NEURAL NETWORK ACOUSTIC MODEL WITH DECISION TREE CLUSTERED TRIPHONES

Tomáš Pavelka and Pavel Král

Dept. of Computer Science and Engineering
University of West Bohemia
Plzeň, Czech Republic
{tpavelka,pkral}@kiv.zcu.cz

ABSTRACT

This article tries to compare the performance of neural network and Gaussian mixture acoustic models (GMMs). We argue that using a multi layer perceptron as an emission probability estimator in hidden Markov model based automatic speech recognition can lead to better results than when the more traditional Gaussian mixtures are applied. We present a solution on how to model triphone phonetic units with neural networks and we show that this also leads to better performance in comparison with GMMs. The superior performance of the neural networks comes at a cost of extremely long training times.

1. INTRODUCTION

The most widely used mathematical framework for automatic speech recognition are the continuous density hidden Markov Models (CDHMMS). Despite of their success these models make various assumptions that are not true for speech data (see [1]). There are attempts to solve some of the drawbacks of the CDHMM paradigm by employing neural networks. The research into the so called *hybrid systems* (see e.g. [2], [3]) has shown that it can be advantageous to use neural networks (instead of the more traditional Gaussian mixtures) as emission probability estimators for hidden Markov model based automatic speech recognizers. Our results presented in [4] demonstrate that there are two main benefits:

- The application of neural networks to emission probability estimation does not place any constraints on the form of its inputs (as opposed to GMM models with diagonal covariance matrices which add delta and acceleration coefficients to the input vector because the elements of the final composed vector are loosely uncorrelated). This is usually exploited by presenting several subsequent speech frames to the input of the neural network and thus allowing the network to "see" a larger context of the speech signal.
- When compared to Gaussian mixture based acoustic models the neural networks need less trainable parameters to achieve similar or better recognition accuracy. As we will show, this can lead to faster recognition speed.

While the above stated can be said about context independent phonetic units (monophones) the experiments presented in [5] make clear that much better results can be gained with adding context dependency (e.g. by using triphones). This article's aim is to explore the possibilities of modeling triphone phonetic units by a neural network, namely the multi layer perceptron.

2. SPEECH CORPORA

All the available speech data is in Czech language, recorded in quiet environment at 16 kHz sampling rate and 16 bits per sample. The corpora are divided into sentences; each sentence is stored in a separate file. The training set consists of three parts:

- **Train Schedule Queries**. This corpus consists of questions about train schedules and related information. An example of such question would be "When does the train for Plzeň leave".
- **LAC-HP Chess**. Stands for LASER Audiocorpus High Precision. The corpus was recorded in an audio studio; all the audio files have been verified during the recording. This set consists of voice commands for a chess game. The commands could be either chess moves (e.g. "Move the king to b5") or miscellaneous commands like "I want to start a new game".
- **LAC-HP Phonetic**. This is a set of nonsense sentences with words containing infrequent phonetic units.

This work was supported by grant no. 2C06009 Cot-Sewing.

The testing corpus for the train schedules is a subset taken out from the original corpus. The testing corpus for the chess game contains only move commands because we have found out that other commands can skew the recognition results (the move commands are much harder to recognize). This means that if other commands are present in the training data the resulting accuracy is highly correlated with moves / other commands ratio. Table 1 shows statistics for all the speech data used in our experiments.

Training Corpus	**Vocabulary size [words]**	**Total Length [hours]**
Train Schedules	1490	11:28:06
LAC-HP Chess	96	1:51:50
LAC-HP Phonetic	115	1:33:02
Testing Corpus	**Vocabulary size [words]**	**Total Length [hours]**
Train Schedules	1490	0:31:34
Chess Moves	96	1:18:28

Table 1. Training and testing corpora

3. GAUSSIAN MIXTURE ACOUSTIC MODELS

In order to make a comparison the Gaussian mixture models (GMMs) were trained by the Hidden Markov Toolkit (HTK, [6]). Only models with diagonal covariance matrices were tested. The parameter estimation was done by a flat start embedded training which only requires the phonetic transcriptions of the training utterances to be available. On the other hand the neural network needs exact locations of phonetic units in the training data. This leads to the second reason for having a set of trained GMM models: the GMM based recognizer can be used to label the training data for the neural network (by the means of forced Viterbi alignment).

In the case of the GMMs the whole set consists of 36 (35 context independent units + silence) phonetic unit models. Each phonetic unit is a three state HMM, each state has its own mixture of Gaussians. The training starts with one Gaussian per state. In each training cycle embedded reestimation is performed four times (our tests show that the error decrease after four iterations is negligible). After the cycle is completed the number of Gaussians for each state is increased twofold. We have trained models with up to 256 Gaussians per state.

The training of triphone GMM models is described in detail in [5]. The process is similar to the training of the monophones, the difference is that after the models with single Gaussian per state are trained, the decision tree clustering (see [7] for details on decision tree clustering) of all the states is performed. The result is that the triphone models which do not have sufficient amount of training data available are tied together with all the other models in their respective clusters. The clustering provides a mapping between the logical models (i.e. any triphone) and the physical models (actual Gaussian mixtures). The number of physical models is much lower than the total number of logical models. The convenience of the decision tree clustering process is in its ability to assign physical models even to triphones that were not present in the training data.

4. NEURAL NETWORK ACOUSTIC MODELS

There are various neural network architectures that have been successfully tested on speech recognition but the most widely used architecture is the multi layer perceptron (MLP). It has been proved that a three layer[1] perceptron can approximate any continuous function given a sufficient number of neurons in the hidden layer. A proof also exists (see e.g. [2]) that says that if an MLP is trained with summed squared error or a similar criterion as a 1-of-N classifier then the activations of the output neurons can be treated as class posterior probabilities. All networks discussed in this paper are multi layer perceptrons.

Note that hidden Markov models work with likelihoods $p(\mathrm{input}|\mathrm{class})$ instead of the class posteriors $p(\mathrm{class}|\mathrm{input})$ that we get on the output of the neural network. These can be converted using the Bayes theorem:

$$P(\mathrm{input}|\mathrm{class}) = \frac{P(\mathrm{class}|\mathrm{input}) \cdot P(\mathrm{input})}{P(\mathrm{class})} \quad . \qquad (1)$$

Since the probability of an input $P(\mathrm{input})$ is the same for all HMM states examined in a given frame it can be discarded from the equation without affecting the result. Whether this is actually beneficial for speech recognition accuracy will be discussed in section 6.

Unlike the training of GMMs the training of a multi layer perceptron requires the training data to be labeled, i.e. for each training input vector the desired output vector must be known. Since we have a trained GMM based recognizer it is possible to label the data by employing *forced Viterbi alignment* which works in the following way: First a HMM of the training utterance is constructed by concatenating all phonetic unit HMMs that correspond to the phonetic units found in the phonetic transcription of the utterance. After that the Viterbi algorithm is run resulting in the state sequence with the highest probability. This way it is known which state a given frame belongs to and from this information a phonetic unit that the frame belongs to can be found.

The incremental version of the backpropagation algorithm is used for training. Incremental means that the weights

[1] Because there is no clear agreement on whether a layer means a layer of neurons or a layer of weights it should be stated that we count the number of neuron layers. This means that our neural network has three layers of neurons and two layers of weights.

of the network are adjusted immediately after a training vector is processed (as opposed to batch training where the weights are adjusted only after all the training vectors have been processed). In order for the incremental backpropagation to converge it is necessary to present the training vectors in random order. There is a problem with data size because the training data is so large it would not fit into memory. To accommodate for this problem the data preparation is done in the following way: First the list of input files is shuffled. Second the list is split into parts (so that each part of training data is about 1 GB in size). After that the training vectors in each part are shuffled.

An alternative error criterion is used during the backpropagation training instead of the more usual summed squared error: the *cross entropy error*. If E^p is the error on training vector p, $Y^p = y_1^p, y_2^p, \ldots, y_N^p$ is the vector of network's outputs and $D^p = d_1^p, d_2^p, \ldots, d_n^p$ is the vector of desired network outputs then the cross entropy error can be computed as

$$E^p = -\sum_{o=1}^{N_o} (d_o^p \log y_o^p) + (1 - d_o^p)\log(1 - y_o^p) \ . \qquad (2)$$

We have found (similarly to what has been reported in [2]) that using this error criterion leads to faster convergence of both the training and the validation error.

In GMM based acoustic models each phonetic unit is modeled by a three state HMM where each state has its own mixture of Gaussians. In neural networks we use a so called *state duplication*: there is only one neuron representing a phonetic unit but the phonetic unit HMM has again three states. All these states share the same emission probability computed by the neuron. We have found that using duplicated three state phonetic unit HMMs instead of single state HMMs significantly increases recognition accuracy.

5. CONTEXT DEPENDENCY IN NEURAL NETWORK ACOUSTIC MODELS

Context dependent neural network acoustic models suffer from the same problems as their GMM counterparts, namely the sparse data problem and the rising computational costs. There are some works [8, 9] trying to solve this by factoring out the context probability and combining context probabilities and context independent probabilities computed by separate neural networks. Our approach was the same as in the case of the GMM models, i.e. to apply decision tree clustering to reduce the total amount of physical context dependent models.

To prepare the training data for the triphone neural network it was necessary to have a GMM classifier which would label the training data. After the initial monophones were converted to triphones based on the training data there was a total of 2535 state models. After the decision tree clustering the number of physical state models was reduced to 517 due to parameter tying. These models were further trained and the number of Gaussians for each state was increased to 32. This system was then used to label the training data for the neural network. The names of the 517 physical state models were converted to indexes and the correct model index for each frame was produced by forced Viterbi alignment. The desired output vectors consisted of all zeros with a single one at the index which corresponds to the label. The network can be trained in the same way as monophone network. The decision trees can be used to find the correct neuron for any given triphone.

6. EXPERIMENTAL RESULTS

Even though the training of the GMM models was done by the HTK software the testing of both the GMM and MLP acoustic models was carried out with the JLASER [10] recognizer. For both acoustic models 13 Mel-frequency cepstral coefficients (MFCCs) served as input. In the case of GMM models these were augmented by the delta and acceleration coefficients computed in the same way as in HTK. In the case of the MLP acoustic model MFCC coefficients from nine consecutive speech frames were used as the input for the network (altogether there were 117 input neurons).

There was a grammar representing all the possible utterances in the chess moves test corpus but the tests with the train schedule corpus were run without any language model. We do not consider the lack of language model to be a problem since our main goal is to compare the two kinds of acoustic models.

For all tests pruning was performed during the decoding phase. For the train schedule corpus a word insertion penalty was applied. Both the pruning threshold and the word insertion penalty were tuned for each acoustic model in order to achieve the highest possible speed while maintaining the highest recognition accuracy[2]. Decoding with pruning means that the emission probabilities are usually not needed only for all phonetic units (this is especially true for triphones). Some computation can be avoided by computing only those emission probabilities that are requested by the decoding algorithm. This is quite straightforward in the case of GMM models. In the case of neural networks the activations of all the hidden neurons need to be computed for every speech frame. But the computation of the output layer neuron activations can be delayed until those are requested by the decoder. For triphones more than 80% of the

[2] During the measurement of the recognition accuracy each recognized utterance is compared to its respective transcription. If N is the total number of words in the transcribed utterance, S the number of substitutions, D the number of deletions, and I the number of insertions needed to transform the recognized utterance into its transcription then the recognition accuracy is computed as $Acc = \frac{N-S-D-I}{N}$.

network's weights are between the hidden and the output layer so this can significantly speed up the recognition.

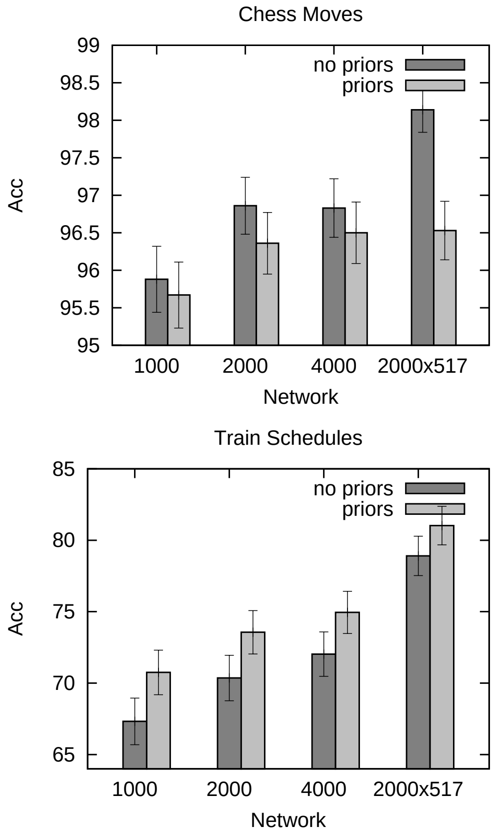

Fig. 1. Neural network acoustic model with and without division by phonetic unit priors. The error bars represent the confidence intervals for 99% probability (computed as the binomial proportion confidence interval).

As has been shown in the previous section the output neuron activations should be divided by the phonetic unit priors (these can be computed as relative frequencies) in order to transform the posterior probabilities into likelihoods. Figure 1 shows the results for different networks on both corpora. It can be seen that while the division by priors is beneficial for the train schedule corpus it is actually harmful in the case of the chess corpus. One fact about the difference between the two testing corpora is that while the distribution of phonemes in the train schedule corpus is similar in both the training and testing data, the distribution for the testing data for chess (only move commands) is different. To test whether this can play a role we have carried a second experiment where the priors were computed directly on the testing data, but the results were almost exactly the same as in the case of priors computed on the training data. In further tests all results for the chess corpus are obtained without the division by priors. The division by priors is only performed for the train schedule corpus.

MLP		**GMM**	
Model	Parameters	Model	parameters
1000x36	153000	mono32mix	269568
2000x36	306000	mono64mix	539136
4000x36	612000	mono128mix	1078272
2000x517	1268000	triph32mix	1290432

Table 2. The number of trainable parameters for different acoustic models.

In order to compare the two kinds of acoustic models tests were performed with models with different numbers of trainable parameters. The general idea is that better recognition accuracy can be achieved by increasing the number of trainable parameters at the expense of recognition speed. In the case of GMMs the trainable parameters can be increased by increasing the number of Gaussians in each mixture. For neural networks the number of hidden neurons can be increased. The results are displayed in Figure 2. The neural networks are denoted by the numbers of hidden neurons and output neurons. For example 2000x36 represents network with 2000 hidden neurons and 36 output neurons representing monophones. The triphone neural networks have 517 output neurons. The GMM models are denoted by the number of Gaussians in a mixture and an indication whether they are monophone or triphone models.

Besides showing the recognition accuracy the figure also shows the recognition speed measured as a percentage of real time processing power on a referential machine needed for the recognition. To compare how the recognition speed relates to the number of trainable parameters of each model see Table 2.

7. CONCLUSIONS

In our tests neural network acoustic models outperform Gaussian mixture models (with diagonal covariance matrices) in both recognition accuracy and recognition speed. It should be noted that the confidence intervals for triphone models overlap so the better accuracy of neural networks may not be statistically significant. It can be seen from Figure 2 that neural network models lead to similar or better recognition accuracy than their GMM counterparts while achieving higher recognition speed. This is related to the total number of parameters that can be adjusted during training (see Table 2): For a given recognition accuracy neural network models have lower number of trainable parameters. We conclude that the claim that neural networks need less trainable parameters to perform acoustic modeling and that this results in lower computational costs is true for monophone models. In the case of triphones the number of trainable parameters is close, but the neural network still outperforms the GMM

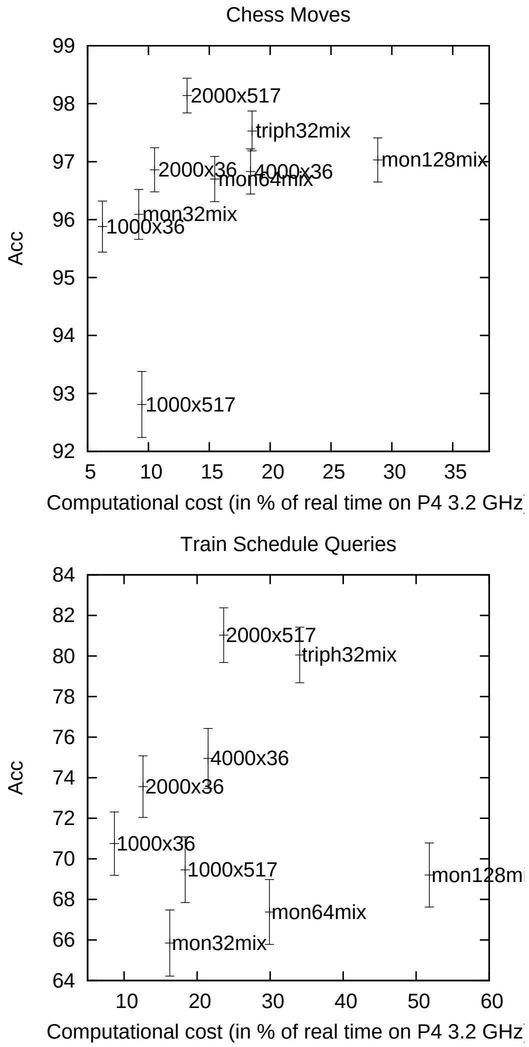

Fig. 2. Comparison of recognition speed and percentage of correct results for all tested acoustic models. The error bars represent the confidence intervals for 99% probability (computed as the binomial proportion confidence interval).

model in terms of computational costs[3].

But neural networks also have disadvantages. The most striking example can be observed in time needed to train the models: While the training of the triphone GMM model is finished in a matter of hours, the training of the best performing triphone neural network (denoted 2000x517 in Figure 2) took 22 days. We also suspect that there may be some kind of limit on the minimal number of hidden neurons, because the triphone network with only 1000 hidden neurons performs so badly. This may pose a problem if the training data increases and this leads to a larger number of output neurons.

[3] The computational costs are dependent on the pruning threshold and the absolute values of the threshold may have a different impact on different kinds of models (GMM models require a higher threshold). To compensate for this the thresholds were set in such way that both kinds of acoustic models had the same average amount of active (non-pruned) states during the decoding phase.

8. REFERENCES

[1] Lawrence R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, 1989.

[2] Herve Bourlard and Nelson Morgan, "Hybrid HMM/ANN systems for speech recognition: Overview and new research directions," in *Summer School on Neural Networks*, 1997, pp. 389–417.

[3] Joe Tebelskis, *Speech Recognition using Neural Networks*, Ph.D. thesis, Carnegie Mellon University, School of Computer Science, Pittsburgh, 1995.

[4] Tomáš Pavelka and Kamil Ekštein, "Neural network acoustic model for recognition of Czech speech," in *PhD Workshop Systems & Control*, 2005.

[5] Jan Hejtmánek and Tomáš Pavelka, "Use of context-dependent units in Czech speech," in *PhD Workshop 2007*, 2007.

[6] Steve Young et al., *The HTK Book (for HTK v. 3.3)*, Cambridge University Engineering Dept., 2002.

[7] Julian Odell, *The Use of Context in Large Vocabulary Speech Recognition*, Ph.D. thesis, Cambridge University Engineering Department, 1995.

[8] Michael Cohen, David Rummelhart, Nelson Morgan, Horacio Franco, Victor Abrash, and Yochai Konig, "Combining neural networks and hidden Markov models," in *DARPA Speech and Natural Language Workshop*, 1992.

[9] Hervé Bourlard, Nelson Morgan, Chuck Wooters, and Steve Renals, "CDNN: a context dependent neural network for continuous speech recognition," in *Acoustics, Speech, and Signal Processing ICASSP-92*, 1992, vol. 2, pp. 349 – 352.

[10] Tomáš Pavelka and Kamil Ekštein, "JLASER: An automatic speech recognizer written in Java," in *XII International Conference Speech and Computer (SPECOM'2007)*, 2007.