After the initial bootstrapping phase has set the states to an approximate solution, a new training procedure is started that fine tunes the word states. In this phase:

- Backpropagation is done from the word output level.
- Positive training is done at each keyword occurrence. The best path is found between the known word beginning and ending in the state layer, and the word output unit is effectively connected to the states in this path. The output unit target is set to 'ON' for this frame and the error is backpropagated through this path.
- Discriminatory training is accomplished by finding the best path wherever a false hit is about to occur, setting the word target to 'OFF', and backpropagating the error from the word output unit through this path, to the rest of the network.

Note: For both bootstrap and best path phases, training is only performed on base keywords. Words with suffixes (-s, -ed, -ing) are not used as training examples during training, but are still tested for during the test phase. Although this severely limits the number of training examples for some words, it was done to keep the system complexity down.

6. RESULTS

A meaningful comparison of results with other groups' attempts at building word spotting systems requires several things, including a standard test set and a standard scoring procedure for proper evaluation of results. A standard is currently under development that addresses these issues.

6.1. Data Set

Our current training corpus for the neural network word-spotter consists of conversational speech of 24 males from the Stonehenge database, read keyword paragraphs of 35 males from the Stonehenge database, and read keyword paragraphs of 14 males from the Waterloo extension set. Our Testing set consists of conversations of 8 males (not in the training set) from the Stonehenge database. We have tested the system using both the original Stonehenge speech, and a bandpass filtered version of the corpus designed to simulate telephone quality speech.

6.2. Performance Measurements

The system performance is measured in two ways; a detection rate (measured as a percentage) and a false alarm rate per keyword per hour (fa/(kw*hr)). The detection rate is calculated by finding the number of correctly spotted keyword occurrences and dividing by the number of keyword occurrences actually present. The false alarm rate is calculated by first finding the number of keyword insertions found by the system. This is then normalized by dividing by the number of unique keywords (our system has 20

Test Data Set	Performance(FOM)
Wideband	**82.5%**
Telephone Bandpassed	**81.3%**

Table 1. Experimental Results

keywords) and by the total amount of test speech given the system.

By changing the threshold of the word-output units, one of these measures can be increased at the expense of the other. A proposed standard to combine these two values is to find the Figure Of Merit (FOM). We find the detection rate at all false alarm rates from 0 to 10 fa/(kw*hr). The FOM is then the detection rate averaged over the false alarm rates from 0 to 10 fa/(kw*hr). Tests on our system are summarized in Table 1. These results indicate that our system's recognition performance compares favorably with results obtained by other word spotting systems [1,2,4] in the literature. The fact that the results for the bandpassed data are very close to the wideband data results indicate that the system is useful for spotting keywords in low quality, noisy speech environments, as well as in high quality recording conditions.

7. DISCUSSION

7.1. Balanced Training

The ratio of keyword occurrences to extraneous speech is fairly small in this database. This implies that the word spotting system has much more discriminatory data than positive examples from which to train itself. Since neural networks seem to work best when the number of negative and positive training cases is balanced, we have added several features to our system with this idea in mind.

During the bootstrap training phase:

- The learning rate for the negative training cases is only a fraction of the learning rate for the positive cases.
- Training is done only on short sections of speech containing the keyword. Approximately one second of extraneous speech is taken before and after each keyword.

During the best path training phase:

- All training speech is used, and positive training is done whenever a keyword is present.
- Negative training is done only when the word output layer activation is above a certain Threshold. This directly limits the amount of negative training that is performed.

7.2. Error Analysis

As was expected, our system performed much better on the longer keywords than on the shorter ones. As seen in Table 2, the best results were obtained on the words "retrace", "waterloo" and "springfield" while the system performed worst on "look" and "want". This is due in part to several facts. First, these short words have less training examples in the speech database than most of the longer words. Compounding this problem is the fact that they are usually used with suffixes, and our system does not learn on cases with added suffixes. Also, they are by nature more confusable than the longer words. Finally, our system is geared more towards longer words, because of nature of the state layer.

Keyword	Perf. (FOM)
boonsboro	91.5%
conway	78.9%
interstate	89.7%
middleton	96.9%
mountain	65.1%
look	9.1%
primary	92.3%
secondary	92.3%
sheffield	96.2%
westchester	89.4%
backtrack	N/A
thicket	82.7%
retrace	100%
waterloo	100%
springfield	100%
chester	73.2%
minus	72.2%
road	67.0%
track	81.0%
want	4.3%

Table 2. Individual performance scores for all keywords, using bandpassed speech test set.

7.3. CONCLUSION

Our novel speaker independent continuous speech word spotting system integrates both neural networks and dynamic programming techniques in an attempt to utilize the strengths of both. Our system performs well on the fairly noisy, poor quality database known as Stonehenge. Our research on this system is far from complete. Many issues can still be addressed and further improvements are expected.

8. REFERENCES

[1] Rose, R.C. and Paul, D.B., "A Hidden Markov Model Based Keyword Recognition System," ICASSP'90.

[2] Wilpon, J.G., Miller, L.G. and Modi, P., "Improvements and Applications for keyword Recognition Using Hidden Markov Modeling Techniques," ICASSP'91.

[3] Waibel, A.H., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K., "Phoneme Recognition Using Time-Delay Neural Networks," in IEEE Transactions on Acoustics, Speech and Signal Processing. 1989.

[4] Morgan, D.P., Scofield, C.L., and Adcock, J.E., "Multiple Neural Network Topologies Applied to Keyword Spotting," ICASSP'91.

[5] Haffner, P., Franzini, M., and Waibel, A., "Integrating Time Alignment and Neural Networks for High Performance Continuous Speech Recognition," to be published.