Application of Pretrained Deep Neural Networks to Large Vocabulary Speech Recognition

Navdeep Jaitly[1]*, *Patrick Nguyen*[2], *Andrew Senior*[3], *Vincent Vanhoucke*[2]

[1]Dept. of Computer Science, University of Toronto, Toronto, ON, Canada
[2]Google, Inc, MoutainView, CA, USA
[3]Google, Inc, New York, NY, USA

ndjaitly@cs.toronto.edu,drpng@google.com,andrewsenior@google.com,vanhoucke@google.com

Abstract

The use of Deep Belief Networks (DBN) to pretrain Neural Networks has recently led to a resurgence in the use of Artificial Neural Network - Hidden Markov Model (ANN/HMM) hybrid systems for Automatic Speech Recognition (ASR). In this paper we report results of a DBN-pretrained context-dependent ANN/HMM system trained on two datasets that are much larger than any reported previously with DBN-pretrained ANN/HMM systems - 5870 hours of Voice Search and 1400 hours of YouTube data. On the first dataset, the pretrained ANN/HMM system outperforms the best Gaussian Mixture Model - Hidden Markov Model (GMM/HMM) baseline, built with a much larger dataset by 3.7% absolute WER, while on the second dataset, it outperforms the GMM/HMM baseline by 4.7% absolute. Maximum Mutual Information (MMI) fine tuning and model combination using Segmental Conditional Random Fields (SCARF) give additional gains of 0.1% and 0.4% on the first dataset and 0.5% and 0.9% absolute on the second dataset.

Index Terms: Deep Belief Networks, Acoustic Modeling, Artificial Neural Network, ANN/HMM

1. Introduction

The ANN/HMM hybrid model was first used for ASR over two decades ago [1]. This model computes generative emission probabilities for acoustic data from the states of an HMM using Bayes rule to invert discriminative probabilities from a neural network trained to predict posterior states of the HMM from acoustic data. In spite of their early promise, ANN/HMM hybrids were eventually overtaken by GMM/HMM systems because of several factors which led to the superior performance and accuracy of GMM/HMM systems. These included a less computationally demanding, easily parallelizable training procedure which enabled the training of large models on large datasets, the ability to perform speaker adaptation and the development of discriminative techniques to train the GMM/HMM models. The performance of neural networks based approaches could theoretically have been improved further by using neural networks with more parameters. It has long been suspected that deep neural networks could model complex higher order statistical structure effectively but training deep neural nets is difficult and until recently such models were not used for ASR. The ANNs used in ASR systems were typically trained with only one hidden layer.

Recent advances in Machine Learning have led to the development of algorithms which can be used to train deep models [2, 3]. One of these approaches is the Deep Belief Network (DBN), a multi-layered generative model which can be trained greedily, layer by layer, using a model known as a Restricted Boltzmann Machine at each layer [2]. It has been empirically observed that using the parameters of a Deep Belief Network to initialize (a.k.a "pretrain") a deep neural network before fine tuning with backpropagation leads to improved performance of the deep neural network on discriminative tasks [4, 5]. This idea has been recently applied to pretrain deep neural networks for use in ANN/HMM hybrid speech recognition systems [6, 7, 8, 9]. State of the art results have been reported on phone recognition on TIMIT for a speaker independent, context independent system using a neural network with 8 layers [5]. Significant improvements have also been reported on context-dependent systems without speaker adaptation on a much larger dataset (Bing voice search data with about 300 hours of data) using a neural network with 9 layers [9].

Several issues about the use of DBNs and ANN/HMMs in ASR need further explorations. These include assessment of the models on large datasets with large language models which are now standard in the community. The recently reported results on using Context-Dependent (CD) ANN/HMMs with 309 hours of data and 9304 tied states is a significant step in that direction [9], but GMM/HMM systems are now routinely trained on much larger datasets. In addition, further studies are required to explore whether such systems can improve on GMM/HMM baselines that

*Performed work at Google

leverage speaker adaptive training (SAT) and whether these systems can gain from model combination. In this

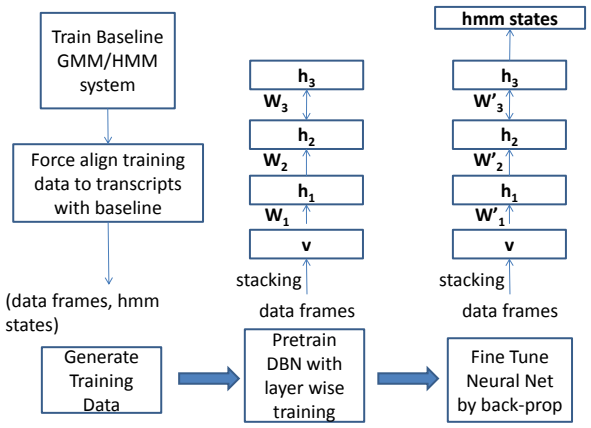

Figure 1: Pipeline for training ANN/HMM hybrid system

paper, we report the results of using a DBN pretrained ANN/HMM model for large vocabulary continuous speech recognition on two different datasets - 5780 hours of Voice Search and Android Voice Input data [1] using a CD system with 7969 target states, and 1400 hours of data from YouTube using a CD system with speaker adapted features and 17552 target states. Both systems significantly outperformed the GMM/HMM baseline systems. On the Voice Search dataset, the ANN/HMM system outperformed the best GMM/HMM system, built with a significantly larger amount of data, by 3.7% absolute (23% relative) Word Error Rate (WER). On the YouTube dataset the ANN/HMM system outperformed the best GMM/HMM system by 4.7% absolute improvement. This gap was further increased by 0.5% absolute by using MMI to fine tune the neural network using a procedure similar to [10] and another 0.9% absolute from model combination by combining results from the GMM/HMM and ANN/HMM systems on the YouTube dataset using Segmental Conditional Random Fields [11]. For Voice Search, a smaller improvement of 0.1% absolute was observed using MMI and an improvement of 0.4% absolute was observed from model combination.

2. Model Training

The ANN/HMM hybrid models were trained in three stages as shown in figure 1. First, a baseline GMM/HMM system was trained and forced alignment was used to associate each frame of data with a target HMM state. Then, a DBN was trained on the acoustic data (which may be MFCC vectors, log filterbanks, or speaker adapted features stacked together) and the weights of the DBN were used to initialize a neural network, which was then trained to predict the HMM state from the acoustic data, using backpropagation. This procedure is the same as has been previously reported in [7, 5, 9, 8] and the reader is referred there for further details. Inasmuch as our system is based on context-dependent models, the reader is referred to [6, 9] as these studies also used context-dependent systems.

Name	# of hours	CMLLR?	WER
Voice Search	$\gg$6K	No	16.0
YouTube	$\gg$1400	Yes	52.3

Table 1: Baselines used for the study

3. Experimental Setup

Table 1 shows a summary of the baseline systems used for the study. The trainining data consisted of unsupervised data that was mostly untranscribed. These were confidence-filtered for optimal unsupervised training. The test sets on the other hand were hand-transcribed.

3.1. Voice Search

The training data for the Voice Search system consisted of approximately 5780 hours of data from mobile Voice Search and Android Voice Input. The baseline model used was a triphone hmm with decision tree clustered states. The acoustic data was contiguous frames of PLP features that were transformed by Linear Discriminant Analysis (LDA). Semi-Tied Covariances (STC) [12] were used in the GMM's to model the LDA transformed features. Boosted-MMI was used to train the model discriminatively [13]. This generated a CD model with 7969 states.

The training data for our hybrid ANN/HMM model was generated by performing a forced alignment of the transcripts to the acoustic observations. This resulted in frames of data being assigned an HMM state. 11 contiguous frames of acoustic vectors were then modeled with a DBN. The DBN's were trained using a greedy layer by layer procedure similar to that of [2] except that the data vector was continuous, 440 dimensional (=11*40, since we used 40 dimensional log filter-banks computed over 25ms with a stride of 10 ms), rather than binary, so the bottom layer was a Gaussian binary RBM, similar to that in [7]. Note that the input representation used here was not the same same as that used in the baseline - we assumed that the ANN would discover relevant features automatically from filter banks. The trained DBN was used to initialize a neural network that was trained by back-propagation to predict the HMM state assigned to the central frame of the stacked acoustic frames from the acoustic vectors used as input to the neural network. Based on our experiments with the Broadcast News task

[1] We will refer to this as the Voice Search data

(not reported here) we chose to use four hidden layers with 2560 nodes per layers as the architecture of choice. Because of computational speed limitations, a model was trained for 6 epochs with approximately one third of the data, and trained a further four epochs on the entire 5780 hours data.

Note that the model we used to generate the forced alignments and unsupervised labels was a Voice Search model built from a much larger dataset with a baseline WER of 16.0% on this testset.

3.2. YouTube

The training data for the YouTube system consisted of approximately 1400 hours of data from YouTube. The system used 9-frame MFCCs that were transformed by LDA and SAT was performed. Decision tree clustering was used to obtain 17552 triphone states, and STCs were used in the GMMs to model the features. The acoustic models were further improved with BMMI [13]. During decoding, Constrained Maximum Likelihood Linear Regression (CMLLR) and Maximum Likelihood Linear Regression (MLLR) transforms were applied.

Training data for the ANN/HMM was again generated from forced alignment using the SAT models to generate the target states. The acoustic data used for the ANN/HMM system were the CMLLR transformed features. The large number of HMM states added significantly to the computational burden, since most of the computation is done at the output layer. As a result, we chose to use 2000 nodes at the lowest layer, but used 1000 nodes in the layers above, to make the training faster.

3.3. Training Procedure

The models were trained on a dual CPU Intel Xeon DP Quad Core E5640 machine with Ubuntu OS equipped with four NVIDIA Tesla C2070 Graphics Processing Units. Each job was performed on a single CPU with a single GPU board. Data were loaded on to CPU memory in big mini-batches of 20 hours for Voice Search, and 17.5 hours for YouTube. These were then loaded into the GPU, and randomly permuted. Mini-batches of size 200 for Voice Search and 500 for YouTube were built by cycling through these permuted vectors. Model parameters were all kept and updated on GPU memory itself. Average gradients were computed on the mini-batches and parameters were updated with a learning rate of .04 for the top two layers of the network and 0.02 for the others, with a momentum of 0.9. Each DBN layer was pre-trained for one epoch as an RBM and then the resulting ANN was discriminatively fine-tuned for one epoch. Weights with magnitudes below a threshold were then permanently set to zero before a further quarter epoch of training.

All the computations involved in training the DBN (matrix multiplications, sampling etc) and the Neural Network (matrix multiplications, etc) were performed on the GPU using the Cudamat library [14].

3.4. Decoding Procedure

Decoding was done on the Google clusters with MapReduce [15]. For this the Google speech recognition engine was modified to incorporate a neural network frontend, which was used to compute the log-likelihoods for the different HMM states, using acoustic data, and the previously trained models. The likelihoods were scaled by the appropriate priors for the states, estimated empirically from the forced alignment state labels. The scaled likelihoods were used in the lattice search during decoding, as was done previously in [7, 6], instead of the typical GMM emission probabilities. With the Voice Search data it was observed that a smoothing of the estimated priors was essential to performance. Smoothing of the priors was performed by rescaling the log(priors) with a multiplier that was chosen by jointly optimizing the language model weight, word insertion penalty and smoothing factor in a grid search.

For decoding YouTube data, we first ran a GMM/HMM decoding pass to obtain a hypothesis transcript, which was used to compute a CMLLR transform to normalize the features. The CMLLR transformed features were then decoded with the ANN/HMM system.

While decoding can be slow with a naive implementation of neural networks, we employed strategies described in [16] with which decoding in x86 CPU architectures can be done in real time.

4. Experimental Results

Table 2 shows a summary of the results. Performance is reported at a large pruning beam to eliminate the impact of search errors.

4.1. Training Time

Since each system had a different number of target states and architecture, the amount of time to process the same duration of speech signal was different for each system. An epoch of the YouTube model trained at the rate of approximately 55 hours per epoch and an epoch of the entire Voice Search data trained at the rate of approximately 321 hours per epoch. As such, computational speed of training remains an important issue for this method.

4.2. Voice Search and YouTube

For the Voice Search dataset an absolute improvement of 3.7% WER was observed over the baseline. For YouTube an improvement of 4.7% was observed over the baseline system after the fourth epoch. Further epochs of training were not beneficial.

4.3. MMI Fine Tuning of Neural Network

Sequence level discriminative fine tuning of the neural network was performed using MMI, similar to the

Name	Model	WER(%)
Voice Search	GMM-HMM baseline	16.0
	DBN pretrained ANN/HMM with sparsity	12.3
	+ *MMI*	12.2
	+ *system combination with SCARF*	**11.8**
YouTube	GMM-HMM baseline	52.3
	DBN pretrained ANN/HMM with sparsity	47.6
	+ *MMI*	47.1
	+ *system combination with SCARF*	**46.2**

Table 2: Summary of Results

method proposed in [10]. This produced a gain of 0.5% for the YouTube dataset, and 0.1% on the Voice Search data.

4.4. Model Combination

Model combination, achieved by using the SCARF framework [11] to combine results from the GMM/HMM system and ANN/HMM system, resulted in further absolute improvement of 0.9% WER in performance of the YouTube system and 0.4% WER improvement for the Voice Search system.

5. Conclusion and Future Challenges

The results from this study indicate that ANN/HMM hybrids pretrained with DBNs can indeed outperform GMM/HMM systems significantly, even when the GMM/HMM systems are built with well established recipes for speaker adaptive training (as was the case for the YouTube GMM/HMM baseline) and discriminative training (both GMM/HMM baselines), using much more data. We have discovered and reported several novel findings that can further improve the accuracy of ANN/HMM hybrids, including gains from prior smoothing, MMI fine-tuning and system combination. We have also shown that speaker adaptive features that can be leveraged within the ANN/HMM hybrid system, by using them as the input to the neural network, as was done in [8].

However, a significant hurdle in the wide scale adoption of these methods in LVCSR is the time taken to discriminatively fine-tune the neural networks (the pretraining using DBN's is much faster, and much less of an issue). Batch-based methods using second order approximations remain a promising way to solve this problem. Further studies are required to select methods and neural network architectures that allow for fast computation without loss of accuracy.

6. Acknowledgements

We would like to acknowledge Will Neveitt for starting this collaboration. We would also like to thank Olivier Siohan and Geoffrey Hinton for helpful discussions.

7. References

[1] N. Morgan and H. Bourlard, "Continuous Speech Recognition using Multilayer Perceptrons with Hidden Markov Models," in *Proc. ICASSP '90*, vol. 1, pp. 413–416.

[2] G. Hinton, S. Osindero, and Y. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[3] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and Composing Robust Features with Denoising Autoencoders," in *Proc. 25th Int. Conf. Machine learning*, 2008, pp. 1096–1103.

[4] G. Hinton and R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[5] G. Dahl, M. Ranzato, A. Mohamed, and G. Hinton, "Phone Recognition with the Mean-Covariance Restricted Boltzmann Machine," in *Advances in Neural Information Processing Systems 23*, 2010, pp. 469–477.

[6] G. Dahl, D. Yu, L. Deng, and A. Acero, "Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition," in *IEEE Trans. Audio, Speech, and Language Processing*, June 2012.

[7] A. Mohamed, G. Dahl, and G. Hinton, "Acoustic Modeling using Deep Belief Networks," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 99, June 2012.

[8] A. Mohamed, T. Sainath, G. Dahl, B. Ramabhadran, G. Hinton, and M. Picheny, "Deep Belief Networks using Discriminative Features for Phone Recognition," in *Proc. ICASSP '11*, pp. 5060–5063.

[9] F. Seide, G. Li, and D. Yu, "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks," in *Interspeech*, 2011, pp. 437–440.

[10] B. Kingsbury, "Lattice-based Optimization of Sequence Classification Criteria for Neural-Network Acoustic Modeling," in *Proc. ICASSP '09*, pp. 3761–3764.

[11] G. Zweig, P. Nguyen, D. Compernolle, K. Demuynck, L. Atlas, P. Clark, G. Sell, M. Wang, F. Sha, H. Hermansky, D. Karakos, A. Jansen, S. Thomas, G. Sivaram, S. Bowman, and J. Kao, "Speech Recognition with Segmental Conditional Random Fields: A summary of the JHU CLSP 2010 Summer Workshop," in *Proc. ICASSP '11*, pp. 5044–5047.

[12] M. Gales, "Semi-tied Covariance Matrices for Hidden Markov Models," in *IEEE Trans. Speech and Audio Processing*, vol. 7, 1999, pp. 272–281.

[13] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for Model and Feature-space Discriminative Training," in *Proc. ICASSP '08*, pp. 4057–4060.

[14] V. Mnih, "Cudamat: a CUDA-based Matrix Class for Python," Department of Computer Science, University of Toronto, Tech. Rep. 004, 2009.

[15] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communic. ACM*, vol. 51, pp. 107–113, January 2008.

[16] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on CPUs," in *Deep Learning and Unsupervised Feature Learning Workshop. NISP 2011*, 2011, pp. 272–281.