A Recurrent Error Propagation Network Speech Recognition System*

Tony Robinson and Frank Fallside
Cambridge University Engineering Department,
Trumpington Street, Cambridge, England.
Enquiries to: ajr@eng.cam.ac.uk

Submitted Computer Speech and Language November 1990
To appear in Volume 5, Number 3, July 1991

Abstract

This paper describes a speaker independent phoneme and word recognition system based on a Recurrent Error Propagation Network (REPN) trained on the TIMIT database.

The REPN is a fully recurrent error propagation network trained by the propagation of the gradient signal backwards in time. A variation of the stochastic gradient descent procedure is used which updates the weights by an adaptive step size in the direction given by the sign of the gradient.

Phonetic context is stored internal to the network and the outputs are estimates of the probability that a given frame is part of a segment labelled with a context-independent phonetic symbol.

During recognition, a dynamic programming match is made to find the most probable string of symbols. This is done at a single level for phoneme recognition and at two levels for word recognition.

The phoneme recognition rate for all 61 TIMIT symbols is 70.0% correct (63.5% accuracy including insertion errors) and on a reduced 39 symbol set the recognition rate is 76.5% correct (69.8%). This compares favourably with the results of other methods, such as HMMs, on the same database (Lee and Hon, 1989; Levinson et al., 1989).

Analysis of the phoneme recognition results shows that information available from bigram and durational constraints is adequately handled within the network allowing for efficient parsing of the network output. For comparison, there is less

*The foundation of the work described in this paper was presented in June 1988 (Robinson and Fallside, 1989). The contents of this paper are based on a technical report of March 1990 (Robinson and Fallside, 1990a), which has been extended to include later work up to November 1990 (Robinson et al., 1990; Robinson and Fallside, 1990b) computation involved in the resulting scheme than in a one-state-per-phoneme HMM system. This is demonstrated by applying the recogniser to the DARPA 1000-word Resource Management task. Parsing the network output to the word level with no grammar and no pruning can be carried out in faster than real time on a SUN 4/330 workstation.

1 Introduction

The most promising approach to the problem of large vocabulary automatic speech recognition is to build a recogniser which has an intermediate level at which phonemes are represented and which is subsequently mapped onto a string of words. Phonemes are the smallest linguistic unit that can be used to distinguish meaning (Ladefoged, 1982, p 23). By their symbolic nature they provide a natural boundary for speech recognition systems between the lower level distributed representations such as the acoustic waveform and its transformations, and the higher level symbolic representations such as words and the representation of syntactic and semantic knowledge. The phoneme recognition approach is practical because the number of phonemes is small (about 45) compared with the number of words in a large vocabulary task (at least 1000). Thus speaker independent phoneme models may be trained with a much smaller speech corpus than would be required to train speaker independent word models.

Currently the best established technique for large scale automatic speech recognition uses Hidden Markov Models (HMMs) (Levinson et al., 1983; Rabiner et al., 1983; Rabiner and Juang, 1986). Recently, connectionist models (Rumelhart and McClelland, 1986; Kohonen, 1988) and more particularly, error propagation networks (Rumelhart et al., 1986) have been used with some success in this field (Bourlard and Wellekens, 1987; Waibel et al., 1987; Franzini et al., 1989; Lippmann, 1989). The main differences between the HMM and connectionist approach using error propagation networks are:

- Error propagation networks provide a discriminant decision, i.e. the training minimises the distance to the target class and maximises the distance to the other classes. Standard HMMs lack this ability although work is now being done to develop discriminant HMMs (Bahl et al., 1986; Young, 1990).

- Recurrent nets can use internal storage for short term context information. Thus the output can represent context-independent phonemes. This contrasts with the HMM approach where context-dependent phonemes, such as generalised triphones, are needed to achieve good performance (Lee, 1989).

- Recurrent nets have an inherent mechanism for adapting to speaker variability. Information relating to type of speaker (e.g. female/male) can be gathered from the input and accumulated over time in the state vector. The recognition process can then use this slowly varying information to make a more accurate classification. There is no such mechanism in word HMMs which consist of concatenated independent phoneme models, although a similar effect can be achieved through an external mechanism such as the remapping of codebooks.

- Error propagation networks are trained by a gradient descent procedure which is considerably slower than HMM Baum-Welch parameter reestimation.
- The sequential nature of the speech signal at the phoneme level is more naturally expressed by the state transitions in a Markov model than by the development of the state vector in a recurrent net. As a result, the state sequence of phoneme HMMs can be concatenated to yield the state sequence for word models but no equivalent operation has been applied to recurrent nets.

The first three points may yield a higher recognition accuracy for recurrent nets and the last two points may be overcome with sufficient computational resources and the use of Markov models for higher level processing. This suggests that recurrent error propagation networks are worth investigating as an alternative to HMMs.

The strategy adopted here is to pass frames of windowed speech through a preprocessor which are then fed to a recurrent net. This net is trained to model the frame-by-frame classification of the TIMIT database. A dynamic programming postprocessor is then used to convert this distributed representation into a string of phoneme and word symbols representing the sentence.

1.1 The TIMIT and Resource Management Databases

Accurate comparison of different speech recognition systems is a difficult task. It is therefore important to evaluate recognisers on a standard database. The DARPA TIMIT Acoustic Phonetic Continuous Speech Database (Garofolo, 1988) (hereafter referred to as the TIMIT database) has been designed to be used for training recognisers at the phoneme level. It has become the most widely available database of its size and type.

At the time of writing, only the December 1988 Prototype CD-ROM was available. This contains all the training material of the full database but none of the test material. Thus, it was necessary to partition this database into training and testing portions. There are 420 speakers in total which were divided into 317 speakers for training and 103 speakers for testing. Eight sentences were used per speaker (the `si` and `sx` sentences). The identity of the test speakers are given in table 1, those marked with an asterisk are believed to have been used by Lee and Hon for testing their Hidden Markov model recogniser (Lee and Hon, 1989). The authors are grateful to Vassilios Digalakis and Mari Ostendorf of Boston University for their help in compiling this list.

In order to compare with other techniques and databases, the 61 TIMIT symbols were mapped onto a set of 50 symbols (Lee, 1989) and a set of 39 symbols (Lee and Hon, 1989). The TIMIT symbols, the reduced sets and the IPA symbols are given in table 2 which is an adaptation of a similar table by Seneff and Zue (Seneff and Zue, 1988; Pullum and Ladusaw, 1986). All occurrences of the the glottal stop, `q`, were discounted for the 39 symbol set.

fdmy0*	fsmm0	mrds0	msfh1	mtkd0*
fjlr0*	fspm0	mree0	msfv0	mtlb0
fkdw0*	fsrh0	mrfl0	msjk0	mtlc0
fmbg0	fsxa0	mrgm0	msjs1	mtmr0
fmcm0	ftaj0	mrjm0	mslb0*	mtmt0
fnkl0	ftbr0	mrlj0	msmc0	mtpf0
fntb0*	ftbw0	mrlr0*	msmr0	mtpg0
frew0	ftlh0	mrms1	msrg0	mtpp0
frll0	futb0*	mroa0	msrr0	mtrr0
fsah0	fvkb0	mrpc0	msvs0	mtwh0*
fsak0	fvmh0	mrpc1	mtaa0	mtwh1
fscn0	mbjv0*	mrre0	mtab0	mvlo0
fsdj0	mdem0*	mrtj0	mtas0	mwbt0
fsem0*	mdlm0*	mrtk0	mtat0	mwdk0
fsgf0	mdss0*	mrvg0	mtbc0	mwem0
fsjg0	mejs0*	mrws0	mtdb0	mwew0
fsjs0	mfwk0*	mrws1	mteb0	mwjg0
fsjw0	mjee0*	mrxb0	mter0	mwsh0
fskp0	mpam0*	msas0	mtjm0	mzmb0
fslb1	mpfu0*	mses0	mtjs0*	
fsma0	mrab1	msfh0	mtju0	

Table 1: Identity of speakers used in the test set

TIMIT	50SET	39SET	IPA	TIMIT	50SET	39SET	IPA
p	p	p	p	b	b	b	p
t	t	t	t	d	d	d	d
k	k	k	k	g	g	g	g
pcl	pcl	sil	p°	bcl	bcl	sil	b°
tcl	tcl	sil	d°	dcl	dcl	sil	d°
kcl	kcl	sil	k°	gcl	gcl	sil	g°
dx	dx	dx	ɾ	q	pau		ʔ
m	m	m	m	em	em	m	m̩
n	n	n	n	en	en	n	n̩
ng	ng	ng	ŋ	eng	ng	ng	ŋ̩
nx	n	n	ɾ̃				
s	s	s	s	sh	sh	sh	š
z	z	z	z	zh	z	sh	ž
ch	ch	ch	č	jh	jh	jh	ǰ
th	th	th	θ	dh	dh	dh	ð
f	f	f	f	v	v	v	v
l	l	l	l	el	l	l	l̩
r	r	r	r	w	w	w	w
y	y	y	y	h#	h#	sil	□
pau	pau	sil	□	epi	epi	sil	▯
hh	hh	hh	h	hv	hh	hh	ɦ
eh	eh	eh	ɛ	ih	ih	ih	ɪ
ao	ao	aa	ɔ	ae	ae	ae	æ
aa	aa	aa	ɑ	ah	ah	ah	ʌ
uw	uw	uw	u	uh	uh	uh	ʊ
er	er	er	ɝ	ux	uw	uw	ü
ay	ay	ay	ɑʸ	oy	oy	oy	ɔʸ
ey	ey	ey	eʸ	iy	iy	iy	iʸ
aw	aw	aw	ɑʷ	ow	ow	ow	oʷ
ax	ax	ah	ə	axr	er	er	ɚ
ix	ix	ih	ɨ	ax-h	ax	ah	ə̥

Table 2: The TIMIT symbol set with the two reduced sets and IPA symbols

In order to demonstrate word recognition, the network was tested on the DARPA 1000-word Resource Management database (Price et al., 1988). All six speakers on the first CD-ROM of the speaker-dependent training data (September 1989 release) were used for testing, with 610 sentences per speaker, (the `sb` and `sr` sentences).

2 Preprocessor

The preprocessor used in this paper was a result of a comparison of many preprocessors for this system (Robinson et al., 1990). Linear Predictive Coding (LPC), Fast Fourier Transform (FFT), filterbank and auditory model techniques were compared by deriving a form of normalised power spectrum plus a power channel for each. In the case of LPC and FFT, this power spectrum was also represented as a cepstrum. The addition of other features, such as zero crossings and estimates of the pitch and formant positions and amplitudes were also investigated. In all cases a 32ms Hamming window was used with a frame spacing of 16ms. The conclusion was reached that most preprocessors which were based around a power spectrum gave similar performance. The simplest of these used the cube root of the powers in twenty channels derived from the FFT. This design was arrived at by simplifying the auditory model presented by Bladon and Lindblom (Bladon and Lindblom, 1981), and is the preprocessor used in this paper.

For practical reasons, (memory limitations on disk and in RAM), the preprocessed data was scaled to fit into 8 bits per channel. This was done by computing a histogram and scaling so that no more than one in 500 samples lies outside the central 15/16th of the range. Typically this meant that one sample in 1000 would be thresholded.

The preprocessor truncated initial and final silences longer than 160ms. This was done to reduce size of the training data and provide a more even distribution of symbols amongst frames.

It was also found to be advantageous to preprocess the training data with several different offsets to better cover the variability in the windowed speech. In a preliminary experiment, this improved the frame-by-frame recognition rate by about 5%, as can be seen in table 3, although it should be noted that part of the increase is as a result of increasing the time constant for smoothing the weight changes used in training (the "momentum" term (Rumelhart et al., 1986)).

no. of offsets	frame-by-frame recognition rate
1	61.1%
2	64.2%
4	66.0%

Table 3: Effect of multiple offsets on frame-by-frame recognition rate

3 The Recurrent Net

A recurrent net can be considered as a sequence of error propagation networks (Rumelhart et al., 1986) where the input and output vectors are divided into external and internal portions. The external input vector, $u_{0\ldots L-1}$, consists of the 21 channels from the preprocessor; and the external output vector, $y_{0\ldots M-1}$ has 61 dimensions, one per phoneme label, and is fed to the postprocessor. The internal output forms a state vector, $x_{0\ldots N-1}$, of 192 dimensions and is fed to the same network in the next time period as shown in figure 1.

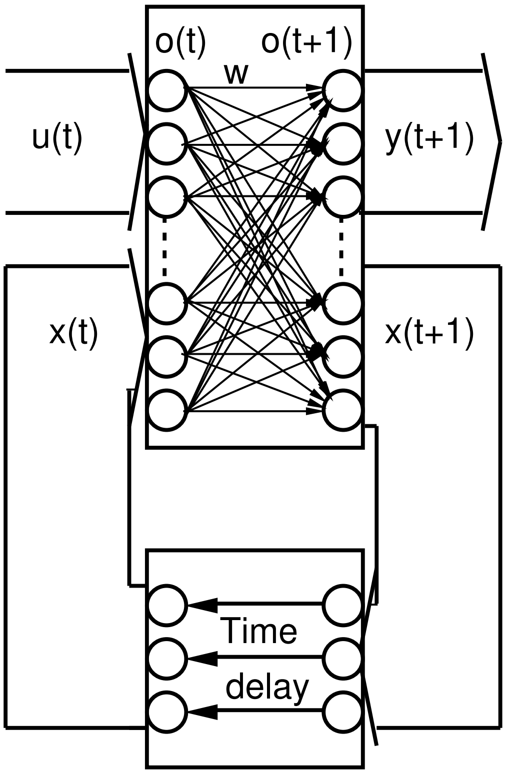

Figure 1: The Recurrent Error Propagation Network

This network operates by concatenating the current external input and the last internal output vectors to give the complete input vector at time t:

$$o_i^{(t)} = \begin{cases} 1 & \text{for } i = 0 \\ u_{i-1}^{(t)} & \text{for } 1 \leq i \leq L \\ x_{i-L-1}^{(t)} & \text{for } L+1 \leq i \leq N+L \end{cases} \tag{1}$$

which is then passed forwards through the network by performing a matrix multiplication by the weights, w_{ij}, followed by the application of a non-linear squashing function:

$$x_i^{(t+1)} = \frac{1}{1 + \exp\left(-\sum_{j=0}^{L+N} w_{ij} o_j^{(t)}\right)} \quad \text{for } 0 \leq i \leq N-1 \tag{2}$$

$$y_{i-N}^{(t+1)} = \frac{1}{1+\exp\left(-\sum_{j=0}^{L+N} w_{ij} o_j^{(t)}\right)} \quad \text{for } N \le i \le N+M-1 \tag{3}$$

The resulting output is compared with the desired output vector, $d_{0\ldots M-1}$, according to a cost function. Following Hinton (Hinton, 1987), Baum and Wilczek (Baum and Wilczek, 1987) and Solla, Levin and Fleisher (Solla et al., 1988) the outputs are treated as probabilities and the cross-entropy cost function is used:

$$\log(p^{(t)}) = \sum_{i=0}^{M-1} d_i^{(t)} \log(y_i^{(t)}) + (1-d_i^{(t)}) \log(1-y_i^{(t)}) \tag{4}$$

Training is performed on a 64-processor array of T800 transputers with the training data distributed evenly over the processors, each of which has a copy of all the weights. Of the possible schemes for training recurrent networks (for example, see (Robinson, 1989; Pearlmutter, 1990)), the "unfolding in time" approach is used as it is computationally efficient whilst computing a good estimate of the error signal. On each processor, the forward pass for 32 consecutive frames is made and the activations are stored in a buffer. The backward pass over this buffer is then performed and the resulting partial derivatives are summed over all processors to give an estimate of the gradient based on 2048 frames. There is a trade off in deciding the number of frames to be processed before updating the weights; a large number gives a more accurate gradient signal, and a small number allows for more frequent weight updates. Typical training time was three days or about 10^{13} floating point operations.

The network used a novel algorithm for updating the weights. A positive step size for each weight is defined, and the weights are changed by this step size multiplied by the sign of the estimated gradient. The estimated gradients are averaged with a first order filter, the "momentum" term (Rumelhart et al., 1986), which started with a small time constant and increased over the first few passes through the training set until it is sufficient to smooth the estimated gradient over the whole of the training set. Initially all steps sizes are equal, and the step is adapted by multiplying (or dividing) by a scaling factor if the estimated gradient agrees (or disagrees) in sign with the smoothed gradient. The scaling factor used was 1.1. The step sizes were hard limited to be not greater than a factor of 16 above or below the mean step size. This method has a theoretical disadvantage in that changes in the magnitude of the step size can occur more rapidly than changes in the smoothed gradient. Thus it is possible to have a large smoothed gradient which consistently disagrees with the sign of the estimated gradient and which may result in a rapid reduction of the step size to the lower threshold, so inhibiting further motion of that weight. In spite of this potential disadvantage, this method was found to converge faster for this problem than any other method tried, in particular the technique of Chan and Fallside (Chan and Fallside, 1987) used in the first version (Robinson and Fallside, 1989), and the similar technique developed by Jacobs (Jacobs, 1988).

The internal representation used to store context in the state units has no external constraints, and as such it is only dependent on the initial weights and their subsequent adaptation. Whilst it is possible and perhaps worthwhile to study this internal representation, such work is outside the scope of this paper. However, it is important to show that long term context can be stored in this structure. To do this, a network was constructed with an extra input that was set high or low depending whether the speaker was female or male, and this was assumed to be an appropriate format for this information. This network displayed slightly faster learning but no significant increase in performance. From this experiment it was concluded that the one additional bit of information was already contained in the state vector in some form, and therefore that long term context information could be trained into the recurrent network.

Example output of the model is given in figure 2 plotted against time as a variable width line per output channel. The hand labels are indicated on the horizontal axis of the diagram and the recogniser labels on the vertical axis. The shaded rectangles represent the target outputs. The sentence is "The viewpoint overlooked the ocean" (TIMIT file: train/dr7/flas0/sx228/sx228.adc) which is of 2 seconds duration.

The network contains 56026 weights which are processed at every 16ms frame yielding 3.5 million multiply and accumulate operations per second. This may be carried out in three times real time using a SUN 4/330 workstation. Thus a real time implementation should be possible on a processor which has support for the multiply and accumulate operation (e.g. i860, DSP32C, TMS32C30), with sufficient spare processing power to perform the subsequent parsing operation and the necessary data movement.

4 Phoneme Recognition

In the hand labelled data, a phoneme symbol is realised as a set of consecutive frames, each of which has the target output distribution for that symbol. A state transition model to generate the possible sequences of frame labels for a six phoneme example is shown in figure 3. The model is in one of the states which corresponds to the frame label at that time. The self loops allow for the label to be repeated and the other transitions (marked by solid and dashed lines) allow for any of the possible phoneme labels to come next. The transitions can be restricted, say to those marked with a solid line, which restricts the sequence of phonemes which can occur. In the example of figure 3 these are restricted to the symbols for the words "she had", namely /sh iy hh ae dcl d/

Any target sequence of frame labels can be compared with the actual output of the REPN by summing the frame-by-frame cost function of equation 4 for each frame. Thus it would be possible to generate every possible sequence of target frame labels, compare these sequences with the output of the network for an utterance, and pick the smallest distance to give the best match sequence of phonemes for that utterance. Alternatively, a more efficient method is that of dynamic programming (Aho et al., 1983, p311) which may be used to find the best match. This method is the basis of speech recognition using Dynamic Time Warping (DTW) (Ney, 1984), Hidden Markov Models (HMM) employing a Viterbi search and is also used in most hybrid HMM/connectionist systems (for example (Morgan and Bourlard, 1990)).

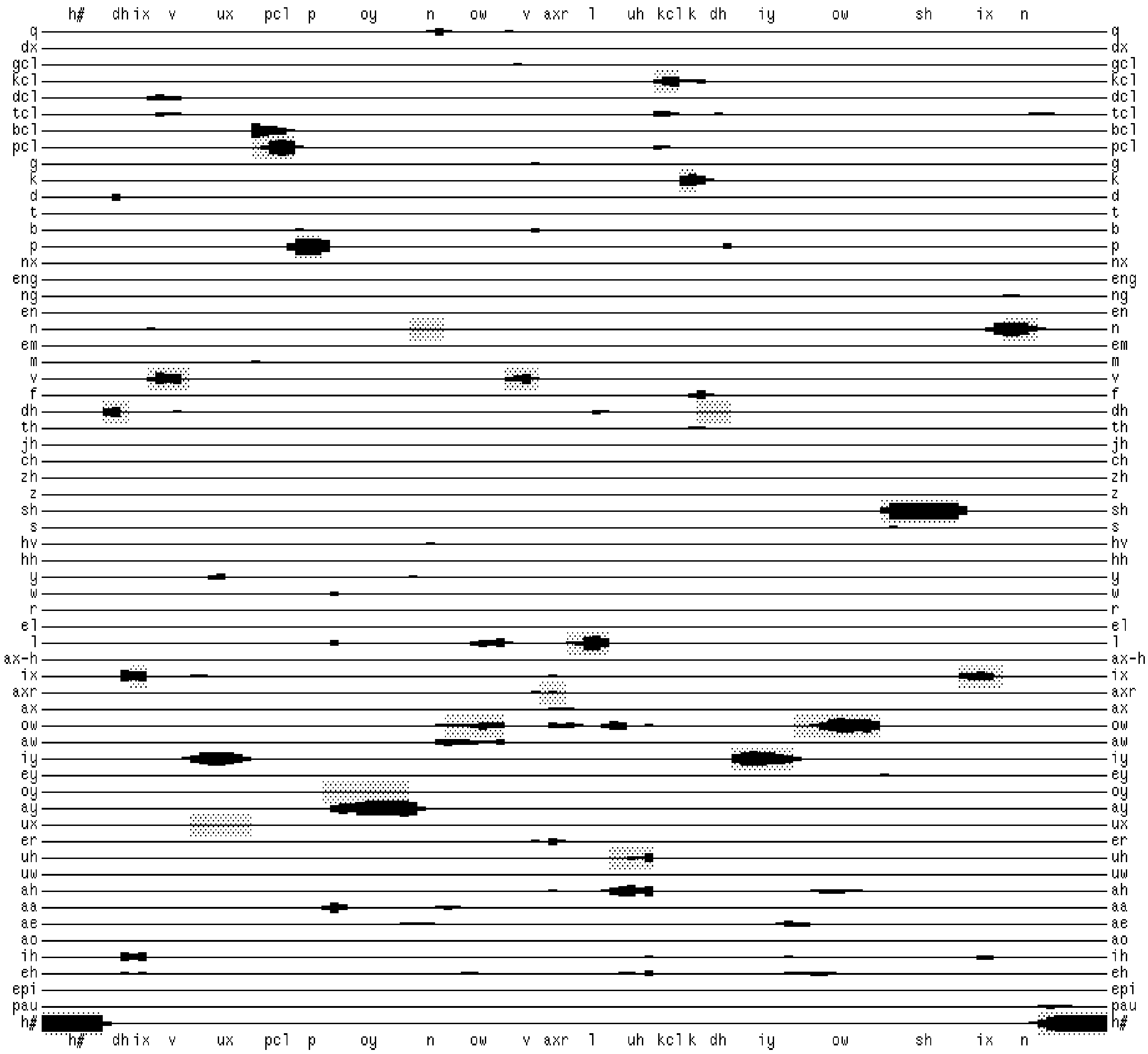

Figure 2: Example output from the recurrent net.

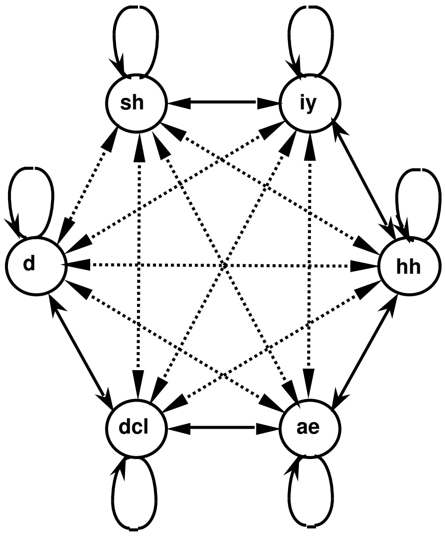

Figure 3: State transition diagram for frame label generation

Central to the dynamic programming method is the "principle of optimality" which means that if the best sequence has been found between any two points in the speech, say from A to B, and also from point B to another point, C, then the best sequence from A to C is the sequence from A to B followed by the sequence from B to C. This can be applied to the phoneme recognition problem if the least value of the accumulated cost function is associated with each node in figure 3. Initially, at time $t = 0$, the least accumulated distance, $D_n^{(t)}$, is zero for every valid stating node, n. At any time, equation 4 defines the distance, $C_n^{(t)}$, between the actual output at t and the target output for node n. The best accumulated distance to node m at time $t + 1$ is then just the minimum of the possible accumulated distances plus and transition cost, T_{mn}:

$$D_m^{(t+1)} = \min_n \left(D_n^{(t)} + C_n^{(t)} + T_{mn} \right) \tag{5}$$

At the end of the sentence, the possible legal final nodes are searched to give the best accumulated distance for the whole sentence. If the best transition is noted at each time step, then this information can be used to trace the best route backwards through the sentence, yielding the best match phoneme string.

Increasing the average transition cost, $\langle T_{mn} \rangle$, increases the average duration of the phonemes and so results in fewer insertion errors and more deletion errors. In a complete speech recognition system it is assumed that insertion and deletion errors are equally harmful and thus the transition cost provides a useful mechanism for balancing these errors. The simplest non-trivial form for the transitional cost is:

$$T_{mn} = \begin{cases} 0 & \text{for } m \neq n \\ \beta & \text{otherwise} \end{cases} \tag{6}$$

Figure 4 shows the trade off of insertion for deletion errors for a range of β. The recognition accuracy is also shown which is defined to be 100% minus the percentage of insertion, substitution and deletion errors.

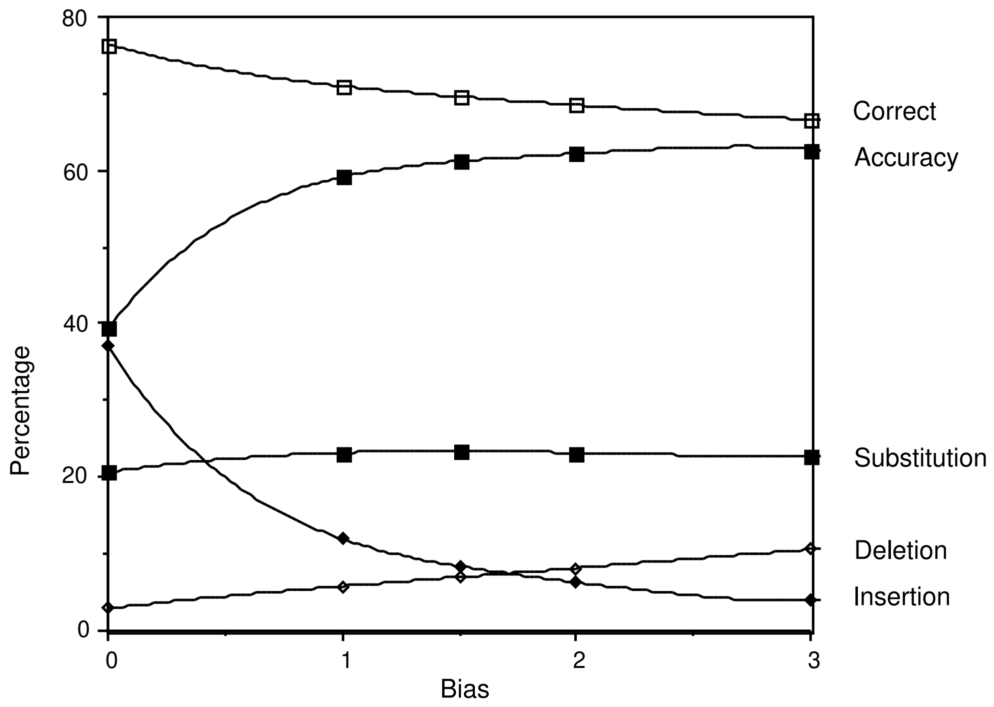

Figure 4: Changes in recognition rate for variation in the bias, β

In addition, other forms of T_{mn} have been considered and the recognition rates when insertion and deletion errors cancel is shown in table 4. Three forms for the transition cost based on bigram probabilities, B_{mn}, were tried, and in addition durational probabilities were used. The bigram probability distribution was calculated from a matrix of the number of co-occurrences of the two symbols in the training set. Similarly the duration probability distribution of was calculated from a histogram of the duration in frames of all the occurrences of that symbol in the training set. To avoid zero probabilities, a small constant (0.5) was added to each frequency count before normalisation.

Parsing with explicit durational constraints requires one storage location per phoneme as opposed to a single storage location if duration constraints are not used. From table 4 it can be seen that the durational information makes little difference to the recognition results. This is a very important for fast parsing to the word level, as described in the next section.

A confusion matrix for the $T_{mn} = \beta * \log B_{mn}$ results is given in figure 5. The hand labels are on the vertical axis and the recogniser labels are on the horizontal axis. The null symbol, -, is added so that insertion and deletion errors may be shown. The area of the square at the intersection of two symbols is proportional

T_{mn}	correct	insert	subst.	delet.	accur.
β	69.1%	7.5%	23.3%	7.5%	61.7%
$\beta + \log B_{mn}$	69.8%	6.2%	24.0%	6.3%	63.6%
$\beta * \log B_{mn}$	70.0%	6.6%	23.5%	6.5%	63.5%
$\beta +$ duration	69.5%	6.9%	23.7%	6.9%	62.6%
$\beta + \log B_{mn} +$ duration	69.8%	5.9%	24.0%	6.1%	63.9%

Table 4: Recognition rates for difference transition functions

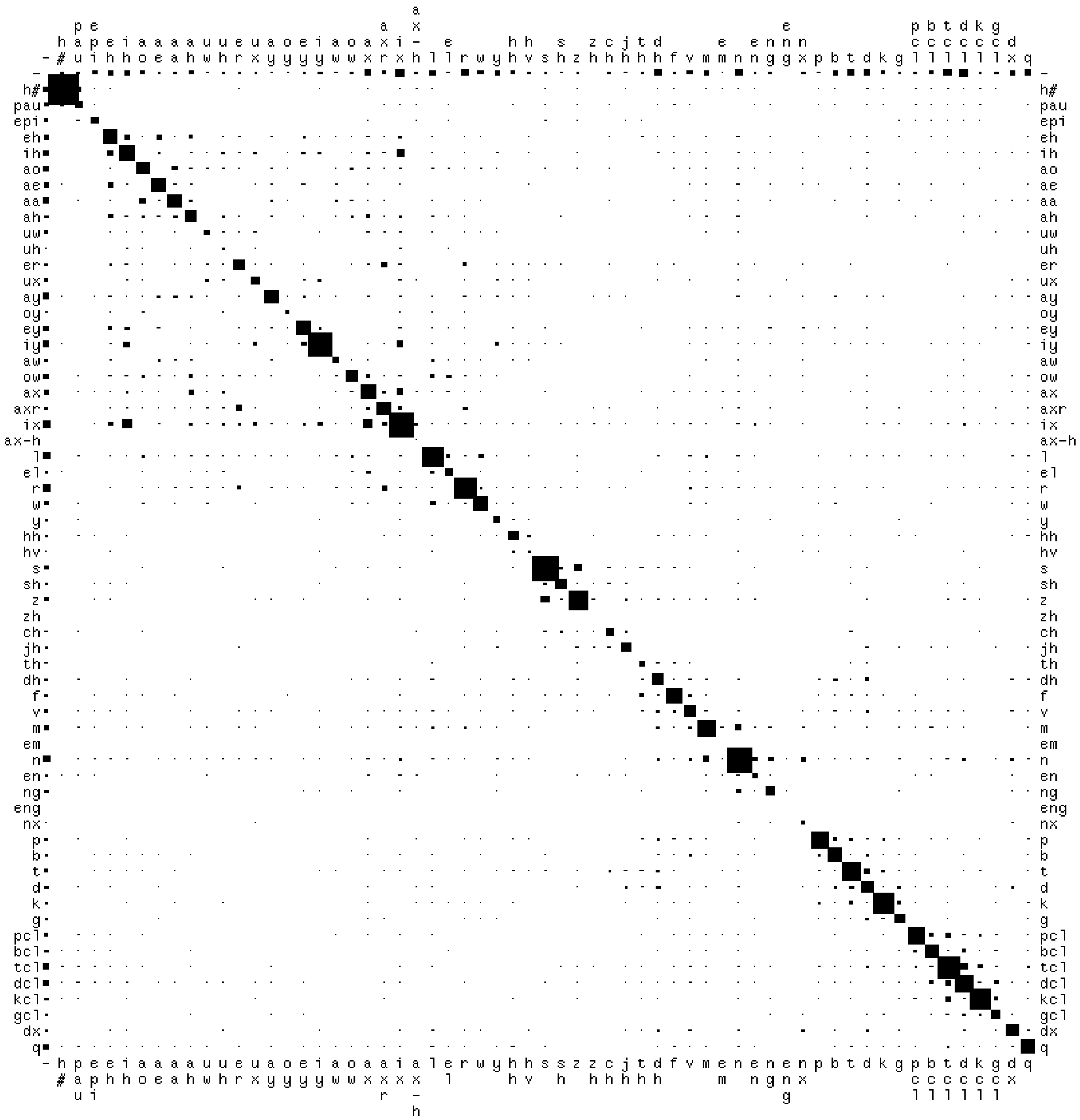

Figure 5: Confusion matrix

to the number of such points in the test set. The resolution is three phonemes per pixel rounded up (about 0.01% of the total). The strong diagonal represents the 70.0% of symbols recognised correctly, and the remainder are errors. The top sixteen errors from this matrix are given in table 5. The most common substitution errors are between symbols belonging to the same broad class and often involve reduced vowels.

hand label	recogniser label	percentage of all errors
ix	ih	1.50%
ix	ax	1.23%
-	dcl	1.13%
-	ix	1.07%
r	-	0.94%
ih	ix	0.94%
ix	-	0.92%
-	tcl	0.85%
z	s	0.84%
n	-	0.83%
tcl	dcl	0.81%
l	-	0.77%
-	dh	0.76%
s	z	0.74%
-	r	0.73%
-	n	0.73%

Table 5: The sixteen most common errors

4.1 Comparison with the SPHINX phone recogniser

The best conventional technique for phoneme recognition is that of HMMs using multiple codebooks and generalised triphones, in particular the phoneme recognition of the CMU SPHINX system (Lee and Hon, 1989). In order to compare the results of this paper with the HMM approach, the 61 symbols set is mapped to a 39 symbol set according to table 2. Also the test set is reduced to those sentences marked with and asterisk in table 1. The recognition rates for the two sizes of symbol set and the two test sets of sentences are given in table 6.

5 Word Recognition

This section outlines the extension of the REPN phoneme recogniser to the problem of word recognition. The 1000 word DARPA Resource Management task is used to demonstrate the system.

tla	test set	nsymbol	correct	insert	subst.	delet.	accur.
REPN	large	61	70.0%	6.6%	23.5%	6.5%	63.5%
REPN	large	39	76.5%	6.7%	17.0%	6.6%	69.8%
REPN	small	61	69.4%	7.5%	24.6%	5.9%	61.9%
REPN	small	39	76.4%	7.6%	17.7%	5.9%	68.9%
SPHINX	small	39	73.8%	7.7%	19.6%	6.6%	66.1%

Table 6: Comparison with the SPHINX phone recogniser

The dictionary was based on that used in the SPHINX system (Lee, 1989, Appendix II). The SPHINX phoneme set was expanded according to table 7 and the output of the recogniser was reduced according to table 2. In addition, multiple closures, such as `/kcl tcl/` were reduced to the first symbol. No attempt was made to deal with glottal stops, epenthetic silences or pauses.

SPHINX	50SET	SPHINX	50SET
B	bcl b	BD	bcl
D	dcl d	DD	dcl
G	gcl g	GD	gcl
P	pcl p	PD	pcl
T	tcl t	TD	tcl
K	kcl k	KD	kcl
TS	tcl t s	SIL	h#

Table 7: SPHINX to 50SET symbols

The computation may be decreased if the dictionary is ordered as a tree structure as in figure 6 as phonemes that occur at the beginning of many words need only be searched for once. Ordering the dictionary in this way gave a search space of 3058 nodes as compared with 6366 nodes with no shared phonemes.

Two level dynamic programming can be used to search for the best word string in much the same way as in the phoneme recognition. Every node in figure 6 has an associated least value of the accumulated cost function. At every time step, equation 5 can be used to update this value. However, with a tree structured dictionary the possible transitions are limited to either looping to the same state, or descending down the tree. This limits the number of possible transitions into a node to two, so the dynamic programming may be carried out with a simple comparison of these values for each node. Some care must be taken to deal with the terminal nodes correctly during backtracking. The computational load of this approach is considerably less than a multi-state HMM. This advantage is borne out in practice as the complete 1000 word parsing stage without grammar can be carried out faster than real time on a SUN 4/330 workstation with no pruning. Results for this

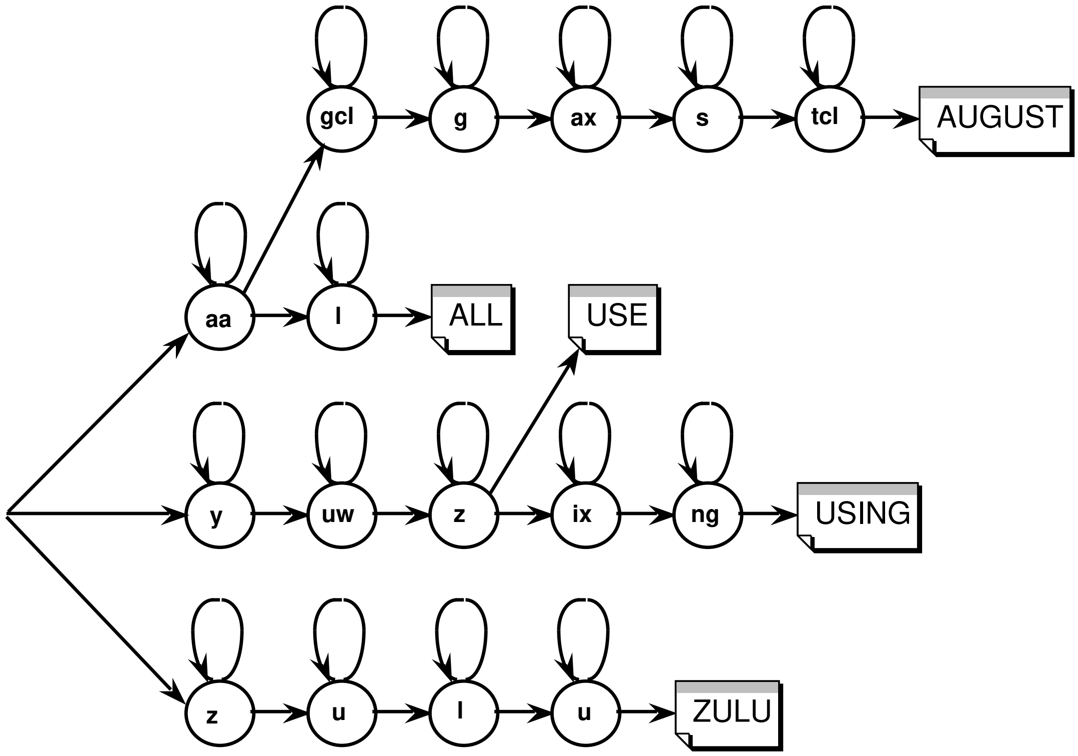

Figure 6: Part of the tree structured 1000 word dictionary

task are given in table 8 (Lee, 1989; Lee et al., 1990). The "Imp." results include the bilinear transform, multiple codebook and implicit insertion/deletion modelling improvements over the baseline SPHINX system, but do not include other improvements such as word duration modelling, context-dependent phones or function-word dependent phones. As such they represent roughly the baseline state of the REPN system.

Method	database	stage	perp.	correct	accur.	perp.	correct	accur.
REPN	TIMIT	baseline	992	52.6%	44.1%	60	76.4%	71.2%
SPHINX	TIMIT	Imp.	997	38.5%	32.1%	60	–	–
SPHINX	RM	Imp.	997	50.0%	45.3%	60	91.2%	90.6%
SPHINX	RM	Jan90	997	–	81.9%	60	–	96.2%

Table 8: Resource Management word recognition rates

Further insight into the difference in performance in table 8 can be obtained by examining the relative performance of the phoneme recogniser on the TIMIT and the Resource Management databases on the 50 symbol set. The "correct" Resource Management transcriptions were generated by concatenating the pronunciations for each word giving a phoneme recognition rate of 62.6% correct (50.5% accuracy) compared with the TIMIT results of 71.1% correct (65.5% accuracy). This is a

45% increase in the number of errors when porting between these databases. The increased error rate has two main sources: firstly, there may be variations in the recording conditions of the two databases; and secondly, there are a range of pronunciations which are acceptable for a given word, so limiting the transcription to a single pronunciation will necessarily introduce errors. In addition, there are fewer possible phonetic contexts in the 1000 word task, so there are variations present in the training of the REPN which are absent in the testing conditions.

6 Conclusion

This paper has described the use of a Recurrent Error Propagation Network (REPN) for the task of speaker independent phoneme and word recognition from continuous speech.

The phoneme recognition results have been shown to be slightly better that the best known HMM results which use multiple-codebooks, restricted HMM transitions and context-dependent phoneme models. In contrast, the REPN approach considers all types of preprocessor output in the same way, has no constraints on the internal structure and produces context independent phoneme information. Thus the REPN method is considerably simpler than the HMM method. Good phoneme recognition is likely to underlie future large vocabulary systems and this paper has presented a viable alternative to HMM phoneme recognition.

Investigation of durational constraints in phoneme recognition has shown that the internal context in the REPN is sufficient to adequately model phoneme duration. This leads to a very efficient parsing strategy for the output as only one state per phoneme is necessary. To demonstrate this, a baseline word recognition system has been built for the 1000 word DARPA Resource Management task. Initial results have been presented which are similar to the SPHINX system in its early stages of development.

7 Acknowledgements

The work described in this paper was carried out as part of an ESPRIT Basic Research Action project (3207). The authors would like to acknowledge NIST for the provision of the DARPA TIMIT and Resource Management databases and the ParSiFal project IKBS/146 which developed the transputer array. They also wish to thank all members of the Speech, Vision and Robotics group of Cambridge University Engineering Department for their advice, and in particular Mike Chong, Patrick Gosling, Mahesan Niranjan, Tim Marsland, Mark Plumbley, Richard Prager, Georges Wong and Phil Woodland.

References

Aho, A. V., Hopcroft, J. E., and Ullman, J. D. (1983). *Data Structures and Algorithms*. Addison-Wesley.

Bahl, L. R., Brown, P. F., de Souza, P. V., and Mercer, R. L. (1986). Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proc. ICASSP*, pages 49–52.

Baum, E. B. and Wilczek, F. (1987). Supervised learning of probability distributions by neural networks. In Anderson, D. Z., editor, *Proceedings of Neural Information Processing Systems*, Denver. American Institute of Physics.

Bladon, R. A. W. and Lindblom, B. (1981). Modeling the judgement of vowel quality differences. *Journal of the Acoustical Society of America*, 69(5):1414–1422.

Bourlard, H. and Wellekens, C. J. (1987). Multilayer perceptrons and automatic speech recognition. In *Proceedings of the IEEE First Annual International Conference on Neural Networks*, pages IV:407–416, San Diego.

Chan, L. W. and Fallside, F. (1987). An adaptive training algorithm for back propagation networks. *Computer Speech and Language*, 2(3/4):205–218.

Franzini, M. A., Witbrock, M. J., and Lee, K.-F. (1989). A connectionist approach to continuous speech recognition. In *Proc. ICASSP*, pages 425–428.

Garofolo, J. S. (1988). *Getting Started with the DARPA TIMIT CD-ROM: An Acoustic Phonetic Continuous Speech Database*. National Institute of Standards and Technology (NIST), Gaithersburgh, MD.

Hinton, G. E. (1987). Connectionist learning procedures. Technical Report CMU-CS-87-115, Computer Science Department, Carnegie-Mellon University.

Jacobs, R. A. (1988). Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1:295–307.

Kohonen, T. (1988). *Self-Organization and Associative Memory*. Springer-Verlag, New York, second edition.

Ladefoged, P. (1982). *A Course in Phonetics*. Harcourt Brace Jovanovich, New York, second edition.

Lee, K.-F. (1989). *Automatic Speech Recognition: The Development of the SPHINX System*. Kluwer Academic Publishers, Boston.

Lee, K.-F. and Hon, H.-W. (1989). Speaker-independent phone recognition using hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(11):1641–1648.

Lee, K.-F., Hon, H.-W., Hwang, M.-Y., and Mahajan, S. (1990). Recent progress and future outlook of the SPHINX speech recognition system. *Computer Speech and Language*, 4:57–69.

Levinson, S. E., Liberman, M. Y., Ljolje, A., and Miller, L. G. (1989). Speaker independent phonetic transcription of fluent speech for large vocabulary speech recognition. In *Proc. ICASSP*, pages 441–444.

Levinson, S. E., Rabiner, L. R., and Sondhi, M. M. (1983). An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *The Bell System Technical Journal*, 62(4):1035–1074.

Lippmann, R. P. (1989). Review of neural networks for speech recognition. *Neural Computation*, 1(1):1–38.

Morgan, N. and Bourlard, H. (1990). Continuous speech recognition using multilayer perceptrons with hidden Markov models. In *Proc. ICASSP*, pages 413–416.

Ney, H. (1984). The use of a one-stage dynamic programming algorithm for connected word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):263–271.

Pearlmutter, B. A. (1990). Dynamic recurrent neural networks. Technical Report CMU-CS-90-196, School of Computer Science, Carnegie-Mellon University.

Price, P., Fisher, W. M., Bernstein, J., and Pallett, D. S. (1988). The DARPA 1000-word Resource Management database for continuous speech recognition. In *Proc. ICASSP*, pages 651–654.

Pullum, G. K. and Ladusaw, W. A. (1986). *Phonetic Symbol Guide*. University of Chicago Press, Chicago.

Rabiner, L. R. and Juang, B. H. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16.

Rabiner, L. R., Levinson, S. E., and Sondhi, M. M. (1983). On the application of vector quantization and hidden Markov models to speaker-independent, isolated word recognition. *The Bell System Technical Journal*, 62(4):1075–1105.

Robinson, A. J. (1989). *Dynamic Error Propagation Networks*. PhD thesis, Cambridge University Engineering Department.

Robinson, A. J. and Fallside, F. (1989). A dynamic connectionist model for phoneme recognition. In *Neural Networks from Models to Applications: Proceedings of nEuro'88*, pages 541–550. I.D.S.E.T., Paris.

Robinson, T. and Fallside, F. (1990a). Phoneme recognition from the TIMIT database using recurrent error propagation networks. Technical Report CUED/F-INFENG/TR.42, Cambridge University Engineering Department.

Robinson, T. and Fallside, F. (1990b). Word recognition from the DARPA resource management database with the Cambridge recurrent error propagation network speech recognition system. In *Third Australian International Conference on Speech Science and Technology*, Melbourne.

Robinson, T., Holdsworth, J., Patterson, R., and Fallside, F. (1990). A comparison of preprocessors for the Cambridge recurrent error propagation network speech recognition system. In *Proceedings of the International Conference on Spoken Language Processing*, Kobe, Japan.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E. and McClelland, J. L., editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. I: Foundations.*, chapter 8. Bradford Books/MIT Press, Cambridge, MA.

Rumelhart, D. E. and McClelland, J. L. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. I: Foundations.* MIT Press, Cambridge, MA.

Seneff, S. and Zue, V. W. (1988). Transcription and alignment of the TIMIT database. In Garofolo, J. S., editor, *Getting Started with the DARPA TIMIT CD-ROM: An Acoustic Phonetic Continuous Speech Database*. National Institute of Standards and Technology (NIST), Gaithersburgh, MD.

Solla, S. A., Levin, E., and Fleisher, M. (1988). Accelerated learning in layered neural networks. *Complex Systems*, 2.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. (1987). Phoneme recognition using time–delay neural networks. Technical report, ATR Interpreting Telephony Research Laboratories.

Young, S. J. (1990). Competitive training in hidden Markov models. In *Proc. ICASSP*, pages 681–684. Expanded in the technical report CUED/F-INFENG/TR.41, Cambridge University Engineering Department.