Conversational Speech Transcription Using Context-Dependent Deep Neural Networks

Frank Seide[1], *Gang Li,*[1] *and Dong Yu*[2]

[1]Microsoft Research Asia, Beijing, P.R.C.
[2]Microsoft Research, Redmond, USA
{fseide,ganl,dongyu}@microsoft.com

10.21437/Interspeech.2011-169

Abstract

We apply the recently proposed Context-Dependent Deep-Neural-Network HMMs, or CD-DNN-HMMs, to speech-to-text transcription. For single-pass speaker-independent recognition on the RT03S Fisher portion of phone-call transcription benchmark (Switchboard), the word-error rate is reduced from 27.4%, obtained by discriminatively trained Gaussian-mixture HMMs, to 18.5%—a 33% relative improvement.

CD-DNN-HMMs combine classic artificial-neural-network HMMs with traditional tied-state triphones and deep-belief-network pre-training. They had previously been shown to reduce errors by 16% relatively when trained on tens of hours of data using hundreds of tied states. This paper takes CD-DNN-HMMs further and applies them to transcription using over 300 hours of training data, over 9000 tied states, and up to 9 hidden layers, and demonstrates how sparseness can be exploited.

On four less well-matched transcription tasks, we observe relative error reductions of 22–28%.

Index Terms: speech recognition, deep belief networks, deep neural networks

1. Introduction

Since the early 90's, artificial neural networks (ANNs) have been used to model the state emission probabilities of HMM speech recognizers [1]. While traditional Gaussian mixture model (GMM)-HMMs model context dependency through tied context-dependent states (e.g. CART-clustered crossword triphones [2]), ANN-HMMs were never used to do so directly. Instead, networks were often factorized, e.g. into a monophone and a context-dependent part [3], or hierarchically decomposed [4]. It has been commonly assumed that hundreds or thousands of triphone states were just too many to be accurately modeled or trained in a neural network. Only recently did Yu *et al.* discover that doing so is not only feasible but works very well [5].

Context-dependent deep-neural-network HMMs, or CD-DNN-HMMs [5, 6], apply the classical ANN-HMMs of the 90's to traditional tied-state triphones directly, exploiting Hinton's deep-belief-network (DBN) pre-training procedure. This was shown to lead to a very promising and possibly disruptive acoustic model as indicated by a 16% relative recognition error reduction over discriminatively trained GMM-HMMs on a business search task [5, 6], which features short query utterances, tens of hours of training data, and hundreds of tied states.

This paper takes this model a step further and serves several purposes. First, we show that the exact same CD-DNN-HMM can be effectively scaled up in terms of training-data size (from 24 hours to over 300), model complexity (from 761 tied triphone states to over 9000), depth (from 5 to 9 hidden layers), and task (from voice queries to speech-to-text transcription). This is demonstrated on a publicly available benchmark, the Switchboard phone-call transcription task (2000 NIST Hub5 and RT03S sets). We should note here that ANNs have been trained on up to 2000 hours of speech before [7], but with much fewer output units (monophones) and fewer hidden layers.

Second, we advance the CD-DNN-HMMs by introducing weight sparseness and the related learning strategy and demonstrate that this can reduce recognition error or model size.

Third, we present the statistical view of the multi-layer perceptron (MLP) and DBN and provide empirical evidence for understanding which factors contribute most to the accuracy improvements achieved by the CD-DNN-HMMs.

2. The Context-Dependent Deep Neural Network HMM

A deep neural network (DNN) is a conventional multi-layer perceptron (MLP, [8]) with many hidden layers, optionally initialized using the DBN pre-training algorithm. In the following, we want to recap the DNN from a statistical viewpoint and describe its integration with context-dependent HMMs for speech recognition. For a more detailed description, please refer to [6].

2.1. Multi-Layer Perceptron—A Statistical View

An MLP as used in this paper models the posterior probability $P_{\mathbf{s}|\mathbf{o}}(s|o)$ of a class s given an observation vector o, as a stack of $(L+1)$ layers of log-linear models. The first L layers, $\ell = 0...L-1$, model posterior probabilities of hidden binary vectors h^ℓ given input vectors v^ℓ, while the top layer L models the desired class posterior as

$$P^\ell_{\mathbf{h}|\mathbf{v}}(h^\ell|v^\ell) = \prod_{j=1}^{N^\ell} \frac{e^{z^\ell_j(v^\ell)\cdot h^\ell_j}}{e^{z^\ell_j(v^\ell)\cdot 1} + e^{z^\ell_j(v^\ell)\cdot 0}}\ ,\ 0 \le \ell < L$$

$$P^L_{\mathbf{s}|\mathbf{v}}(s|v^L) = \frac{e^{z^L_s(v^L)}}{\sum_{s'} e^{z^L_{s'}(v^L)}} = \mathrm{softmax}_s(z^L(v^L))$$

$$z^\ell(v^\ell) = (W^\ell)^T v^\ell + a^\ell$$

with weight matrices W^ℓ and bias vectors a^ℓ, where h^ℓ_j and $z^\ell_j(v^\ell)$ are the j-th component of h^ℓ and $z^\ell(v^\ell)$, respectively.

The precise modeling of $P_{\mathbf{s}|\mathbf{o}}(s|o)$ requires integration over all possible values of h^ℓ across all layers which is infeasible. An effective practical trick is to replace the marginalization with the "mean-field approximation" [9]. Given observation o, we set $v^0 = o$ and choose the conditional expectation $E^\ell_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}^\ell|v^\ell\} = \sigma\big(z^\ell(v^\ell)\big)$ as input $v^{\ell+1}$ to the next layer, where $\sigma_j(z) = 1/(1+e^{-z_j})$.

MLPs are often trained with the *error back-propagation* procedure (BP) [10] with stochastic gradient ascent

$$(W^\ell, a^\ell) \quad \leftarrow \quad (W^\ell, a^\ell) + \epsilon \frac{\partial D}{\partial (W^\ell, a^\ell)} \quad , \quad 0 \le \ell \le L,$$

for an objective function D and learning rate ϵ. If the objective is to maximize the total log posterior probability over the T training samples $o(t)$ with ground-truth labels $s(t)$, i.e.

$$D \quad = \quad \sum_{t=1}^{T} \log P_{\mathbf{s}|\mathbf{o}}(s(t)|o(t)), \tag{1}$$

then the gradients are

$$\begin{aligned}
\frac{\partial D}{\partial W^\ell} &= \sum_t v^\ell(t) \cdot (e^\ell(t))^T \quad ; \quad \frac{\partial D}{\partial a^\ell} = \sum_t e^\ell(t) \\
e^L(t) &= (\log \mathrm{softmax})'(z^L(v^L(t))) \\
e^{\ell-1}(t) &= W^\ell \cdot e^\ell(t) \cdot \mathrm{diag}\left(\sigma'(z^\ell(v^\ell(t))\right)
\end{aligned}$$

with error signals $e^\ell(t)$, the component-wise derivatives $\sigma_j'(z) = \sigma_j(z) \cdot (1 - \sigma_j(z))$ and $(\log \mathrm{softmax})_j'(z) = \delta_{s(t),j} - \mathrm{softmax}_j(z)$, and Kronecker delta δ.

BP, however, can easily get trapped in poor local optima for deep networks. This can be somewhat alleviated by growing the model layer by layer, or more effectively by using the DBN pre-training procedure described next.

2.2. DBN Pre-Training

The deep belief network (DBN), proposed by Hinton [11], provides a new way to train deep generative models. The layer-wise greedy pre-training algorithm developed in DBN was later found to be also effective in training DNNs.

The DBN pre-training procedure treats each consecutive pair of layers in the MLP as a *restricted Boltzmann machine* (RBM) [11] whose joint probability is defined as

$$P_{\mathbf{h},\mathbf{v}}(h, v) \quad = \quad \frac{1}{Z_{h,v}} \cdot e^{v^T W h + v^T b + a^T h}$$

for the Bernoulli-Bernoulli RBM applied to binary v with a second bias vector b and normalization term $Z_{h,v}$, and

$$P_{\mathbf{h},\mathbf{v}}(h, v) \quad = \quad \frac{1}{Z_{h,v}} \cdot e^{v^T W h + (v-b)^T (v-b) + a^T h}$$

for the Gaussian-Bernoulli RBM applied to continuous v. In both cases the conditional probability $P_{\mathbf{h}|\mathbf{v}}(h|v)$ has the same form as that in an MLP layer.

The RBM parameters can be efficiently trained in an *unsupervised* fashion by maximizing the likelihood $\mathcal{L} = \prod_t \sum_h P_{\mathbf{h},\mathbf{v}}(h, v(t))$ over training samples $v(t)$ with the approximate *contrastive divergence* algorithm [11, 12]. We use the specific form given in [12]:

$$\begin{aligned}
\frac{\partial \log \mathcal{L}}{\partial W} &\approx \sum_t v(t) \cdot E_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}|v(t)\}^T - \sum_t \hat{v}(t) \cdot E_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}|\hat{v}(t)\}^T \\
\frac{\partial \log \mathcal{L}}{\partial a} &\approx \sum_t E_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}|v(t)\} - \sum_t E_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}|\hat{v}(t)\} \\
\frac{\partial \log \mathcal{L}}{\partial b} &\approx \sum_t v(t) - \sum_t \hat{v}(t)
\end{aligned}$$

with $\hat{v}(t) = \sigma(W\hat{h}(t) + b)$, where $\hat{h}(t)$ is a binary random sample from $P_{\mathbf{h}|\mathbf{v}}(\cdot|v(t))$.

To train multiple layers, one trains the first layer, freezes it, and uses the conditional expectation of the output as the input to the next layer and continue training next layers. Hinton and many others have found that initializing MLPs with pretrained parameters never hurts and often helps [11].

2.3. Integrating DNNs with CD-HMMs

Following the traditional ANN-HMMs of the 90's [1], we replace the acoustic model's Gaussian mixtures with an MLP and compute the HMM's state emission likelihoods $p_{\mathbf{o}|\mathbf{s}}(o|s)$ by converting state posteriors obtained from the MLP to likelihoods:

$$p_{\mathbf{o}|\mathbf{s}}(o|s) \quad = \quad \frac{P_{\mathbf{s}|\mathbf{o}}(s|o)}{P_{\mathbf{s}}(s)} \cdot \mathrm{const(s)}. \tag{2}$$

Here, classes s correspond to HMM states, and observation vectors o are regular acoustic feature vectors augmented with neighbor frames (5 on each side in our case). $P_{\mathbf{s}}(s)$ is the prior probability of state s.

However, unlike earlier ANN-HMM systems, we model tied triphone states directly. It had long been assumed that the thousands of triphone states were too many to be accurately modeled by an MLP, but [5] has shown that doing so is not only feasible but works very well. This is a critical factor in achieving the unusual accuracy improvements in this paper. The resulting model is called the Context-Dependent Deep Neural Network HMM, or CD-DNN-HMM.

3. Training CD-DNN-HMMs

In this section, we will describe the process and some practical considerations in training CD-DNN-HMMs.

3.1. Basic Training Process

DNN model learning begins with the DBN pre-training (section 2.2), using one full sweep through the 309 hours of training data for all hidden layers but the first, where we use two full sweeps. Slight gains may be obtained if the pre-training procedure sweeps the data additional times. However, this seems to be not critical [5]. RBMs are not scale-invariant, so the training corpus is normalized to zero mean and unit variance [13].

After pre-training, the DNN is fine-tuned using BP with state labels obtained through forced alignment. The model structure (phone set, HMM topology, tying of context-dependent states) and the HLDA transform are inherited from our best GMM-HMM model ML-trained on the same data, which is also used to initialize the state alignment $s(t)$ (Eq. (1)). The alignment is updated once during training.

3.2. Gradient Ascent

The gradient ascents are done in mini-batches. The mini-batch size trades off noisy gradients with slow convergence. In our case, around 1000 frames generally worked best for back-propagation (except for the first mini-batches, where we reduced it by a factor of 4), and 100-300 for pre-training.

For pre-training, we used a learning rate $\epsilon \approx 1$ per minute of speech, and for early BP iterations about 1 per 3 seconds which was reduced 40-fold after 3 sweeps. Following the original BP paper [10], we also smoothe the gradients with a first-order low-pass ("momentum," time constant $\approx$ 25 seconds). The minibatch size and learning rates used in this work are very close to what have been used in other tasks [5, 6].

Table 1: *Standard GMM-HMM vs. CD-DNN-HMM for single-pass speaker-independent recognition on five speech-to-text test sets. Also shown are our group's best-ever GMM-HMM result for three of the test sets. Transcription word-error rates are given in %.*

acoustic model & training	recognition mode	RT03S		Hub5'00	voicemails		tele-
		FSH	SW	SWB	MS	LDC	conf
GMM 40-mix, ML, SWB 309h	single-pass SI	30.2	40.9	26.5	45.0	33.5	35.2
GMM 40-mix, BMMI, SWB 309h	single-pass SI	27.4	37.6	23.6	42.4	30.8	33.9
CD-DNN 7 layers x 2048, SWB 309h, this paper	single-pass SI	18.5	27.5	16.1	32.9	22.9	24.4
(rel. change GMM BMMI $\rightarrow$ CD-DNN)		(-33%)	(-27%)	(-32%)	(-22%)	(-26%)	(-28%)
GMM 72-mix, BMMI, Fisher 2000h	multi-pass adaptive	18.6	25.2	17.1	-	-	-

Gradient ascent with small mini-batches cannot be meaningfully parallelized across multiple servers. Instead, we utilize multiple NVidia Tesla GPGPU devices connected to a single host. With mini-batches, much of the algorithm can be written as operations on large matrices, for which GPGPUs are an excellent match [11]. To reduce data transfer, the model parameters (W, a, b, related accumulators) are stored distributed in disjoint stripes across the GPGPU devices, while node values are scattered/gathered during computation. The speed-up from using GPGPUs is so large that even seemingly simple operations like tracking the training likelihood notably impact runtime if they are not also moved to the GPGPU.

Gradient ascent in mini-batches works best if data is presented in random order, but that is problematic because relevant speech corpora do not fit into RAM. As a compromise, we limit randomization to within a rolling window of 48 hours of data which is held in RAM.

3.3. Weight Sparseness

Enforcing weight sparseness is equivalent to incorporating L0 and approximate L1 regularizations to the training criterion. We have found, however, that enforcing sparseness during pre-training or at the early stage of BP is harmful. The strategy adopted in this paper, which proved to be effective, is to fine-tune the models without sparseness constraint until the weights are relatively stable. In our experiments, we started enforcing sparseness after sweeping the data at least four times. The strategy is to force all weights smaller than a threshold to be 0, and to continue fine-tuning the model with BP while keeping all weights smaller than half of the threshold to be 0. This is to prevent reappearance of small weights while avoiding sudden removal of weights that shrink below the original threshold.

4. Experimental Results

4.1. Setup and Core Results

We evaluate the effectiveness of CD-DNN-HMMs on the task of speech-to-text transcription using the 309-hour Switchboard-I training set [14]. The system uses 13-dimensional PLP features with windowed mean-variance normalization and up to third-order derivatives, reduced to 39 dimensions by HLDA. The speaker-independent crossword triphones use the common 3-state topology and share 9304 CART-tied states.

The GMM-HMM baseline system has 40 Gaussian mixtures trained with maximum likelihood (ML) and refined discriminatively with the boosted maximum-mutual-information (BMMI) criterion. Using more than 40 Gaussians did not improve the ML result.

The CD-DNN-HMM system replaces the Gaussian mixtures with likelihoods derived from the MLP posteriors (Eq. (2)), while leaving everything else the same. One question was whether a softmax layer for 9304 tied states would work, which to our knowledge has never been tried before.

The primary test set is the FSH half of the 6.3h Spring 2003 NIST rich transcription set (RT03S), while the 1831-segment SWB part of the NIST 2000 Hub5 eval set was used for system development. The trigram language model was trained on the 2000h Fisher-corpus transcripts and interpolated with a written-text trigram. Test-set perplexity with the 58k dictionary is 84.

Recognition is done in a single-pass without any speaker adaptation. Table 1 shows the core result: The word-error rate (WER) is reduced from the GMM-HMM's 27.4% (BMMI) to CD-DNN-HMM's 18.5%—a 33% relative reduction, which is quite significant. Note that even with our best CD-DNN-HMM system, the average frame level state error rate is only slightly below 60%, and the average log posterior probability is -1.76.

The 309-hour CD-DNN-HMM system also outperforms our best speaker-adaptive multi-pass system (18.6%, last row), which uses additional acoustic training data (the 2000h Fisher corpus), VTLN, and ROVER, by 0.1% absolute. For comparison, [7] reports around 17% on this task, and results on more recent but similar test sets are around 17-18% [15].

Much of the gain also carries over to four less well-matched test sets—the cell-phone SW portion of RT03S, 110 in-house voice-mails, 249 LDC voicemails from [16], and a 55-minute in-house set of wideband recordings of teleconferences. The relative improvements are 22–28% over BMMI using the same acoustic and language models (except for voice-mails, where the LM includes additional domain data).

4.2. Influence of Layers and Training Procedures

Table 2 shows the effect of layers and training procedures for our development set (Hub5'00). The ANNs were trained using state alignments generated from the GMM-HMM ML model (whereas in Table 1, alignments were updated with CD-DNN-HMMs after 7 passes through the data). Increasing from 1 to 9 layers (2048 hidden nodes at each layer) reduces WER from 24.2% to 17.0% if DBN pre-training is exploited. The same WER is reached by a five-layer model with 3072 hidden nodes, whereas a 9-layer model with only 1024 nodes reaches 17.9%.

Pure BP training (random initialization without DBN pre-training) is nearly 2 percentage points worse for 2 and 3 layers, but the gap reduces to below 1 point for 4, 5, and 7 layers. The gap is smaller for layer-growing BP (LBP, replacing the respective top layer after BP convergence with a new random hidden layer). It stays within 0.5 points and reaches the DBN for 4 layers, at a cost of an $\mathcal{O}(L)$ times larger training time.

Our results seem to suggest that although pre-training helps, it is not critical for up to 5 hidden layers. Even without pre-training, CD-DNN-HMMs with two or more hidden layers easily outperform the discriminatively trained GMM-HMMs. We believe that direct modeling of tied triphone states is the enabler

Table 2: *Comparing different number of layers/hidden nodes and training algorithms: BP with DBN pre-training, pure back-propagation (BP), and layer-wise BP-based model growing (LBP). Shown are word-error rates in % for Hub5'00 SWB.*

$L \times N^{\ell}$	DBN	BP	LBP	$1 \times N^{1}$	DBN
1×2k	24.2	24.3	24.3	1×2k	24.2
2×2k	20.4	22.2	20.7	-	-
3×2k	18.4	20.0	18.9	-	-
4×2k	17.8	18.7	17.8	-	-
5×2k	17.2	18.2	17.4	1×3772	22.5
7×2k	17.1	17.4	17.4	1×4634	22.4
9×2k	17.0	-	-	-	-
9×1k	17.9	-	-	-	-
5×3k	17.0	-	-	-	-
-	-	-	-	1×16k	22.1

of the superior performance of CD-DNN-HMMs.

The right column in Table 2 shows that a single-hidden layer model with the same number of parameters as the deep network performs significantly worse, and even a significant increase of hidden nodes to 16k cannot make good on the difference. This suggests that the deep models' additional factorization by means of intermediate layers is indeed important.

4.3. Other Influence Factors

Table 3 shows the system development for a CD-DNN-HMM with 7 hidden layers to illustrate the impact of several factors. Starting with a 1-hidden-layer MLP with about as many parameters as the 7-hidden-layer CD-DNN-HMM, we see that the use of context frames yields a 14% relative improvement, nearly twice than what is achieved using neighbor frames with GMM-HMMs (LDA, fMPE). The effective exploitation of neighbor frames is one of the DNN's strengths.

Rearranging the parameters from 1 into 7 layers shows the modeling power of intermediate layers: Errors are reduced by 24% relative. Updating the state alignment using the CD-DNN-HMM yields another 4% relative reduction.

Finally, moderate sparsification of the weight matrices that zeroes about two thirds of the weights yields a small improvement of 0.3 points. Trade-offs can be further made between the size of the model (and thus the decoding time).

The WER results presented may further be improved by tuning transition parameters [6].

5. Conclusion

In this paper, we have scaled up CD-DNN-HMMs [5, 6] to large data sets for speech-to-text transcription. We have shown feasibility of converting state posteriors to emission likelihoods for a softmax layer of over 9,000 nodes, and of training a DNN on hundreds of hours of data using multiple GPGPUs. We have demonstrated that recent improvements from using CD-DNN-HMMs on smaller tasks [5, 6] do carry over to large corpora and speech-to-text transcription, achieving up to 33% relative WER reduction over a discriminatively trained GMM-HMM.

Our results suggest that the three critical factors that contribute to the remarkable accuracy gains from the CD-DNN-HMMs are (1) the direct modeling of tied triphone states through the DNN; (2) the effective exploitation of neighbor frames by the DNN; and (3) the efficient and effective modeling ability of deeper networks. DBN pre-training leads to best results, but only by a margin below 1 point for relevant setups.

Table 3: *Comparing different influence factors on CD-DNN-HMM accuracy. 'nz' means 'non-zero.' Word-error rates in % for Hub5'00 SWB.*

acoustic model	#params	WER (r. chg.)
GMM 40 mix, BMMI	29.4M	23.6
CD-DNN 1 layer×4634 nodes	43.6M	26.0 (+10%)
+ 2×5 neighbor frames	45.1M	22.4 (-14%)
CD-DNN 7 layers×2048 nodes	45.1M	17.1 (-24%)
+ updated state alignment	45.1M	16.4 (-4%)
+ sparsification 66%	15.2M nz	16.1 (-2%)

The two main challenges we are facing are to improve efficiency of the training to scale up further to thousands of hours of data, and to develop effective speaker-adaptation techniques for CD-DNN-HMMs.

6. Acknowledgements

We would like to thank Asela Gunawardana, Kit Thambiratnam, Jasha Droppo, Nikko Ström, and Andreas Stolcke for their help with the baseline system and fruitful discussion. Our special thanks go to Ajith Jayamohan and Igor Kouzminykh of the MSR Extreme Computing Group for access to a Tesla server farm, without which this work would not have been possible.

7. References

[1] S. Renals, N. Morgan, H. Bourlard, M. Cohen, and H. Franco, "Connectionist Probability Estimators in HMM Speech Recognition," IEEE Trans. Speech and Audio Proc., January 1994.

[2] H.-W. Hon, "Vocabulary-Independent Speech Recognition—The VOCIND system," Ph.D. thesis, Carnegie Mellon University, Pittsburgh, 1992.

[3] H. Franco *et al.*, "Context-Dependent Connectionist Probabilty Estimatation in a Hybrid Hidden Markov Model–Neural Net Speech Recognition System," Computer Speech and Language, vol. 8, pp. 211–222, 1994.

[4] J. Fritsch *et al.*, "ACID/HNN: Clustering Hierarchies of Neural Networks for Context-Dependent Connectionist Acoustic Modeling," Proc. ICASSP 1998, May 1998.

[5] D. Yu, L. Deng, and G. Dahl, "Roles of Pretraining and Fine-Tuning in Context-Dependent DNN-HMMs for Real-World Speech Recognition," in Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Dec. 2010.

[6] G. Dahl, D. Yu, L. Deng, and A. Acero, "Context-Dependent Pre-Trained Deep Neural Networks for Large Vocabulary Speech Recognition", IEEE Trans. Speech and Audio Proc., Special Issue on Deep Learning for Speech and Lang. Processing, 2011 (to appear).

[7] A. Stolcke *et al.*, "Recent Innovations in Speech-to-Text Transcription at SRI-ICSI-UW," IEEE Trans. on Audio, Speech, and Lang. Processing, vol. 14, No. 5, Sep. 2006

[8] F. Rosenblatt, "Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms", Spartan Books, Wash. DC, 1961.

[9] L. K. Saul, T. Jaakkola, and M. I. Jordan, "Mean Field Theory for Sigmoid Belief Networks", Journal: Computing Research Repository–CORR, pp. 61-76, 1996.

[10] D. Rumelhart, G. Hinton, and R. Williams, "Learning Representations By Back-Propagating Errors," Nature, vol. 323, Oct. 1986.

[11] G. Hinton, S. Osindero, and Y. Teh, "A Fast Learning Algorithm for Deep Belief Nets", Neural Computation, vol. 18, pp. 1527–1554, 2006.

[12] G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines", Technical Report UTML TR 2010–003, University of Toronto, 2010.

[13] A. Mohamed, G. Dahl, and G. Hinton, "Deep Belief Networks for Phone Recognition," in Proc. NIPS Workshop Deep Learning for Speech Recognition, 2009.

[14] J. Godfrey and E. Holliman, "Switchboard-1 Release 2," Linguistic Data Consortium, Philadelphia, 1997.

[15] IEEE Trans. Audio, Speech, and Lang. Processing, Special Section on Rich Transcription, vol. 14, No. 5, pp. 1490-1608, 2006.

[16] M. Padmanabhan et al., "Voicemail Corpus Part I and II," Linguistic Data Consortium, Philadelphia, 1998 and 2002.