LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION WITH CONTEXT-DEPENDENT DBN-HMMS

*George E. Dahl**

University of Toronto
Department of Computer Science
Toronto, ON, Canada

Dong Yu, Li Deng, Alex Acero

Speech Research Group
Microsoft Research
Redmond, WA, USA

ABSTRACT

The context-independent deep belief network (DBN) hidden Markov model (HMM) hybrid architecture has recently achieved promising results for phone recognition. In this work, we propose a context-dependent DBN-HMM system that dramatically outperforms strong Gaussian mixture model (GMM)-HMM baselines on a challenging, large vocabulary, spontaneous speech recognition dataset from the Bing mobile voice search task. Our system achieves absolute sentence accuracy improvements of 5.8% and 9.2% over GMM-HMMs trained using the minimum phone error rate (MPE) and maximum likelihood (ML) criteria, respectively, which translate to relative error reductions of 16.0% and 23.2%.

***Index Terms*—** Speech recognition, deep belief network, context-dependent phone, LVCSR, DBN-HMM

1. INTRODUCTION

Despite many years of successful research and advances in discriminative training for hidden Markov models (HMMs), e.g. [1, 2, 3, 4], the performance of automatic speech recognition (ASR) systems in real usage conditions remains unsatisfying. Recently, context-independent (CI) deep belief network (DBN) hidden Markov model hybrid architectures have shown promising results for phone recognition (see [5, 6, 7]). In this paper, we propose a novel context-dependent (CD) extension of the previous work on DBN-HMMs. Using the challenging Bing mobile voice search dataset, we demonstrate that the context-dependent DBN-HMMs we propose here can significantly outperform strong discriminatively trained Gaussian mixture model (GMM)-HMMs on a real-world, large vocabulary, continuous speech recognition (LVCSR) task.

DBN-HMMs are closely related to the artificial neural network (ANN)-HMM hybrid architectures that were first explored as an alternative paradigm for ASR in the late 1980s and early 1990s. In both architectures, each output unit of the neural network is trained to estimate the posterior probability of a continuous density HMMs' state given the acoustic observations. Our work differs from earlier CD-ANN-HMMs [8] in two key respects. First, we used deeper, more expressive neural network architectures and thus employed the unsupervised DBN pre-training algorithm to make sure training would be effective. Second, we used posterior probabilities of senones [9] as the output of the DBN/ANN, instead of the combination of context-independent phone and context class used previously. This second difference also distinguishes our work from earlier uses of DBNs to produce distributions over monophone HMM states (CI-DBN-HMMs) for phone recognition [5, 6, 7].

*This work was performed while the author was working as an intern at Microsoft Research.

2. CONTEXT-DEPENDENT DBN-HMMS

An HMM is a generative model in which the observation sequence is assumed to be generated from a hidden Markov process that transitions between states $S = \{s_1, \cdots, s_K\}$. The parameters of the HMM are the initial state probability distribution $\pi = \{p(q_0 = s_i)\}$, where q_t is the state at time t, the transition probabilities $a_{ij} = p(q_t = s_j | q_{t-1} = s_i)$, and a model to estimate the observation probabilities $p(\mathbf{x}_t | s_i)$. The key difference between DBN-HMMs and GMM-HMMs is the use of DBNs (instead of GMMs) to estimate the observation probabilities given senones (CI-DBN-HMMs use CI-phone states instead). We actually use the DBN to model $p(q_t|\mathbf{x}_t)$, the posterior probability of the senones given the observation vector $\mathbf{x}_t$, which is possible since $p(q_t)$ is easy to estimate from an initial senone-level alignment of the training set.

This idea of using senones as the modeling unit has been proposed in [10] where the posterior probability of senones were estimated using deep-structured conditional random fields (CRFs). Modeling senones using DBNs provides two primary advantages. First, we can implement a DNN-HMM system with only minimal modifications to an existing GMM-HMM system. Second, any improvements in modeling units that are incorporated into the GMM-HMM baseline system, such as cross-word triphone models, will be accessible to the DNN through the use of the shared training labels.

In our CD-DBN-HMMs, the decoded word sequence $\hat{w}$ is given by: $\hat{w} = \operatorname{argmax}_w p(\mathbf{x}|w)p(w)$, where $\mathbf{x}$ is the observation sequence, $p(w)$ is the language model (LM) probability, and $p(\mathbf{x}|w) \cong \max_q \pi(q_0) \prod_{t=1}^{T} a_{q_{t-1}q_t} \prod_{t=0}^{T} p(\mathbf{x}_t|q_t)$ is the acoustic model (AM) probability. Note that in the observation probability $p(\mathbf{x}_t|q_t) = p(q_t|\mathbf{x}_t)p(\mathbf{x}_t)/p(q_t)$, $p(\mathbf{x}_t)$ is independent of the word sequence and thus can be ignored during decoding. Note also that although dividing by the prior probability $p(q_t)$ may not give improved recognition accuracy under some conditions, we have found it to be very important in alleviating the label bias problem, especially when the training utterances contain long silence segments.

CD-DBN-HMMs can be trained using the embedded Viterbi algorithm, which involves two main steps. First, we must generate a senone-level alignment using the Viterbi algorithm, and then we must train the DBN to predict the senone in each frame. If DBNs provide better senone predictions at each *frame*, then CD-DBN-HMMs can achieve better *sentence* recognition accuracy than tri-phone GMM-HMMs.

3. DEEP BELIEF NETWORKS

Deep belief networks (DBNs) are probabilistic generative models with multiple layers of stochastic hidden units above a single bottom layer of observed variables that represent a data vector. There is an efficient unsupervised algorithm, first described in [11], for learning the connection weights in a DBN that is equivalent to training each adjacent pair of layers as a restricted Boltzmann machine (RBM). There is also a fast, approximate, bottom-up inference algorithm to infer the states of all hidden units conditioned on a data vector. After the unsupervised, or *pre-training* phase, [11] used the *up-down* algorithm to optimize all of the DBN weights jointly using labeled data.

In what is now a standard abuse of terminology, we will also refer to feed-forward neural nets whose weights have been initialized with the DBN pre-training algorithm as DBNs. Thus in this work, we train deep, but otherwise standard, neural networks by first using the unsupervised DBN pre-training algorithm to initialize the weights and then simply using standard backpropagation to fine-tune the network weights.

Insufficiently deep architectures can require an exponential blow-up in the number of computational elements needed to represent certain functions satisfactorily. Thus one primary motivation for using deep models, such as DBNs, is that they are generally much more representationally efficient than shallower models like GMMs. Furthermore, GMMs as used in ASR typically have a large number of highly localized Gaussians. Since GMMs in ASR depend on local generalization, their generalization ability is likely to be limited when modeling rapidly varying distributions. One of our hopes in this work is to demonstrate that replacing GMMs with more powerful models can reduce recognition error in a difficult LVCSR task.

Restricted Boltzmann Machines (RBMs) are the building blocks of DBNs, and are undirected graphical models with a layer of observed, or visible, variables and a layer of hidden variables, with each layer forming one part of a bipartite graph. For the purposes of this work, the hidden units will always be binary and the visible units will either be Bernoulli or Gaussian distributed, conditional on the hidden units. An RBM assigns an energy to every configuration of visible and hidden state vectors, denoted $\mathbf{v}$ and $\mathbf{h}$ respectively. For binary visible units, the RBM energy function is:

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^{\mathrm{T}}\mathbf{v} - \mathbf{c}^{\mathrm{T}}\mathbf{h} - \mathbf{v}^{\mathrm{T}}\mathbf{W}\mathbf{h}, \tag{1}$$

where $\mathbf{W}$ is the matrix of visible/hidden connection weights, $\mathbf{b}$ is a visible unit bias, and $\mathbf{c}$ is a hidden unit bias. For any type of RBM, the probability of any particular setting of the visible and hidden units is given in terms of the energy of that configuration by:

$$P(\mathbf{v}, \mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z}, \tag{2}$$

where the normalization factor $Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$ is known as the partition function. To deal with real-valued speech input, we use an RBM with Gaussian visible units (GRBM) whose energy function is given by:

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2}(\mathbf{v} - \mathbf{b})^{\mathrm{T}}(\mathbf{v} - \mathbf{b}) - \mathbf{c}^{\mathrm{T}}\mathbf{h} - \mathbf{v}^{\mathrm{T}}\mathbf{W}\mathbf{h}. \tag{3}$$

Note that equation 3 implicitly assumes that the visible units have a diagonal covariance Gaussian noise model with a variance of 1 on each dimension given the hidden units.

We train all the RBMs with one-step contrastive divergence, exactly as done in the recent work described in [5, 6, 7, 12].

4. EXPERIMENTS

We evaluated our CD-DBN-HMM system by conducting a series of experiments on the data collected from the Bing mobile voice search (BMVS) application (formerly known as Live Search for mobile [14]) – a real-world, large-vocabulary, spontaneous, continuous speech recognition task.

4.1. Description of Dataset and GMM-HMM Baselines

The Bing mobile voice search application allows users to do US-wide location and business lookup from their mobile phones via voice. This is a challenging task since the dataset contains all kinds of variations: noise, music, side-speech, accents, sloppy pronunciation, hesitation, repetition, interruption, and different audio channels. The dataset was split into a 24-hour (32,057 utterance) mini training set, which was a subset of the full 2100-hour (3M utterance) training set, a 6.5-hour (8,777 utterance) development set, and a 9.5-hour (12,758 utterance) test set. We ran all experiments on the mini training set in this study. To facilitate performance comparisons with the work in [13], which uses the same dataset and task, we used the public lexicon from Carnegie Mellon University. The language model (LM) used in the evaluation contains a 65K word vocabulary, 3.2 million word bi-grams, and 1.5 million word tri-grams. Performance on this task was evaluated using sentence accuracy (SA) instead of word accuracy for a variety of reasons. First, in order to compare our results with [13], we would need to compute sentence accuracy. Second, the average sentence length is 2.1 tokens, so sentence accuracy is highly correlated with the word accuracy. Third, the users care most about whether they can find the business or location they seek in the fewest attempts. They typically will repeat the whole sentence if one of the words is mis-recognized. Fourth, there is significant inconsistency in spelling that makes using sentence accuracy more convenient.

To compare our proposed DBN-HMM model with the standard state-of-the-art systems, we have trained clustered cross-word triphone GMM-HMMs with the maximum likelihood (ML), maximum mutual information (MMI), and minimum phone error (MPE) criteria using the mini training set. The 39-dim features used in the experiments include the 13-dim static Mel-frequency cepstral coefficient (MFCC) (with C0 replaced with energy) and its first and second derivatives. The features were pre-processed with the cepstral mean normalization (CMN) algorithm. We also standardized the frames so that each dimension would have mean zero and unit variance over the entire training set.

The performance of the best GMM-HMM configurations is summarized at the bottom of table 1. The systems trained using the mini training set have 53K logical and 2K physical tri-phones with 761 shared states (senones), each of which is a 24-mixture GMM. The system trained using the full training set has 11K physical triphones with 2.8K 96-mixture senones. Note that the accuracy of the system trained using the full training set was quoted from [13] and is only 2.5% better than the system trained with the mini training set when both systems were trained using the ML criterion, which indicates that the baselines we trained on the mini training set are not weak. However, while the entire mini training set was manually transcribed, only 130 hours of the full training set were manually transcribed. The rest of the full training set used the user's implicit confirmation as supervision, a procedure which yields a correct transcription 90% of the time.

Model Type	Depth	Alignment	Labels	Tuned Trans.	Dev Acc.	Test Acc.
DBN-HMM	1	from ML GMM-HMM	Monophone States	no	59.3%	-
DBN-HMM	3	from ML GMM-HMM	Monophone States	no	64.2%	-
ANN-HMM	1	from ML GMM-HMM	Triphone Senones	no	68.0%	-
ANN-HMM	2	from ML GMM-HMM	Triphone Senones	no	68.2%	-
DBN-HMM	1	from ML GMM-HMM	Triphone Senones	no	68.1%	-
DBN-HMM	2	from ML GMM-HMM	Triphone Senones	no	69.5%	-
DBN-HMM	3	from ML GMM-HMM	Triphone Senones	no	69.6%	-
DBN-HMM	4	from ML GMM-HMM	Triphone Senones	no	70.2%	-
DBN-HMM	5	from ML GMM-HMM	Triphone Senones	no	70.3%	68.4%
DBN-HMM	5	from MPE GMM-HMM	Triphone Senones	no	70.7%	68.8%
DBN-HMM	5	from MPE GMM-HMM	Triphone Senones	yes	71.0%	69.0%
DBN-HMM	5	from DBN-HMM	Triphone Senones	no	71.7%	69.6%
DBN-HMM	5	from DBN-HMM	Triphone Senones	yes	71.8%	69.6%
ML GMM-HMM baseline					62.9%	60.4%
MMI GMM-HMM baseline					65.1%	62.8%
MPE GMM-HMM baseline					65.5%	63.8%
ML GMM-HMM baseline 2100 hours of training data (transcription is 90% accurate)					-	62.9% [13]

Table 1. Experimental results

4.2. DBN-HMM Results and Analysis

Our experimental results, summarized in table 1, show the effect on recognition accuracy of a few of the important choices one must make when training DBN-HMMs. In all DBN-HMM experiments presented in the table, we used 2048 units in each hidden layer and 11 (5-1-5), 39-dimensional frames of MFCCs as the input to the network. During pre-training we used a learning rate of 0.004 and a momentum of 0.9 for all layers. For fine-tuning, we used a learning rate of 0.08 for the first 6 epochs and a learning rate of 0.002 for the last 6 epochs, also always with momentum of 0.9. In all our experiments, gradients were averaged over minibatches of 256 cases.

Using triphone senones as the DBN outputs provides a large gain over monophone state DBN outputs, or in other words CD-DBN-HMMs significantly outperform CI-DBN-HMMs on this task. Note that once we use senones as the network outputs, even with only a single hidden-layer and no pre-training, we obtain a dev set accuracy of 68.0%, which is already 2.5% higher than the best GMM-HMM baseline. We believe this perhaps unusually strong single layer result indicates that using senones for the neural network outputs is quite beneficial and that the dataset we use has substantial acoustic variability, giving an advantage to a model that does not make strong assumptions about the data distribution. As can be seen from the 3 hidden-layer results in table 1, the gain from using triphone senones persists even with a deeper architecture. DBN pre-training is not necessary for single hidden-layer models (68.1% vs 68.0% on the dev set), but it is very important for deeper architectures (although all our experiments in this work with three or more hidden layers were performed with pre-training). In general, we expect that when deeper models are used, the pre-training step in the DBN-HMM will become even more important.

In all our experiments, adding DBN hidden layers improved recognition accuracy, but with more layers these improvements diminished. Overall, using models with five hidden layers provides us with a 2.2% accuracy improvement over a single hidden-layer system when the same alignment is used. Since training a five-layer model is already quite expensive, we did not explore architectures with more than five hidden layers, so we cannot rule out additional gains from using more layers.

In order to demonstrate the efficiency of parameterization enjoyed by deeper neural networks, we have also trained a single hidden-layer neural network with 16K hidden units, a number chosen to guarantee that the weights required somewhat more space to store than the weights for our 5 hidden-layer models. We were able to obtain an accuracy of 68.6% on the development set, which is slightly more than the 2K hidden unit single layer result of 68.1% in figure 1, but well below even the two layer result of 69.5% (let alone the five layer result of 70.3%).

In addition to the importance of using DBN pre-training and training the DBN to produce a distribution over senones, we found that the higher the accuracy of the model producing the forced alignment used to generate frame-level labels for DBN training, the better the performance of the trained DBN. Thus training a five hidden-layer DBN on the alignment from the MPE trained GMM-HMM produced a superior result (70.7% dev set accuracy) to the same DBN trained on labels derived from the ML GMM-HMM alignment (70.3% dev set accuracy). We also found that re-tuning the HMM transition probabilities using maximum likelihood training also provided a slight improvement in accuracy. After tuning the transition probabilities for the five hidden-layer DBN-HMM trained on labels derived from the MPE GMM-HMM alignment, we used the DBN-HMM to produce another forced alignment and generated new training labels for the frames. After fine-tuning the DBN on these labels and re-tuning the transition probabilities, we were able to achieve our best result of 71.8% accuracy on the dev set and 69.6% accuracy on the test set.

Overall, our proposed CD-DBN-HMMs obtained 69.6% accuracy on the test set, which is 5.8% (or 9.2%) higher than those obtained using the MPE (or ML)-trained GMM-HMMs. This improvement translates into a 16.0% (or 23.2%) relative error rate reduction over the MPE (or ML)-trained GMM-HMMs.

4.3. Training and Decoding Time

DBN-HMMs significantly outperform GMM-HMMs in terms of recognition accuracy on our task and although DBN training is asymptotically quite scalable, training can be quite time consuming in practice. All timing results reported here are based on a trainer written in Python running on a workstation with a 2.66GHz Intel Xeon 5600 series processor and an NVIDIA Tesla C1060 general

purpose graphical processing unit. We used the CUDAMat library [15] to perform matrix operations on the GPU from our Python code.

We used 50 epochs of pretraining for the first hidden layer, 20 epochs for each layer thereafter, and 12 epochs of backpropagation. For larger datasets, reasonable results might be possible with fewer epochs of training than what we used here. Pre-training a five-layer DBN-HMM took about 62 hours and fine-tuning took about 16.8 hours. To achieve the best result reported in this paper, we had to run two passes of fine-tuning, one with the MPE GMM-HMM alignment, and one with the DBN-HMM alignment. The total fine-tuning time is thus 33.6 hours. The total time spent to train the system from scratch is about four days. Note that we have observed that using a GPU speeds up training by about a factor of 30 compared to just using the CPU in our setup.

While training is considerably more expensive than for GMM-HMM systems, decoding is still very efficient. We can decode in real time even using a five hidden-layer DBN-HMM with 2K units per layer, both with and without using GPUs.

5. CONCLUSION AND FUTURE WORK

We have described a novel context-dependent version of DBN-HMMs for LVCSR that achieves substantially better results on the challenging BMVS dataset than strong discriminatively trained GMM-HMMs. Although our experiments show that DBN-HMMs provide dramatic improvements in recognition accuracy, training DBN-HMMs is quite expensive compared with training GMM-HMMs, primarily because training the former is not easy to parallelize across computers and needs to be carried out on a single GPU machine. However, decoding in DBN-HMMs is very efficient so test time is not an issue in real-world applications. We believe our work on context-dependent DBN-HMMs is only the first step towards a more powerful acoustic model for LVCSR; many issues remain to be resolved. We hope that the encouraging results we presented in this work will inspire other researchers to investigate related approaches to acoustic modeling in LVCSR, to tackle some of the numerous research challenges deeper acoustic models present, and to provide possible solutions to some of the grand challenges in speech recognition and understanding [16].

6. ACKNOWLEDGMENTS

We would like to thank Dr. Patrick Nguyen for preparing the BMVS dataset and providing the ML-trained GMM-HMM baseline, Dr. Chaojun Liu for his assistance in getting the discriminatively-trained GMM-HMM baselines, Dr. Jasha Droppo for his parallel computing support, Prof. Geoff Hinton for advice and encouragement of this work, and Prof. Nelson Morgan for discussions on prior work on ANN-HMM approaches.

7. REFERENCES

[1] D. Povey, *Discriminative training for large vocabulary speech recognition*, Ph.D. thesis, Cambridge University Engineering Dept, 2003.

[2] E. McDermott, T. Hazen, J. L. Roux, A. Nakamura, and S. Katagiri, "Discriminative training for large vocabulary speech recognition using minimum classification error," *IEEE Transactions on Speech and Audio Processing*, vol. 15, no. 1, pp. 203–223, 2007.

[3] X. He, L. Deng, and W. Chou, "Discriminative learning in sequential pattern recognition — a unifying review for optimization-oriented speech recognition," *IEEE Signal Processing Magazine*, vol. 25, no. 5, pp. 14–36, 2008.

[4] D. Yu, L. Deng, X. He, and A. Acero, "Large-margin minimum classification error training for large-scale speech recognition tasks," in *Proc. ICASSP*, 2007, vol. 4, pp. 1137–1140.

[5] A. Mohamed, G. E. Dahl, and G. E. Hinton, "Deep belief networks for phone recognition," in *NIPS Workshop on Deep Learning for Speech Recognition and Related Applications*, 2009.

[6] G. E. Dahl, M. Ranzato, A. Mohamed, and G. E. Hinton, "Phone recognition with the mean-covariance restricted Boltzmann machine," in *Advances in Neural Information Processing Systems 23*, J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, Eds., pp. 469–477. 2010.

[7] A. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *Proc. Interspeech*, 2010, pp. 2846–2849.

[8] S. Renals, N. Morgan, H. Boulard, M. Cohen, and H. Franco, "Connectionist probability estimators in HMM speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 1, pp. 161–174, 1994.

[9] M. Hwang and X. Huang, "Shared-distribution hidden Markov models for speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 1, no. 4, pp. 414–420, 1993.

[10] D. Yu and L. Deng, "Deep-structured hidden conditional random fields for phonetic recognition," in *Proc. Interspeech*, 2010, pp. 2986–2989.

[11] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[12] L. Deng, M. Seltzer, A. Mohamed, A. Acero, D. Yu, and G. Hinton, "Binary coding of speech spectrograms using a deep auto-encoder," in *Proc. Interspeech*, 2010, pp. 1692–1695.

[13] G. Zweig and P. Nguyen, "A segmental CRF approach to large vocabulary continuous speech recognition," in *Proc. ASRU*, 2009, pp. 152–155.

[14] A. Acero, N. Bernstein, R. Chambers, Y. Ju, X. Li, J. Odell, P. Nguyen, O. Scholtz, and G. Zweig, "Live search for mobile: Web services by voice on the cellphone," in *Proc. ICASSP*, 2008, pp. 5256–5259.

[15] V. Mnih, "Cudamat: a CUDA-based matrix class for python," Tech. Rep. UTML TR 2009-004, Department of Computer Science, University of Toronto, November 2009.

[16] J. M. Baker, L. Deng, J. Glass, S. Khudanpur, C. Lee, N. Morgan, and D. O'Shaugnessy, "Research developments and directions in speech recognition and understanding, part 1," *IEEE Signal Processing Magazine*, vol. 26, no. 3, pp. 75–80, 2009.