Several Improvements to a Recurrent Error Propagation Network Phone Recognition System

Tony Robinson
`ajr@eng.cam.ac.uk`

CUED/F-INFENG/TR82

30 September 1991

Abstract

Recurrent Error Propagation Networks have been shown to give good performance on the speaker independent phone recognition task in comparison with other methods (Robinson and Fallside, 1991). This short report describes several recent improvements made to the existing recogniser for the TIMIT database.

The improvements are: an addition to the preprocessor to represent voicing information; use of histogram normalisation on the input channels of the network; normalisation of the output channels to enforce unity sum; a change in the cost function to give equal weighting to each target symbol; a change in the representation of the outputs to reduce quantisation errors; retraining on the complete TIMIT training set; and the better estimation of HMM phone models.

Most of these changes decrease the number of arbitrary parameters used and allow for the integration of the system with standard HMM techniques. The result of these changes is a decrease in the number of errors by about 16% (from 36.5% to 30.7% when all 61 TIMIT phones are used and from 30.2% to 25.0% on a reduced 39 phone set).

1 Introduction

It is beyond the scope of this report to describe the recurrent network phone recognition system in full detail. Instead a brief outline will be given and more details can be found in Robinson and Fallside (1991).

A recurrent error propagation network can be considered as a sequence of standard error propagation networks (Rumelhart et al., 1986) with the addition of feedback. The input and output vectors are divided into external and internal portions. The internal output, $x(t)$, forms the state vector and is fed back to the internal input of the next frame, as in figure 1. For practical reasons, training the network is performed by unfolding in time, although other training schemes are possible.

The external input, $u(t)$, to the network consists of 16ms frames of preprocessed speech. A comparison of preprocessors for this task can be found in Robinson et al. (1990). The conclusion of this study was that one power channel and 20 power spectral channels were

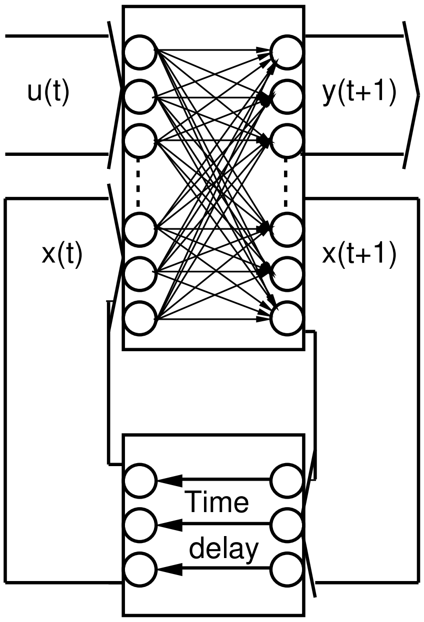

Figure 1: The Recurrent Error Propagation Network

preferable considering performance and simplicity. This preprocessor forms the basis of this report although further discussion is presented later.

The external output, $y(t)$, of the network represents the best estimate of the probabilities that the specified frame belongs to a given symbol. Currently it is assumed that there is only one "correct" frame, and that the labelling process assigned this label to the frame. As the ideal output from the network is one unit on and the remainder off for the duration of the phone, this can be used as a template when doing phone recognition. Searching for the best sequence of templates can be efficiently accomplished using dynamic programming:

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

where $D_n^{(t)}$ is the minimum total cost incurred to get to state n at time t, $C_n^{(t)}$ is the cost associated with accepting the frame at time t by state n, and T_{mn} is a transition cost incurred when moving from state n to m. Changing the transition cost allows the control of insertion and deletion errors. By incurring no cost on template boundaries ($T_{mn} = 0$) the algorithm reduces to picking the largest output unit and (optionally) merging adjacent frames with the same label. This leads to a large number of insertion errors which can be reduced by increasing the transition cost. This is very similar to finding the maximum likelihood state sequence in a HMM using the viterbi algorithm, and the similarity is discussed further in section 13.

The rest of the report will describe a set of changes from the preprocessor stage through to the final symbol recognition. In order that the changes may be rapidly evaluated, and for consistency with earlier work, the number of state units is set to 96 for most of the evaluations but finally increased to 192. The motivation behind the changes is not only to increase recognition accuracy, but also to clean up the framework of the connectionist recogniser so that higher level probabilistic constraints, such as word models, may be properly incorporated.

The first experiments use the prototype version of the DARPA TIMIT Acoustic Phonetic Database (Lamel et al., 1987). This has 420 talkers of American English speaking a total 4200 sentences recorded in a noise free environment at 16kHz. The concluding experiments use the complete version (October 1990) with 420 training speakers and 210 test speakers.

2 A more efficient implementation of the baseline system

The computational cost needed to train the connectionist recogniser is significant in spite of changes made to the basic error propagation algorithm to decrease the learning time. In order to train in reasonable time, a processor array of 64 T800 transputers is used with the training data partitioned over the processors and one copy of the complete network per processor. It was found that by placing the code for the inner multiply-and-accumulate loop in on-chip RAM, and overlapping communication and processing, a factor of two was gained in speed over the previous implementation. Slight algorithm changes were incurred but these had no significant effect on the performance as can be seen in table 1.

Version	correct	insertion	substitution	deletion	total errors
Original	65.5%	6.1%	27.5%	7.0%	40.6%
Faster	65.7%	6.3%	27.3%	6.9%	40.6%

Table 1: Effect of implementation change on performance

3 Recognition in noise

Noise was added to the clean speech to get an idea of the tolerance of the recogniser. Gaussian white noise passed though a low pass first order filter with 3dB point at 600Hz was used. The recogniser was trained and tested on the same noise levels. Figure 2 and table 2 show that there is relatively little degradation when the signal to noise ratio is greater than 30dB, but significant degradation when less than 20dB.

SNR/dB	correct	insertion	substitution	deletion	total errors
0	43.8%	12.0%	44.5%	11.7%	68.2%
10	55.4%	8.7%	36.0%	8.6%	53.3%
20	61.8%	7.3%	30.8%	7.4%	45.5%
30	64.4%	6.3%	28.3%	7.3%	41.9%
40	65.6%	6.3%	27.1%	7.2%	40.7%
∞	66.2%	6.3%	26.8%	7.0%	40.0%

Table 2: Performance degradation in noise

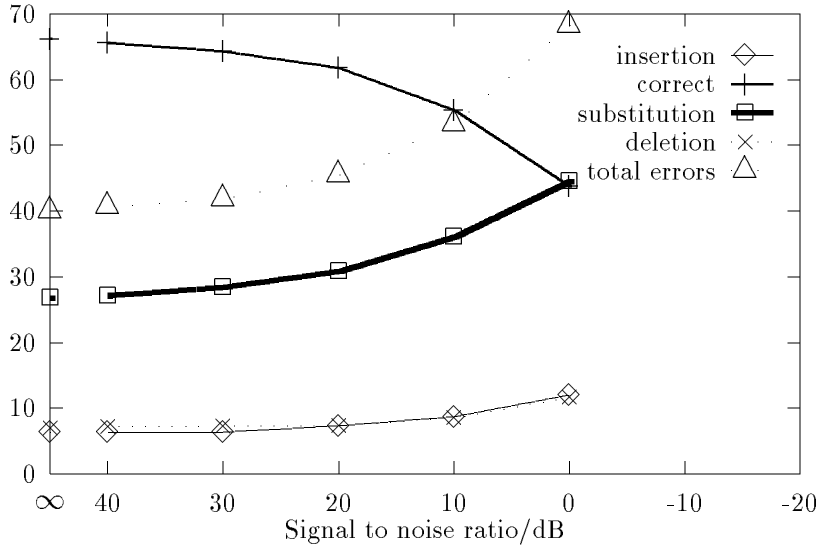

Figure 2: Performance degradation in noise

4 Addition of fundamental frequency and degree of voicing information

For recordings in a low noise background, unnatural but intelligible speech may be reconstructed from the power output of a bank of filters (about 20 are required). However, this requires the estimation of whether the frame is voiced or not, and for voiced frames a suitable fundamental frequency, $f0$. Using a voicing decision and an estimate of $f0$ from the original speech improves the intelligibility and the naturalness of the resynthesised speech. Thus an estimation of the power spectra envelope (possibly in LPC form) along with degree of voicing and $f0$ frequency can be considered the basics for natural speech synthesis.

Version	correct	insertion	substitution	deletion	total errors
Original	65.5%	6.1%	27.5%	7.0%	40.6%
$f0$ only	65.6%	6.1%	27.6%	6.9%	40.5%
voicing only	65.8%	6.3%	27.2%	7.1%	40.6%
$f0$ and voicing	65.6%	6.2%	27.2%	7.2%	40.6%

Table 3: Effect of adding $f0$ and voicing information

However, in speech recognition often only the shape of the power spectra is used, the other parameters are often regarded as insignificant at best - and introducing a significant statistical bias at worst. This problem is the result of training on small data sets and the difficulty of combining differing data types as speech recogniser input. In the earlier preprocessor search it was found that addition of fundamental frequency and degree of voicing information makes no difference to the recognition rate. Table 3 repeats this result by computing voicing parameters from the the peak in the autocorrelation function of a single frame within in the range of reasonable pitch values (62.5Hz to 400Hz). The degree of voicing is taken as the relative height of this peak to the value a zero delay, and the pitch is taken as the position of the peak smoothed by a first order filter whose time constant was varied in proportion to the degree of voicing. Thus the filter averaged over a short time period when the speech was voiced and over a long time period for weakly voiced or unvoiced speech.

5 Input normalisation

In the previous work it was shown that the power to which the input activations are raised affects the performance. Perviously the output channels of the preprocessor were linearly scaled to fit into a single byte so that only a few values were thresholded (about one in a thousand at either end of the range).

An alternative approach is to use a monotonically increasing function which transforms the output from the preprocessor into a standard distribution. As there are many data samples (about 400,000), this may be achieved by computing a histogram and dividing the total range into equi-probable regions. 256 regions were chosen so that the value could be stored in a single byte. Just before the channels are presented to the network they are transformed using a look-up table such that the probability density function is a zero-mean unit-variance Gaussian. The results of doing this are presented in table 4. It is not clear whether the improvement in performance is due to a reduction in the quantisation error on the input, or because this transformation results in more easily separable patterns in the input space of the network.

Version	correct	insertion	substitution	deletion	total errors
Original	65.5%	6.1%	27.5%	7.0%	40.6%
i/p normalised	68.8%	6.2%	25.3%	5.9%	37.4%

Table 4: Effect of input normalisation

6 Output normalisation

Previously the training algorithm assumed that the output units were measures of probabilities of independent events. Thus the range of the output values (activations) was 0.0 to 1.0 independent of the other outputs. However, this is clearly not the case as the output probabilities are constrained to sum to unity. This property can be enforced by using the Potts or softmax (Bridle, 1989) function as the activation function on the output units:

$$f(x_i) = \frac{\exp(x_i)}{\sum_i \exp(x_i)} \tag{2}$$

Like the linear activation function for the least-mean-squares distance metric, and the sigmoidal activation function applied to unnormalised output with the cross-entropy distance metric, the softmax activation function results in the derivative of the per pattern cost function, E, with respect to the summed input, x_i, is:

$$\frac{\partial E}{\partial x_i} = x_i - t_i \tag{3}$$

where t_i is the target activation for unit i. The result of training with this form of normalised output in given in table 5

Version	correct	insertion	substitution	deletion	total errors
Original	65.5%	6.1%	27.5%	7.0%	40.6%
o/p normalised	67.0%	6.8%	26.9%	6.2%	39.9%

Table 5: Effect of output normalisation

7 Weighting the cost function

Previously the cost function assigned equal weight to each frame of speech. However the objective is to maximise the number of phones recognised correctly - thus it would be better to weight each phone equally and thus give a relatively higher frame weighting to shorter phones. This weighting of the cost function does not change the desired output of the network for a particular frame, but only the modelling power allocated to producing that output.

In this implementation the length of the phone was calculated as a number of frames and the gradient signal was weighted by the inverse of the length. Thus the long periods of silence at the start and end of sentences made an equal contribution to the cost function as a one-frame stop release. The results are presented in table 6.

Version	correct	insertion	substitution	deletion	total errors
Original	65.5%	6.1%	27.5%	7.0%	40.6%
Weighted	66.1%	6.2%	26.5%	7.4%	40.1%

Table 6: Effect of the weighted cost function

It appears that this technique would also be useful for word recognition - if all words are to be scored equally then they should receive the same weighting in the cost function during training. Further dividing the cost function by the number of phones in a word would have the desired effect of weighting words equally, thereby increasing the modelling power allocated to phones in the context of short words which are the cause of many word level errors.

8 Combination of changes

Table 7 gives the result of combining all the modifications described above. The reduction in the total error rate is more that the sum of the reductions when each modification was applied in isolation. Combinations of the inclusion and removal of the pitch and degree voicing information were searched. As can be seen in table 7, the incorporation of pitch information alone made little difference, although the change was much bigger when pitch was added on top of the degree of voicing information.

Version	correct	insertion	substitution	deletion	total errors
Original	65.5%	6.1%	27.5%	7.0%	40.6%
Sections [5-7]	68.8%	6.2%	25.3%	5.9%	37.4%
[5-7] and $f0$	68.8%	6.1%	25.2%	6.1%	37.3%
[5-7] and voicing	69.2%	6.1%	24.8%	6.0%	36.9%
[5-7], $f0$ and voicing	69.3%	5.8%	24.8%	5.9%	36.5%

Table 7: Effect of the combination of changes

9 A larger network

All the previous results were obtained with a network of 96 state units in order to reduce the training time (about 3.10^{12} floating point operations). However, as shown previously the performance increases significantly with more state units. This value is limited by the quadratic increase in the number of parameters to train with increasing number of units, which results in slower training time and less space per processor for training data storage. The results with the maximum practical value, 192, are shown in table 8.

Version	correct	insertion	substitution	deletion	total errors
Sections [4-7]	69.3%	5.8%	24.8%	5.9%	36.5%
192 state units	72.9%	5.5%	21.8%	5.3%	32.6%

Table 8: From 96 to 192 state units

10 Retraining

Once trained, a forced alignment of the network output with the hand labels may be made. This results in the redefinition of the phone boundaries, perhaps to more suitable positions for network training. However, investigation of the new boundary positions showed that they were very close to the originals. This is reflected in table 9 – after retraining the network on the new boundaries for two iterations there was no change in performance. However, this is major advantage when using an unlabelled database in that the TIMIT trained model can be used to give a first rough segmentation which can be refined by iteration.

Retrain	correct	insertion	substitution	deletion	total errors
0	72.9%	5.5%	21.8%	5.3%	32.6%
1	72.8%	5.4%	21.9%	5.2%	32.6%
2	72.7%	5.2%	21.8%	5.5%	32.5%

Table 9: Effect of forced alignment and retraining

11 Final prototype TIMIT results

For compatibility with future word recognition results, the cost function used in the recognition phase was changed from minimising the sum of the cross-entropy terms (as used in training) to maximising the sum of log probabilities. In practice this makes negligible difference to the recognition score but it does ease interfacing to higher level grammatical constraints. In addition, the use of the inter-symbol bias was changed from balancing insertion and deletion errors to minimising the total number of errors. Results are presented in table 10 for the combinations of including and excluding duration and phoneme transition probabilities. The duration probabilities were calculated from a histogram of the lengths of the hand labels in the training set measured to the nearest frame. Similarly, the transition probabilities were obtained by counting the relative frequencies of transition from one phone to the next. As adjacent phones are always different, the transition probability from a phone to itself is zero. These probabilities are incorporated by adding the log of the value in on a transition. From the table it can be seen that the addition of bigram statistics reduces the number of errors by 1% but the addition of durational information makes no difference.

bigram	duration	bias	correct	insertion	substitution	deletion	total errors
no	no	4.0	70.6%	3.6%	21.0%	8.4%	33.0%
no	yes	3.0	70.6%	3.6%	21.2%	8.2%	33.0%
yes	no	1.0	71.4%	3.3%	21.4%	7.2%	32.0%
yes	yes	0.0	71.3%	3.3%	21.6%	7.1%	32.0%

Table 10: Final prototype TIMIT results

It should be noted that the use of bigram probabilities for phoneme transitions is quite a weak constraint. Any word recognition system imposes much stronger constraints on the possible transitions and in this case it is useful to use durational information.

12 Full TIMIT results

All previously reported work used the prototype TIMIT CD-ROM. The full database has 420 speakers for training and 210 speakers for testing. The analysis of section 11 was repeated using all 420 training speakers, about 1/3 more data than previously. However, due to memory limitations on the hardware used this meant that the number of state units had to be reduced to 184. Table 11 shows an improvement of 0.9% in the error rate over the previous results. This could be the effect of more training data or due to changes in the labelling between the two versions.

bigram	duration	bias	correct	insertion	substitution	deletion	total errors
no	no	4.0	71.3%	3.4%	20.5%	8.2%	32.1%
no	yes	3.0	71.6%	3.6%	20.7%	7.7%	32.0%
yes	no	1.0	72.2%	3.3%	21.1%	6.7%	31.1%
yes	yes	0.0	72.4%	3.5%	21.3%	6.3%	31.1%

Table 11: Full TIMIT results

13 A Markov model framework

The systems without durational modelling used in the previous sections can be considered as simple Markov models with one state per phone and with the emission probabilities of the states being estimated by the recurrent network. As is conventional, the transition probability from state i to state j will be designated as a_{ij}. As durational information is not used, the self loop probabilities, a_{ii}, are all equal, and thus may be factored out of the dynamic programming (viterbi search) used to find the maximum likelihood state sequence.

In the case when no bigram or durational probabilities are used, a HMM system can be constructed with the bias, β, being the difference between the log of the self loop probabilities and the log state exiting probabilities. In addition we have the constraint that the sum of the probabilities over all transitions (including the self loop) is unity:

$$\begin{aligned} \log a_{ii} - \log a_{ij} &= \beta \qquad i \neq j & (4) \\ \sum_j a_{ij} &= 1 & (5) \end{aligned}$$

For a Markov model of N states these may be solved to give:

$$a_{ij} = \begin{cases} \frac{e^{\beta}}{e^{\beta}+N-1} & \text{if } i = j \\ \frac{1}{e^{\beta}+N-1} & \text{otherwise} \end{cases} \tag{6}$$

The case where the bigram probabilities, B_{ij}, are incorporated leads to the following constraints and solution:

$$\begin{aligned} \log a_{ii} - \log a_{ij} &= \beta - \log B_{ij} \qquad i \neq j & (7) \\ a_{ij} &= \begin{cases} \frac{e^{\beta}}{1+e^{\beta}} & \text{if } i = j \\ \frac{B_{ij}}{1+e^{\beta}} & \text{otherwise} \end{cases} & (8) \end{aligned}$$

The original formulation has the advantage that by factoring out the self loop probability, no cost is incurred on a self loop transition. This avoids one addition per state. Although this computational saving may be of value when there are very many states, as in large vocabulary word recognition, parsing the network output to the phoneme level can be easily accomplished and so speed is not a consideration.

The HMM transition probabilities may also be estimated in the standard way:

$$a_{ij} \quad = \quad \frac{\sum_t \gamma_{ij}^{(t)}}{\sum_t \sum_k \gamma_{ik}^{(t)}} \tag{9}$$

where $\gamma_{ij}^{(t)}$ is the probability of being in state i at t and j at $t+1$, $\sum_t \gamma_{ij}^{(t)}$ is the expected number of transitions from i to j and $\sum_t \sum_k \gamma_{ik}^{(t)}$ is the expected number of times state i is occupied.

In the case of hand labelled data, as in the TIMIT task, the identity of the state at every time frame is known. Thus equation 9 reduces to counting the number of transitions from i to j and dividing this by the total number of frames labelled as i. This may be easily accomplished and a comparison of this automatic setting of a_{ij} and the manual setting via β is given in table 12. Whilst the performance gain is slight, it is important to be able to automatically estimate these parameters.

Version	correct	insertion	substitution	deletion	total errors
manual	72.4%	3.5%	21.3%	6.3%	31.1%
automatic	72.8%	3.5%	20.9%	6.3%	30.7%

Table 12: Parameter setting using HMM estimation

14 Comparison with other systems

A comparison with other systems may be made if the 61 TIMIT symbols are reduced to a common set. In this section the mapping is done on the symbolic output of the recogniser. There may be a small advantage in training the recogniser on fewer symbols as more training data is available for each one, but this will not be pursued here.

The first HMM results on this task were provided with the phone recognition component of the CMU SPHINX recognition system (Lee and Hon, 1989). This used multiple codebooks and right-context HMMs. The output from this recogniser may be emulated by first mapping all closures to the silence symbol, sil, then performing the 16 reductions given in table 13 and finally deleting all instance of the glottal stop symbol, q, from the output. The comparison is presented as entries "39" and "SPHINX" in table 14.

The latest HMM results are provided by the Bell Laboratories system which uses single Gaussian continuous density triphone HMMs with durational constraints and trigram phonotactic constraints (Ljolje, 1991). A comparison with these results is a little more involved. Firstly any sentence initial or sentence final silence, h#, is deleted. These are easy to recognise and provide no information. The performance with this modification is given under the heading "61′" in table 14. Secondly, any closure-release pair is converted

h#	pau	epi	ax-h	hv	em	eng	ux	el	axr	en	nx	zh	ao	ih	dx
sil	sil	sil	ax	hh	m	ng	uw	l	er	n	n	sh	aa	ix	sil

Table 13: Mapping to 39 symbols

to a single symbol, so "tcl t" becomes a single phone whereas "dcl b" remains as two. This was accomplished by mapping {bcl, dcl, gcl, pcl, tcl, kcl} to {b, d, g, p, t, k} and then deleting multiple instances of members of the second set. Thirdly the mappings of table 13 are performed. Finally "q" was mapped to "dx" to give the comparison under the headings "39′" and "CVDHMM".

Version	correct	insertion	substitution	deletion	total errors
61	72.8%(72.1%)	3.5%(3.4%)	20.9%(21.0%)	6.3%(6.9%)	30.7%(31.3%)
61′	71.3%(70.6%)	3.7%(3.6%)	22.1%(22.2%)	6.6%(7.3%)	32.4%(33.1%)
39	78.6%(77.5%)	3.6%(3.6%)	15.0%(15.5%)	6.4%(6.9%)	25.0%(26.1%)
SPHINX	73.8%	7.7%	19.6%	6.6%	33.9%
39′	74.3%(73.1%)	3.6%(3.4%)	18.0%(18.5%)	7.7%(8.4%)	29.2%(30.3%)
CVDHMM	74.8%	5.4%	19.6%	5.6%	30.6%

Table 14: Comparison with other TIMIT phone recognisers: The main percentages are an evaluation over the whole of the test set, the numbers in parentheses are the evaluation over the smaller "core test set".

It is to be expected that better HMM results could be obtained by using tied mixtures to estimate the emission probabilities, and that better connectionist results could be obtained by using trigram phonotactic constraints.

15 Conclusion

This report has presented many changes to an existing connectionist system, most of which have decreased the number of arbitrary parameters used or allowed for the integration of the system with standard HMM techniques. Altogether the changes have made a modest decrease in the error rate. The previous reported result was 36.5% errors on 61 phones and 30.2% on 39 phones so this work represents a decrease in the number of errors by about 16%.

The resulting system is competitive with the best implemented and published HMM technology. However, further improvements in HMM systems seem likely with techniques such as tied mixtures and discriminative training which are well established for HMM word recognition. The recurrent network remains a comparatively simple system compared with the mature HMM technology.

Acknowledgements

The work described in this paper was carried out as part of an ESPRIT Basic Research Action project (3207). The author would like to acknowledge NIST for the provision of the DARPA TIMIT database and the ParSiFal project IKBS/146 which developed the transputer array. Thanks also to Andrej Ljolje for discussions on a common symbol set and all members of the Speech, Vision and Robotics group of Cambridge University Engineering Department for their advice, and in particular Mark Plumbley who inspired much of this work.

References

Bridle, J. S. (1989). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In Fougelman-Soulie, F. and Heŕault, J., editors, *Neuro-computing: Algorithms, Architectures and Applicatations*, pages 227–236. Springer-Verlag.

Lamel, L. F., Kasel, R. H., and Seneff, S. (1987). Speech database development: Design and analysis of the acoustic-phonetic corpus. In *Proceedings of the DARPA Speech Recognition Workshop*, pages 26–32.

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.

Ljolje, A. (1991). New developments in phone recognition using an ergodic hidden markov model. Technical memorandum TM-11222-910829-12, A T and T Bell Laboratories. Submitted to IEEE transactions on Signal Processing as: High Accuracy Phone Recognition Using Context Clustering and Quasi-triphonic Models.

Robinson, T. and Fallside, F. (1991). A recurrent error propagation network speech recognition system. *Computer Speech and Language*, 5(3).

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.