DETECTION OF WORD-BOUNDARIES FROM CONTINUOUS PHONEME STREAMS USING SIMPLE RECURRENT NEURAL NETWORKS

L.R. Leerink*, M.A. Jabri*, A.E. Dutech+

*Systems Engineering and Design Automation Laboratory
Department of Electrical Engineering
The University of Sydney

+Ecole Nationale Superieure de l'Aeronautique et de l'Espace
Toulouse, France

ABSTRACT - This paper describes a word boundary detection system based on recurrent neural networks. We show that by using a more effective training algorithm the performance of previous research in this area can be matched by a significantly smaller and simpler network. It is then shown that the performance of this network can be further improved by varying the forward context. For the architecture used in our simulations, empirical results indicate that the optimal amount of forward context depends on the number of recurrent units in the network.

INTRODUCTION

Word boundary detection plays an important role in most continuous speech recognition systems. In a typical speech recognition system word boundary detection takes place after segmentation and before word verification, and as such directly determines the amount of processing required by higher levels.

Word identification in continuous speech is complicated by the fact that coarticulation or phonetic/phonological recoding takes place at word boundaries. For word boundaries to be recognised properly these encoding rules have to known and applied to the input in real-time. The advantage of using neural networks for this task is that recurrent networks can use certain neurons to store short-term context information, and can automatically learn these phonetic decoding rules. Recurrent networks also have the capability of recognizing and adapting to different speakers (Robinson and Fallside, 1991).

Allen & Kamm (1990) have used a partially recurrent network to locate word boundaries and to recognize words from phoneme sequences. Their network, whose architecture is shown below in figure 1, achieved 87.5% correct word boundary detection on a stream of phonemes from the DARPA Acoustic-Phonetic speech corpus (TIMIT).

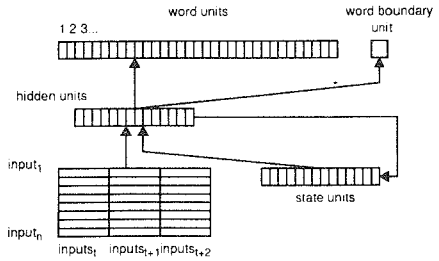

Figure 1. Allen & Kamm Network Architecture.

ARCHITECTURE ANALYSIS

In their architecture Allen & Kamm (1990) combine two well-known methods of recognizing time sequences. The first is that of time-delay neural networks which use a tapped delay line to turn the time sequence into a spatial pattern (Elman and Zipser, 1988; Kohonen, 1989). This is done at the input where the inputs for time *t, t+1* and *t+2* are presented in parallel to the network. The second concept is

that of partially recurrent networks, often referred to as Jordan (1989) or Elman (1990) networks. This method uses decaying state or context units, which function as a short-term memory and enables the network to remember events from the recent past. Referring to figure 1, the output of each state unit is calculated from a weighted sum of the output of the hidden units and the previous value of the state units. By combining these two methods, the network can base present decisions on phonemes that have been presented to it in the past (backward context) and the next two phonemes that are forthcoming (the forward context of *t+1* and *t+2*).

Allen & Kamm (1990) used several implementation specific additions to the standard back-propagation algorithm. The end of the phoneme sequence was padded with codes representing silence, and the state unit activations were reset to zero at the end of each sentence.

ARCHITECTURE USED IN EXPERIMENTS

The architecture used in our experiments is shown below in figure 2. It can be classified as a two-layer partially recurrent network, and is a simplified version of the Elman (1990) network which has the hidden layer removed. Robinson and Fallside (1991) have named it a recurrent error propagation network and have applied this network to the problem of phoneme recognition.

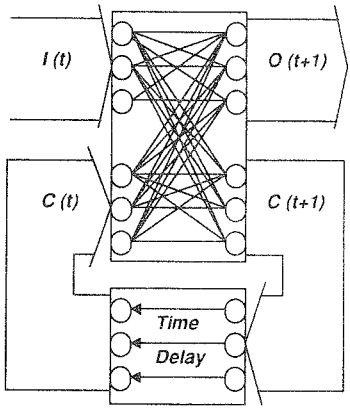

Figure 2. The Error Propagation Network.

In figure 2 the context units are represented by the *C(t)* neurons. During normal operation the input vector at time *t* are applied to nodes *I(t)*, and during the feedforward pass values are produced at both the output nodes *O(t+1)* and the context units *C(t+1)*. The values of the context units are then copied back to the input layer *C(t)* for use as input in the following time period.

Compared to the architecture in figure 1, this network

- has no hidden units
- has no tapped delay line and only the input at time *t* is applied to the inputs at any point in time
- the value of the context units do not decay with time; the context nodes in the input layer are pins and those in the second layer are standard neurons

The majority of the simulations were run using 50 context units. In this configuration the network contains 5561 weights, which is 32% of the weights used in figure 1.

PROCEDURE

In order to compare results with that obtained by Allen & Kamm (1990), as far as was possible to determine the same data has been used. The training set consisted of 5 utterances of each of 50 sentences

(spoken by different talkers). The testing set consisted of another utterance of each of the 50 sentences. The accuracy of the word boundary detection was determined by the percentage of correct outputs. The two criteria used by Allen & Kamm (1990) i.e. the percentage of word boundaries detected correctly and the false alarm rate are also used for comparisons.

TRAINING ALGORITHM

Several training algorithms exist for training partially recurrent neural networks, but for non-trivial tasks with large training sets the back-propagation through time (BTT) or unfolding in time (Minsky and Papert, 1969), (Werbos, 1990) is usually used. This method is computationally efficient and does not use any approximations in following the gradient. For an application where the time information is spread over T input patterns, the algorithm simply duplicates the network T times - which results in a feedforward network which can be trained by a variation of the standard backpropagation algorithm.

In our simulations the standard BTT algorithm was used. However, the standard quadratic cost function was replaced by the entropic cost function (Solla et al. 1988). The entropy cost function has previously been shown (Denker et al., 1988) to solve some problems that cannot be solved using the quadratic cost function. Our simulations have shown that for the problem of boundary detection significantly better results are achieved using this cost function.

The second modification to the standard BTT algorithm was to add the Jacobs (1988) Delta-Bar-Delta learning rate adaptation heuristic. In our application this heuristic not only increased convergence by approximately an order of magnitude, but also proved to be efficient in avoiding local minima. In several cases training converged where without the heuristic the BTT algorithm could make no further progress.

RESULTS AND DISCUSSION

In the architecture as is shown in figure 2 the only parameter that can be modified is the number of context units N_c. However, if the network is implemented and trained in this manner the network has no forward context to base it's decisions on. The context units can be used to store backward context, but for the problem of word boundary detection both contexts are equally important.

This problem is solved by simply delaying the target pattern for the classification output $O(t+1)$ by a variable number of time steps. If e.g. the delay is one, the network is effectively recognizing the beginning of words, and not the ends. This classification delay of T_d provides the network with exactly T_d steps of forward context. However, in order to use this information efficiently it must be extracted and stored until the classification is eventually made. There are thus two variables to explore, and our simulations have been targeted towards obtaining the importance of and interaction between N_c and T_d.

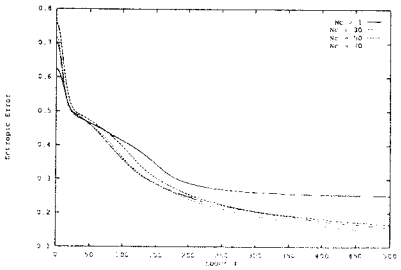

Figure 3. Entropic error during training for N_c=1,30,50,70.

If the classification delay T_d is kept constant and the number of context units increased, the performance increases as would be expected. In figure 3 the entropic error is shown during training for 4 different values of N_c for the case where T_d is 1.

The corresponding performance curve is shown in figure 4. The speed with which all networks, including the one with only one context unit, gets 80% of the training vectors correct seems remarkable but is because the average word contains four phonemes. The gradient decent algorithm rapidly detects that the network can get 80% of the outputs correct by simply keeping the output permanently low. All networks then enter a plateau for about 50 epochs until criteria for boundary detection are gradually extracted

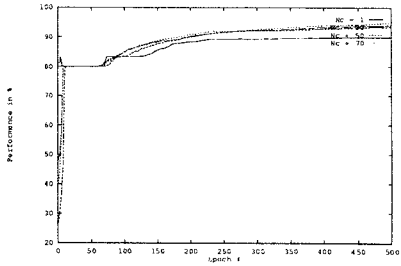

Figure 4. Performance during training for N_c=1,30,50,70.

From figure 4 is seems that the performance can be increased by simply increasing the number of context units. However, this increase is performance is less than linear compared to the increase in context units, while the increase in the computational load is quadratic.

For studying the effect of the time delay parameter T_d only the network with 50 context units were used. The changes in performance for T_d ranging from 0 to 11 are shown in figure 5. As expected, the performance increases initially and then decreases as T_d is increased. The decrease in performance at epoch 3 would probably disappear if the simulation were run several times with different starting points.

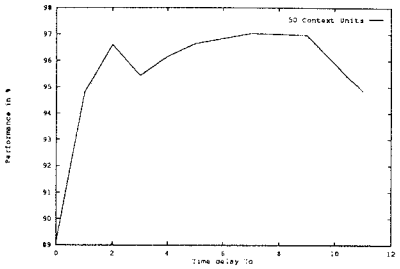

Figure 5. Performance for N_c=50 and varying T_d

Due to the fact that the network is trained on 5 pronounciations of the 50 sentences, the generalization to the 6th version of the sentence in the test set was good, performance was nearly always within 1% of that obtained during training. In fact, the best training performance of 99.33% was obtained using a network with N_c=100 and T_d=7. On the test set, a performance of 98.49% was obtained, which corresponds to 98.2% correct word boundary detection and a false alarm rate of 0.31%.

The remaining word boundary errors have been examined to see where further improvement is possible. Most of the errors occur with single phonemes which are a word but which can also appear at the beginning or end of larger words. A typical example is the phoneme /a/, in the two sequences /a/ /lone/ and /alone/. It is obvious that when humans distinguish between these two sequences higher level semantic knowledge is used. Without this knowledge there is a maximum to what can be achieved by any recognition system.

CONCLUSIONS

This paper has examined the architecture used by previous researchers for the problem of word boundary detection. It has been demonstrated that their performance can matched by a significantly smaller and simpler network. This has been achieved through the use of a different cost function and the addition of a learning rate heuristic to the training algorithm.

The effect of changing the number of context units N_c and the time delay T_d on performance has been analysed empirically. It has been shown that the performance increases slowly with increase in N_c, but that for every N_c value of the there is a range of optimal T_d values.

Analysis of the remaining errors show that the majority require higher level information for correct classification. This indicates that there is an upper bound to the performance achievable in a system which does not include knowledge of the higher domains. Present research is directed at overcoming this problem by using neural networks to perform knowledge integration between different domains.

REFERENCES

Allen, R.B. and C.A. Kamm (1990). *A Recurrent Neural Network for Word Identification from Continuous Phoneme Strings.* Advances in Neural Information Processing Systems 3. Morgan Kaufmann.

Elman, J.L. (1990). *Finding Structure in Time.* Cognitive Science 14, 179-211.

Elman, J.L. and D. Zipser (1988). *Learning the Hidden Structure of Speech.* Journal of the Acoustical Society of America 83, 1615-1626.

Jacobs, R.A. (1988). *Increased Rates on Convergence Through Learning Rate Adaptation.* Neural Networks 1, 295-307.

Jordan, M.I. (1989). *Serial Order: A Parallel, Distributed Processing Approach.* Advances in Connectionist Theory: Speech, eds. J.L. Elman and D.E. Rumelhart. Hillsdale: Erlbaum.

Kohonen, T. (1989). *Self-Organization and Associative Memory.* Berlin: Springer Verlag.

Robinson, A.J. and F. Fallside (1991). *A error propagation network speech recognition system.* Computer Speech and Language 5, 259-274.

A COMBINED NEURAL NETWORK AND CONTOUR METHOD FOR MOUTH IMAGE LOCATION FOR SPEECH-DRIVEN IMAGE ENHANCEMENT

S-H Luo and R.W.King

Speech Technology Research Group
Department of Electrical Engineering
The University of Sydney

ABSTRACT - This paper describes a new and effective mouth locating method to locate automatically the mouth shape in a human head with shoulder image. This work forms part of our research aimed at to improving the quality of image compression for very low bit rate videotelephony where the motion of the mouth should correspond exactly to what is being uttered, and is not excessively smoothed by any more general purpose data compression method.The paper outlines how we intend to integrate phonetic information with the mouth shape and motion.

INTRODUCTION

It is of current interest in speech and image processing to focus on the developing of videophone and videoconference systems operating over very low bit rate channel (Whybray, 1990). Compared to other video utilities, videophone and videoconference are intended primarily for person-to-person or group-to-group audiovisual communications. It is desirable to operate these at low data rates, ideally at the 64 kbits/s rate offered by a single ISDN channel. Existing video coders operating at this rate can exhibit very poor performance where there is some degree of motion. Accordingly, so-called 'intelligent' image coding (Kaneko, 1991) has recently come in for a great deal of attention for future videophone and videoconference services. Compared to conventional coding techniques, which were designed to transmit the waveform signals, intelligent image coding methods utilize knowledge about the shape and structure of objects and images, and to some extent handle the meaning or content of visual information (Aizawa, 1987, Welsh,1990,Walden et al., 1977).

There exists much mutual information between the acoustic speech signal (speech) and the 'visual' speech signal (mouth movement). In normal speech perception, acoustic speech is reinforced to some degree by observation of the speaker's mouth (Massaro, 1987). Visual speech signals are used in lipreading; a complementary process is to use parameters derived from acoustic speech signals to drive an image of mouth movement. Such a process would provide intelligent enhancement of visual image compression methods and provide the basis for image animation by intelligent coding. Different mouth dimensions and motion (including tongue and teeth position) correspond to different speech utterances. For example, vowels in English differ primarily in terms of the visible tongue height and tongue advancement. Walden (1977) has described an experiment in which all the English consonants were divided into 9 visually discriminable categories, which provide the basis of any technique for speech-enhanced image coding.

To generate mouth movement from speech, as is required in intelligent video coding, a crucial requirement is an automatic and accurate mouth locating method. The method we give here can meet this demand. Mouth location from a head and shoulders image is most conveniently performed in two similar steps: first, the location of the head, and secondly, location of the mouth, using data derived from the head boundary.

Two major processes are undergone in both of these two steps. In the first process, by using an active contour model, called 'snake' (Kass, 1988; Waite & Welsh, 1990), which originates from the way the contour changes its shape), we can derive a shape which has a high probability of being a head or mouth contour. The snake achieves this by means of minimizing three defined 'energy' measures: the image energy, the elastic energy due to snake's stretching, and another elastic energy due to snake's bending. Since the snake computation is a local energy minimizing process, its result may be a head/mouth shape or not. At this stage we introduce a trained multi-layer perceptron net as a pattern recognizer to determine the validity