INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center Street • Suite 600 • Berkeley, California 94704 • 1-510-642-4274 • FAX 1-510-643-7684

Connectionist Probability Estimation in HMM Speech Recognition

Steve Renals and Nelson Morgan

TR-92-081

December 1992

ABSTRACT

This report is concerned with integrating connectionist networks into a hidden Markov model (HMM) speech recognition system, This is achieved through a statistical understanding of connectionist networks as probability estimators, first elucidated by Hervé Bourlard. We review the basis of HMM speech recognition, and point out the possible benefits of incorporating connectionist networks. We discuss some issues necessary to the construction of a connectionist HMM recognition system, and describe the performance of such a system, including evaluations on the DARPA database, in collaboration with Mike Cohen and Horacio Franco of SRI International. In conclusion, we show that a connectionist component improves a state of the art HMM system.

Part I

INTRODUCTION

Over the past few years, connectionist models have been widely proposed as a potentially powerful approach to speech recognition (e.g. Makino et al. (1983), Huang et al. (1988) and Waibel et al. (1989)). However, whilst connectionist methods have performed well in discrete utterance recognition addressed as a static pattern recognition problem (e.g. Fanty et al. (1992)), architectures and associated training algorithms have not yet been developed that can adequately model the temporal structure of speech.

State of the art continuous speech recognition systems are statistical in nature, based around hidden Markov models (HMMs) (e.g. Lee (1988), Cohen et al. (1990)). Within this statistical framework, connectionist methods have been used to improve continuous speech recognition systems (Renals et al., 1992; Austin et al., 1992; Robinson, 1992). Such improvements have resulted from an integration of the connectionist and statistical components, based upon a statistical understanding of the computations being performed by connectionist networks.

This report discusses such a connectionist–statistical speech recognition system, developed at ICSI, in collaboration with SRI International. We shall review the HMM approach to speech recognition, and through a discussion of possible training criteria and probability estimators, describe how a probabilistic understanding of feed-forward networks enables the principled construction of a hybrid connectionist–HMM system. We shall report experimental results that we have achieved using this system with various network architectures.

Part II

STATISTICAL SPEECH RECOGNITION

HIDDEN MARKOV MODELS

Hidden Markov modelling of speech, assumes that speech is a *piecewise stationary* process. That is, an utterance is modelled as a succession of discrete stationary states, with instantaneous transitions between these states. A simple HMM is illustrated in figure 1. Essentially, a HMM is a stochastic automaton, with a stochastic output process attached to each state.[1] Thus we have two concurrent processes: a Markov process modelling the temporal structure of speech; and a set of state output processes modelling the instantaneous character of the speech signal. Note a wider class of models, hidden semi-Markov models, include a third stochastic process, modelling state duration. We shall not deal with models of this class, concerning ourselves only with time-synchronous HMMs.

Ideally, there would be a unique HMM for all allowable sentences in the language being modelled; this is clearly infeasible for any but the most trivial of cases. A hierarchical modelling scheme is usually adopted. Each sentence is modelled as a sequence of words. The number of total models required is now much smaller—rather than one model per possible sentence, there is now one model per vocabulary word. However, for large vocabularies, a very large training set would be required to learn models for each word: some words occur very infrequently. Thus a further decomposition is required into subword units.[2] Although there are good linguistic arguments for choosing units such as syllables or demi-syllables, the unit most commonly used is the phone[3] and is the unit used here. Thus we have around 60 basic phone HMMs (for English), and from these we construct word and sentence models. For any given sentence we may write down the corresponding HMM; each state in that HMM is contributed by a constituent phone HMM.

HMM SPEECH RECOGNITION

The basic problem of speech recognition is to be able to output the correct sentence corresponding to a spoken utterance. A general approach to this problem is to output the most probable sentence given the acoustic data. Thus we must choose sentence $\mathbf{S}$, for which the probability[4] $P(\mathbf{S}|\mathbf{X})$, is a maximum, where $\mathbf{X}$ is a sequence of acoustic data vectors.[5] If we choose to use hidden Markov models, then a sentence is represented by a particular state sequence, Q_1^N, and the probability we

1. More generally, the stochastic process could be regarded as being attached to each transition. If the processes attached to each transition exiting a particular state are constrained to be identical, then this is equivalent to that process being attached to the state. In practice, state processes, rather than transition processes, are used in most speech recognition systems.
2. Although many large vocabulary continuous speech recognition systems do have individual models for frequently occurring words—particularly function words, such as “and” and “the”.
3. A phone is an acoustic category, whereas a phoneme is a perceptual category. For example, an utterance of the word “citizenship” may be phonemically transcribed as /s ih dx ih z en sh i p/, although the /z/ may be voiced or unvoiced. A phone transcription would represent an unvoiced /z/ as [s]. The distinction between phones and phonemes is often confused in the speech recognition literature.
4. We use P to represent a probability, and p to represent a probability density.
5. This probability should actually be written as $P(\mathbf{S}|\mathbf{X}, \Theta)$, where Θ represents the model parameters. For now we shall ignore this conditioning on the model.

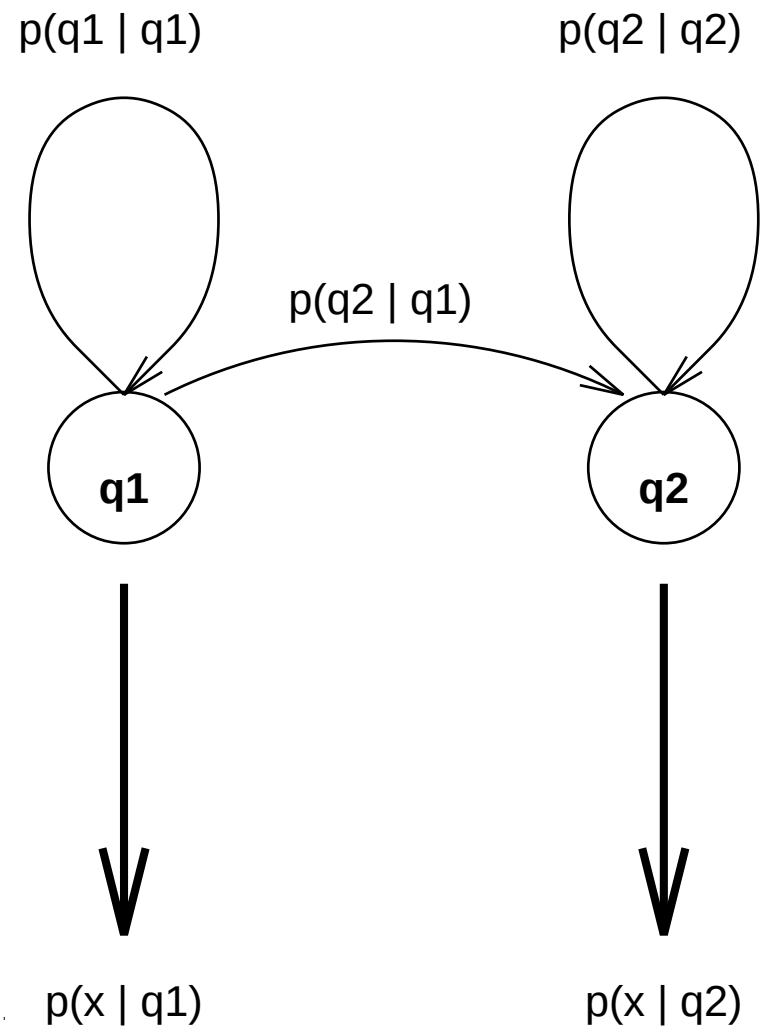

Figure 1: A schematic of a two state, left-to-right hidden Markov model (HMM). A hidden Markov model is a stochastic automaton, consisting of set of states, and corresponding transitions between states. HMMs are "hidden" because the state of the model, q, is not observed; rather the output, $\mathbf{x}$, of a stochastic process attached to that state is observed. This is described by a probability distribution $p(\mathbf{x}|q)$. The other set of pertinent probabilities are the state transition probabilities, $P(q_i|q_j)$.

require is $P(Q_1^N|\mathbf{X})$. It is not obvious how to estimate this probability directly (but see section IV); however we may re-express this probability using Bayes' rule:

$$P(Q_1^N|\mathbf{X}) = \frac{p(\mathbf{X}|Q_1^N)P(Q_1^N)}{p(\mathbf{X})} \quad . \tag{1}$$

This separates the probability estimation process into two parts: *acoustic modelling*, in which the data dependent probability density $p(\mathbf{X}|Q_1^N)$ / $p(\mathbf{X})$ are estimated; and *language modelling* in which the prior probabilities of state sequences, $P(Q_1^N)$, are estimated. Thus, using the HMM assumptions, we are able to treat acoustic modelling and language modelling independently, using the data dependent and prior probability estimates.

ACOUSTIC DATA MODELLING

DENSITY ESTIMATION

If we treat the probability of the data, $p(\mathbf{X})$, as a normalising constant across models, then we can estimate the desired posterior $P(Q_1^N|\mathbf{X})$ using the joint density of the data and the state sequence,

$$P(Q_1^N|\mathbf{X}) \propto P(Q_1^N, \mathbf{X}) \quad . \tag{2}$$

Using a language model to specify $P(Q_1^N)$, the acoustic modelling problem then becomes the estimation of the likelihood of the acoustic data given the state sequence, $p(\mathbf{X}|Q_1^N)$.

The usual HMM training approach is to construct a density estimator using a maximum likelihood training criterion (e.g. Bahl et al. (1983)). Given a training state sequence, the parameters of the stochastic process on each state are adjusted to maximise the likelihood of that state sequence producing the observed acoustic data. Since we are using a hierarchical modelling approach, different state sequences share states, thus constraining the optimisation process. A powerful, provably convergent (to a local minimum) algorithm for this optimisation is the forward-backward algorithm (Baum et al., 1970; Liporace, 1982).

Assumptions. In the course of training an acoustic model, various assumptions are made:

- Piecewise stationarity—we can model speech using a Markov chain;
- The prior probability of a model can be separately estimated—a language model may be derived without reference to the acoustic data;
- Observation independence—the current data vector is conditionally independent of previously emitted data vectors;
- First order Markov process—the state of the process, $q(t)$, depends only on the previous state, $q(t-1)$;
- State emission—the current data vector is dependent only on the current state of the process.

At any given time, we wish to compute the joint density of a model emitting a data vector $\mathbf{x}(t)$ while in state $q(t)$, given the previous state sequence Q_1^{t-1} and acoustic vector sequence $\mathbf{X}_1^{t-1}$, $p(\mathbf{x}(t), q(t)|\mathbf{X}_1^{t-1}, Q_1^{t-1})$. The above assumptions allow us to simplify this probability:

$$p(\mathbf{x}(t), q(t)|\mathbf{X}_1^{t-1}, Q_1^{t-1}) \;=\; p(\mathbf{x}(t)|q(t))\, P(q(t)|q(t-1)) \quad . \tag{3}$$

The probability of a particular state emitting a particular data vector, $p(\mathbf{x}(t)|q(t))$, is referred to as the state *output* or *emission* probability. The other probability, $P(q(t)|q(t-1))$, is referred to as the state *transition* probability.

Thus training an acoustic model by density estimation involves estimating the state transition probabilities, and the probability density function (pdf) from which the state output likelihoods are drawn. A key design decision in acoustic modelling (given a training criterion, in this case maximum likelihood density estimation) is the choice of functional form for the state output pdfs.

Most HMM speech recognition systems use a parametric form of output pdf. In this case a particular functional form is chosen for the set of pdfs to be estimated. Typical choices include Gaussians, mixtures (linear combinations) of Gaussians and binned histograms (for vector quantised data). The parameters of the pdf are then adapted according to the training data, so as to optimally model the data. If we are dealing with a family of models within which the correct model falls, then this is an optimal strategy. However, in the case of modelling speech using HMMs, both the HMM assumptions and the output pdfs used for the HMMs are not good models of speech. In this case, producing the best possible model of each unit of speech, will not necessarily lead to the best speech recognition performance.

An alternative approach is non-parametric density estimation. Although this does not address the problem of the HMM being an incorrect model, it does attempt to use the data to choose the family of output pdfs. In non-parametric density estimation, the family of pdfs under consideration changes as more data is seen. An example is Parzen window estimation (Parzen, 1962), a kernel-based method. In this technique, as new data points occur, new kernels corresponding to those data points are added to the estimator. This has been used in HMM speech recognition by Soudoplatoff (1986).

DISCRIMINATIVE TRAINING

Ultimately in speech recognition, we are not concerned with estimating the joint density of the speech data and word sequence; we are interested in the posterior probability of a word sequence, given the acoustic data. More informally, we are not finally concerned with modelling the speech signal, but with correctly choosing the sequence of words that was uttered. At a local level, rather than constructing the set of pdfs that best describe the data (within the constrained family of functions being optimised), we are interested in ensuring that the correct HMM state is the most probable (according to the model) for each frame.

This leads us to a discriminative training criterion. Discriminative training attempts to model the class boundaries—learn the distinctions between classes—rather than construct as good a model as possible for each class. In practice this results in an algorithm that minimises the likelihood of incorrect, competing models and maximises the likelihood of the correct model. This differs from density estimation maximum likelihood training, in which each data point is used to update the density model of the class from which it has been assigned.[6] Thus, in discriminative training, the parameters of the correct class will be adjusted towards a data point (as in density estimation), but those of the incorrect classes are moved away from the data point.

6. In the case of "soft" density estimation, each data point is shared out amongst classes (depending on the likelihood of generation by each class), so several class densities are updated. But there is no sense of discrimination.

There are many discriminative pattern recognition methods in the literature including the method of Potential Functions (Aizerman et al., 1964), learning vector quantisation (LVQ) (Kohonen et al., 1988) and the multi-layer perceptron (MLP) (see e.g. Hertz et al. (1991)). Unlike the other methods, the MLP may be used directly to compute class-conditional posterior probabilities (see section III).

THE VITERBI ALGORITHM

So far we have discussed the models we use, and how their parameters are estimated. In this section we outline how trained HMMs are used to recognise speech.

HMMs are generative stochastic models of speech. To recognise speech we take an input speech signal, and compute the most probable sequence of models to have generated the speech signal. An efficient algorithm for computing this state sequence is a dynamic programming algorithm known as Viterbi decoding. The Viterbi algorithm essentially traces the minimum cost (or maximum probability) path through a time-state lattice (Ney, 1984) subject to the constraints imposed by the acoustic and language models.

The Viterbi algorithm may also be used in training. In this case a Viterbi alignment is performed for a known word model sequence to obtain the optimal state segmentation. Given this optimal segmentation the output pdf parameters (e.g. means and variances of Gaussians, weights of a MLP, etc.) may be re-estimated. In the case of Gaussian pdfs, this process is known as the *segmental k-means* algorithm (Rabiner et al., 1989).

PRIOR PROBABILITIES

The combination of phone models to form word models is constrained by a phone-structured lexicon that details the allowed pronunciations for each word. Likewise the construction of sentences from words is constrained by a language model, such as a stochastic grammar or (more simply) a wordpair grammar that lists all allowable pairs of words. In a statistical speech recognition system, the language model will thus assign a prior probability to all allowable sentences in the language. Since sentences are composed of words, a prior probability is specified for each sentence. Using the allowable pronunciations for each word (which may be probabilistic), prior probabilities are also specified for each phone (and for each state of each phone model). So the specification of the language model, phone-structured lexicon and basic phone HMMs, sets the prior probabilities for HMM states, phones, words and sentences.

These prior probabilities are encoded in the topology and associated transition probabilities of the hidden Markov word and sentence models. It will be important later to distinguish these prior probability estimates from the prior probability estimates from the phone relative frequencies obtained from the training data. We generally do not wish to use the latter since a typical speech training database is much smaller than a typical textual corpus from which the language model is derived.

Part III

MLP PROBABILITY ESTIMATION

MULTI-LAYER PERCEPTRONS

A multi-layer perceptron (MLP), $\mathcal{F}$, is a layered feed-forward network. In the simplest case, considered in this work, there is an input (data) layer, a hidden layer and an output layer. In this case the network may be described as a function of the data $\mathbf{X}$:

$$\mathcal{F} = \mathbf{g}(\mathbf{W}^{OH} f(\mathbf{W}^{HI}\mathbf{X})). \tag{4}$$

Where $\mathbf{W}^{OH}$ is a weight matrix between the hidden and output layers, and $\mathbf{W}^{HI}$ a weight matrix between input and hidden layers. f and $\mathbf{g}$ are vectors of transfer functions for the hidden and output layers, typically sigmoidal in character.

POSTERIOR PROBABILITY ESTIMATION

MLPs may be used to estimate probabilities. As originally shown by Bourlard and Wellekens (1989), and expanded by Gish (1990) and Richard and Lippmann (1991), a MLP trained to perform classification is a class-conditional posterior probability estimator. That is, after a '1-from-N' training, where there is a one-to-one correspondence between output units and classes, a MLP output value, given an input $\mathbf{x}$, will be an estimate of the posterior probability of the corresponding class q_i given the input, $P(q_i|\mathbf{x})$. This result holds for training with various error functions (including relative entropy or sum squared). The output units should be constrained to be non-negative and less than 1 (e.g. using a sigmoid transfer function).

Note that a sigmoid transfer function does not constrain the sum (over all classes) of class-conditional posterior probabilities to equal 1. However, our computational experiments have shown that the sum of estimated posteriors is usually extremely close to 1, for test vectors drawn from a region of space well-sampled for training. However in some applications (e.g. when combining or comparing the estimates from different networks) it is desirable to enforce a 'sum-to-1' constraint. One way of achieving this is by adopting a normalising output transfer function such as the normalised exponential or 'softmax' (Bridle, 1990b).

In estimating posteriors using a MLP, we are using a discriminative training criterion and not performing density estimation. Assuming equal class priors, for simplicity, we have the relation:

$$P(q_i|\mathbf{x}) \propto \frac{p(\mathbf{x}|q_i)}{p(\mathbf{x})} \quad . \tag{5}$$

Thus directly estimating the posterior tells us how probable it is that a particular vector belongs to a particular class, but gives us no information on how likely it is to observe that vector in the first place. The full joint density estimation, on the other hand, tells us both how likely we are to observe a particular vector, as well as its class membership probabilities. This is illustrated in figure 2. So although, a MLP does not provide a full probability model, if we are interested in discriminating between classes it may provide a "better" use of parameters.

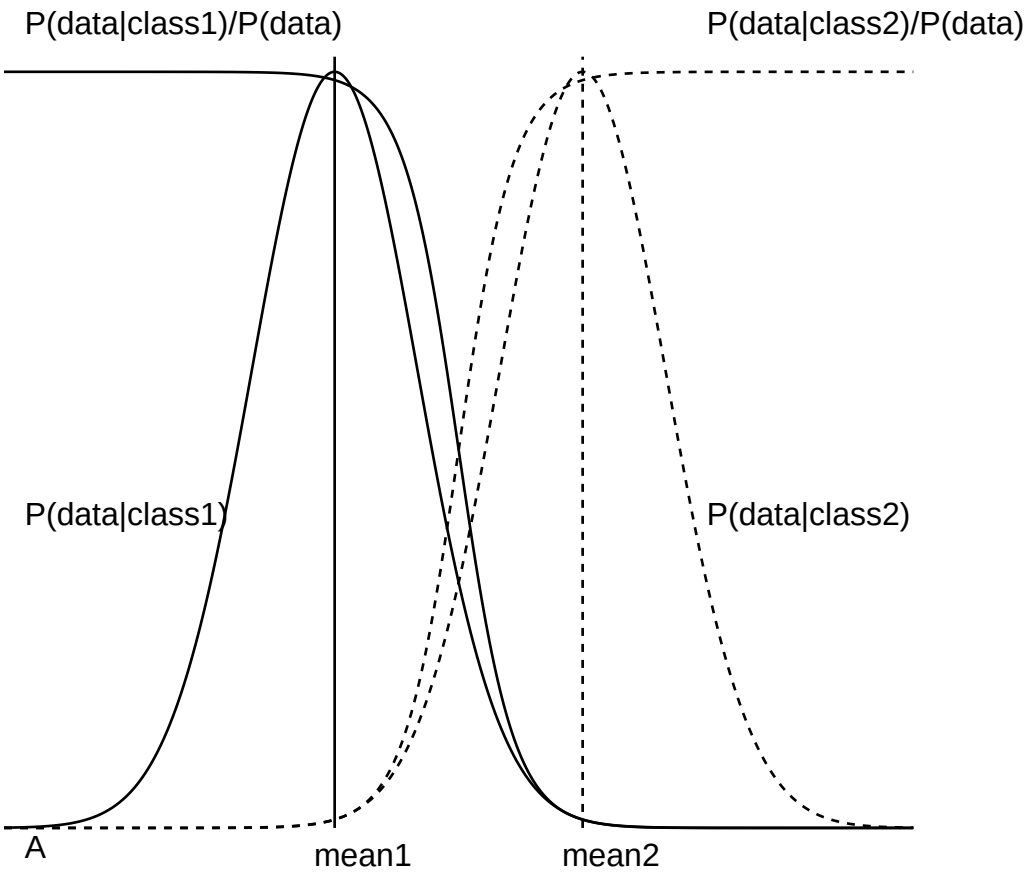

Figure 2: Posterior and likelihood estimation. In this example we have 2 classes, described by Gaussians, with equal priors. Consider point A. If we use Gaussian likelihood estimators, then we can tell that there is a low probability that A was generated by either class 1 or class 2. However, comparing these small probabilities, we can tell that it is much more likely that A was generated by class 1 compared with class 2. A direct posterior estimator, on the other hand, tells us whether it is more probable that A is from class 1 or class 2, but does not tell us how likely it is to observe A in the first place. The posterior probability estimate tells us that given a piece of data was observed, how likely it is to be a member of a particular class.

The posterior probability estimation theorem holds only for a global minimum of the error function. In practice, this is not attainable: indeed, using a cross-validation training schedule a local minimum is not reached, as training is stopped early to avoid overfitting (see section V). However, empirical results indicate that good posterior probability estimates are still achieved.

OBTAINING LIKELIHOOD ESTIMATES

Applying a trained MLP to new data gives us the posterior probabilities of each class, given that data. These probabilities depend on the relative class frequencies, which may be regarded as the data-estimated priors. However, as discussed earlier, we want to use the language model estimated priors at recognition time. Thus we need to factor out the relative class frequencies, before using these probabilities at recognition time. We use (scaled) likelihoods rather than MLP posterior probability estimates at recognition time.[7]

It is easy to convert posteriors to scaled likelihoods using (5), but with non-equal priors:

$$\frac{p(\mathbf{x}|q_i)}{p(\mathbf{x})} = \frac{P(q_i|\mathbf{x})}{P(q_i)} \quad . \tag{6}$$

Thus dividing each MLP output by its relevant frequency results in a scaled likelihood estimate, suitable for use in at recognition time.

HMM PROBABILITY ESTIMATION

The above sections have shown a MLP can be used to estimate HMM output probabilities. Estimating probabilities in this way enables us to make weaker assumptions than standard HMM systems.

- Although a MLP is a parametric model, a large network defines an extremely flexible set of functions. In this manner we do not make strong assumptions about the input statistics. Hence, multiple sources of evidence may be combined as the input to a MLP. For example a single MLP may be trained using input data that mixes samples drawn from several distributions, discrete or continuous.
- By training discriminatively, and not constructing a complete model of the joint density, we are making weaker assumptions about the functional form of this density.
- Maximum likelihood estimation of HMM parameters requires the assumption of conditional independence of observations. MLPs can model correlations across an input window of adjacent frames.

A further benefit of using MLPs comes from the regularity of the resulting recognition time computations, enabling an efficient implementation using parallel hardware.

7. This substitution is not forced upon us; there is no technical reason why a Viterbi search cannot be carried out using posterior probabilities (Bourlard & Wellekens, 1990).

PRIORS AND BIASES

If we use a softmax output transfer function then:

$$P(q_i|\mathbf{x}) = \frac{\exp(\sum_j w_{ij}\text{hid}_j + \text{bias}_i)}{\sum_k \exp(\sum_j w_{kj}\text{hid}_j + \text{bias}_k)} \tag{7}$$

$$\propto \exp(\sum_j w_{ij}\text{hid}_j + \text{bias}_i) \tag{8}$$

$$\propto \exp(\log(p(\mathbf{x}|q_i)))\exp(\log(P(q_i)))\ , \tag{9}$$

where hid_j is the output of hidden unit j, w_{ij} is the weight from hidden unit j to output unit i and bias_i is the bias value for output unit i. It is tempting to identify the weighted sum of hidden unit outputs as the data part (and so the log likelihood, $\log(p(\mathbf{x}|q_i))$) and the bias as the prior part (the log prior $\log(p(\mathbf{x}|q_i))$) of each output unit. Observation of the output biases of a trained network does indeed show a correlation with the log priors (relative frequencies). For example, the output biases are uniformly negative.

However this relationship is too facile. Hervé Bourlard pointed out that the case is not so simple, using the example of a multivariate Gaussian classifier. As is well known we can write down the corresponding linear discriminant function (see e.g. Duda and Hart (1973)) for a Gaussian classifier with equal covariances:

$$g_i(\mathbf{x}) = \mathbf{w}_i^T\mathbf{x} + w_{i0}\ . \tag{10}$$

Where $\mathbf{w}$ is a weight matrix expressible in terms of the mean (μ_i) and covariance (Σ_i) of class i and w_{i0} is the bias for class i. In this case we have:

$$\mathbf{w}_i = \Sigma_i^{-1}\mu_i \tag{11}$$

and

$$w_{i0} = -\mu_i^T\Sigma_i^{-1}\mu_i + \log(P(q_i))\ . \tag{12}$$

Here the bias is influenced by the class mean and covariance (a data term) as well as the log prior term.

So, it seems likely that the biases of a trained MLP will also contain a data component, rather than just encoding prior information. One way we attempted to minimise the data component was to replace the usual sigmoid hidden unit transfer function ($1 / 1 + \exp(-x)$) with a $\tanh(x)$ function. This has a $[-1, +1]$ range (rather than $[0, 1]$); therefore in the case of random input, the expected output would be 0. Hence, it might be hoped that the data portion of the biases would be minimal. Of course, hidden unit outputs resulting from speech input are not random, and as reported in section VI, the network biases were not good replacements for the priors at recognition time. However, this postulated relationship was used to speed training time and improve generalisation (section VI).

Part IV
DISCRIMINATIVE HMMs

As discussed earlier, traditional HMMs incorporate a complete probabilistic model of the joint density, $p(\mathbf{X}, Q_1^N)$. This is generally estimated using a prior language model $P(Q_1^N)$ and an acoustic model $p(\mathbf{X}|Q_1^N)$, optimised by a maximum likelihood process. This joint density is assumed proportional to the posterior, $P(Q_1^N|\mathbf{X})$: the normalising denominator $p(\mathbf{X})$ is ignored.

All these probabilities should be conditioned on the model parameters, Θ. Thus we should express the probability of the data as $p(\mathbf{X}|\Theta)$. At recognition time this is constant across all models. At training time the parameters of the models are being adapted by the training algorithm: thus $p(\mathbf{X}|\Theta)$ is not constant across models. We may rewrite this probability as:

$$p(\mathbf{X}|\Theta) = \sum_{Q_1^N} p(\mathbf{X}|Q_1^N, \Theta) P(Q_1^N|\Theta) \tag{13}$$

$$= p(\mathbf{X}|C_1^N, \Theta) P(C_1^N|\Theta) + \sum_{I_1^N} p(\mathbf{X}|I_1^N, \Theta) P(I_1^N|\Theta) \ , \tag{14}$$

where C_1^N represents the correct state sequence and the sum over I_1^N represents a sum over all incorrect state sequences.

Thus if we take our acoustic model as $p(\mathbf{X}|Q_1^N, \Theta)/p(\mathbf{X}|\Theta)$, rather than $p(\mathbf{X}|Q_1^N, \Theta)$, we must employ a different training criterion. Rather than using maximum likelihood estimation to train each model, we must maximise the likelihood of the correct state sequence, whilst simultaneously reducing the likelihood of competing, incorrect state sequences. This corresponds to discriminative training, and we refer to an HMM trained in this fashion as a *discriminative* HMM.

FRAME-LEVEL DISCRIMINATION

In this paper, we deal mainly with discriminative training of HMMs at the frame level, rather than the state sequence level. In practice this means we are concerned with the estimation of the state output probabilities $p(\mathbf{x}|q_i)$. As discussed earlier these are obtained using a MLP trained for framewise phone classification, which estimates the posterior $P(q_i|\mathbf{x})$. After dividing by the relative frequencies (acoustic model phone priors) and invoking Bayes' rule we have a discriminative acoustic model at the frame level, an estimate of $p(\mathbf{x}|q_i) \ / \ p(\mathbf{x})$.

In practice, rather than using a single frame of acoustic input, we use $2c + 1$ frames, giving c frames of left and right context. In this case the MLP estimates the not-so-local probability $P(q_i|\mathbf{x}(t-c), \ldots, \mathbf{x}(t), \ldots, \mathbf{x}(t+c))$.

In this approach the transition probabilities are not estimated discriminatively. The maximum likelihood estimate may be used, or some duration constraint may be encoded using model states and constant transition probabilities.[8] An alternative approach was the discriminative HMM, originally

8. In practice, transition probabilities have a much smaller dynamic range than output probabilities, and are so less important. Hence good speech recognisers can be built using these ad hoc transition probability settings, provided more principled methods are used for the output probabilities.

defined by Bourlard and Wellekens (1990). Here the probabilities estimated by the network are the local posteriors $P(q(t)|q(t-1), \mathbf{x}(t))$. This posterior combines the modelling of the output probabilities (now transition-specific, rather than state-specific) and the transition probabilities. These probabilities may be estimated by a MLP, with the addition of binary input units, representing the previous state. At recognition time, all possible previous states must be cycled through in order to compute all relevant local posteriors. Significant experiments have not yet been performed using this scheme.

GLOBAL DISCRIMINATION

There have been two basic approaches suggested for the optimisation of a discriminative HMM at a state sequence (or global) level.

Bahl et al. (1986) presented a training scheme for continuous HMMs in which the mutual information between the acoustic evidence and the word sequence was maximised using gradient descent. This is a discriminative objective function, maximised by gradient descent. More recently, Bridle introduced the "alphanet" representation (Bridle, 1990a) of HMMs, in which the computation of the HMM "forward" probabilities $\alpha_{jt} = P(\mathbf{X}_1^T, q(t) = j)$ is performed by the forward dynamics of a recurrent network. Alphanets may be discriminatively trained by minimising a relative entropy objective function. This function incorporates the negative log of the global posterior probability of the correct state sequence given the acoustic evidence $P(Q_1^N|\mathbf{X}, \Theta)$, rather than the local posterior of a state given one frame of acoustic evidence. This posterior is the ratio of the likelihood of the correct model to the sum of the likelihoods of all models. For continuous speech, a model refers to a sentence model. The numerator of this ratio is the quantity computed by the forward-backward algorithm in training mode (when the word sequence is constrained to be the correct word sequence, so only time-warping variations are considered). The denominator involves a sum over all possible models: this is equivalent to the sum computed if the forward-backward algorithm were to be run at recognition time (with the only constraints over the word sequence provided by the language model). Computation of this quantity would be prohibitive for both training and recognition. A simpler quantity to compute is just the sum over all possible phoneme sequences (unconstrained by language model). This is not desirable as it assumes uniform priors rather than those specified by the language model.

Initial work in using global optimisation methods for continuous speech recognition has been performed by Bridle and Dodd (1991), Niles (1991) and Bengio et al. (1992). Bridle and Bengio used this approach to optimise the input parameters via some (linear or non-linear) transform, training the parameters of the HMM by a maximum likelihood process.

A second approach to global optimisation of discriminative HMMs is analagous to segmental k-means training and has been referred to as *embedded training* (Bourlard & Morgan, 1990) or connectionist Viterbi training (Franzini, Lee, & Waibel, 1990). In this method a frame level optimization is interleaved with a Viterbi re-alignment. It should be noted that the transition probabilities are still optimised by a maximum likelihood criterion (or the Viterbi approximation to it). It may be proved that performing a Viterbi segmentation using posterior local probabilities will also result in a global optimisation (Bourlard & Wellekens, 1990): however, there is a mismatch between model and acoustic data priors, as discussed earlier.

Part V

A CONNECTIONIST–HMM RECOGNITION SYSTEM

The previously described components may be put together to form a hybrid connectionist–HMM continuous speech recognition system (figure 3).

The first stage in a speech recognition system is known as the *front end*. This consists of sampling the time-amplitude waveform, followed by an analysis process designed to give a concise, non-redundant representation of the speech signal. This is usually a frame based analysis in which a window of speech (typically 20 ms wide) is analysed by some kind of spectral or linear predictive analysis, the window being advanced at discrete intervals (typically 10 ms). The resulting speech signal is then characterised by a series of feature vectors at 10 ms intervals. This method of analysis embeds piecewise stationarity, since it assumes that the speech signal may be represented by a sequence of discrete "snapshots".

In this paper we are mainly concerned with building statistical models of the speech signal in the feature vector domain. We use a set of basic HMMs, corresponding to phones. These are concatenated or built into networks, to form words and sentences, according to the lexicon and language model.

When training a traditional HMM system, the topologies and output pdfs of the HMMs are chosen and initialised, then their parameters are estimated. In our connectionist HMM system, we follow the same basic approach. We choose the topologies of the HMMs and we choose a MLP (with a given architecture) to be our output pdf estimator. In the case where we have a single pdf per phone, the MLP may be viewed as being trained to perform phonetic discrimination. We initialise the models and perform a Viterbi alignment (using this bootstrap recogniser), producing a time-aligned state segmentation (subject to the language model). From the state segmentation we can, of course, obtain phone and word segmentations. This segmentation is used to produce the training targets for the MLP. Thus the MLP targets implicitly contain information about the model. The MLP is then trained using the back-propagation algorithm. Here, the temporal and static parts of the training problem are separated, in contrast to the forward-backward algorithm. The process maybe iterated, alternating between training the MLP and re-estimating the transition probabilities, an embedded training process.

The transition probabilities are not trained discriminatively, but are estimated using a maximum likelihood method. This mismatch between discriminative output pdfs and maximum likelihood transition probabilities is a weakness of our method, but does not seem to have a large adverse effect on performance, since the system is much more dependent on the output pdfs. In practice, the transition probabilities are mainly used to encode average duration.

The usual time-synchronous Viterbi decoding algorithm is used for recognition. For each frame of speech, a vector of HMM state conditional posterior probabilities is produced by the MLP. These are converted to scaled likelihoods, by dividing by the data priors (relative state frequencies in the training data). In combination with the transition probabilities, this enables us to compute the HMM state sequence most likely to have produced the speech signal, given the lexical and language model constraints. An additional parameter used in the recognition is the word transition probability, which is usually set by hand using a validation set. This parameter penalises (lowers the probability) of

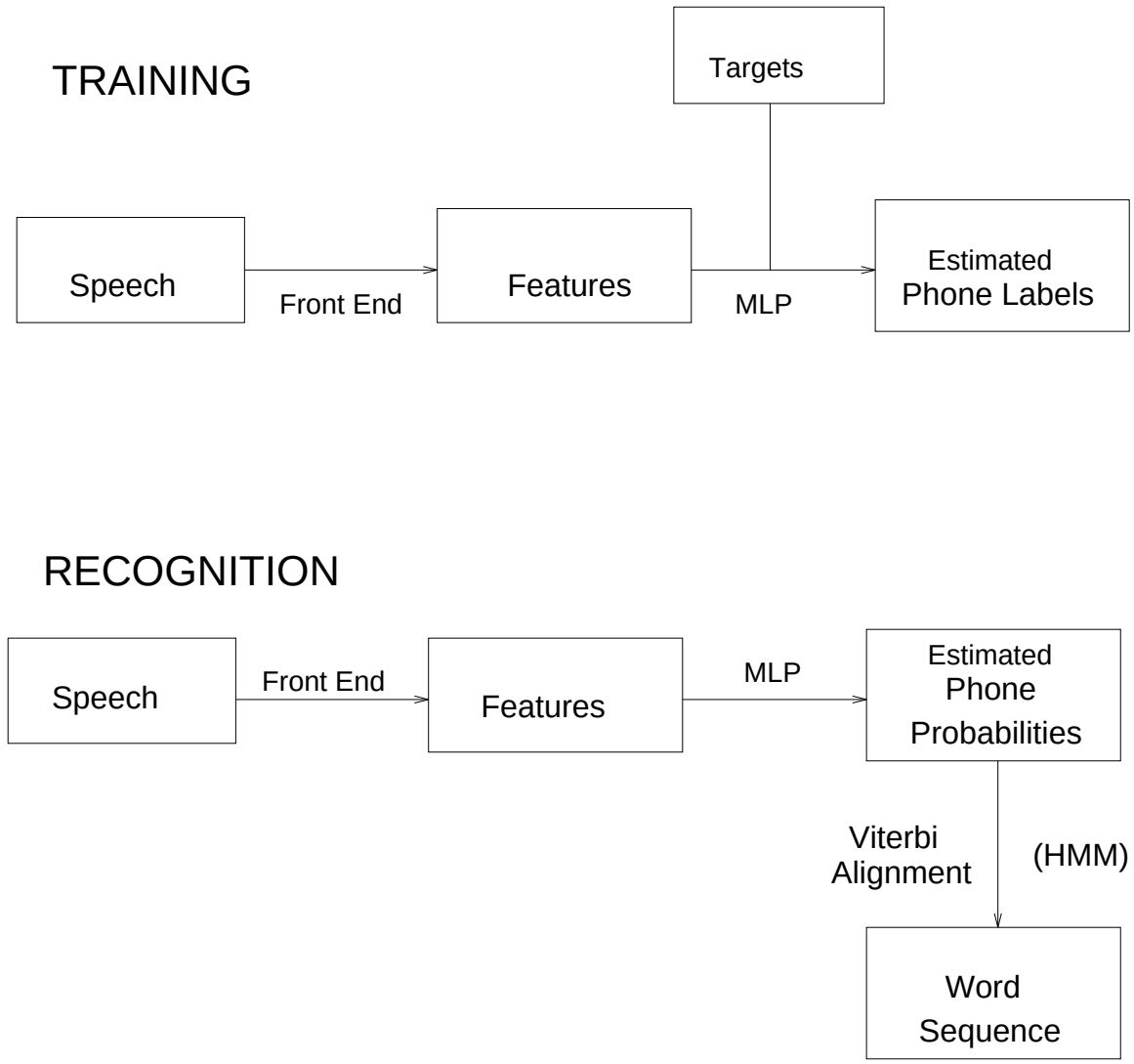

Figure 3: A schematic of the training and recognition processes. At both training and recognition times, the speech is processed by a front end (e.g. a mel cepstral transform) that extracts a concise description of the speech every frame (typically every 10 ms). When training, using alignments produced using a previously trained recogniser (or bootstrapping on time-aligned labelled data, such as the TIMIT database), a MLP is trained to phonetically classify frames of data. At recognition time a trained MLP is used to estimate phone probabilities in a Viterbi dynamic programming search. This search uses the constraints of allowed word pronunciations and the language model to produce the most probable string of words (according to the model).

word transitions, thus reducing the tendency to favour short words over longer ones.[9]

9. This is a result of the observation independence assumption, causing an under-estimate of the joint observation density (Brown, 1987).

Part VI
EXPERIMENTS

METHODS

Most of our experiments have been performed using the DARPA Resource Management (RM) speaker independent continuous speech database, in collaboration with Mike Cohen and Horacio Franco of SRI International. This is a very well studied database with a vocabulary of 991 words. The standard training set contains 3990 sentences from 109 speakers, or about 1.5 million frames of data. Evaluation is performed over various test sets, typically containing 300 sentences from 10–12 new speakers.

The difficulty of a speech recognition task is strongly affected by the perplexity, which is essentially the geometric mean of the number of words that can follow any other word. When no grammar is used, the RM task has a perplexity of 991—any word can follow any other. With a simple word-pair grammar listing allowable pairs of words, the perplexity is reduced to about 60. Note that the perplexity of the deterministic source grammar used to generate sentences for the RM task was about 6.

The front end used in these experiments was a mel cepstral[10] analysis, producing 12 coefficients, plus energy for each 10ms frame of speech. In addition to these 13 coefficients, we estimate their temporal derivatives, giving 26 coefficients per frame.

We chose the basic subword unit to be the phone. Each phone was represented as a two or three state left-to-right HMM. However the output pdfs were tied within each phone model. The pdfs were estimated using a MLP with N outputs corresponding to N phones. This MLP, when trained as a phonetic classifier, output class conditional posterior probabilities.

The networks trained to perform these estimations were typically large; in addition to the current frame (26 inputs), 4 frames of left and right context were typically appended, giving a total of 234 inputs. We usually used networks with about 1000 hidden units, and with 69 output classes, giving a total of over 300,000 weights. Training a network with this many weights is not trivial.[11] We have used a cross-validation training procedure combined with stochastic gradient descent (per-pattern update). Cross-validation training is essential for good generalization and preventing over-training, especially when using large networks. In our training schedule we cross-validate by withholding a certain proportion of the training data (typically 5–10%) and using this to validate the training after each epoch. When the classification performance on the validation set first fails to improve by a certain amount (typically 0.5%) the gradient descent step-size is reduced, typically by a factor of 2. After each succeeding epoch the step size is further reduced, until once again there is no improvement on the validation set. Training is then halted.

---

10. The mel cepstrum is the Fourier transform of the logarithm of a mel spectrum. This spectrum, usually computed using FFTs, is equivalent to an unequally spaced filter bank that approximates the "critical bands" of human hearing. The cepstrum is usually truncated (higher order terms set to zero) to smooth the representation, essentially removing closely spaced variations from the log spectrum.

11. Computation was done using the RAP, a ring array processor, that was configured with from four to twenty TMS320C30 DSPs. This provided matrix-vector operations that were 2 orders of magnitude faster than the Sparc 2 workstations ordinarily used for program development. Training still took 1-2 days.

Following the discussion in part III, we have found empirically that the output biases of a trained network may approximate the log odds of the priors (for a sigmoid transfer function) or the log of the priors (for a softmax transfer function): certainly, they are all negatively valued. We have found that training is speeded (by over 25%) and generalisation improved by initialising the output biases to the log odds (or log) of the relative frequencies of the corresponding classes. Another training improvement was to use random (with replacement) pattern presentation. These two methods together improved the speed of training by a factor of 2 (training time of 10 passes[12] through the training set reduced to 5) and also improved generalisation.

We integrated these probability estimates into SRI's system, DECIPHER (Cohen et al., 1990). DECIPHER is a much richer system than the previous baseline systems we have used. It includes both multiple probabilistic word pronunciations, cross-word phonological models, multiple densities per phone, and context dependent phone modelling.[13] The basic DECIPHER system uses tied Gaussian mixtures to estimate pdfs.

We initially worked with a context-independent form of DECIPHER; this was the complete DECIPHER system, except that the basic models were 69 context-independent phone models, rather than over 3000 context-dependent models. In this work we replaced the tied mixture pdfs by a MLP. We also worked with the full, context-dependent DECIPHER system; in this case the context dependent tied mixture pdfs were augmented by the context independent MLP.

We used DECIPHER to initialise our models. Our initial training targets were obtained by using the tied mixture DECIPHER segmentation, and we retained the estimated transition probabilities.

ARCHITECTURES

Some experiments were performed, using the simple single pronunciation ICSI system, to assess the affect of varying MLP architectures. In particular the effect of varying the hidden unit transfer function from a sigmoid to a hyperbolic tangent was investigated, as was changing the output unit transfer function from a softmax.

The single pronunciation ICSI system uses 69 phone HMMs, with a single pronunciation lexicon (constructed by SRI, and obtainable from the DARPA CD-ROM) using the most likely pronunciations for each word. We use a single MLP pdf for each phone model, shared by all states. The number of states in these left-to-right HMMs is equal to the average duration of the phone i (in frames) divided by 2 (N_i / 2), with all the transition probabilities set equal to 0.5. This gives a minimum duration of N_i / 2 frames and a most likely duration of N_i frames for each phone.

Using the same network as in the DECIPHER experiments above, the basic single pronunciation system returned a word error rate of 11.9% on the February 1991 test set. Changing to a softmax output transfer function lowered the test error to 10.3%.[14] Although the fact that a $\tanh(x)$ hidden unit transfer function is symmetric around 0 adds theoretical appeal, retraining the network with this kind of transfer function resulted in an error of 10.8%.

---

12. Note that when training with random pattern selection, training does not involve a series of passes through the complete training set. We regard a pass as training on P randomly chosen patterns, when there are P patterns in the training set.
13. A *context independent* phone model, is one which models a phone in whatever context it may occur. *Context dependent* phone models, however, model phones in specific phonetic contexts, to take account of the acoustic effect of context. Thus there may be several different models for a single phone, depending on the surrounding context.
14. Varying the word transition probability from 10^{-12} to 10^{-9} resulted in a lower test error of 10.1%.

Following the discussion earlier of priors and biases, an experiment was tried in which the relative frequency estimate of network priors was replaced by the trained output biases, which were hoped to give a smoothed approximation to the log priors in the case of softmax. Note that in the case of random pattern presentation, the relative frequencies are not necessarily the correct acoustic data phone priors for the network. It was hoped that the biases might represent a better estimate. This was implemented, by taking the weighted sum of hidden unit outputs as a log likelihood estimate (no addition of biases, equivalent to addition of log prior estimates) and no application of the softmax at recognition time. However, when tested on a network with 512 tanh units (tanh was chosen following the discussion in part III), the word recognition error was increased from 11.9% to 16.4%.

RESULTS

Our principal experiments were performed using a network containing 234 inputs, 1000 sigmoid (0,1) hidden units and 69 softmax outputs. Random (without replacement) pattern presentation was used in the stochastic gradient descent training and the output biases were initialised to be the log of the class relative frequencies. This MLP probability estimator was used inside the full DECIPHER system. Training a network of around 300,000 weights required around 5 passes through the training database of 1.3 million training patterns (200,000 patterns were held out for cross-validation). These networks used sigmoidal transfer functions for hidden and output units.

To train a MLP we require a bootstrap model to produce time-aligned phonetic labels. In this case we used a context-independent version of the DECIPHER system to perform the forced alignment between the training data and word sequence.

The baseline DECIPHER system modelled the output distributions using tied Gaussian mixtures. Training used the forward-backward algorithm to optimize a maximum likelihood criterion.

We used two sets of test sentences for evaluation. A 300 sentence development set (the June 1988 RM speaker independent test set), separate from the network training cross-validation set, was used to tune the HMM recognition parameters, such as the word transition penalty. The results reported here were obtained from a 300 sentence evaluation test set (the February 1991 RM speaker independent test sets); no tuning of parameters was performed using this set.

A context independent form of the DECIPHER system was used as a baseline. It was trained using the maximum likelihood forward-backward procedure, and gave a word error of 11.0%, using the RM word-pair grammar (perplexity 60). When the usual HMM output probability estimators were replaced with a MLP trained to classify its input acoustic vectors into 1 of 69 classes (and outputting posterior probability estimates) the word recognition error improved to 6.2%. In a later experiment, we further reduced the error to about 5% by re-aligning the data using the MLP for probability estimation and then retraining the MLP with the new alignment.

In a related experiment, the MLP probability estimates were smoothed together with probabilities from the full context-dependent DECIPHER tied-mixture system. This reduced the error from 4.0% to 3.3%, a results that was slightly better than the best systems that had previously been evaluated on this data set. These results are summarised in figure 4. The relationship between the number of parameters and recognition performance is graphed in figure 5.

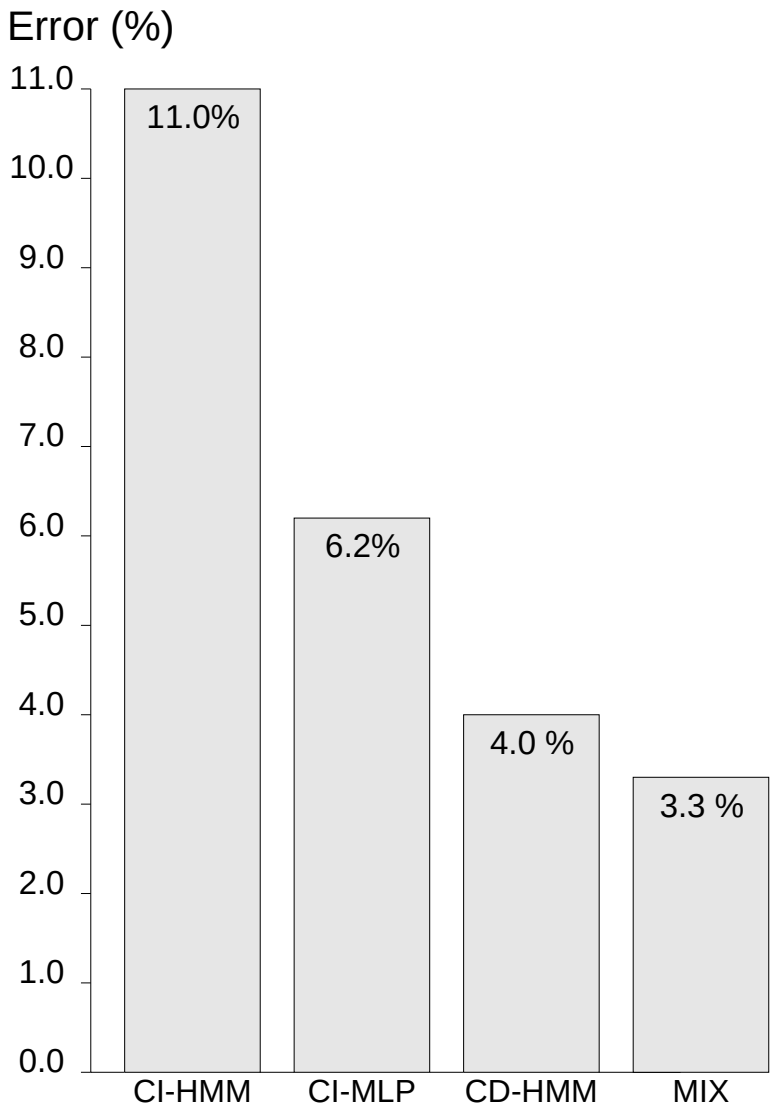

Figure 4: Results using the February 1991 test set, using the perplexity 60 wordpair grammar. The CI-HMM system is the baseline, context independent DECIPHER system. The CI-MLP is the analgous hybrid system in which the tied mixture output probability estimator is replaced by a MLP, estimating probabilities for the same models. The CD-HMM is the full context-dependent DECIPHER system. The MIX system is a simple interpolation between the CD-HMM and the CI-MLP. The previous best result on this test set was 3.6% reported by CMU.

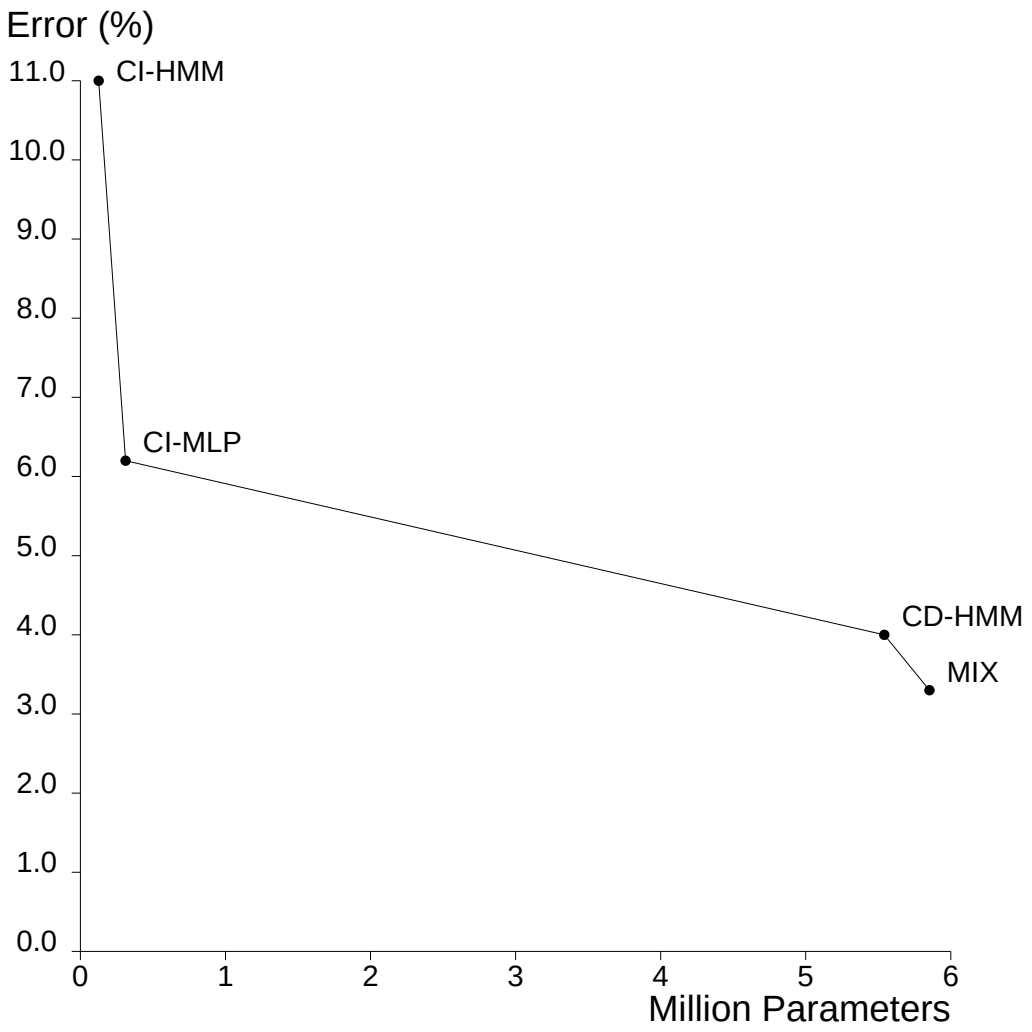

Figure 5: For the same systems as the previous figure, a plot of recognition error, versus number of parameters. Note that a MLP with the same number of parameters as the CI-HMM (500 hidden units) achieves about 8.0% recognition error.

Part VII
CONCLUSION

In this report we have reviewed the basis of statistical (HMM) speech recognition methods, paying attention to the assumptions embedded in standard techniques. In particular, we have considered density estimation methods compared with discriminative methods. Using the result that feed-forward networks may discriminatively estimate probabilities, we have constructed a connectionist HMM speech recognition system. Experiments on the DARPA speaker independent Resource Management task have demonstrated that these connectionist methods improved a state of the art HMM speech recognition system. These results arise from weakening two underlying HMM assumptions:

- Model correctness—By estimating only the class-conditional posterior probability, we have not attempted to optimise an inappropriate model of the joint density. This discriminative approach seems to be a better use of parameters for a recognition task.
- Observation independence—the acoustic context was increased by presenting a multi-frame input to the network. Thus the probabilities estimated were conditioned on a sequence of data vectors, rather than a single data vector, weakening the observation independence assumption.

Furthermore, we are finding the connectionist HMM framework a good one in which to explore issues such as speaker adaptation, consistency modelling and context-dependent phone modelling.

ACKNOWLEDGEMENTS

Much of this work arises from an ongoing collaboration with Hervé Bourlard. The experiments reported in part VI were carried out in collaboration with Mike Cohen and Horacio Franco of SRI International. This work benefitted from the input of Phil Kohn, Yochai Konig and Chuck Wooters. Thanks to Mike Hochberg and Tony Robinson for a critical reading of an earlier draft of this work.

REFERENCES

Aizerman, M. A., Braverman, E. M., & Rozonoer, L. I. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, *25*, 821–837.

Austin, S., Zavaliagkos, G., Makhoul, J., & Schwartz, R. (1992). Speech recognition using segemental neural nets. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 625–628 San Francisco.

Bahl, L. R., Brown, P. F., de Souza, P. V., & Mercer, R. L. (1986). Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 49–52 Tokyo.

Bahl, L. R., Jelinek, F., & Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-5*, 179–190.

Baum, L. E., Petrie, T., Soules, G., & Weiss, N. (1970). A maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, *41*, 164–171.

Bengio, Y., de Mori, R., Flammia, G., & Kompe, R. (1992). Global optimization of a neural network–hidden Markov model hybrid. *IEEE Transactions on Neural Networks*, *3*, 252–259.

Bourlard, H., & Morgan, N. (1990). A continuous speech recognition system embedding MLP into HMM. In Touretzky, D. S. (Ed.), *Advances in Neural Information Processing Systems*, Vol. 2, pp. 413–416. Morgan Kaufmann, San Mateo CA.

Bourlard, H., & Wellekens, C. J. (1989). Links between Markov models and multi-layer perceptrons. In Touretzky, D. S. (Ed.), *Advances in Neural Information Processing Systems*, Vol. 1, pp. 502–510. Morgan Kaufmann, San Mateo CA.

Bourlard, H., & Wellekens, C. J. (1990). Links between Markov models and multilayer perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-12*, 1167–1178.

Bridle, J. S. (1990a). Alpha-nets: a recurrent neural network architecture with a hidden Markov model interpretation. *Speech Communication*, *9*, 83–92.

Bridle, J. S. (1990b). Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In Touretzky, D. S. (Ed.), *Advances in Neural Information Processing Systems*, Vol. 2, pp. 211–217. Morgan Kaufmann, San Mateo CA.

Bridle, J. S., & Dodd, L. (1991). An Alphanet approach to optimising input transformations for continuous speech recognition. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 277–280 Toronto.

Brown, P. F. (1987). *The Acoustic-Modelling Problem in Automatic Speech Recognition*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Cohen, M., Murveit, H., Bernstein, J., Price, P., & Weintraub, M. (1990). The DECIPHER speech recognition system. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 77–80 Albuquerque.

Duda, R. O., & Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. Wiley Interscience, New York.

Fanty, M., Cole, R. A., & Roginski, K. (1992). English alphabet recognition with telephone speech. In Moody, J. E., Hanson, S. J., & Lippmann, R. P. (Eds.), *Advances in Neural Information Processing Systems*, Vol. 2, pp. 199–206. Morgan Kaufmann, San Mateo CA.

Franzini, M. A., Lee, K. F., & Waibel, A. (1990). Connectionist Viterbi training: a new hybrid method for continuous speech recognition. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 425–428 Albuquerque.

Gish, H. (1990). A probabilistic approach to the understanding and training of neural network classifiers. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1361–1364 Albuquerque.

Hertz, J., Krogh, A., & Palmer, R. (1991). *Introduction to the theory of neural computation.* Addison-Wesley, Redwood City, CA.

Huang, W., Lippmann, R., & Gold, B. (1988). A neural net approach to speech recognition. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 99–102 New York.

Kohonen, T., Brna, G., & Chrisley, R. (1988). Statistical pattern recognition with neural networks: benchmarking studies. In *Proceedings Second IEEE International Conference on Neural Networks*, Vol. 1, pp. 61–68 San Diego.

Lee, K. F. (1988). *Large Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System.* Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Liporace, L. A. (1982). Maximum likelihood estimation for multivariate observations of Markov sources. *IEEE Transactions on Information Theory*, *IT-28*, 729–734.

Makino, S., Kawabata, T., & Kido, K. (1983). Recognition of consonants based on the perceptron model. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 738–741 Boston MA.

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*, 263–271.

Niles, L. T. (1991). Timit phoneme recognition using an HMM-derived recurrent neural network. In *Proceedings European Conference on Speech Communication and Technology*, pp. 559–562 Genova, Italy.

Parzen, E. (1962). On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, *33*, 1065–1076.

Rabiner, L. R., Wilpon, J. G., & Soong, F. K. (1989). High performance connected digit recognition using hidden Markov models. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *37*, 1214–1225.

Renals, S., Morgan, N., Cohen, M., & Franco, H. (1992). Connectionist probability estimation in the DECIPHER speech recognition system. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 601–604 San Francisco.

Richard, M. D., & Lippmann, R. P. (1991). Neural network classifiers estimate Bayesian a posteriori probabilities. *Neural Computation*, *3*, 461–483.

Robinson, T. (1992). Recurrent nets for phone probability estimation. In *Proceedings DARPA Workshop on Continuous Speech Recognition*, p. In press.

Soudoplatoff, S. (1986). Markov modelling of continuous parameters in speech recognition. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 45–48 Tokyo.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., & Lang, K. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech and Signal Processing*, *37*, 328–339.