HIDDEN NEURAL NETWORKS: A FRAMEWORK FOR HMM/NN HYBRIDS

Søren Kamaric Riis[1] *Anders Krogh*[2,*]

[1]Dept. of Mathematical Modelling, Technical University of Denmark, 2800 Lyngby, DK (`sr@imm.dtu.dk`)
[2]The Sanger Centre, Wellcome Trust Genome Campus, Hinxton, Cambs, CB10 1SA, GB (`krogh@cbs.dtu.dk`)

ABSTRACT

This paper presents a general framework for hybrids of Hidden Markov models (HMM) and neural networks (NN). In the new framework called Hidden Neural Networks (HNN) the usual HMM probability parameters are replaced by neural network outputs. To ensure a probabilistic interpretation the HNN is normalized globally as opposed to the local normalization enforced on parameters in standard HMMs. Furthermore, all parameters in the HNN are estimated simultaneously according to the discriminative conditional maximum likelihood (CML) criterion. The HNNs show clear performance gains compared to standard HMMs on TIMIT continuous speech recognition benchmarks. On the task of recognizing five broad phoneme classes an accuracy of 84% is obtained compared to 76% for a standard HMM. Additionally, we report a preliminary result of 69% accuracy on the TIMIT 39 phoneme task.

1. INTRODUCTION

Among speech research scientists it is widely believed that HMMs are one of the best and most successful modelling approaches for acoustic events in speech recognition. However, common assumptions like state conditional observation independence and time independent transition probabilities limit the classification abilities of HMMs. These assumptions can be relaxed by introducing neural networks in the HMM framework and recently several approaches for HMM/NN hybrids have been proposed, see *e.g.* [1, 2, 3, 4, 5, 6, 7, 8]. In [6] a standard feedforward NN is trained separately to estimate phoneme posterior probabilities. These posteriors are scaled with observed phoneme frequencies and replace the usual emission densities in a HMM during decoding. A similar approach is taken in [7], but here a recurrent NN is used. Instead of training the HMM and the NN separately several authors have proposed architectures where all parameters are estimated simultaneously. In some hybrids [2, 4, 5] a feedforward NN or recurrent NN [8] performs an adaptive input transformation of the observation vectors. Thus, the network outputs are used as new observation vectors in a continuous density HMM and simultaneous estimation of all parameters is performed by backpropagating errors calculated by the HMM into the NN.

Our approach is somewhat similar to the idea of adaptive input transformations, but instead of retaining the computationally expensive mixture densities we propose to *replace* some or all HMM probability parameters by the output of small state specific neural networks. Simultaneous estimation of all parameters can then be performed similar to what is done for adaptive input transformations. A proper probabilistic interpretation is guaranteed by normalizing the model globally as opposed to the often approximate local normalization enforced in many existing hybrids, *e.g.*, using softmax on neural network outputs or using prior scaled network outputs. However, calculating the normalization term can be avoided if the discriminative CML criterion is used for training.

2. THE HNN

In the HNN it is possible to assign up to two networks to each state: 1) a *match network* estimating the probability that the current observation matches a given state and 2) a *transition network* that estimates transition probabilities conditioned on observations. This is very similar to the IOHMM architecture [3] although training and decoding of the IOHMM differs somewhat from that of the HNN. One of the two types of networks in each HNN state can be omitted and replaced by standard HMM parameters. In fact all sorts of combinations with standard HMM states are possible. Instead of using state specific match networks we could use one big network with the same number of outputs as there are states in the HNN. This would correspond to tying weights between input and hidden units in all the match networks.

More formally the HMM emission probability $\phi_i(x_l)$ of observation vector x_l in state i is replaced by a match network $\phi_i(s_l; w^i)$, which is parameterized by weights w^i with input s_l and only one output. The network input s_l corresponding to x_l will usually be a window of context around x_l, *e.g.*, a symmetrical context window of $2K+1$ observation vectors, $x_{l-K}, x_{l-K+1}, \ldots, x_{l+K}$. It can however be any sort of information related to x_l or even the observation sequence in general. Similarly, the probability θ_{ij} of a transition from state i to j is replaced by the output of a transition network $\theta_{ij}(s_l; u^i)$, which is parameterized by weights u^i. The transition network assigned to state i has $\mathcal{J}_i$ outputs, where $\mathcal{J}_i$ is the number of (non-zero) transitions *from* state i. The neural networks in the HNN can be standard feedforward networks or recurrent networks. In fact, they need not even be neural networks — they can be any smooth mapping defined on the space of observation feature vectors.

In complete analogy with the likelihood $p(x|\mathcal{M})$ of a HMM for observation sequence $x = x_1, \ldots, x_L$, we define the quantity

$$q(x|\mathcal{M}) = \sum_{\pi} q(x, \pi|\mathcal{M}) \tag{1}$$

[*]Present address: Center for Biological Sequence Analysis, Technical University of Denmark, 2800 Lyngby, Denmark

with

$$q(x,\pi|\mathcal{M}) = \prod_{l=1}^{L} \theta_{\pi_{l-1}\pi_l}(s_{l-1}; u^{\pi_{l-1}})\phi_{\pi_l}(s_l; w^{\pi_l}) \quad (2)$$

where $\mathcal{M}$ denotes the whole model, *i.e.*, *all* parameters, and the state sequence $\pi = \pi_1, \ldots, \pi_L$ is a particular path through the model. We define $\pi_0 = 0$ and $\theta_{0i}(s_0; u^0)$ is the probability of initiating a path in state i. s_0 is the context we choose to associate with the beginning of the sequence. The probabilistic interpretation is ensured by explicit normalization of q,

$$p(x|\mathcal{M}) = \frac{q(x|\mathcal{M})}{\int_{x' \in \mathcal{X}} q(x'|\mathcal{M})dx'} \quad (3)$$

The integration in the denominator is taken over the space of observation vector sequences $\mathcal{X}$. For the HNN it is generally not possible to calculate the normalization in (3), since this would require knowledge of the network outputs for all possible inputs. However, as shown below, we do not need to compute the normalization term.

Note that the likelihood definition (3) is also applicable to conventional HMMs. Actually, a discrete HMM with non-normalizing parameters is equivalent to the so called Boltzmann Chain introduced in [9].

Apart from making this model elegant from a mathematical and computational point of view, we also believe that it may be beneficial to give up local normalization of parameters even for standard HMMs. In fact, non-normalizing parameters are already used frequently to increase speech recognition accuracy by introducing so-called "transition biases" and "stream exponents", see *e.g.* [7, 10].

2.1. Training and decoding

Assume that the *complete labeling* is available, *i.e.*, that each observation x_l has an associated label y_l corresponding to the class to which it belongs. In speech recognition the classes could be distinct phonemes. Similarly, assume that each state in the HNN is assigned a class label. To maximize the prediction accuracy we choose parameters so as to maximize the conditional likelihood of the observed labeling[1] $y = y_1, \ldots, y_L$,

$$P(y|x,\mathcal{M}) = \frac{p(x,y|\mathcal{M})}{p(x|\mathcal{M})} \quad (4)$$

as we have previously proposed in [11]. Maximizing (4) is known as Conditional Maximum Likelihood estimation (CML) and is equivalent to Maximum Mutual Information estimation (MMI) [12, 13] if the language model is fixed during training. $p(x,y|\mathcal{M})$ is calculated as a sum over all paths consistent with the labeling, *i.e.*, if observation l is labeled f only paths in which the l-th state has label f are allowed. If the set of these consistent paths is called $\mathcal{A}(y)$ we have,

$$p(x,y|\mathcal{M}) = \frac{q(x,y|\mathcal{M})}{\sum_{y'} \int_{x' \in \mathcal{X}} q(x',y'|\mathcal{M})dx'} \quad (5)$$

with,

$$q(x,y|\mathcal{M}) = \sum_{\pi \in \mathcal{A}(y)} q(x,\pi|\mathcal{M}). \quad (6)$$

[1]The extension to multiple training sequences is straightforward by assuming that training sequences are independent.

Since $\sum_{y'} q(x',y'|\mathcal{M}) = q(x'|\mathcal{M})$ the normalization is the same in (3) and (5), so

$$P(y|x,\mathcal{M}) = \frac{q(x,y|\mathcal{M})}{q(x|\mathcal{M})} \quad (7)$$

and the normalizing factor has conveniently disappeared. Both $q(x|\mathcal{M})$ and $q(x,y|\mathcal{M})$ can be calculated by a straightforward extension of the forward algorithm, see *e.g.* [11].

Using the above framework for phoneme recognition requires that the phoneme segmentation (complete labels) of the spoken utterance is known. However, in continuous speech recognition the desired output of the recognizer is not the phoneme label corresponding to each speech frame (*complete labeling*), but rather the phoneme transcription (*incomplete labeling*) [13]. Similar to procedures for standard HMMs (*e.g.* [5, 10]) we can build a "transcription model" by concatenating phoneme submodels according to the observed phoneme transcription of a given training utterance. Analogous to (4) and (7) the goal is now to maximize,

$$P(w|x,\mathcal{M}) = \frac{p(x,w|\mathcal{M})}{p(x|\mathcal{M})} = \frac{p(x|\mathcal{M}_w)}{p(x|\mathcal{M})} = \frac{q(x|\mathcal{M}_w)}{q(x|\mathcal{M})} \quad (8)$$

where w is the phoneme transcription corresponding to x and $\mathcal{M}_w$ is the transcription model. In this work training using the complete or incomplete labeling is called *complete* and *incomplete label training* respectively.

To maximize (7) or (8) we use stochastic online gradient ascent augmented by a momentum term, where the parameter update is performed after each observation sequence. We found that online gradient ascent with an adaptive stepsize yields considerably faster convergence than batch training (updating after presenting the entire training set). Calculating the derivative of $\log P(y|x,\mathcal{M})$ w.r.t. a weight in the match or transition networks yields backpropagation training of the neural networks based on an error signal calculated by the forward–backward algorithm, see *e.g.* [2, 5]. Thus, for each sentence a forward-backward pass is followed by backpropagation training of the neural networks.

In agreement with [5] we have found that Viterbi decoding does not perform well for discriminatively trained models. A probable reason for this is that the Viterbi path through the model need not correspond to the optimal transcription. Ideally the decoder should output the labeling that maximizes the full likelihood of the model. In this work we use the computationally efficient full likelihood based N-best decoder proposed in [14]. This decoder allows multiple active partial transcriptions in each state during a Viterbi style decoding. For computational reasons it is usually necessary to limit the number of active partial transcriptions in each state. We use an upper limit of 10 active partial transcriptions and only the most likely transcription is considered after decoding.

3. GENERAL EXPERIMENTAL SETUP

Initially the HNN has been evaluated on the task of recognizing five broad phoneme classes in the TIMIT database: Vowels (V), Consonants (C), Nasals (N), Liquids (L) and Silence (S). We use one sentence from each of the 462 speakers in the TIMIT training set for training, and the results are reported for the recommended TIMIT core test set. An independent crossvalidation set is used for selecting the best performing model after training.

The preprocessor outputs an observation vector every 10ms consisting of 26 features: 12 mel scaled cepstral coefficients, 1 log energy coefficient and the corresponding delta coefficients. These vectors are normalized to zero mean and unit variance in order to speed up training of the HNN.

Each of the five classes are modelled by a simple left-to-right three state model. The last state in any submodel is fully connected to the first state of all other submodels. Since we did not observe any improvements using transition networks all models use standard HMM transition probabilities. Note however, that transitions between submodels are also trained discriminatively when using the CML criterion as opposed to the MMI criterion.

Our baseline system is a standard discrete HMM using a codebook of 256 prototype vectors (3856 free parameters). In the HNN we replace the emission distribution by fully connected match networks with a symmetric input window of $2K+1$ observation vectors and a sigmoid output function.

CML training is performed using a maximum of 100 online gradient ascent epochs and the Baum-Welch reestimation algorithm is used for ML training. The best model found within 30 epochs of complete label training is used as initial model for incomplete label training. Training times are less than one day on a fast workstation for all models evaluated on the broad class problem.

3.1. Baseline results

In table 1 the results for complete label training are shown for the baseline system. For the ML trained model it is observed that Viterbi and N-best decoding yields approximately the same accuracy[2]. However, for the CML estimated model a considerably higher accuracy is obtained by the N-best decoder. Probably this is because the CML estimated models are trained so as to maximize the probability of the labeling, which need not correspond to the most probable path found by the Viterbi algorithm.

Table 1 shows that the difference between complete and incomplete label training is negligible for the ML estimated models, but significant for the CML estimated models. The reason for this is that the CML criterion is very sensitive to mislabelings, because it is dominated by those training sequences that have an unlikely labeling. Since the phoneme segmentation in TIMIT is set by hand it is very likely to contain mislabelings which degrade performance of the complete label trained model. Thus, an accuracy of 81.3% is obtained using incomplete label CML training compared to 79.0% for complete label CML training. For comparison the ML estimated discrete HMM only reaches 76.1%. In [5] an accuracy of 69.3% is reported for a ML trained continuous density HMM with a mixture of six diagonal covariance gaussians per state. Thus, for approximately the same number of parameters the ML trained discrete HMM outperforms the continuous density HMM by more than 6%. For a MMI trained model with a single diagonal covariance gaussian per state they report an accuracy of 72.4%, compared to our 81.3% for the CML trained discrete HMM.

3.2. HNN Results

The match networks in the HNN are initially trained by a few iterations of standard backpropagation to classify the observations into the five broad classes. This speeds up training of the HNN considerably and the models are less prone to getting stuck in local minima. All HNNs are estimated using incomplete label CML training.

[2] $\%Acc = 100\% - \%Ins - \%Del - \%Sub$, where $\%Ins$, $\%Del$ and $\%Sub$ denote the percentage of insertions, deletions and substitutions used for aligning the observed and the predicted transcription. The NIST standard scoring package "sclite" ver. 1.1 is used in all experiments.

Table 1. *Discrete HMM recognition accuracies.*

Complete labels	Vit	N-Best
ML	75.9	76.1
CML	76.6	79.0
Incomplete labels	**Vit**	**N-Best**
ML	75.8	75.2
CML	78.4	*81.3*

Table 2. *HNN recognition accuracies. For $K = 1$ a context of one left and one right frame is used,* i.e., *a total of three frames is used.*

0 hidden units	#Parms	N-Best
$K = 0$	436	80.8
$K = 1$	1216	81.7
10 hidden units	**#Parms**	**N-Best**
$K = 0$	4246	*84.0*
$K = 1$	12046	83.6

Even though the HNN with zero hidden units and no context ($K = 0$) contains almost ten times less parameters than the baseline system it achieves a comparable recognition accuracy of 80.8%, see table 2. For a context of one left and right frame ($K = 1$) the HNN outperforms the CML estimated HMM. No further improvement was observed for contexts larger than $K = 1$. It is interesting to note that the match networks in the HNN without hidden units actually just implements linear weighted sums of input features (passed through a sigmoid output function). Adding hidden units to the HNN drastically increases performance even if no context is used, see table 2. Thus, for approximately the same number of parameters as used in the baseline system the HNN with 10 hidden units and no context ($K = 0$) obtains 84.0% recognition accuracy. Context degrades test set performance slightly for the HNN with 10 hidden units. We believe this is due to overfitting, because of the large number of parameters in these models. In [5] a feedforward NN is used as a global adaptive input transformation to a continuous density HMM with a single diagonal covariance gaussian per state. This hybrid is trained by MMI and N-best decoding gives an accuracy of 78.5%. The result has later been improved to 81.3% by using a linear transformation instead of the NN and by using multiple CML training passes [4].

Using the forward-backward algorithm it is straightforward to calculate the posterior probability $P(\pi_l = i|x, \mathcal{M})$ of occupying state i at time l [13]. For a test sentence (si2183: "Books are for schnooks") the state posterior probabilities provided by the HNN with 10 hidden units and no context are shown for the the consonant submodel states in fig. 1. The posteriors for the "begin" and "end" states are only large at the class boundaries, whereas the posteriors for the "middle" state is large only between these boundaries. Hence, the model is very good at discriminating between different classes. This is also verified in fig. 2 showing that more than 50% of the frames have a winning-label posterior probability[3] $P(y_l^*|x, \mathcal{M})$ larger than or equal to 0.9. This

[3] Posterior label probabilities for each frame are obtained by summing state posteriors at time l for states that carry the same label. The winning-label y_l^* at time l is defined by the largest label posterior at time l.

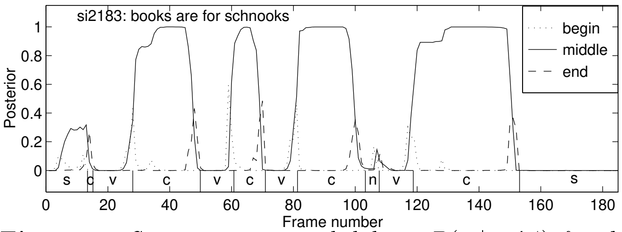

Figure 1. *State posterior probabilities $P(\pi_l|x, \mathcal{M})$ for the three states in the consonant submodel for test sentence "si2183". The broad class segmentation is shown at the bottom of the plot.*

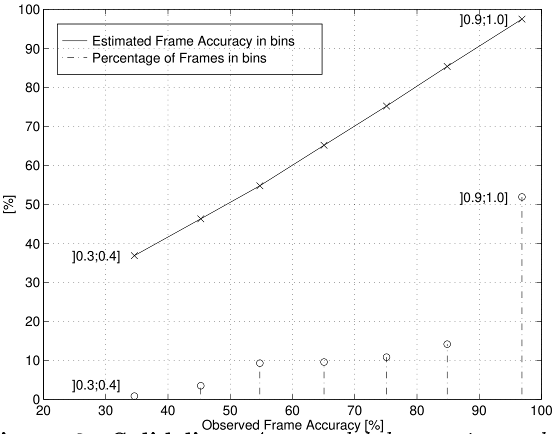

Figure 2. **Solid line:** *Average label posterior probability of winning class $\bar{P}(y_l^*|x, \mathcal{M})$ ranked into 7 equally spaced bins and plotted as function of the observed frame accuracy in these bins.* **Dashed line:** *Percentage of frames falling into the 7 bins located at the corresponding observed frame accuracies.*

corresponds to an observed frame recognition rate of more than 95%. Furthermore, there is an almost perfect correlation between the observed frame accuracy and the average label posterior of the winning class. Thus, the larger $P(y_l^*|x, \mathcal{M})$ the more confident is the prediction.

3.3. TIMIT 39 phoneme recognition

A set of experiments were carried out on recognizing the reduced 39 TIMIT phoneme set also considered in [4, 7, 10]. In these experiments we used the same preprocessor and submodel setup as in the broad class problem, but a total of 39 phoneme submodels were used. The full TIMIT training and testing set (except all "sa-sentences") were used in these experiments. For a HNN using no context, 10 hidden units in the match networks, standard transition probabilities and incomplete label training a preliminary test set result of 69% accuracy is obtained. This is comparable to results on the same task reported in the literature for standard context independent HMMs [10] and for a HMM using a global linear adaptive input transformation [4]. Currently one of the best results reported is 75% accuracy and is obtained by a HMM/recurrent NN hybrid [7].

4. CONCLUSION

It has been shown that the HNN combination of neural networks and a Hidden Markov model yields a better recognition accuracy than a standard HMM on TIMIT continuous phoneme recognition. In addition it was illustrated that the HNN provides very accurate estimates of phoneme posterior probabilities. The particular architecture introduced here is characterized by: 1) All parameters are trained discriminatively at the same time 2) Proper probability distributions are obtained by global normalization as opposed to local normalization, and 3) Large flexibility in that not all of the HMM parameters need to be replaced by neural networks. Future work includes further experiments on the 39 phoneme task, other optimization methods, *e.g.*, second order methods, and careful network design to reduce model complexity.

ACKNOWLEDGMENTS

The authors would like to thank Steve Renals and Finn T. Johansen for valuable comments and suggestions to this work. The Sanger Centre is supported by the Wellcome Trust.

REFERENCES

[1] P. Baldi and Y. Chauvin, "Hybrid Modeling, HMM/NN Architectures, and Protein Applications," *Neural Computation*, vol. 8, pp. 1541–65, 1996.

[2] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe, "Global optimization of a neural network-Hidden Markov model hybrid," *IEEE Trans. on Neural Networks*, vol. 3, no. 2, pp. 252–9, 1992.

[3] Y. Bengio and P. Frasconi, "An input output HMM architecture," in *Advances in Neural Information Processing Systems - 7*, pp. 427–34, Morgan-Kaufmann, 1995.

[4] F. Johansen, "Global optimisation of HMM input transformations," in *Proceedings of ICSLP'94*, vol. I, pp. 239–42, 1994.

[5] F. Johansen and M. Johnsen, "Non-linear input transformations for discriminative HMMs," in *Proceedings of ICASSP'94*, vol. I, pp. 225–28, 1994.

[6] S. Renals, N. Morgan, H. Bourlard, M. Cohen, and H. Franco, "Connectionist probability estimators in HMM speech recognition," *IEEE Trans. on Speech and Audio Proc.*, vol. 2, no. 1, pp. 161–74, 1994.

[7] A. Robinson, "An application of recurrent nets to phone probability estimation," *IEEE Trans. on Neural Networks*, vol. 5, pp. 298–305, 1994.

[8] V. Valtchev, S. Kapadia, and S. Young, "Recurrent input transformations for Hidden Markov models," in *Proceedings of ICASSP'93*, pp. 287–90, 1993.

[9] L. Saul and M. Jordan, "Boltzman Chains and Hidden Markov models," in *Advances in Neural Information Processing Systems - 7*, pp. 435–42, Morgan-Kaufmann, 1995.

[10] S. Kapadia, V. Valtchev, and S. Young, "MMI training for continuous phoneme recognition on the TIMIT database," in *Proceedings of ICASSP'93*, vol. 2, pp. 491–4, 1993.

[11] A. Krogh, "Hidden Markov models for labeled sequences," in *Proceedings of the 12th ICPR'94*, pp. 140–4, 1994.

[12] L. Bahl, P. Brown, P. de Souza, and R. Mercer, "Maximum mutual information estimation of Hidden Markov model parameters for speech recognition," in *Proceedings of ICASSP'86*, pp. 49–52, 1986.

[13] B. Juang and L. Rabiner, "Hidden Markov models for speech recognition," *Technometrics*, vol. 33, no. 3, pp. 251–72, 1991.

[14] R. Schwarz and Y.-L. Chow, "The N-best algorithm: An efficient and exact procedure for finding the N most likely hypotheses," in *Proceedings of ICASSP'90*, pp. 81–84, 1990.