CONTEXT-DEPENDENT DEEP NEURAL NETWORKS FOR AUDIO INDEXING OF REAL-LIFE DATA

Gang Li[1], *Huifeng Zhu*[1,3], *Gong Cheng*[1], *Kit Thambiratnam*[2], *Behrooz Chitsaz*[4], *Dong Yu*[4], *and Frank Seide*[1]

[1] Microsoft Research Asia, [2] Microsoft Search Technology Center, 5 Danling Street, Haidian District, Beijing 100080, P.R.C.
[3] Chinasoft, 55 South College Road, Haidian District, Beijing 100080, P.R.C.
[4] Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
{ganl,gocheng,behroozc,dongyu,fseide}@microsoft.com

ABSTRACT

We apply Context-Dependent Deep-Neural-Network HMMs, or CD-DNN-HMMs, to the real-life problem of audio indexing of data across various sources. Recently, we had shown that on the Switchboard benchmark on speaker-independent transcription of phone calls, CD-DNN-HMMs with 7 hidden layers reduce the word error rate by as much as one-third, compared to discriminatively trained Gaussian-mixture HMMs, and by one-fourth if the GMM-HMM also uses fMPE features.

This paper takes CD-DNN-HMM based recognition into a real-life deployment for audio indexing. We find that for our best speaker-independent CD-DNN-HMM, with 32k senones trained on 2000h of data, the one-fourth reduction does carry over to inhomogeneous field data (video podcasts and talks). Compared to a speaker-adaptive GMM system, the relative improvement is 18%, at very similar end-to-end runtime.

In system building, we find that DNNs can benefit from a larger number of senones than the GMM-HMM; and that DNN likelihood evaluation is a sizeable runtime factor even in our wide-beam context of generating rich lattices: Cutting the model size by 60% reduces runtime by one-third at a 5% relative WER loss.

Index Terms— speech recognition, deep neural networks, deep learning, audio indexing

1. INTRODUCTION

This paper describes our experience of deploying *Context-Dependent Deep Neural Network Hidden Markov Models*, or CD-DNN-HMMs, in an existing commercial service for LVCSR-based audio indexing.

The CD-DNN-HMM [1, 2] is a recent acoustic-modeling technique for HMM-based speech recognition that can greatly outperform conventional Gaussian-mixture based HMMs. Like the ANN-HMMs of the 90's [3], CD-DNN-HMMs replace GMMs by an artificial neural network, but they differ in significantly increased network depth (7 or more hidden layers) and in that they *directly* model tied *context-dependent* states (senones) [4, 1] instead of factorizing the networks [5, 6]. [1] and [2] achieved relative error reductions of up to 33% on the Switchboard benchmarking task of transcribing telephone calls. It is not known yet, however, how far these gains carry over to tasks with much larger acoustic mismatch and variety, such as a public audio-indexing service.

This paper describes our experience with deploying CD-DNN-HMMs in a real-life application, the commercially available Microsoft Research Audio/Video Indexing Service [7], or MAVIS. The specific questions addressed include:

- Are parameter settings optimal for the GMM-HMM also optimal for DNN (particularly, the number of senones)?
- What is the runtime impact of the different likelihood evaluation, and what are first approaches to optimize it?
- What are obtainable accuracy improvements from DNN on real-life audio/video data encountered in the MAVIS service?

This paper is organized as follows. After a review of the CD-DNN-HMM, we will discuss practical issues of deploying CD-DNN-HMMs in Section 3, followed by a system description of the MAVIS system in Section 4. Section 5 will then present evaluation results regarding the above three questions.

2. THE CONTEXT-DEPENDENT DEEP-NEURAL-NETWORK HMM

A deep neural network (DNN) is a conventional multi-layer perceptron (MLP [8]) with many layers, where training is typically initialized by a pretraining algorithm. Below, we describe the DNN, briefly touch upon its training, and describe its integration with context-dependent HMMs for speech recognition. Extra details can be found in [9].

2.1. Deep Neural Network

A DNN 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 conditionally

independent hidden binary units h^ℓ given input vectors v^ℓ, while the top layer L models the desired class posterior as

$$\begin{aligned} P^\ell_{\mathbf{h}|\mathbf{v}}(h^\ell|v^\ell) &= \prod_{j=1}^{N^\ell} \frac{e^{z^\ell_j(v^\ell)\cdot h^\ell_j}}{e^{z^\ell_j(v^\ell)\cdot 1} + e^{z^\ell_j(v^\ell)\cdot 0}} \quad , \quad 0 \le \ell < L \\ P^L_{\mathbf{s}|\mathbf{v}}(s|v^L) &= \frac{e^{z^L_s(v^L)}}{\sum_{s'} e^{z^L_{s'}(v^L)}} = \mathrm{softmax}_s(z^L(v^L)) \\ z^\ell(v^\ell) &= (W^\ell)^T v^\ell + a^\ell \end{aligned} \tag{1}$$

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" [10]. 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, with component-wise sigmoid $\sigma_j(z) = 1/(1+e^{-z_j})$.

2.2. Training

DNNs, being 'deep' MLPs, can be trained with the well-known *error back-propagation* procedure (BP) [11]. Because BP can easily get trapped in poor local optima for deep networks, it is helpful to 'pretrain' the model in a layer-growing fashion. In [12] we have shown that two pretraining methods, *deep belief network* (DBN) pretraining [13, 14, 15] and *discriminative pretraining*, are approximately equally effective.

The CD-DNN-HMM's model structure (phone set, HMM topology, tying of context-dependent states) is inherited from a matching GMM-HMM model that has been ML-trained on the same data. That model is also used to initialize the class labels $s(t)$ through forced alignment. This alignment is updated a few times during training using the DNN model.

DNN training is an expensive operation. To give an indication, the main model used in this paper has N=3072 hidden nodes and J=32k senones. The total number of parameters is 157 million, with the majority being concentrated in the output layer. Using a single server equipped with a high-end NVidia Tesla S2070 GPGPU, it took 59 days to train this model. (The pipeline technique described in [16] had not been available yet in the work at hand.)

2.3. Speech Recognition with CD-DNN-HMMs

As in the traditional ANN-HMMs of the 90's [3], the acoustic model's Gaussian mixtures are replaced with an MLP, which computes the HMM's state emission likelihoods $p_{\mathbf{o}|\mathbf{s}}(o|s)$ by converting state posteriors from the MLP to likelihoods:

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

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, and $\mathrm{const}(s)$ denotes a constant w.r.t. s, i.e. a value that does *not* depend on s.

However, unlike earlier ANN-HMM systems, the CD-DNN-HMM models 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 [1] 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. Hence the name Context-Dependent Deep Neural Network HMM, or CD-DNN-HMM.

3. PRACTICAL USE OF CD-DNN-HMMS

In the following we describe two practical considerations; Model size and likelihood evaluation in the live system.

3.1. Model Size

In our earlier work [2], we experimented with different hidden-layer dimensions N on a 309-hour training set, and found 2048 to be a good choice. The number of senones for the CD-DNN-HMM model had just been chosen to be the same as for our best GMM-HMM. However, there is no reason to assume that this choice is optimal.

To find out, we conducted a parameter sweep over the number of senones on our basic Switchboard setup (309 hours of training, ML-trained GMM-HMM), reported in Section 5.

Our deployed system was, however, going to be trained on the full 2000h Fisher+Switchboard set, which takes up to two months on a high-end GPGPU. Not having the compute capacity for a full enumeration of number of hidden dimensions and senones, we instead chose a number of parameters that has roughly the same relative increase from the 309h model as the optimal GMMs for the respective training-set sizes.

3.2. Likelihood Evaluation at Runtime

The cost of likelihood evaluation is an important factor for real-life deployments. Unlike our research environment, we do not assume that GPGPUs are available in our deployment environment, but instead rely on standard Intel CPUs.

We exploit that from the output layer we do not actually need normalized posteriors, because the softmax denominator is identical for all states. To compute the log likelihood values used in our LVCSR decoder, we rewrite Eq. (2) as:

$$\log p_{\mathbf{o}|\mathbf{s}}(o|s) - \mathrm{const}(s) = (W^L_{*s})^T v^L + a^L_s - \log P_{\mathbf{s}}(s)$$

where $\mathrm{const}(s)$ is unknown but cancels out in any pruning and best-path decision and derived word posteriors.[1]

[1]The *absolute* acoustic scores in the generated word lattices will be skewed, but absolute acoustic scores are rarely—if ever—needed directly.

This allows for two optimizations similar to those used for GMMs: First, no logs or exponential functions are computed ($\log P_{\mathbf{s}}(s)$ is tabulated), and secondly and more importantly, likelihoods can be computed *on demand*. E.g., in our system, due to pruning, only about 30% of all likelihoods are ever needed. For a DNN, however, this optimizaton only applies to the output layer—the hidden layers still need to be computed in full. Yet, considering that the output layer makes up over half of the model parameters, savings are still considerable.

To take advantage of the CPU cache, we borrow another technique from GMMs; we compute likelihoods in *batches* of multiple frames, such that the model parameters are only read from main RAM once per batch of frames. In an on-demand context, this leads to some excess computation, yet a batch size of 4 is a beneficial compromise (as it is for Gaussians).

The inputs to the output layer, the v^L, must be fully computed according to the sigmoid cascade in Eqs. (1). Thus, we can choose larger batches (like 8 frames); we are only limited by their induced latency.[2] A complication is that with this, batching between the hidden and the output layer is not synchronized, which adds to code complexity.

4. MAVIS SYSTEM DESCRIPTION

The Microsoft Research Audio/Video Indexing System (MAVIS), is a set of software components that use speech recognition to enable searching of digitized spoken content [7]. Target domains include meetings, conference calls, voice mails, presentations, online lectures, and Internet video. We want to briefly touch upon some of its aspects: its architecture; a technique for vocabulary adaptation; and the indexing of word lattices with a standard off-the-shelf full-text search engine.

4.1. Architecture

MAVIS performs speech recognition "in the cloud," while the actual video-search engine runs locally at the customer site. This split was chosen because many enterprises already have compatible search engines and related web services deployed in their intranet or web presence, but regularly lack the compute capacity to perform the expensive speech recognition on large back-logs of hundreds or thousands of hours of data (on the other hand, customers often have local capacity to continue indexing of newly created data).

The MAVIS speech-recognition web service (running in Windows Azure) implements a processing queue that receives user-uploaded media and processes it into word lattices, which are returned to the user as opaque binary objects (called Audio Indexing BLOBs or AIBs).

The processing has two phases, with parallel processing in each phase: First, vocabulary adaptation (described next) based on the input file's metadata, which involves wait times for web downloads, is executed in parallel to a first-pass recognition to get segmentation information and feature extraction followed by acoustic adaptation. The second phase then performs the main recognition with updated models and wider beams, to generate the rich lattices that form the AIBs.

These AIBs are then ingested into the search engine that is the back-end for the video-search and viewing experience. Fig. 1 illustrates the process.

[2]For use in a real-time system, it would even be straight forward to parallelize computation of the lower layers across multiple cores if needed.

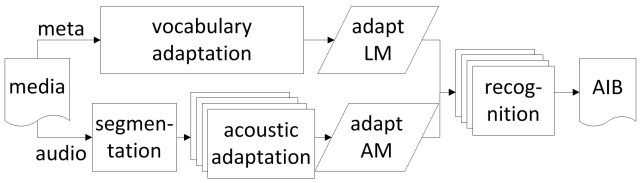

Fig. 1. *MAVIS speech recognition block diagram.*

4.2. Vocabulary Adaptation

Domain-independent speech recognition and Spoken Document Retrieval (SDR) systems are plagued by LM mismatch and out-of-vocabulary (OOV) issues. MAVIS uses a technique for per-file vocabulary adaptation to allow blind indexing and transcription of inhomogeneous speech collections.

The idea is to use available supporting metadata of a media file to generate keywords that are sent to a web search engine to retrieve related content. That content is then used to augment the language model and dictionary. In [17], we show that a Figure of Merit of 48% is achieved for (originally) OOV words, while in-vocabulary queries improve from 68 to 75%.

4.3. Word-Lattice Indexing with Full-Text Search Engines

Like many full-text search engines, the engine used for MAVIS, Microsoft SQL Server, does not have a native capability of indexing word lattices. To address this, we use an approximation dubbed "TALE" (Time-Anchored Lattice Expansion, [18]) which constructs word-position aligned lattices similar to confusion networks, which are then ingested through an input plug-in. Through a "creative abuse" of the plug-in APIs, specifically the word breaker, we are able to index all alternative word tokens that occur at a given word position, even though SQL Server's full-text engine was not designed for that [19]. Word confidence (posteriors probabilities) is encoded through mangling of word names.

5. EXPERIMENTAL RESULTS

5.1. Setup and Core Results

The speech recognizers used in this paper are trained for the task of speech-to-text transcription, on 2000 hours worth of transcribed telephone conversations from the Switchboard

Table 1. *Basic improvement of CD-DNN-HMM over the GMM-HMM (L=7 hidden layers, N=hidden dimensions, J=number of senones). Word-error-rates (WERs) in percent.*

setup	WER[%] Hub5'00 SWB	WER[%] RT03S FSH
GMM-HMM (DT), 309h SWBD-I, J=9304	23.6	27.4
CD-DNN-HMM N=2048, J=32k, re-aligned	15.8	18.9
(rel. change from GMM)	(-33%)	(-31%)
GMM-HMM (ML), 2000h Fisher, J=18k	24.3	25.7
+ fMPE + DT	19.6	20.5
CD-DNN-HMM N=3k, J=32k	14.4	15.6
(rel. chg. from GMM-fMPE-HMM)	(-27%)	(-24%)

and Fisher benchmark collections [20]. There was no data specific to the audio-indexing task available to us, and thus, the same Switchboard acoustic models are also used for recognizing the audio/video content. For the senone-tuning experiment, we used a smaller training set consisting only of the 309h SWBD-I subset.

The systems use 13-dimensional PLP features with rolling-window mean-variance normalization and up to third-order derivatives, which for the GMM-HMM systems is reduced to 39 dimensions by HLDA, and the usual 3-state cross-word triphones with CART-tied triphone states (senones).

The GMM-HMM baseline models are trained with maximum likelihood (ML) and refined discriminatively (DT) with the boosted maximum-mutual-information (BMMI) criterion. For the 2000h set, we also use the fMPE feature transform.

The CD-DNN-HMM systems are derived from matching GMM-HMM systems with the same number of senones and trained on the same data, which are also used to create forced state alignments for use as the initial ground truth. The CD-DNN-HMMs completely inherit the GMM-HMM's structure and topology, and just replace the likelihood-emission GMMs with a DNN. One difference to the GMM-HMMs is that the features used by the DNNs skip the HLDA transform, because we found that they are able to learn the transform implicitly as part of the first layer [12], at a marginally better result.

The trigram language model was trained on the 2000h Fisher transcripts and interpolated with a written-text trigram for broader coverage to account for inhomogenous topics. Test-set perplexity on our dev set with the 58k lexicon is 84.

The models are evaluated on Switchboard data as well as field data from our MAVIS deployment. The Switchboard data are the 1831-segment SWB part of the NIST 2000 Hub5 set (we use this as our development set) and he FSH half of the 6.3h Spring 2003 NIST rich transcription set (RT03S).

The MAVIS field data consists of 42 presentation-style videos (total duration 54h) from 5 sources of different topical areas: video broadcasts of legislative hearings of several US states; medical reports and training videos (including hearty demonstrations of open-heart surgery); popular science talks e.g. about robotics or the large hadron collider; internal talks given at Microsoft Research; and presentations from Microsoft's developers conference (PDC).

Table 2. *Effect of model size. Word-error rates in percent. The respective optimal choices are marked in bold-face for the development set (Hub5'00-SWB).*

#sen. (J)	GMM-HMM (ML) Gaus-sians	Hub5'00 SWB	RT03S FSH	CD-DNN-HMM #hid. (N)	Hub5'00 SWB	RT03S FSH
309h (SWBD-I)						
9.0k	60	26.2	29.9	2k	17.2	19.8
11k	48	**26.1**	30.3	2k	17.1	19.5
15k	40	26.1	30.1	2k	17.2	19.5
18k	40	26.1	30.3	2k	16.7	19.3
22k	36	26.3	31.2	2k	16.7	19.4
27k	28	26.5	31.7	2k	**16.4**	19.4
32k	24	27.5	32.2	2k	16.4	19.5
2000h (SWBD+Fisher)						
18k	72	24.3	25.7	2k	15.2	16.3
32k	n/a	n/a	n/a	3k	**14.4**	15.6

This paper being about acoustic modeling, accuracy is evaluated as word error rates (WER) (rather than, say, search-accuracy metrics like Figure of Merit).

Table 1 shows the effectiveness of DNNs. Using 309h of training data, our best CD-DNN-HMM reduces word errors by about one-third from GMM-HMM DT baseline (from 23.6 and 27.4 to 15.8 and 18.9%, respectively). This result is slightly better than what we reported in [2] due to increased model size. Compared to [2], this paper now also includes results for training on the full 2000h Fisher corpus. Compared to the *ML*-trained baseline GMM system, a 40% rel. reduction is achieved, but the fair baseline to compare to is the one that uses DT + fMPE: For that, the reduction is still about one-fourth (from 19.6 and 20.5 to 14.4 and 15.6%, respectively). We believe that this is one of the best published results for single-pass speaker-independent recognition on this task.

5.2. Model Size

We raised the question whether it is indeed optimal to choose the number of senones for the DNN to be the same as the optimal one for the GMM-HMM. Table 2 shows WERs for a range of models trained[3] with the 309h SWBD-I set, and it indicates that this is indeed *not* so: CD-DNN-HMM can accomodate twice as many senones or more. While the GMM has an optimum at J=11k senones and shows effects of over-training for 22k, the DNN benefits from up to 27k.

[3]The experiments shown in Table 2 use slightly simpler setups compared to Table 1: The GMM-HMMs are not discriminatively trained, and for 309h, the CD-DNN-HMMs did not use state-label re-alignment. The shown numbers of Gaussians are those that led to the respective best WER.

Table 3. *ASR engine runtime (time cost per 10-ms frame) and WER for both our SWBD baseline and the MAVIS data (L=7 hidden layers, N=2k hidden nodes per layer, J=18k senones) vs. (N=3k, J=32k). Since the softmax output layer ($\ell = L$) is computed on-demand, we show its realtime factor separately from the fully-computed hidden layers ($\ell < L$).*

test set	model size			runtime [ms per frame]			WER
				DNN layer(s)		Viterbi	[%]
	N	J	MB	$\ell < L$	$\ell = L$	expans.	
Hub5'00-SWB	2k	18k	63	5.8	3.1	10.1	15.2
	3k	32k	157	12.6	5.6	10.1	14.4
MAVIS	2k	18k	63	6.4	4.0	17.4	31.0
	3k	32k	157	12.1	6.3	14.4	29.8

Due to the runtime cost, the 2000h model was dimensioned by extrapolating from the 309h results. The resulting model has a hidden-layer dimension of N=3072 and J=32k senones (157 million parameters). It took 59 days to train.

The 309h and 2000h GMM-HMM baseline models in Table 1 have 29 and 102 million parameters, respectively.

5.3. Runtime

Likelihood evaluation with DNNs at present requires significantly higher computational effort. While for our configuration with J=18k senones, the total number of multiply-and-accumulate operations for a 72-mixture GMM model is 101 million compared to 63 for a respective DNN (with hidden-layer dimension N=2048), one must consider that for the DNN only the output layer can be computed on-demand. The lower layers, which comprise nearly half of all operations, must be fully computed. If, as in our system, only 30% of all state likelihoods are evaluated on average, this difference matters. In addition, there exist numerous speed-up techniques for GMMs (not to mention highly-tuned cache-optimal implementations), for which DNN equivalents remain to be found.

Table 3 shows processing time per frame broken down into DNN hidden-layer computation, DNN output-layer computation, and Viterbi state expansion. For the smaller 2000h DNN above (N=2k, J=18k), DNN computation makes up about 40-50% of the total computation. For the full-size model (N=3k, J=32k), this grows to around 60%. This is despite the fact that the decoder itself uses very wide beams in order to generate rich lattices, which is important for audio indexing.

Introducing MAVIS data, the table also shows the effect of the data mismatch—for the small model, total computation time increases by 45% compared to the well-matched Switchboard test set. This is a natural side effect of score-based beam pruning (both setups use identical beam widths). For the full-size model, the runtime for MAVIS data is only 16% higher; this is because here, the hidden layers, which are independent of search space, constitute a larger part of computation.

5.4. Recognizing Real-Life Audio/Video Data

Table 4 shows the overall results on the five MAVIS corpora, for four relevant setups: A ML-trained baseline GMM-HMM system, denoted as GMM-HMM (ML); an enhanced version using fMPE and DT (GMM-fMPE-HMM (DT)); and two CD-DNN-HMM configurations of different model sizes (not using fMPE). All models are Switchboard models trained on the 2000h Fisher/SWBD training data, as no domain-specific acoustic training data was available. SI denotes speaker-independent recognition, while AA and VA stand for acoustic and vocabulary adaptation (cf. section 4.2), respectively.

First, we see that the general range of word-error rates on the real-life deployment data is broadly twice as high (in the 30s) as for the very well matching SWBD database (15-20%, Table 1). The legislative-hearings set has the worst accuracy, probably due to recording conditions and data compression. We also see that using fMPE and DT (column labeled GMM-fMPE-HMM) reduces word errors, but the relative gain is significantly less (from 36.6 to 34.4% for the fully adapted system, 6% relative) than for the, again, well-matched SWBD setup (Table 1, 2000h GMM setup), where it is about 20% relative. The two adaptation techniques have a worthwhile effect, about 13% gain, for the GMM-fMPE-HMM.[4]

The table then shows results for two configurations of CD-DNN-HMM, a smaller one (N=2k, J=18k) and the full-size one (N=3k, J=32k). We include the smaller configuration to show the impact of the very large number of DNN model parameters on runtime—cutting the model size by 60% (Table 3) sacrifices less than one point of WER (29.1 instead of 28.2%) while reducing the end-to-end runtime by one-third (from 3.8 to 2.6 times realtime).[5] This is of relevance for our commercial web-service deployment, since runtime is proportional to hosting cost.

In terms of accuracy, compared to GMM-fMPE-HMM, the full-size CD-DNN-HMM model reduces WER by a relative 25% for speaker-independent recognition (from 39.5% to 29.6%). This is neatly consistent with our observation on Switchboard (Table 1), where we saw gains of 27 and 24%. And remember that those GMM baselines use fMPE and discriminative sequence-level learning, while the DNN systems do not. The relative gain from using CD-DNN-HMMs observed on the Switchboard benchmark does indeed carry over to the more varied and mismatched real-life audio/video data.

Lastly, for deployment, we should use adaptation. Both the GMM and the DNN systems use vocabulary adaptation (VA). Acoustic adaptation (AA), however, was only available

[4] Note that vocabulary adaptation has a larger benefit when measured by audio-search accuracy [17]. This is because the content words that users tend to be looking for are more likely to be rare and out of vocabulary.

[5] We found this surprising, but consider that these times are actual measurements in the deployment environment, and thus somewhat unreliable. More reliable are the offline measurements in Table 3. Extrapolating from those, we would expect the small model to be around 15% faster. We suspect that the discrepancy is caused by paging. Further investigation is warranted.

Table 4. *Comparison of GMM-HMM and CD-DNN-HMM configurations for a number of real-life data sets (word-error rates in percent and end-to-end realtime factor). SI=Speaker Independent, AA=Acoustic Adaptation, VA=Vocabulary Adaptation.*

test set (topic area)	meta data?	data size	GMM-HMM (ML)			GMM-fMPE-HMM (DT)			CD-DNN-HMM N=2k, K=18k		CD-DNN-HMM N=3k, J=32k	
			SI	+ AA	+ VA	SI	+ AA	+ VA	SI	+ VA	SI	+ VA
state legislative hearings	partial	25.5h	51.1	48.9	47.7	52.8	47.5	46.8	39.7	38.8	38.5	37.8
medical topics and training	partial	11.1h	41.1	39.5	37.0	39.7	36.8	33.4	31.5	28.4	30.7	27.9
popular science talks	partial	5.8h	28.7	26.1	25.8	28.9	25.6	23.8	20.4	19.9	19.6	19.0
MS Research internal talks	yes	5.5h	39.6	37.8	37.0	37.7	35.0	33.9	31.1	30.3	30.0	29.2
MS developers conference	yes	5.7h	40.6	37.8	35.4	38.6	35.2	34.1	30.5	28.1	29.2	27.4
total		53.6h	40.2	38.0	36.6	39.5	36.0	34.4	30.6	29.1	29.6	28.2
realtime factor					3.7×			3.7×		2.6×		3.8×

to the GMM systems, because our implementation of adaptation for DNNs [12] requires a GPGPU to be efficient, which we do not want to assume for web-service deployment. With adaptation in this form, the final relative gain shrinks to 18% (improvement from 34.4% to 28.2%), which is still rather worthwhile.

6. CONCLUSION

We have applied Context-Dependent Deep-Neural-Network HMMs, or CD-DNN-HMMs, to the real-life problem of audio indexing of data across various sources.

Across an inhomogeneous range of field data (video podcasts and talks) from a commercial deployment, our best speaker-independent CD-DNN-HMM—which has 7 layers of 2048 hidden nodes and 32k senones, and was trained on 2000 hours of Fisher and Switchboard data—achieves *the same one-fourth relative error reduction* that we observe on the Switchboard test sets, compared to a matching, discriminatively trained GMM-HMM, which in addition uses HLDA, fMPE features, and discriminative sequence-level learning. This is despite larger variability and training/test mismatch, and an error-rate level nearly twice as high.

Compared to a GMM that also uses speaker adaptation, the relative gain from using DNNs is a respectable 18%.

We further find that the optimal number of senones for the DNN is larger (double or more) than that for the GMM-HMM; and that the likelihood-evaluation cost for DNNs is a significant portion of overall runtime, partially because only part of it lends itself to on-demand computation. Yet, the end-to-end runtime is the same since the DNN does not use speaker adaptation and the associated additional recognition pass. By using a 60% smaller DNN, end-to-end runtime can be cut by 30% at a 5% relative WER loss.

7. REFERENCES

[1] 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.

[2] F. Seide, G. Li, and D. Yu, "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks," Interspeech, 2011.

[3] 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.

[4] B. Kingsbury, "Lattice-based Optimization of Sequence Classification Criteria for Neural-Network Acoustic Modeling," Proc. ICASSP, Taipei, 2009.

[5] 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.

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

[7] The Microsoft Research Audio Video Indexing System. `http://research.microsoft.com/projects/mavis`.

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

[9] 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).

[10] L. Saul *et al*., "Mean Field Theory for Sigmoid Belief Networks", Journal: Computing Research Repository–CORR, pp. 61-76, 1996.

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

[12] F. Seide, G. Li, X. Chen, and D. Yu, "Feature Engineering in Context-Dependent Deep Neural Networks for Conversational Speech Transcription," Proc. ASRU, Waikoloa Village, 2011.

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

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

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

[16] X. Chen, A. Eversole, G. Li, D. Yu, and F. Seide, "Pipelined Back-Propagation for Context-Dependent Deep Neural Networks," Proc. Interspeech, 2012.

[17] S. Meng, K. Thabiratnam, Y. Lin, L. Wang, G. Li, and F. Seide, "Vocabulary and Language Model Adaptation Using Just One Speech File," Proc. ICASSP, 2010.

[18] P. Yu, Y. Shi, and F. Seide, "Approximate Word-lattice Indexing with Text Indexers: Time-anchored Lattice Expansion," ICASSP, 2008.

[19] P. Yu *et al*.. "Word-Lattice Based Spoken-Document Indexing with Standard Text Indexers," Proc. ACM-SIGIR Second Workshop on Searching Spontaneous Conversational Speech, Singapore, 2008

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