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 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)

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 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.

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

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.

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

TIMIT	50SET	39SET	IPA	TIMIT	50SET	39SET	IPA
p	p	p	**p**	b	b	b	**b**
t	t	t	**t**	d	d	d	**d**
k	k	k	**k**	g	g	g	**g**
pcl	pcl	sil	**p**$^{\text{o}}$	bcl	bcl	sil	**b**$^{\text{o}}$
tcl	tcl	sil	**d**$^{\text{o}}$	dcl	dcl	sil	**d**$^{\text{o}}$
kcl	kcl	sil	**k**$^{\text{o}}$	gcl	gcl	sil	**g**$^{\text{o}}$
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	ɑ$^{\text{y}}$	oy	oy	oy	ɔ$^{\text{y}}$
ey	ey	ey	**e**$^{\text{y}}$	iy	iy	iy	**i**$^{\text{y}}$
aw	aw	aw	ɑ$^{\text{w}}$	ow	ow	ow	**o**$^{\text{w}}$
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

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.

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

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

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:

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

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

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, 1988) and Solla, Levin and Fleisher (Solla