Connectionist Probability Estimators in HMM Speech Recognition

Steve Renals, Nelson Morgan, Hervé Bourlard, Michael Cohen and Horacio Franco

***Abstract*—We are concerned with integrating connectionist networks into a hidden Markov model (HMM) speech recognition system. This is achieved through a statistical interpretation of connectionist networks as probability estimators. We review the basis of HMM speech recognition and point out the possible benefits of incorporating connectionist networks. Issues necessary to the construction of a connectionist HMM recognition system are discussed, including choice of connectionist probability estimator. We describe the performance of such a system, using a multi-layer perceptron probability estimator, evaluated on the speaker-independent DARPA Resource Management database. In conclusion, we show that a connectionist component improves a state-of-the-art HMM system.**

I. Introduction

Over the past few years, connectionist models have been widely proposed as a potentially powerful approach to speech recognition (e.g., [1,2,3]). However, while connectionist methods have performed well in discrete utterance recognition addressed as a static pattern recognition problem (e.g., [4]), 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 on hidden Markov models (HMMs) (e.g., [5,6,7]). Within this statistical framework, connectionist methods have been used to improve continuous speech recognition systems [8,9,10]. Such improvements have resulted from an integration of the connectionist and statistical components, based upon a statistical interpretation of the computations being performed by connectionist networks.

This paper discusses such a connectionist–statistical speech recognition system, developed at the International Computer Science Institute (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 connectionist networks enables the construction of a hybrid connectionist–HMM system. We shall discuss the performance of this system evaluated on the DARPA Resource Management database, a 991 word speaker-independent continuous speech recognition task [11].

II. Statistical Speech Recognition

A. Hidden Markov Models

Hidden Markov modeling of speech assumes that speech is a *piecewise stationary* process. That is, an utterance is modeled 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 stochastic processes: a Markov process modeling the temporal structure of speech; and a set of state output processes modeling the stationary character of the speech signal. Note that a wider class of models, hidden semi-Markov models, includes a third stochastic process modeling state duration [12]. We shall not deal with models of this class, concerning ourselves only with time-synchronous HMMs.

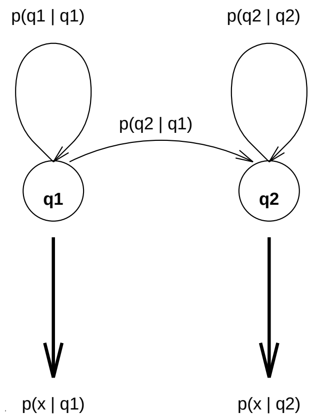

Figure 1: A schematic of a two state, left-to-right hidden Markov model (HMM). *A hidden Markov model is a stochastic automaton, consisting of a set of states and corresponding transitions between states. HMMs are "hidden" because the state of the model, q, is not observed; rather the output, $\mathbf{x}$, 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, $P(q_i|q_j)$.*

Ideally, there would be a unique HMM for each allowable sentence in the language being modeled; this is clearly unfeasible for any but the most trivial of languages. A hierarchical modeling scheme is usually adopted. Each sentence is modeled as a sequence of words. The number of total models required is now much smaller—rather than one model per possible sen-

Steve Renals was with the International Computer Science Institute, Berkeley CA 94704, USA; he is now with Cambridge University Engineering Department, Cambridge CB2 1PZ, UK.
Nelson Morgan is with the International Computer Science Institute, Berkeley CA 94704, USA.
Hervé Bourlard is with Lernout & Hauspie Speechproducts, Ieper B-8900, Belgium.
Michael Cohen and Horacio Franco are with SRI International, Menlo Park CA 94025, USA.

[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 *tied* (*i.e.* constrained to be equal), 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.

tence, 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 sub-word units. Although there are good linguistic arguments for choosing units such as syllables or demi-syllables, the unit most commonly used is the phone[2] and this is the unit used here. There are around 60 basic phone HMMs (for English), and from these word and sentence models may be constructed. For any given sentence we may write down the corresponding HMM; each state in that HMM is contributed by a constituent phone HMM.

B. HMM Speech Recognition

The basic problem of speech recognition is to be able to transcribe the sequence of words corresponding to a spoken utterance. A general approach to this problem is to output the most probable sentence given the acoustic data. Thus we choose sentence $\mathbf{S}$ (and, consequently, the associated sequence of HMMs), for which the probability[3] $P(\mathbf{S}|\mathbf{X})$ is a maximum, where $\mathbf{X}$ is a sequence of N acoustic data vectors, $\{\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(t), \ldots, \mathbf{x}(N)\}$.[4] If we use hidden Markov models, then a sentence is represented by a particular sequence of models, $\mathbf{M}$, and the probability we require is $P(\mathbf{M}|\mathbf{X})$.

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

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

This separates the probability estimation process into two parts: *acoustic modeling*, in which the data dependent probability $p(\mathbf{X}|\mathbf{M})/p(\mathbf{X})$ is estimated; and *language modeling* in which the prior probabilities of sentence models, $P(\mathbf{M})$, are estimated. Thus we are able to treat acoustic modeling and language modeling independently, using the data dependent and prior probability estimates.

If we use the criterion referred to as the maximum likelihood criterion, then estimation of the acoustic model reduces to estimating $p(\mathbf{X}|\mathbf{M})$, as $p(\mathbf{X})$ is assumed to be equal across models. Calculation of this probability involves the sum of the probabilities of all possible paths of length N through $\mathbf{M}$. In this case, training may be performed by the Baum-Welch (or forward-backward) algorithm [13,14,15]. Another criterion, usually referred to as the Viterbi criterion, only considers the best path through $\mathbf{M}$, leading to simplifications of the algorithms involved. This criterion also generates, as a by-product of training or recognition, the word (or. possibly, sub-word unit) segmentation. In this case, a sentence is then represented by a particular state sequence, $Q_1^N = \{q(1), \ldots, q(t), \ldots, q(N)\}$, where $q(t)$ represents the particular state (out of the set of possible HMM states) visited at time t, and we estimate $p(\mathbf{X}|Q_1^N)$. Training using the Viterbi criterion is sometimes known as the *segmental k-means* algorithm [16].

Using the maximum likelihood criterion, recognition can be carried out using a best-first search strategy via the stack decoding algorithm [17] or, equivalently, by an A* search [18]. Recognition may be performed using the Viterbi criterion, by computing the state sequence, Q_1^N, that maximizes the posterior $P(Q_1^N|\mathbf{X})$. The Viterbi algorithm essentially traces the minimum cost (or maximum probability) path through a time-state lattice [19] subject to the constraints imposed by the acoustic and language models.

C. Acoustic Data Modeling

Density Estimation. The usual HMM training approach is to construct a density estimator that maximizes the likelihood $P(\mathbf{X}|\mathbf{M})$ (or $P(\mathbf{X}|Q_1^N)$ if the Viterbi criterion is used).

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

- Piecewise stationarity—we can model speech using a Markov chain;
- The prior probability of a model can be separately estimated—a language model including syntactic constraints about word sequences and phonological rules about sub-word unit sequences, $P(\mathbf{M})$, may be derived without reference to the acoustic data (although some attempts have been made to relax this assumption [20]);
- Observation independence—the current data vector, $\mathbf{x}(t)$, is conditionally independent of previously emitted data vectors $\mathbf{X}_1^{t-1} = \{\mathbf{x}(1), \ldots, \mathbf{x}(t-1)\}$;
- First order Markov process—the current state of the process, $q(t)$, depends only on the previous state, $q(t-1)$;
- State emission—the current data vector, $\mathbf{x}(t)$, is dependent only on the current state of the process, $q(t)$.

$p(\mathbf{X}|\mathbf{M})$ can be computed in terms of local joint densities $p(\mathbf{x}(t), q(t)|\mathbf{X}_1^{t-1}, Q_1^{t-1}, \mathbf{M})$, of a model $\mathbf{M}$ 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 $\mathbf{X}_1^{t-1}$:

$$p(\mathbf{X}|\mathbf{M}) = \sum_s p(q_s(t), \mathbf{X}|\mathbf{M}) \tag{2}$$

$$\begin{aligned} &p(q_s(t), \mathbf{X}|\mathbf{M}) = \\ &\quad p(q_r(t-1), \mathbf{X}_1^{t-1}|\mathbf{M}) p(q_s(t), \mathbf{x}(t)|\mathbf{X}_1^{t-1}, q_r(t-1), \mathbf{M}). \end{aligned} \tag{3}$$

(This is the forward recurrence of the Baum-Welch algorithm.) The above assumptions allow us to simplify the local density:

$$\begin{aligned} &p(\mathbf{x}(t), q(t)|\mathbf{X}_1^{t-1}, Q_1^{t-1}, \mathbf{M}) = \\ &\quad p(\mathbf{x}(t)|q(t), \mathbf{M})\ P(q(t)|q(t-1), \mathbf{M}) \quad . \end{aligned} \tag{4}$$

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

Training an acoustic model by density estimation involves estimating the state transition probabilities and the pdf from which

[2] A phone is an acoustic category, whereas a phoneme is a linguistic category. For example, an utterance of the word "citizenship" may be phonemically transcribed as /s ih t 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.

[3] We use P to represent a probability, and p to represent a probability density.

[4] 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.

the state output likelihoods are drawn. A key design decision in acoustic modeling (given a training criterion, such as 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 Laplacians, Gaussians and mixtures (linear combinations) of these. The parameters of the pdf are then estimated so as to optimally model the training data. If we are dealing with a family of models within which the correct model falls, this is an optimal strategy. However, in the case of modeling 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 (within the HMM constraints) 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 [21], 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 [22].

Discriminative Training. Ultimately in speech recognition we are not concerned with estimating the joint density of the speech data and word sequence, but are interested in the posterior probability of a word sequence given the acoustic data. More informally, we are not finally concerned with modeling the speech signal, but with correctly choosing the sequence of words that was uttered.

We may translate this concern to a local level, if we assume the Viterbi criterion. Rather than constructing the set of pdfs that best describe the data (within the constrained family of functions being optimized), 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 accurate a model as possible for each class. In practice this results in an algorithm that minimizes the likelihood of incorrect, competing models and maximizes the likelihood of the correct model. This differs from maximum likelihood density estimation, in which each data point is used to update the density model of the class to which it has been assigned.[5] Thus, in discriminative training, the parameters of a class pdf will be forced towards training examples from that class (as in maximum likelihood training), but will also be pushed away from training examples from competing classes.

There are many discriminative pattern recognition methods in the literature including the method of Potential Functions [23], learning vector quantization (LVQ) [24] and the multi-layer perceptron (MLP) (see e.g., [25]). Unlike the other methods, the MLP may be used directly to compute class-conditional posterior probabilities (see section III. B).

[5]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.

D. 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 or, more generally, by the phonotactics of the language. 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 assigns a prior probability to each allowable sentence in the language. 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 sentences, words, phones and HMM states.

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 of the phone relative frequencies observed in 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; in any event we are forced to use the latter since it is implicit to the models used in the recognition process.

III. MLP Probability Estimation

A. Multi-layer Perceptrons

Multi-layer perceptrons (MLPs) are probably the best studied class of neural networks. They have a layered feedforward architecture with an input layer, zero or more hidden layers and an output layer. Each layer is connected to the previous via a weight matrix, and operates according to the relation

$$y_i^L = f\left(\sum_j w_{ij}^{L,L-1} y_j^{L-1}\right) , \tag{5}$$

where y_i^L is the output of unit i in layer L, $w_{ij}^{L,L-1}$ is an element of the weight matrix between layers $L-1$ and L and f is the transfer function of a unit, typically a sigmoid:

$$f(x) = \frac{1}{1+\exp(-x)} . \tag{6}$$

Equation (5) can incorporate a bias for y_i^L by assuming a unit in layer $L-1$ with a fixed output of 1. The equation may also be modified to allow layer L to receive input from multiple lower layers, via additional weight matrices.

MLPs are trained to associate an input vector with a desired output vector. Both classification and regression may be performed in the same framework. In the case of N-class classification, a network with N outputs would be used, one for each class. A '1-from-N' training scheme would thus be used, where the desired output vector would contain a one for the correct class and zero for all other classes.

Training is accomplished via the back-propagation algorithm (e.g., [26]), a steepest descent procedure. For large problems, a stochastic approximation procedure is usually adopted (per sample update, rather than batch update).

B. Posterior Probability Estimation

MLPs may be used to estimate probabilities. Several authors have discussed the behavior of feedforward networks in terms of learning probability distributions (e.g., Hopfield [27]). Bourlard and Wellekens [28,29] proved that a MLP trained to perform classification is a class-conditional posterior probability estimator.[6] That is, after a '1-from-N' training, a MLP output value, given an input $\mathbf{x}$, will be an estimate of the posterior probability, $P(c_i|\mathbf{x})$, of the corresponding class c_i given the input. This result holds for training with various error functions (including relative entropy and mean square error). The output units should be constrained to be non-negative and less than one (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 one. However, computational experiments have shown that the sum of estimated posteriors is usually extremely close to one, for test vectors drawn from a region of space well-sampled by the training data [32,33]. 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 normalizing output transfer function such as the normalized exponential or 'softmax' [34],

$$f(x_i^L) = \frac{\exp(x_i^L)}{\sum_{j \in L} \exp(x_j^L)} , \tag{7}$$

where x_i^L is the activation (pre-transfer function output) of unit i in layer L.

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:

$$P(c_i|\mathbf{x}) = \frac{p(\mathbf{x}|c_i)}{p(\mathbf{x})} . \tag{8}$$

A posterior probability estimate 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 estimate, on the other hand, tells us both how likely we are to observe a particular vector, as well as its class membership probabilities. 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.

The theorem that shows that an MLP may be used as a posterior probability estimator is valid only when a global minimum of the error function is attained. 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 VI). Cross-validation is a sensible approach, however, since we do not wish to compute the density for the training set, but estimate this density for an unseen test set. Empirical results indicate that good posterior probability estimates are achieved with this method [35].

C. Obtaining Likelihood Estimates

Applying a trained MLP to test data gives us estimates of the conditional posterior probabilities of each class. These probabilities depend on the relative class frequencies, which may be regarded as estimates of $p(c_i)$. However, as discussed earlier, we want to use the language model priors at recognition time. Thus, the relative class frequencies are factored out at recognition time, to give (scaled) likelihood estimates rather than posterior probability estimates.[7]

It is easy to convert posteriors to scaled likelihoods using (8), but with non-equal priors:

$$\frac{p(\mathbf{x}|c_i)}{p(\mathbf{x})} = \frac{P(c_i|\mathbf{x})}{P(c_i)} . \tag{9}$$

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

D. HMM Probability Estimation

Using the above framework, we may use a MLP to estimate HMM output probabilities. As an example, consider a system with P single-state phone models. If a MLP is trained to classify its inputs into 1 of P phone classes, then the ith output of the MLP is as an estimate of $P(c_i|\mathbf{x})$. This may be used to estimate the output probability of the single state of phone HMM c_i. This posterior probability estimate implicitly uses the relative frequencies in the training set as phone priors, so we use (9) to convert the posterior probability estimates to scaled likelihood estimates. In general, the output probabilities of multiple-state HMMs may be estimated using a MLP, with an output corresponding to each independent state.

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 the output 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.

[6] This result was expanded by Gish [30], Hampshire and Pearlmutter [31] and Richard and Lippmann [32], among others.

[7] This substitution is not forced upon us; there is no theoretical reason why a Viterbi search cannot be carried out using posterior probabilities [29] (but, see section II. D).

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

E. Priors and Biases

We would like to have a statistical understanding of the parameters of a connectionist network. If we use a softmax transfer function at the output layer then:

$$(10)\qquad P(c_i|\mathbf{x}) = \frac{\exp(\sum_j w_{ij}\text{hid}_j + \text{bias}_i)}{\sum_k \exp(\sum_j w_{kj}\text{hid}_j + \text{bias}_k)}$$

$$(11)\qquad \propto \exp(\sum_j w_{ij}\text{hid}_j + \text{bias}_i)\,,$$

also,

$$(12)\qquad P(c_i|\mathbf{x}) \propto \exp\left[\log(p(\mathbf{x}|c_i)) + \log(P(c_i))\right],$$

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, $\log(p(\mathbf{x}|c_i))$) and the bias as the prior part (the log prior, $\log(p(c_i))$) of each output unit. Our observations of the output biases of a trained network do indeed show a correlation with the log priors (relative frequencies).

Note that a similar relationship holds in the case of sigmoid output units. If the output of a sigmoid is a posterior probability then, following the above reasoning, we may identify the bias with the log odds of the prior, $\log[p(c_i)/(1-p(c_i)]$, remembering the inverse of the sigmoid function $f(x)$ is $\log[f/(1-f)]$.

However this relationship is too facile. Let us consider the case of a Gaussian classifier. As is well known, we can write down the corresponding linear discriminant function (see e.g., [36]) for a Gaussian classifier with equal covariances:

$$(13)\qquad g_i(\mathbf{x}) = \mathbf{w}_i^T\mathbf{x} + w_{i0}\,,$$

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

$$(14)\qquad \mathbf{w}_i = \Sigma^{-1}\mu_i$$

and

$$(15)\qquad w_{i0} = -\mu_i^T\Sigma^{-1}\mu_i + \log(P(c_i))\,.$$

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

We may expect the output biases of a trained MLP to be influenced by the acoustic data, as well as prior information. One way we attempted to minimize the influence of the acoustic data on the output biases 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 biases would learn to encode the class relative frequencies, rather than the acoustic data. Of course, hidden unit outputs resulting from speech input are not random and, as reported in [37], 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 generalization (section VII. A).

IV. Alternative Connectionist Estimators

MLPs are not the only connectionist probability estimators we could use for this task. In this section we consider radial basis function (RBF) networks and the recurrent generalization of the MLP. Both these classes of network are posterior probability estimators, when trained appropriately. We also consider MLPs used as predictors, which may be shown to estimate conditional likelihoods.

A. Radial Basis Function Networks

RBF networks are also feedforward networks. Their primary difference from MLPs is that the hidden layer consists of units with local or semi-local transfer functions. They are often referred to as radial basis functions, since they may be radial in nature and are assumed to form a high dimensional basis for the input data set.

The RBF network was originally introduced as a method of function approximation [38,39]. A set of K approximating functions $f_k(\mathbf{x},\theta)$ is constructed from a set of J basis functions $\phi_j(\mathbf{x},\theta)$,

$$(16)\qquad f_k(\mathbf{x},\theta) = \sum_{j=1}^{J} a_{kj}\phi_j(\mathbf{x},\theta) \quad 1 \le k \le K\,.$$

This defines a network with J RBFs (hidden units) and K linear output units, with weights a_{kj}. The form of the RBFs is typically Gaussian, with the "weights" θ between the input and hidden layer specifying the means and covariances of the RBFs.

When using RBF networks, the RBF layer is usually trained in an unsupervised manner (e.g., k-means clustering). The hidden units in such a network may be regarded as modeling the input distribution, with no discriminative component. There are several attractive reasons for using RBF networks:

- Training the RBF layer in an unsupervised manner is computationally efficient.
- If linear output units are used, then the discriminative training of the output layer may be accomplished non-iteratively using a matrix inversion [39,40].
- Since RBFs are local, they are suitable for time-varying input distributions. RBFs that describe input data that no longer occurs will not affect the final classification. This is not the case for MLPs where hidden units have global effects within the input space.
- RBF networks have a structural isomorphism to tied mixture density models, although the training criterion is often different [41]. In both cases the outputs are weighted sums of Gaussians; however in the tied mixture case, the network is trained as a density estimator using the maximum likelihood criteria; in the RBF case it is trained as a discriminative posterior probability estimator.

There is no reason why the means and variances of the RBFs cannot be trained discriminatively, using the back-propagation algorithm. However, this sacrifices computational efficiency at training time. Furthermore, it seems that a full discriminative training of non-local (sigmoid) hidden units is likely to be more effective than training local hidden units, since the global hidden units are in a position to compute global "facts" about the input.

In the case of RBF networks with sigmoid or softmax outputs, we can show that the outputs are posterior probability estimates since the essential conditions are the same as for an MLP. In the case of linear output units the outputs are not guaranteed to be formally probabilities—they may be negative or greater than one, although, it may be proved that the outputs of a trained 1-from-N RBF network will sum to one [42]. In practice (e.g., [43,44]), outputs less than zero or greater than one are not common. Such outputs are usually reset by a post-network hard limiter between zero and one.

Linear output RBF networks are an attractive probability estimator for computational reasons [43,44]; large RBF networks may be trained on a substantial speech database using standard workstations. This is not the case with MLPs. RBF networks with a prior-weighted softmax (or prior-weighted linear normalizing) output transfer function are also potentially attractive, since in this case there is a relationship between the RBF network weights and the coefficients of a tied mixture density system [41].

RBF networks have been used successfully as probability estimators in HMM continuous speech recognition systems [43] and isolated word recognition systems [44]. However, there has been no evidence that RBF networks outperform MLPs, provided there is adequate computational support to train the desired MLP. In a series of experiments at ICSI using the DARPA speaker-dependent Resource Management continuous speech database, the RBF systems offered a considerably inferior performance compared with our standard MLP systems (these MLP experiments are discussed in [45,46], and are similar to the speaker-independent experiments in section VII). Using single frame perceptual linear prediction (PLP) coefficient inputs [47], a 512 hidden unit MLP produced a phone classification rate of 59%, at the frame level. Extensive experiments with RBF networks (using the various training schemes and output transfer functions outlined above) produced a best classification score of 52%. This network had 1000 RBFs and prior-weighted, normalized exponential output units. The RBFs were determined by a k-means clustering process and training of the output weights was performed using back-propagation.

Experiments have also been performed in which the coefficients of a tied mixture density HMM (SRI's DECIPHER system [7]) were used to initialize a RBF network, which was then discriminatively trained using the scheme described in [41]. However, this additional training did not result in an improved performance. We hypothesize that the initial state of the network, as specified by the maximum likelihood trained tied-mixture system, also corresponded (or nearly so) to a local minimum of the error surface of the discriminative objective function of the RBF network.

B. Recurrent Networks

We may also use recurrent networks to estimate posterior probabilities of HMM states. Robinson [48,10] has used such networks to great effect in phone recognition and continuous speech systems.

These recurrent networks are essentially multi-layer perceptrons with added recurrent connections between the hidden units. These feedback connections are advantageous since they provide the network with a time-dependent state. A MLP may be interpreted as a FIR filter with fixed window back in time. A recurrent network, however, is an IIR system with a potentially infinite window back in time. This is due to the network's internal feedback.

A recurrent network may be unfolded in time to give a multi-layer network, with a separate layer for the hidden units at each time. This unfolding enables training using a variant of the back propagation algorithm, referred to as back propagation through time [26,49,50]. Since the output layer at any time may be regarded as the output of a deep feedforward network, the probability estimation proofs for the MLP also hold (given similar training conditions). Thus we may use such networks to estimate probabilities.

C. Predictive MLPs

MLPs may be used for regression. Given the previous p samples or frames of speech we may train a MLP to predict the next sample or frame. Although it is unlikely that a single predictive MLP could be used practically as a general model of speech, it is a powerful way to model stationary dynamics. Thus, we could embed predictive MLPs in a Markov process to give a piecewise stationary model of speech dynamics [51,52,53,54].

The resultant model is a (nonlinear) autoregressive (AR) HMM, in which state output pdfs depend on the prediction error of the MLP. Linear ARHMMs using Gaussian autoregressive densities on each state are well studied [55,56]. An advantage of using this type of model is that it explicitly addresses observation independence by modeling the observations as an AR process. If the AR process is constructed at the sample level, then we have an elegant fusion of linear predictive analysis and hidden Markov modeling, with no distinction between analysis and recognition.

Using MLPs as nonlinear predictors in an ARHMM results in a more powerful, but more computationally expensive, model than a linear ARHMM. The predictive MLPs are trained by the usual gradient descent process embedded in a Viterbi [51,52,53] or forward-backward [54] HMM training algorithm. These nonlinear ARHMMs have generally modeled at the feature vector level, rather than at the sample level. Levin [51] used a single predictive MLP, rather than a MLP for each HMM state; however this MLP had an extra set of "control" inputs, representing HMM state.

Predictive MLPs do not have the discriminative character of direct MLP probability estimators. Rather than estimating posterior probabilities, of the form $P(q_i|\mathbf{x}(t))$, they estimate a *conditional* likelihood, $p(\mathbf{x}(t)|q_i, \mathbf{X}_{t-p}^{t-1})$, which may be used to estimate the global conditional likelihood $p(\mathbf{X}_{p+1}^N, Q_{p+1}^N|\mathbf{X}_1^p, Q_1^p)$ [57,58].

V. DISCRIMINATIVE HMMs

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

All these probabilities should be conditioned on the model parameters, Θ and the probability of the data should be expressed 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, so $p(\mathbf{X}|\Theta)$ is not constant across models. We may rewrite this probability as:

$$
\begin{aligned}
(17) \quad p(\mathbf{X}|\Theta) &= \sum_i p(\mathbf{X}|\mathbf{M}_i, \Theta) P(\mathbf{M}_i) \\
(18) \quad &= p(\mathbf{X}|\mathbf{C}, \Theta) P(\mathbf{C}) + \sum_j p(\mathbf{X}|\mathbf{I}_j, \Theta) P(\mathbf{I}_j),
\end{aligned}
$$

where $\mathbf{C}$ represents the correct model and $\mathbf{I}_j$ an incorrect model.

If we take our acoustic model as the ratio $p(\mathbf{X}|\mathbf{M}, \Theta)/p(\mathbf{X}|\Theta)$, rather than $p(\mathbf{X}|\mathbf{M}, \Theta)$, we must maximize this ratio. This may be carried out by discriminative training, which maximizes the likelihood of the correct model, while simultaneously reducing the likelihood of competing, incorrect models. We refer to an HMM trained in this fashion as a *discriminative* HMM.

A. Frame-level Discrimination

In this paper, we deal mainly with discriminative training of HMMs at the frame level, rather than the model level. In practice this means we are concerned with the estimation of the state output probabilities $p(\mathbf{x}|q_i)$. As discussed earlier we may obtain posterior probability estimates $P(c_j|\mathbf{x})$ using a MLP trained for framewise phone classification. If we consider single output distribution phone HMMs (*i.e.* single state HMMs or multiple state HMMs which have a shared output distribution common to all states in the model), then the probability $P(c_j|\mathbf{x})$ output by the MLP may be identified with the posterior probability $p(q_i|\mathbf{x})$ of a state q_i in the phone HMM modeling c_j. After dividing by the relative frequencies (estimates of $P(c_j)$ and hence $P(q_i)$) and invoking Bayes' rule we have a discriminative acoustic model at the frame level, *i.e.* an estimate of $p(\mathbf{x}|q_i)/p(\mathbf{x})$.

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

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. An alternative approach was the discriminative HMM, originally defined by Bourlard and Wellekens [29]. Here the network estimates the local posterior probabilities $P(q(t)|q(t-1), \mathbf{x}(t))$. This posterior combines the modeling of the output probabilities (now transition-specific, rather than state-specific) and the transition probabilities.

It is clear that this approach leads to discriminative acoustic models at the frame level only, which does not guarantee discrimination at the word or sentence level. However, it is worth noting that if the local probabilities sum to one over all possible states (which is of course the case for actual posterior probabilities or if a softmax function is used at the output of the MLP, and always approximately true in other cases—see section III B) then the global posteriors $P(S|\mathbf{X})$ are also discriminant, summing to one over all possible sentences. (For a proof of this, see [58].) The local probabilities may be estimated by a MLP, with the addition of binary input units, representing the previous state (one for each possible previous state). At recognition time, posterior probabilities $p(q(t)|q(t-1), \mathbf{x}(t))$ must be computed for all possible transitions. Thus, several forward passes through the network are required for each frame, corresponding to each possible $q(t-1)$. We have not yet performed significant experiments using this scheme.

B. Global Discrimination

There have been two basic approaches suggested for the optimization of a discriminative HMM at a model (or global) level. One involves a direct computation of the posterior probability of a model given the acoustic data; the second is the Viterbi approximation to this.

Bahl et al. [59] presented a training scheme for continuous HMMs in which the mutual information between the acoustic evidence and the word sequence was maximized. This approach used a discriminative objective function, locally maximized by gradient ascent. More recently, Bridle introduced the "alphanet" representation [60] 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 minimizing a relative entropy objective function. This function incorporates the negative log of the posterior probability of the correct model given the acoustic evidence $P(\mathbf{M}|\mathbf{X}, \Theta)$, 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. 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). Direct computation of this quantity for continuous speech 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 optimization methods for continuous speech recognition has been performed by Bridle [61], Niles [62] and Bengio [63]. Bridle and Bengio used this approach to optimize the input parameters via some (linear or nonlinear) transform, training the parameters of the HMM by a maximum likelihood process.

The Viterbi approximation to this is analogous to segmental k-means training and has been referred to as *embedded training* [64] or connectionist Viterbi training [65]. In this method a frame level optimization is interleaved with a Viterbi re-alignment. It should be noted that the transition probabilities are still optimized 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 optimization [29]. There is, however, a mismatch between model and acoustic data priors, as discussed earlier.

VI. A Connectionist–HMM Continuous Speech Recognition System

The previously described components may be put together to form a hybrid connectionist–HMM continuous speech recognition system (figure 2).

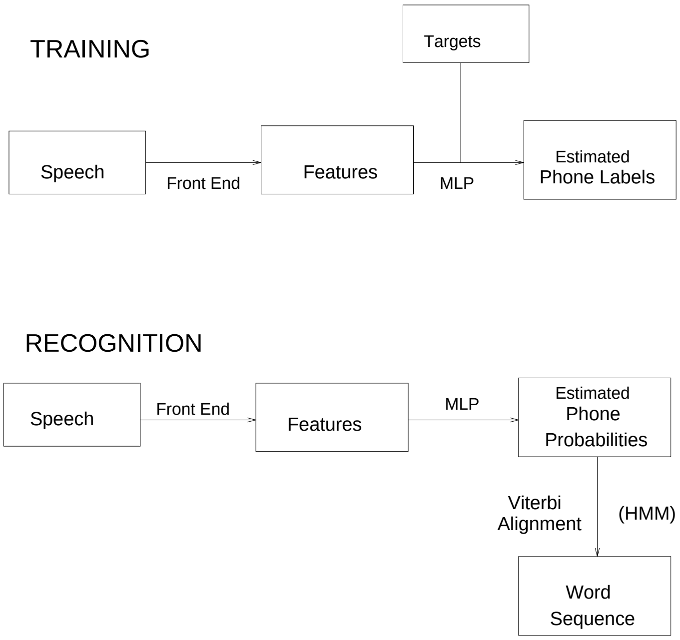

Figure 2: A schematic of the training and recognition processes. *At both training and recognition times, the speech is processed by a front end (e.g., a mel cepstral or a PLP transform) that extracts a concise description of the speech every frame (typically every 10 ms). Using alignments produced by a previously trained recognizer (or bootstrapping on time-aligned labeled data, such as the TIMIT database), a MLP is trained to phonetically classify frames of data. The alignment/training process may be iterated to provide an "embedded training" process. For recognition 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).*

The front end consists of sampling the time-amplitude waveform, followed by an analysis process designed to give a concise representation of the speech signal. This is usually a frame based analysis in which a window of speech (typically 20 ms wide) is analyzed by some kind of spectral analysis and the window is advanced at discrete intervals (typically 10 ms). The resulting speech signal is then characterized 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 spectral "snapshots".

In this paper we are 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 initialized and their parameters 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 (*i.e.* $p(\mathbf{x}|q_i) = p(\mathbf{x}|c_j)$ for all states q_i of phone c_j), the MLP may be viewed as being trained to perform phonetic discrimination. We initialize the models and perform a Viterbi alignment (using a bootstrap recognizer), producing a time-aligned state segmentation, subject to the language model. From the state segmentation we can, of course, obtain phone and word segmentations. The state segmentation is used to produce the training targets for the MLP. The MLP targets implicitly contain information about the model. The MLP is then trained using the back-propagation algorithm. In this Viterbi training scheme, the temporal and static parts of the training problem are separated, in contrast to the forward-backward algorithm. The process may be iterated, alternating between training the MLP and re-estimating the transition probabilities, an embedded training process.

The usual time-synchronous Viterbi decoding algorithm is used for recognition. For each frame of speech, a vector of HMM state conditional posterior probabilities ($p(q_i|\mathbf{x})$ for all states q_i) is produced by the MLP. These are converted to scaled likelihoods by dividing by the relative state frequencies of the training data. In combination with the transition probabilities, we may use the Viterbi algorithm 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 penalizes (lowers the probability) of word transitions, thus reducing the tendency to favor short words over longer ones.[8]

Modeling Context. Thus far, we have only considered context-independent[9] phone modeling using MLPs. However, traditional HMM systems have experienced a much improved performance by modeling phones in context [67,6,7].

Consider a set of N phone classes $\mathcal{C} = \{c_1, \ldots, c_N\}$ and D context classes $\mathcal{D} = \{d_1, \ldots, d_D\}$. A naive approach to estimating context-dependent probabilities using a MLP would require a network containing $D \times N$ outputs. This approach scales badly, as it would result in extremely large networks for anything other than a trivial set of context classes. The traditional context-dependent HMM approach also suffers from the need to estimate an excessive number of parameters.

However, without any simplifying assumptions, we may estimate context-dependent posterior probabilities using networks that are not substantially larger than our context-independent

[8] This is a result of the observation independence assumption, causing an under-estimate of the joint observation density [66].

[9] 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.

MLPs. The method uses a network decomposition [68], based on the definition of conditional probability:

$$p(c_i, d_j|\mathbf{x}) = p(c_i|\mathbf{x})p(d_j|c_i, \mathbf{x}) \quad (19)$$

The first term on the right hand side is estimated by our usual context-independent MLP. The second term is the estimate of the posterior probability of the context class, given the input and the phone class. This second term may also be estimated using an MLP with the usual speech input, plus a dependence on the phone class.

This dependence on the phone class may be represented in two basic ways:

1. By an extra set of 1-from-N binary inputs, one for each possible phone class;
2. By having a replicated network (mapping from input data to context class) for each possible phone class.

The first approach has been investigated in [46,69]. This method of network decomposition may be viewed as trading space for time during recognition. At recognition time, probabilities must be computed for all phones in all contexts (assuming there is no pruning). This means that for each frame of data, the second network must be run multiple times for all possible phone classes. Fortunately, with a possible constraint on representability, we may increase computational efficiency by having a direct connection from the binary units to the output units, bypassing the hidden units.

We may compute a table of the binary units' effect on the output, with these vectors being added on directly before the output transfer function. Thus the bulk of computation is only performed once per frame.

A variant of the second approach [70,71] uses an equivalent decomposition:

$$p(c_i, d_j|\mathbf{x}) = p(d_i|\mathbf{x})p(c_j|d_i, \mathbf{x}) \ . \quad (20)$$

The second network maps acoustic data to the phone class in a particular context. In this case we have one network for each possible context. The number of parameters is reduced by constraining the set of networks to share a hidden layer. Furthermore, the hidden-to-output transformation of the context-dependent networks may be initialized with the context-independent weights. Cross-validation training may then be used to ensure that context-dependent training does not decrease performance. This leads to a smoothing of the context-independent and context-dependent parameters.

VII. Experiments

A. Methods

Most of our experiments have been performed using the DARPA Resource Management (RM) speaker-independent continuous speech database [11]. This is a very well studied database with a vocabulary of 991 words. The standard training set contains 3990 sentences from 109 speakers. 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 branching factor, or *perplexity*, which is 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 [72], 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 2 or 3 state output pdfs in each model were tied. Thus each phone model contained a single output distribution, with the multiple states acting as a duration model. The output 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.

These probability estimates were integrated into SRI's system, DECIPHER [7]. It includes multiple probabilistic word pronunciations, cross-word phonological models, multiple densities per phone, and context-dependent phone models. The basic DECIPHER system uses tied Gaussian mixture state output pdfs.

Initial work used a context-independent form of DECIPHER. This was the complete DECIPHER system, except that there 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. Additionally, we also worked with the context-dependent DECIPHER system. In this case the context-dependent tied mixture pdfs were augmented by the context-independent MLP pdfs.

The context-dependent DECIPHER system was used to bootstrap our models. Our initial training targets were obtained by using the tied mixture DECIPHER segmentation, and we retained the estimated transition probabilities.

B. Architectures and Training

The networks trained to perform these estimations were large. In addition to the current frame (26 inputs), 4 frames of left and right context were appended, giving a total of 234 inputs. We usually used networks with about 1000 hidden units and 69 output classes, giving a total of over 300,000 weights. Training a network with this many weights is not trivial.[11] We 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 testing phonetic classification on an inde-

[10] The mel cepstrum is the Fourier transform of the logarithm of a mel spectrum. This spectrum, usually computed using DFTs, 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 [73], 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 using a 16 processor system.

pendent test set after each epoch [12]. 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.

Following the discussion in section 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). We have found that training is speeded (by over 25%) and generalization improved by initializing 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 epochs through the training set reduced to 5) and also improved generalization.

C. Results

Our experiments were performed using an MLP with sigmoid outputs, trained on the RM training set. Training the 300,000 weight network required around 5 passes through the training database of 1.5 million training patterns. Around 225,000 patterns were used for cross-validation (a combination of the February 89 and October 89 RM speaker-independent test sets).

We used two classes of test sentences for evaluation. A 600 sentence development set (the February 1989 and October 1989 RM speaker-independent test sets), the same as the network cross-validation set, was used to tune the HMM recognition parameters, such as the word transition probability. Final word recognition results were obtained using unseen test sets of 300 sentences. Three such sets were used here, the February 1991 and the two September 1992 RM speaker-independent test sets. No tuning of parameters was performed using these sets.

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% on the February 1991 test set, 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 one of 69 classes (and outputting posterior probability estimates) the word recognition error improved to 5.8%. This network was trained from context-dependent alignments, which may give a slight advantage to the MLP. A similar experiment using context-independent alignments resulted in a slight degradation in performance (to 6.1%) but still showed a large improvement over the baseline. 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. Two heuristics were tried for combining the MLP and tied mixture estimates of the state output probabilities. In the first weighted logs of the MLP and tied mixture likelihood estimations were used:

$$(21)\ \log(P(\mathbf{x}|q_j)) = \lambda_1 \log\left(\frac{P_{mlp}(q_j|\mathbf{x})}{P(q_j)}\right) + \lambda_2 \log(P_{tm}(\mathbf{x}|q_j))$$

where P_{mlp} denotes the MLP estimate of a probability and P_{tm} the tied mixture estimate. A single set of λs was used over all the states: they were optimized for minimum recognition error over the 300 sentence development set.

In the second heuristic, the log of a weighted average of the state output probabilities estimated by the MLP and the tied Gaussian mixtures was used:

(22)
$$\log(P(\mathbf{x}|q_j)) = \log\left(\lambda_1 \frac{P_{mlp}(q_j|\mathbf{x})P_{tm}(\mathbf{x})}{P(q_j)} + \lambda_2 P_{tm}(\mathbf{x}|q_j)\right).$$

In this approximation, the probability of the data $P(\mathbf{x})$ was required to ensure that the two likelihood estimates are scaled similarly. This cannot be obtained from the MLP, and was approximated by summing over the state conditional tied Gaussian likelihoods:

$$(23)\qquad P_{tm}(\mathbf{x}) = \sum_i P_{tm}(\mathbf{x}|q_i)P(q_i).$$

The best results on the development set were obtained using the first method, which was adopted when evaluating over the three test sets. This reduced the error significantly in both cases. These results are summarized in tables I and II and graphed in figure 3 for the February 1991 test set using the wordpair grammar.

	% error		
Test Set	CI-MLP	CD-HMM	MIX
Feb 91	5.8	3.8	3.2
Sep 92a	10.9	10.1	7.7
Sep 92b	9.5	7.0	5.7

Table I: Results using the three test sets with the perplexity 60 wordpair grammar. *CI-MLP is the context-independent MLP-HMM hybrid system, CD-HMM is the full context-dependent* DECIPHER *system and the MIX system is a simple interpolation between the CD-HMM and the CI-MLP.*

	% error		
Test Set	CI-MLP	CD-HMM	MIX
Feb 91	24.7	19.3	15.9
Sep 92a	31.5	29.2	25.4
Sep 92b	30.9	26.6	21.5

Table II: Results using the three test sets using no grammar (perplexity 991).

The relationship between the number of parameters and recognition performance is graphed in figure 4.

[12] An epoch corresponds to a training iteration over the entire training set. Note that when using random pattern selection, training does not involve a series of complete passes through the complete training set. We regard an epoch as training on P randomly chosen patterns, when there are P patterns in the training set.

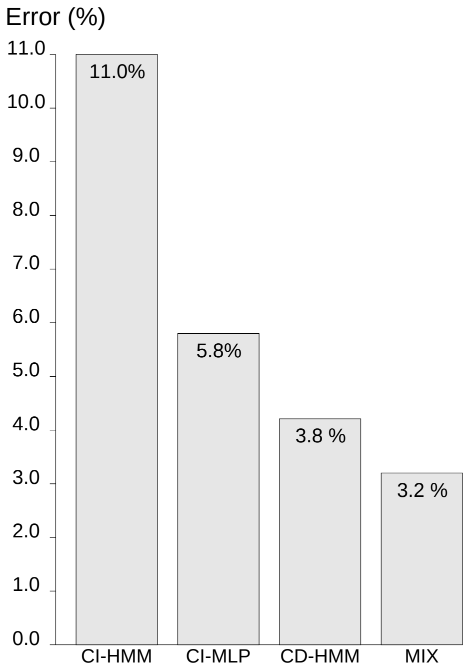

Figure 3: Results using the February 1991 test set, using the perplexity 60 wordpair grammar. *This also includes the performance of the context-independent tied mixture HMM (CI-HMM) system.*

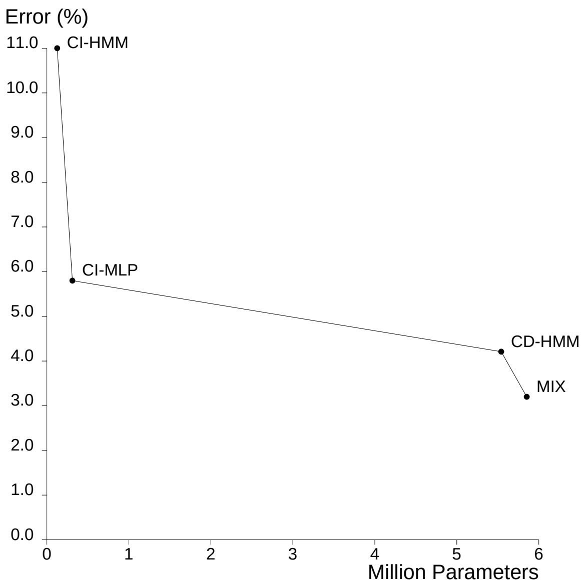

Figure 4: Recognition error versus number of parameters, for the February 1991 test set, using the wordpair grammar. *A MLP with the same number of parameters as the CI-HMM (500 hidden units) achieves about 8.0% recognition error (not shown on this graph).*

VIII. Conclusion

In this paper 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 feedforward 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:

- Comparing like with like, a discriminatively trained connectionist context-independent system performed considerably better than the corresponding maximum likelihood tied mixture system.
- The context-independent MLP–HMM hybrid system had a word accuracy of from 0.8% to 2.5% lower than the context-dependent HMM However the latter system has 50 times the number of models and 35 times the number of parameters compared with the MLP system.
- Interpolating MLP context-independent probabilities with tied mixture context-dependent probabilities produced an increase in word accuracy.

These results arise from weakening two underlying HMM assumptions:

- Model correctness—By estimating only the class-conditional posterior probability, we have not attempted to optimize 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 robust front ends, speaker adaptation, consistency modeling and context-dependent phone modeling.

Acknowledgements

This work benefited from the input of Phil Kohn, Yochai Konig and Chuck Wooters. Thanks to Mike Hochberg, Tony Robinson and an anonymous referee for a critical reading of the manuscript. We acknowledge support from the International Computer Science Institute, from DARPA contract MDA904-90-C-5253 awarded to SRI International, and from ESPRIT Basic Research Action 6487, Wernicke. SR is currently supported by a SERC postdoctoral fellowship. HF was partially supported by a fellowship from CONICET, Argentina.

References

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

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

[3] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, pp. 328–339, 1989.

[4] M. Fanty, R. A. Cole, and K. Roginski, "English alphabet recognition with telephone speech," in *Advances in Neural Information Processing Systems* (J. E. Moody, S. J. Hanson, and R. P. Lippmann, eds.), vol. 4, pp. 199–206, San Mateo CA: Morgan Kaufmann, 1992.

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

[6] K. F. Lee, *Large Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1988.

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

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

[9] S. Austin, G. Zavaliagkos, J. Makhoul, and R. Schwartz, "Speech recognition using segmental neural nets," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (San Francisco), pp. 625–628, 1992.

[10] T. Robinson, "The application of recurrent nets to phone probability estimation," *IEEE Transactions on Neural Networks*. To appear March 1994.

[11] P. Price, W. M. Fisher, J. Bernstein, and D. S. Pallett, "The DARPA 1000-word Resource Management database for continuous speech recognition," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 651–654, 1988.

[12] S. E. Levinson, "Continuously variable duration hidden Markov models for automatic speech recognition," *Computer Speech and Language*, vol. 1, pp. 29–45, 1986.

[13] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains," *Annals of Mathematical Statistics*, vol. 41, pp. 164–171, 1970.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society Series B*, vol. 39, pp. 1–38, 1977.

[15] L. A. Liporace, "Maximum likelihood estimation for multivariate observations of Markov sources," *IEEE Transactions on Information Theory*, vol. IT-28, pp. 729–734, 1982.

[16] L. R. Rabiner, J. G. Wilpon, and F. K. Soong, "High performance connected digit recognition using hidden Markov models," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, pp. 1214–1225, 1989.

[17] F. Jelinek, "Fast sequential decoding algorithm using a stack," *IBM Journal of Research and Development*, vol. 13, pp. 675–685, 1969.

[18] N. J. Nilsson, *Problem-Solving Methods in Artificial Intelligence*. New York: McGraw-Hill, 1971.

[19] H. Ney, "The use of a one-stage dynamic programming algorithm for connected word recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, pp. 263–271, 1984.

[20] L. R. Bahl, S. Das, P. V. de Souza, M. Epstein, R. L. Mercer, B. Merialdo, D. Nahamoo, M. A. Picheny, and J. Powell, "Automatic phonetic baseform determination," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Toronto), pp. 173–176, 1991.

[21] E. Parzen, "On estimation of a probability density function and mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.

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

[23] M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control*, vol. 25, pp. 821–837, 1964.

[24] T. Kohonen, G. Brna, and R. Chrisley, "Statistical pattern recognition with neural networks: benchmarking studies," in *Proceedings Second IEEE International Conference on Neural Networks*, vol. 1, (San Diego), pp. 61–68, 1988.

[25] J. Hertz, A. Krogh, and R. Palmer, *Introduction to the theory of neural computation*. Redwood City, CA: Addison-Wesley, 1991.

[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Procressing* (D. E. Rumelhart and J. L. McClelland, eds.), vol. 1, pp. 318–362, Cambridge MA: MIT Press, 1986.

[27] J. J. Hopfield, "Learning algorithms and probability distributions in feed-forward networks," *Proceedings of the National Academy of Sciences USA*, vol. 84, pp. 8429–8433, 1987.

[28] H. Bourlard and C. J. Wellekens, "Links between Markov models and multi-layer perceptrons," in *Advances in Neural Information Processing Systems* (D. S. Touretzky, ed.), vol. 1, pp. 502–510, San Mateo CA: Morgan Kaufmann, 1989.

[29] H. Bourlard and C. J. Wellekens, "Links between Markov models and multilayer perceptrons," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-12, pp. 1167–1178, 1990.

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

[31] J. B. Hampshire II and B. A. Pearlmutter, "Equivalence proofs for multilayer perceptron classifiers and the Bayesian discriminant function," in *Proceedings of the 1990 Connectionist Models Summer School* (D. Touretzky, J. Elman, T. Sejnowski, and G. Hinton, eds.), pp. 159–172, San Mateo CA: Morgan Kaufmann, 1990.

[32] M. D. Richard and R. P. Lippmann, "Neural network classifiers estimate Bayesian a posteriori probabilities," *Neural Computation*, vol. 3, pp. 461–483, 1991.

[33] H. Bourlard and N. Morgan, "Merging multilayer perceptrons and hidden Markov models: some experiments in continuous speech recognition," in *Neural Networks: Advances and Applications* (E. Gelenbe, ed.), pp. 215–239, North-Holland, 1991.

[34] J. S. Bridle, "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters," in *Advances in Neural Information Processing Systems* (D. S. Touretzky, ed.), vol. 2, pp. 211–217, San Mateo CA: Morgan Kaufmann, 1990.

[35] N. Morgan and H. Bourlard, "Generalization and parameter estimation in feedforward nets: Some experiments," in *Advances in Neural Information Processing Systems* (D. S. Touretzky, ed.), vol. 2, pp. 413–416, San Mateo CA: Morgan Kaufmann, 1990.

[36] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley Interscience, 1973.

[37] S. Renals and N. Morgan, "Connectionist probability estimation in HMM speech recognition," Tech. Rep. TR-92-081, International Computer Science Institute, Berkeley CA, USA, 1992.

[38] M. J. D. Powell, "Radial basis functions for multi-variable interpolation: a review," Tech. Rep. DAMPT/NA12, Dept. of Applied Mathematics and Theoretical Physics, University of Cambridge, 1985.

[39] D. S. Broomhead and D. Lowe, "Multi-variable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321–355, 1988.

[40] S. Renals and R. Rohwer, "Phoneme classification experiments using radial basis functions," in *Proceedings International Joint Conference on Neural Networks*, vol. I, (Washington DC), pp. 461–468, 1989.

[41] S. Renals, N. Morgan, H. Bourlard, H. Franco, and M. Cohen, "Connectionist optimisation of tied mixture hidden Markov models," in *Advances in Neural Information Processing Systems* (J. E. Moody, S. J. Hanson, and R. P. Lippmann, eds.), vol. 4, pp. 167–174, San Mateo CA: Morgan Kaufmann, 1992.

[42] A. R. Webb and D. Lowe, "The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis," *Neural Networks*, vol. 3, pp. 367–375, 1990.

[43] S. Renals, D. McKelvie, and F. McInnes, "A comparative study of continuous speech recognition using neural networks and hidden Markov models," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Toronto), pp. 369–372, 1991.

[44] E. Singer and R. P. Lippmann, "A speech recogniser using radial basis function neural networks in an HMM framework," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (San Francisco CA), pp. 629–632, 1992.

[45] N. Morgan, C. Wooters, H. Bourlard, and M. Cohen, "Continuous speech recognition on the Resource Management database using connectionist probability estimation," in *Proceedings International Conference on Spoken Language Processing*, (Kobe), pp. 1337–1340, 1990.

[46] H. Bourlard and N. Morgan, "Continuous speech recognition by connectionist statistical methods," *IEEE Transactions on Neural Networks*, in press.

[47] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *Journal of the Acoustical Society of America*, vol. 87, pp. 1738–1752, 1990.

[48] T. Robinson and F. Fallside, "A recurrent error propagation network speech recognition system," *Computer Speech and Language*, vol. 5, pp. 259–274, 1991.

[49] A. J. Robinson and F. Fallside, "Static and dynamic error propagation networks with application to speech coding," in *Neural Information Processing Systems, Denver CO 1987* (D. Z. Anderson, ed.), pp. 632–641,

New York: American Institute of Physics, 1987.
[50] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proceedings IEEE*, vol. 78, pp. 1150–1160, 1990.
[51] E. Levin, "Word recognition using hidden control neural architecture," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Albuquerque), pp. 433–436, 1990.
[52] K. Iso and T. Watanabe, "Speaker-independent word recognition using a neural prediction model," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Albuquerque), pp. 441–444, 1990.
[53] J. Tebelskis and A. Waibel, "Large vocabulary recognition using linked predictive neural networks," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Albuquerque), pp. 437–440, 1990.
[54] E. Tsuboka, Y. Takada, and H. Wakita, "Neural predictive hidden Markov model," in *Proceedings International Conference on Spoken Language Processing*, (Kobe, Japan), pp. 1341–1344, 1990.
[55] A. B. Poritz, "Linear predictive hidden Markov models and the speech signal," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Paris), pp. 1291–1294, 1982.
[56] B. H. Juang and L. R. Rabiner, "Mixture autoregressive hidden Markov models for speech signals," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, pp. 1404–1412, 1985.
[57] H. Bourlard, N. Morgan, and C. Wooters, "Conectionist approaches to the use of Markov models for speech recognition," in *Advances in Neural Information Processing Systems* (R. P. Lippmann, J. E. Moody, and D. S. Touretzky, eds.), vol. 3, pp. 213–219, San Mateo CA: Morgan Kaufmann, 1991.
[58] H. Bourlard and N. Morgan, *Connectionist Speech Recognition—A Hybrid Approach*. Kluwer Academic, 1993.
[59] L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Tokyo), pp. 49–52, 1986.
[60] J. S. Bridle, "Alpha-nets: a recurrent neural network architecture with a hidden Markov model interpretation," *Speech Communication*, vol. 9, pp. 83–92, 1990.
[61] J. S. Bridle and L. Dodd, "An Alphanet approach to optimising input transformations for continuous speech recognition," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Toronto), pp. 277–280, 1991.
[62] L. T. Niles, "Timit phoneme recognition using an HMM-derived recurrent neural network," in *Proceedings European Conference on Speech Communication and Technology*, (Genova, Italy), pp. 559–562, 1991.
[63] Y. Bengio, R. de Mori, G. Flammia, and R. Kompe, "Global optimization of a neural network–hidden Markov model hybrid," *IEEE Transactions on Neural Networks*, vol. 3, pp. 252–259, 1992.
[64] H. Bourlard and N. Morgan, "A continuous speech recognition system embedding MLP into HMM," in *Advances in Neural Information Processing Systems* (D. S. Touretzky, ed.), vol. 2, pp. 413–416, San Mateo CA: Morgan Kaufmann, 1990.
[65] M. A. Franzini, K. F. Lee, and A. Waibel, "Connectionist Viterbi training: a new hybrid method for continuous speech recognition," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Albuquerque), pp. 425–428, 1990.
[66] P. F. Brown, *The Acoustic-Modelling Problem in Automatic Speech Recognition*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1987.
[67] R. Schwartz, Y. Chow, O. Kimball, S. Roucous, M. Krasner, and J. Makhoul, "Context-dependent modelling for acoustic-phonetic recognition of continuous speech," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, (Tampa FL), pp. 1205–1208, 1985.
[68] N. Morgan and H. Bourlard, "Factoring networks by a statistical method," *Neural Computation*, vol. 4, pp. 835–838, 1992.
[69] H. Bourlard, N. Morgan, C. Wooters, and S. Renals, "CDNN: A context dependent neural network for continuous speech recognition," in *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, (San Francisco), pp. 349–352, 1992.
[70] M. Cohen, H. Franco, N. Morgan, D. Rumelhart, and V. Abrash, "Hybrid neural network/hidden Markov model continuous-speech recognition," in *Proceedings International Conference on Spoken Language Processing*, (Banff), pp. 915–918, 1992.
[71] M. Cohen, H. Franco, N. Morgan, D. Rumelhart, and V. Abrash, "Context-dependent multiple distribution phone modelling using MLPs," in *Advances in Neural Information Processing Systems* (S. J. Hanson, J. D. Cowan, and C. L. Giles, eds.), vol. 5, San Mateo CA: Morgan Kaufmann, 1993.
[72] S. Furui, "Speaker independent isolated word recognition using dynamic features of speech spectrum," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 34, pp. 52–59, 1986.
[73] N. Morgan, J. Beck, P. Kohn, J. Bilmes, E. Allman, and J. Beer, "The Ring Array Processor (RAP): A multiprocessing peripheral for connectionist applications," *Journal of Parallel and Distributed Computing*, vol. 14, pp. 248–259, 1992.