Gaussian Free Cluster Tree Construction using Deep Neural Network

Linchen Zhu, Kevin Kilgour, Sebastian Stüker, Alex Waibel

International Center for Advanced Communication Technologies - InterACT,
Institute of Anthropomatics and Robotics
Karlsruhe Institute of Technology (KIT), Germany

linchen.zhu@student.kit.edu,
{kevin.kilgour,sebastian.stueker,alexander.waibel}@kit.edu

Abstract

This paper presents a Gaussian free approach to constructing the cluster tree (CT) that context dependent acoustic models (CD-AM) depend on. In resent years deep neural networks (DNN) have supplanted Gaussian mixture models (GMM) as the default method for acoustic modeling (AM). DNN AMs have also been successfully used to flat start context independent (CI) AMs and generate alignments on which CTs can be trained. Those approaches however still required Gaussians to build their CTs. Our proposed Gaussian free CT algorithm eliminates this requirements and allows, for the first time, the flat start training of state of the art DNN AMs without the use of Gaussian. An evaluation on the IWSLT transcription task demonstrates the effectiveness of this approach.

Index Terms: speech recognition, cluster tree, deep neural networks, flat start, context dependent acoustic models

1. Introduction

Due to advances in deep neural networks (DNN), Hidden Markov Models that use DNNs to estimate their state emission probabilities have now become the dominent form of acoustic models in large vocabulary continuous speech recognition (LVCSR) [1]. Automatic speech recognition systems using HMM/DNN AMs have been shown to outperform HMMs that use Gaussiam Mixture Models (GMMs) for estimating their emission probabilities. Word error rates (WERs) are usually reduced by up to 30% relative [2, 3, 4]. The acoustic model of an ASR systems estimates the conditional probability that a certain word sequence $W = w_1, ..., w_n$ has produced a certain sequence of observed feature vectors $F = f_1, ..., f_T$, i.e. $P(F|W)$. Let s_t denote the state that an HMM is in at time t, and let X_t denote the feature vector being emitted at time t. Then, for every state i of an an HMM λ, one needs to calculate the emission probability of that state emitting the feature vector f_t at time t under the assumption that one already is in state i, i.e. $P(X_t = f_t|s_t = i, \lambda)$.

The choice of states in an HMM depends on which units of speech one wants to model. Most commly used are phonemes which are subdivided into, e.g., three, sub-states. In order to be able to build an HMM for an arbitrary word sequence, regardless of whether that word sequence was seen in the training data or not, HMM parameters, such as emission probabilities and transition probabilities, are shared across all states that belong to the same (sub-)phoneme. This kind of acoustic model is called a context-independent (CI) model and its modelling assumption is clearly wrong as phonemes will be pronounced differently depending on the phonetic context in which they are spoken. This lead to the use of polyphones, instead of phonemes as modeling units. A polyphone is defined as a phoneme in a certain phonetic context. Depending on how long the phonetic context is that one considers, we, e.g., speak of tri-phones (one phoneme to the left and right of the phoneme modelled is considered) or quinphones (two phonemes to the left and right are considered).

Given the currently available amounts of acoustic model training data, the number of possible tri-phone or quin-phone models is generally too large as to be able to robustly estimate their emission probability on the given training data. Also, the not uncommon case that a polyphone has not been seen in the training data at all requires a back-off mechanismen. Therefore, polyphones are normally clustered into groups of similarly pronounced polyphones, called generalized polyphones. These cluster then share a common set of HMM parameters (as in the CI case for phonemes), leading to context-dependent (CD) AMs. Often the clustering is being performed with the help of classification and regression trees that ask general properties of the phonetic context of a phoneme. As the questions can be applied to any sequence of phonemes, regardless of whether this sequence has been seen in training or not, these cluster trees also solve the problem of back-off for unseen polyphones, as they will always assign a generalized polyphone to any phoneme given its phonetic context [5].

The use of context dependent AMs is not only of advantage for HMM/GMM models, but also important for HMM/DNN models [6]. The clustering procedure for the CD AMs requires a distant measure for clusters of polyphones. One common distance measure is for example the weighted entropy distance calculated on the mixture weights of a semi-continuous HMM trained on the unclustered polyphones. As this procedure requires GMMs for obtaining the mixture weights on which the entropy distance is calculated, a normal training procedure for HMM/DNN models is to first train an HMM/GMM model, perform the clustering of generalized polyphones and only then to train the actual DNN for use in the recognition system.

Another prerequisite to DNN training is the alignment of feature vectors to states, usually done with the Viterbi algorithm. This is necessary due to the fact that training data is normally only aligned at an utterance level. Despite the superiority of HMM/DNN AMs most ASR system still seem to rely on HMM/GMM AMs to generate this alignment [1]. Senior et. al. [7] show that this does not have to be the case and demonstrate how to flat-start an HMM/DNN AM using a randomly initialized context independent HMM/DNN AM to generate an initial alignment. However, after successfully bootstrapping a CI HMM/DNN AM they *"accumulate sufficient statistics to model*

each context dependent state with a diagonal covariance Gaussian" in order to build the cluster tree. While their setup avoids the use of any HMM/GMM AMs they still train Gaussians during clustering making their setup not Gaussian free.

In this paper we show how a cluster tree can also be built without using Gaussians by adapting the entropy distance metric to work with the probability distribution generated by the output layer of a CI HMM/DNN AM. This then leads to the ability to ab able to perform truly GMM free training of an acoustic model.

Removing the reliance on Gaussians or GMMs simplifies the flat-start training of state of the art ASR systems. Labs and institutions that have no previous experience in building ASR systems now have one less tool that they have to implement if they wish to build a CD HMM/DNN AM.

The structure of this paper is as follows. After an overview of the related work in Section 2 a description of cluster trees is given in Section 3 where our proposed DNN based approach is compared to the baseline GMM based approach. Section 4 presents the ASR system into which the new DNN based cluster trees are integrated. Their effectiveness is evaluated in Section 5. The paper concludes with a summary in Section 6.

2. Related Work

As mentioned in the introduction [7] show how to flat-start a CI DNN AM. Their approach uses a randomly initialized context independent DNN AM to generated an alignment on a batch of data which is then used to train the DNN AM. After several iterations it converges. The cluster tree is built by first collecting each triphone seen in their training data modeling them using a diagonal covariance Gaussian. Using a list of linguistically motivated questions greedy top down splitting is performed to optimize the likelihood gain. They also experiment with various other input features for their cluster trees and show that cluster trees built using CI log posteriors outperform the baseline cluster tree.

[8] also show how a CI DNN can be bootstrapped with equally aligned training data. The classification accuracy of the CI DNNs can be improved by iteratively realigning the data and training new CI DNNs on the realignments. During the construction of the cluster tree they assume the posteriors of the CI DNN to be a Gaussian.

3. Cluster Trees

A language with 40 phonemes and 3 states per phoneme would require 192000 ($= 3 \times 40^3$) triphone states or over 300 million quinphone states. Many of these states are never seen in the training data and many are almost identical to each other. Clustering algorithms are used to group these triphone or quinphone states together and reduce the total number of states that have to be modeled. This directly affects the size of the DNN-AM's output layer.

We refer to a decision trees used to cluster the large number of possible polyphones into a manageable number of classes as a cluster tree. The basic procedure requires a set of yes/no questions that can be asked about phoneme like whether it is a vowel of if it is at a word boundary. An example cluster tree, where a set of triphones is split after asking the question *"is the previous phoneme a vowel"* is shown in figure 1.

The following steps describe how to build a cluster tree:

1. Go over the alignment and get statistics on all existing polyphones.

2. Begin with all polyphones that have the same center phone clustered into one cluster per HMM state (e.g. begin, middle, end).

3. Select a cluster to split. Each yes/no questions will split a cluster into two separate clusters. Only take questions into consideration that produce clusters of a minimum size. Find the *best* questions to pose.

4. Split to cluster using the *best* question and repeat from step 3 until all clusters that can be split are split.

5. Prune the tree back until it has the desired number of leaves.

The key point in this procedure is how to find the *best* question. So given two clusters A and B we require a distance metric $d(A, B)$ that is high for good splits and low for bad splits. A common metric is the weighted entropy distance between A and B. It is the entropy difference between both clusters being joined and them being separate. Let the number of occurrence of each cluster be n_A and n_B, then the weighted entropy distance can be defined as:

$$d(A, B) = (n_A + n_B)H(A \cup B) - n_A H(B) - n_A H(B) \tag{1}$$

where $H(A)$ is the entropy of cluster A, $H(B)$ is the entropy of cluster B and $H(A \cup B)$ is the entropy of the merged cluster. Using the equation

$$H(p) = \sum_{i=1}^{k} p(i) \log p(i) \tag{2}$$

we can compute the entropy of a k dimensional probability distributions. Furthermore given probability distributions (over the same probability space) p_A and p_B for both clusters the probability distribution of the merged cluster can be computed as:

$$p_{A \cup B}(i) = \frac{n_A \, p_A(i) + n_A \, p_B(i)}{n_A + n_B} \tag{3}$$

The required probability distributions can be defined in multiple ways. One method would be to train a single Gaussian on the training examples belonging to a particular cluster. In this case the probability space would be the feature space. Another approach uses the discrete mixture weights of a shared codebook of Gaussians as the feature space.

3.1. GMM based

The GMM based approach requires a fully continuous CI HMM/GMM system, where each phone (monophone) state is modeled using a weighted mixture of a set of Gaussians (codebook) trained specifically for that phone state. For all polyphone states appearing in the training data derived from the same monophone state a new GMM is trained by only learning the mixture weights and keeping the codebook of the monophone. This results in a semi continuous HMM.

In semi-continuous HMMs, although the emission distributions of HMM states are still modeled by GMMs, the emission distributions can be represented by just their mixture weights. Furthermore, the normalized mixture weights of polyphone states can be regarded as the probabilities of a discrete distribution. Each mixture weight can be interpreted as the a priori probability of a codeword (Gaussian). As a result, the similarities between clusters represented as GMMs can be measured using entropy distance via their mixture weights.
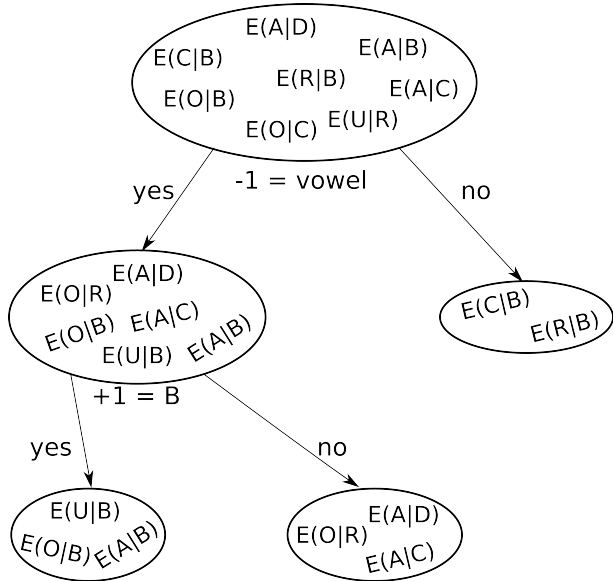
Figure 1: *An example cluster tree for the center phone E. The notation $E(A|D)$ indicated that the phone E is precedes by the phone A and followed by the phone D. The question "-1= vowel" asks if the preceding phone is a vowel.*

3.2. DNN based

Our novel DNN based clustering approach leverages the classification ability of a context independent DNN. For each input feature vector a discrete probability distribution of underlying generative monophone states can be calculated using a CI DNN.

A CI DNN is trained using the aligned phone states as targets. The softmax activation function of the final layer guarantees that it will output a probability distribution. The average CI DNN output of a polyphone state can be calculated by summing the CI DNN output vectors of all the feature vectors assigned to it, then dividing the sum by the count of feature vectors. Our novel approach rests on the idea that similar polyphone states should also have similar average CI DNN outputs, moreover the average CI DNN output can also be regarded as a discrete distribution, therefore entropy distance between polyphone states can be measured based on their average CI DNN outputs:

$$p_A(i) = \frac{1}{n_A} \sum_{j=1}^{n_A} P_{DNN}(s_i, F_j) \qquad (4)$$

$P_{DNN}(s_i, F_j)$ is the probability distribution generated by the CI DNN for the feature F_j. All features F_j are examples of the polyphone cluster A. After calculating it's entropy using equation 2 the weighted entropy distance $d(A, B)$ between two classes can now be computed without using any Gaussians.

4. ASR system

In this work all experiments, including the training of acoustic models, the building of the cluster trees and the evaluation of the ASR systems, are carried out with the Janus Recognition Toolkit (JRTk) which is developed at Karlsruhe Institute of Technology and Carnegie Mellon University [9].

Text corpus	# Words
TED	2,685k
News+Newscrawl	1,500M
Euro Language Newspaper	95,783k
Common Crawl	51,156k
Europarl	49,008k
ECI	14,582k
MultiUN	6,964k
German Political Speeches	5,695k
Callhome	159k
HUB5	20k
Google Web	(118m n-grams)

Table 1: *German language modeling data after cleaning and compound splitting. In total, we used 1.7 billion words, not counting Google Ngrams.*

4.1. Data

We used the following data sources to train the acoustic model:

- 180 hours of Quaero training data from 2009 to 2012.
- 24 hours of broadcast news data

For language model training texts from various sources like webdumps, scraped newspapers and transcripts are used. The in text corpora listed in Table 1 range in size from about 5 MByte to just over 6 GByte and are split into 28 sub corpora.

4.2. Acoustic models

We used a context dependent quinphone setup with three states per phoneme (except silence with is only modeled with one state) and a left-to-right topology without skip states. Our systems are based of the best single system from the 2014 IWSLT evaluation systems [10]. We extract 40lMel features from the audio using a 32ms window and a 10ms frame shift and augment them with tonal features [11]. All models use *vocal tract length normalization* (VTLN).

The context independent CI DNN AM on which the DNN based cluster trees are trained uses an input feature window of 15 and containes four hidden layers with 1200 neurons. Its output layer consists of 139 neurons, one for each of the three states of the 46 phones in our phone set and an extra one for silence. The hidden layers were first pre-trained with stacked denoising autoencoder (SdA) on the training data. After the pretraining of four hidden layers is completed, a logistic regression layer with 139 output neurons is added on top of the hidden layers. Then the DNN is fine-tuned with the supervised BP algorithm. The CI DNN is trained for 13 epochs and the CI ASR system based on it produces a WER of 26.9%. We consider this performance sufficient to estimate the a posteriori probabilities of the underlying monophone states given a feature vector and hence deem it suitable for cluster tree building.

The CI GMM AM on which the GMM based cluster trees are trained uses *incremental splitting of Gaussians* (MAS) training, followed by *optimal feature space* training and 2 iterations of Viterbi training.

The CD DNN AM is built using a modular DNN [12]. This involves stacking the deep bottleneck features over a window of 13 frames as the input to a NN with five 1600 unit hidden layers and an output layer containing as many neurons as its associated cluster tree has leaves, each corresponding to a context

dependent phone state. The deep bottleneck features are extracted using an MLP with five 1600 unit hidden layers prior to the 42 unit bottleneck layer. Its inputs are 40 lMel and 14 tone features stacked over a 13 frame window. Both neural network modules are pretrained as denoising autoencoders. Pretraining and fine-tuning are implemented using *Theano* [13].

4.3. Language models

A tuning set was randomly selected form the AM training data transcripts. The 300k vocabulary is selected by building a Witten-Bell smoothed unigram language model using the union of all the text sources vocabulary as the language models vocabulary (global vocabulary). With the help of the maximum likelihood count estimation method described in [14] we found the best mixture weights for representing the tuning sets vocabulary as a weighted mixture of the sources word counts thereby giving us a ranking of all the words in the global vocabulary sorted by their relevance to the tuning set. Using this 300k vocabulary, a 4gram case sensitive language model with modified Kneser-Ney smoothing was built for each of the sub corpora. This was done using the SRI Language Modeling Toolkit [15].

5. Experimental setup

We evaluate the effectiveness of the proposed DNN based cluster tree by building quinphone cluster trees of various sizes from 3k leaves to 21k leaves and compare them to cluster trees built using the baseline GMM approach. For each cluster tree a modular DNN AM is trained with the appropriate output layer. As can be seen in Figure 2 larger cluster trees outperform smaller cluster trees up to 18k leaves. The reduction in WER appears to be almost linear until about 12k leaves after which more leaves lead to less of an improvement. Since the WER started to deteriorate with cluster trees containing 21k or more leaves we halted our experiment at there. For the smaller cluster trees the baseline GMM based approach performed slightly better than the DNN based approach but for the larger and better cluster trees the DNN based approach consistently outperformed the GMM based approach.

Using the McNemar statistical test we compared the aligned hypothesis of both 18k systems and found the system using the DNN based cluster tree to be significantly better than the GMM based cluster tree with $p < 0.001$.

This shows that our DNN based cluster trees are not only a simple replacement for GMM based cluster trees in situations where CI-GMM AM are not available but can also outperform them.

6. Conclusion

In this work we have proposed a novel DNN based approach to building cluster trees and performed multiple experiments to confirm the functionality of our approach. We show how the entropy distance metric can adapted to work with the probability distribution generated by the output layer of a CI DNN AM thereby allowing us to build a cluster tree without using Gaussians. Eliminating the need for Gaussians in the construction of a cluster tree allows flat-started CI HMM/DNN AM systems to easily bootstrap a CD HMM/DNN AM. For the larger cluster trees with more than 12k leaves that produce CI HMM/DNN AMs with the lowest WER our DNN based approach outperforms the GMM based approach. The overall bwst system used a DNN based cluster tree with 18k leaves. In future work we
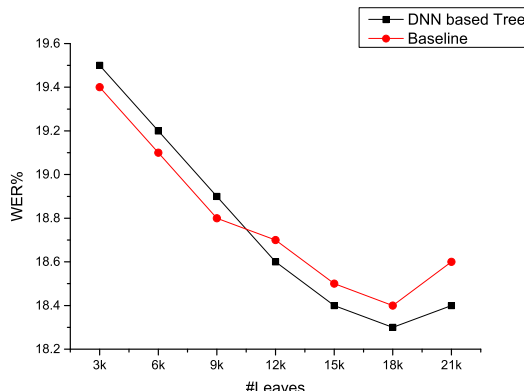

Figure 2: *A comparison of our DNN base cluster tree with a baseline GMM based cluster tree. We built cluster trees of various sizes between 3k leaves to 21k leaves and tested them on the IWSLT 2012 development set.*

hope further analyze the difference between the DNN based cluster trees and the GMM based cluster trees to in particular to find out why the DNN based approach is only better for cluster trees with a large number of leaves. We also wish to evaluate the effectiveness of other probability distributions.

7. Acknowledgements

8. References

[1] A. L. Maas, A. Y. Hannun, C. T. Lengerich, P. Qi, D. Jurafsky, and A. Y. Ng, "Increasing deep neural network acoustic model size for large vocabulary continuous speech recognition," *arXiv preprint arXiv:1406.7806*, 2014.

[2] H. Soltau, G. Saon, and T. N. Sainath, "Joint training of convolutional and non-convolutional neural networks," *to Proc. ICASSP*, 2014.

[3] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30–42, 2012.

[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[5] S. J. Young and P. C. Woodland, "State clustering in hidden markov model-based continuous speech recognition," *Computer Speech & Language*, vol. 8, no. 4, pp. 369–383, 1994.

[6] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Large vocabulary continuous speech recognition with context-dependent dbn-hmms," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4688–4691.

[7] A. Senior, G. Heigold, M. Bacchiani, and H. Liao, "Gmm-free dnn training," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 2014.

[8] C. Zhang and P. Woodland, "Standalone training of context-dependent deep neural network acoustic models," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 5597–5601.

[9] M. Finke, P. Geutner, H. Hild, T. Kemp, K. Ries, and M. Westphal, "The karlsruhe-verbmobil speech recognition engine," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, vol. 1. IEEE, 1997, pp. 83–86.

[10] K. Kilgour, M. Heck, M. Müller, M. Sperber, S. Stüker, and A. Waibel, "The 2014 kit iwslt speech-to-text systems for english, german and italian."

[11] F. Metze, Z. A. Sheikh, A. Waibel, J. Gehring, K. Kilgour, Q. B. Nguyen, and V. H. Nguyen, "Models of tone for tonal and nontonal languages," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 261–266.

[12] J. Gehring, W. Lee, K. Kilgour, I. R. Lane, Y. Miao, A. Waibel, and S. V. Campus, "Modular combination of deep neural networks for acoustic modeling." in *INTERSPEECH*, 2013, pp. 94–98.

[13] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a cpu and gpu math expression compiler," in *Proceedings of the Python for scientific computing conference (SciPy)*, vol. 4. Austin, TX, 2010, p. 3.

[14] A. Venkataraman and W. Wang, "Techniques for effective vocabulary selection," *Arxiv preprint cs/0306022*, 2003.

[15] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *Seventh International Conference on Spoken Language Processing*, 2002.