Feature Engineering in Context-Dependent Deep Neural Networks for Conversational Speech Transcription

Frank Seide[1], Gang Li[1], Xie Chen[1,2], and Dong Yu[3]

[1] *Microsoft Research Asia, 5 Danling Street, Haidian District, Beijing 100080, P.R.C.*
[2] *Department of Electronic Engineering, Tsinghua University, 10084 Beijing, P.R.C.*
[3] *Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA*

{fseide,ganl}@microsoft.com, chenxie09@mails.tsinghua.edu.cn, dongyu@microsoft.com

***Abstract*—We investigate the potential of Context-Dependent Deep-Neural-Network HMMs, or CD-DNN-HMMs, from a feature-engineering perspective. Recently, we had shown that for speaker-independent transcription of phone calls (NIST RT03S Fisher data), CD-DNN-HMMs reduced the word error rate by as much as one third—from 27.4%, obtained by discriminatively trained Gaussian-mixture HMMs with HLDA features, to 18.5%—using 300+ hours of training data (Switchboard), 9000+ tied triphone states, and up to 9 hidden network layers.**

In this paper, we evaluate the effectiveness of feature transforms developed for GMM-HMMs—HLDA, VTLN, and fMLLR—applied to CD-DNN-HMMs. Results show that HLDA is subsumed (expected), as is much of the gain from VTLN (not expected): Deep networks learn vocal-tract length invariant structures to a significant degree. Unsupervised speaker adaptation with discriminatively estimated fMLLR-like transforms works (as hoped for) nearly as well as fMLLR for GMM-HMMs.

We also improve model training by a discriminative pretraining procedure, yielding a small accuracy gain due to a better internal feature representation.

I. Introduction

In this paper, we investigate *Context-Dependent Deep Neural Network Hidden Markov Models*, or CD-DNN-HMMs, from a feature-engineering perspective. CD-DNN-HMMs [1], [2] are a recent very promising acoustic model. For speaker-independent single-pass recognition, it achieved relative error reductions of 16% on a business-search task [1], [2], and of up to one-third on the Switchboard phone-call transcription benchmark [3], over discriminatively trained GMM-HMMs.

Like the classical ANN-HMMs of the 90's [4], CD-DNN-HMMs replace the GMMs by an artificial neural network, but they differ in (1) significantly increased network depth (up to 11 hidden layers so far) and (2) in the way context dependence is handled: Instead of factorizing the networks, e.g., into a monophone and a context-dependent part [5], or decomposing them hierarchically [6], CD-DNN-HMMs *directly* model tied *context-dependent* states (senones). This had long been considered ineffective, until [1] showed that it works and yields large error reductions for deep networks.

This paper aims at evaluating the effectiveness of several feature-engineering techniques developed for GMM-HMMs—HLDA [7], VTLN [8], and fMLLR [9]—when applied to CD-DNN-HMMs. We find that CD-DNN-HMMs subsume not only HLDA (expected) but also most of the gains of VTLN (unexpected). The DNN can be viewed as a complex discriminative feature extractor which extracts environment- and speaker-invariant representations optimized to predict the tied triphone states. Unsupervised fMLLR adaptation, on the other hand, carries over to the DNN.

Lastly, while in [3] CD-DNN-HMM training relied on deep-belief-network pretraining [10], we show that replacing it with a discriminative variant leads to small improvement in recognition accuracy due to improved internal feature representation.

II. The Context-Dependent Deep-Neural-Network HMM

A deep neural network (DNN) is a conventional multi-layer perceptron (MLP [11]) with many layers, where training is typically initialized by a pretraining algorithm. Below, we describe a statistical view and a discriminative feature-learning view of the DNN, discuss its training and parameter initialization, and describe its integration with context-dependent HMMs for speech recognition and the application of feature-space speaker adaptation. Extra details can be found in [2].

A. Deep Neural Network

1) DNN—A Statistical View:
A DNN as used in this paper models the posterior probability $P_{\mathbf{s}|\mathbf{o}}(s|o)$ of a class s given an $\underline{o}$bservation 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 $\underline{h}$idden binary units h^ℓ given input vectors v^ℓ, while the top layer L models the desired class posterior as

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

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" [13]. 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) DNN—A Discriminative Feature-Learning View: In the DNN, the estimation of the posterior probability $P_{\mathbf{s}|\mathbf{o}}(s|o)$ can also be considered as a two step process: In the first step, the observation vector o is transformed into a feature vector v^L through L layers of non-linear transforms. A popular transformation is the sigmoid function:

$$v^{\ell+1} \;=\; \sigma\big(z^\ell(v^\ell)\big) \quad , \quad 0 \le \ell < L$$

In the second step, the posterior probability $P_{\mathbf{s}|\mathbf{o}}(s|o)$ is estimated using the log-linear model with features v^L as

$$P_{\mathbf{s}|\mathbf{o}}(s|o) \;=\; P^L_{\mathbf{s}|\mathbf{v}}(s|v^L) \;=\; \mathrm{softmax}_s(z^L(v^L))$$

If we thought of the first L layers as fixed, learning the parameters in the softmax layer would amount to training a conditional maximum entropy (MaxEnt) model. With conventional MaxEnt models, features are manually designed. E.g., [14] uses first and second-order statistics of the observations.

In DNNs, however, the features defined by the first L layers are also automatically learned. This not only eliminates the tedious and erroneous process of manual feature construction but also has the potential of extracting features that are impossible to construct manually (e.g., through many layers of non-linear transforms) and features that discriminate classes better. This is shown by higher recognition accuracy when using deep networks compared to shallow networks with the same number of parameters [3], or manually constructed features [14].

Viewing DNNs as automatic feature learning plus conditional MaxEnt model construction opens a path to using different feature transformation functions (e.g., radial basis function or pooling functions like max and sum), using different optimization procedures (e.g., alternating between feature extraction and MaxEnt model training), adding additional feature transformation layers borrowed from GMM-HMMs, or learning model parameters with different strategies like discriminative pretraining as described in Section II-B3.

B. Training Deep Neural Networks

DNNs, being 'deep' MLPs, can be trained with the well-known *error back-propagation* procedure (BP) [15]. 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. The following will recap BP and describe two pretraining methods, *deep belief network* (DBN) pretraining [10] and what we call *discriminative pretraining*.

1) Error Back-Propagation: The BP procedure [15] updates MLPs with stochastic gradient ascent

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

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 $O = \{o(t)\}$ with ground-truth labels $s(t)$, i.e.

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

then the gradients are

$$\begin{aligned}
\frac{\partial D}{\partial W^\ell} &= \sum_t v^\ell(t)\,(\omega^\ell(t)\,e^\ell(t))^T \, ; \; \frac{\partial D}{\partial a^\ell} = \sum_t \omega^\ell(t)\,e^\ell(t) \quad (2)\\
e^L(t) &= (\log\mathrm{softmax})'(z^L(v^L(t)))\\
e^{\ell-1}(t) &= W^\ell \cdot \omega^\ell(t) \cdot e^\ell(t) \qquad \text{for} \quad 0 \le \ell < L\\
\omega^\ell(t) &= \begin{cases} \mathrm{diag}\left(\sigma'(z^\ell(v^\ell(t))\right) & \text{for} \quad 0 \le \ell < L \\ 1 & \text{else} \end{cases}
\end{aligned}$$

with error signals $e^\ell(t) = \partial D/\partial v^{\ell+1}(t)$ as back-propagated from networks $\ell+1$ and above; network ℓ's output-non-linearity's derivative $\omega^\ell(t)$ if present; 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.

2) Pretraining as a Deep Belief Network: The deep belief network (DBN), proposed by Hinton [10], provides a new way to train deep generative models. The layer-wise greedy pretraining algorithm developed in DBN was later found to be also effective in training DNNs.

The DBN pretraining procedure treats each consecutive pair of layers in the MLP as a *restricted Boltzmann machine* (RBM) [10] whose joint probability is defined as

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

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

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

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 [10], [18]. We use the specific form given in [18]:

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

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

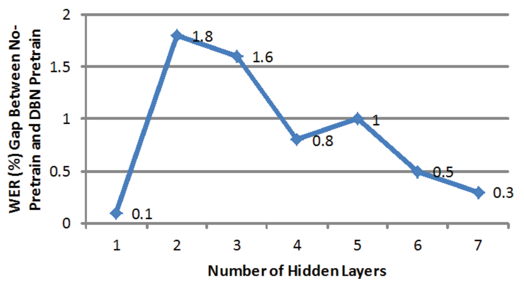

Fig. 1. *Word error rate gap in percentage points between the system without pretraining and that with DBN pretraining.*

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 often helps [10]. Figure 1, which is a different view of results originally presented in [3], illustrates the word error rate (WER) gap between systems without and with DBN pretraining. We can clearly see that DBN-pretraining is not important when only one hidden layer is used and it significantly outperforms the system without pretraining with two hidden layers. When the number of hidden layers increases, the effectiveness decreases as indicated by the decreasing WER gap between the system with and without pretraining. This is because DBN-pretraining employs two approximations. First, the mean-field approximation is used as the generation target when training the next layer. Second, the approximated contrastive divergence algorithm is used to learn the model parameters. Both these approximations introduce modeling errors for each additional layer. As the number of layers increases, the integrated errors increase and the effectiveness of DBN-pretraining decreases.

3) Discriminative Pretraining:
To remedy the modeling inaccuracies in DBN pretraining, we propose an alternative which we call *discriminative pretraining*, or DPT. We took a first step in [3] with "layer-wise BP," in which we first trained a one-hidden-layer DNN to full convergence using senone labels discriminatively with BP, then replaced the softmax layer by another randomly initialized hidden layer and a new random softmax layer on top, again discriminatively trained the network to full convergence, and so on until the desired number of hidden layers was reached. This is similar to 'greedy layer-wise training' [16], but differs in that [16] only updates newly added hidden layers.

While this achieved accuracies close to those obtained with DBN pretraining, we found that it can be further improved by (1) stopping very early by going through the data only once and (2) using large learning rate (0.08 in our experiments). The goal is to bring the weights close to a good local optimum.

As our results will show, DPT outperforms DBN pretraining, full-convergence "layer-wise BP," and no pretraining for deep networks. Yet, we believe better pretraining and/or optimization algorithms exist to achieve even better performance.

4) Weight Sparseness:
DNNs can be improved by moderate sparsification of the weight matrices. Enforcing weight sparseness is equivalent to incorporating L0 and approximate L1 regularizations to the training criterion. We have found, however, that enforcing sparseness during pretraining 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.

C. Speech Recognition with CD-DNN-HMMs

1) Integrating DNNs with CD-HMMs:
Following the traditional ANN-HMMs of the 90's [4], 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 from the MLP to likelihoods:

$$p_{\mathbf{o}|\mathbf{s}}(o|s) \quad = \quad \frac{P_{\mathbf{s}|\mathbf{o}}(s|o)}{P_{\mathbf{s}}(s)} \cdot \text{const(s)}. \tag{3}$$

Here, classes s correspond to HMM <u>s</u>tates, and <u>o</u>bservation 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 [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. The resulting model is called the Context-Dependent Deep Neural Network HMM, or CD-DNN-HMM.

2) Speaker Adaptation With Feature-Space Transforms:
Feature-space transforms for speaker adaptation developed for GMM-HMMs—VTLN and fMLLR—can conceptually be applied to DNNs, although with limits, as results will show.

Vocal-Tract Length Normalization, or VTLN, warps the frequency axis to account for the fact that the precise locations of vocal-tract resonances vary roughly monotonically with the physical size of the speaker. Using 20 quantized warping factors from 0.8 to 1.18, the optimal warping factor per conversation side in training is found EM-like by repeatedly selecting the best factor given the current model and then updating the model. In testing, we pick the best factor by "brute-force," running recognition for all factors and selecting the highest cumulative log probability per conversation side.

Feature-space Maximum-Likelihood Linear Regression, or fMLLR, transforms feature frames with a speaker-specific square matrix M and a bias c. For GMM-HMMs, M and c are estimated to maximize the likelihood of the adaptation data given the model. In this paper, the 'ground truth' are unsupervised recognition transcripts. We iterate four times.

For DNNs, we can either use fMLLR transforms from the GMM, or estimate M and c by maximizing $D'(O) = D(M \cdot O + c)$ on the unsupervised adaptation data with BP [17]. Without augmenting neighbor frames, the gradients are:

$$\frac{\partial D'}{\partial M} = \sum_t o(t) \cdot (W^0 e^0(t))^T \; ; \; \frac{\partial D'}{\partial c} = \sum_t W^0 e^0(t)$$

while *with* neighbor augmentation, M is block-diagonal, with blocks and bias tied across neighbor frames. We call this *Feature-space Discriminative Linear Regression*, or fDLR.

III. Training CD-DNN-HMMs

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

A. Basic Training Process

DNN model learning begins with the pretraining (DBN or discriminative), using one full sweep through the 309 hours of training data for all hidden layers (for DBN, we use two full sweeps for the first layer, and slight gains may be obtained by sweeping the data additional times, but it seems not critical [1]). RBMs are not scale-invariant, so the training corpus is normalized to zero mean and unit variance [19].

After pretraining, 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, if used, 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.

B. 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 pretraining.

For pretraining, 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 [15], we also smooth 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 [1], [2].

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.

C. GPGPU Considerations

Gradient ascent with small mini-batches cannot be meaningfully parallelized across multiple servers. Instead, we utilize one or two NVidia Tesla 448-core 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 [10].

TABLE I
Core results and effect of modeling techniques on CD-DNN-HMM accuracy. 'SI' means speaker-independent single-pass, and 'nz' 'non-zero'. The last row is our 'best-ever' speaker-adaptive multi-pass system trained on 7 times more data. Word-error rates in % for Hub5'00-SWB and RT03S-FSH.

modeling technique	#params [10^6]	WER ($\pm$relative change) Hub5'00-SWB	RT03S-FSH
GMM 40 mix DT 309h SI	29.4	23.6	27.4
CD-DNN 1 layer×4634 nodes	43.6	26.0 (+10%)	29.4 (+7%)
+ 2×5 neighbor frames	45.1	22.4 (-14%)	25.7 (-13%)
CD-DNN 7 layers×2048 nodes	45.1	17.1 (-24%)	19.6 (-24%)
+ updated state alignment	45.1	16.4 (-4%)	18.6 (-5%)
+ sparsification	15.2 nz	16.1 (-2%)	18.5 (-1%)
(total gain GMM→DNN)		*(-32%)*	*(-33%)*
+ VTLN speaker adaptation	16.9 nz	16.1 ($\pm$0%)	17.6 (-5%)
+ fDLR (4 iterations)	16.9 nz	15.5 (-4%)	16.9 (-4%)
GMM 72 mix DT 2000h SA	102.4	17.1 *(-28%)*	18.6 *(-32%)*

To reduce data transfer between CPU and GPGPU(s), model parameters (W, a, b) and accumulators are kept only in GPGPU memory during operation. Thus, when using a single GPGPU device, data transfer is limited to minibatches of features and ground-truth labels. With more than one device, we distribute model parameters and gradients in disjoint stripes across the GPGPU devices, while layer state is scattered/gathered between devices. Due to this, the second GPGPU device only yields an additional 25% speed-up.

IV. Experimental Results

A. Setup and Core Results

We evaluate the effectiveness of CD-DNN-HMMs and feature transforms on the task of speech-to-text transcription using the 309-hour Switchboard-I training set [20]. The system uses 13-dimensional PLP features with rolling-window mean-variance normalization and up to third-order derivatives, reduced to 39 dimensions by HLDA. The speaker-independent 3-state cross-word triphones share 9304 CART-tied states.

The GMM-HMM baseline system has 40-Gaussian mixtures per state, trained with maximum likelihood (ML), and refined discriminatively (DT) 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. (3)), while leaving everything else the same.

The data for system development is the 1831-segment SWB part of the NIST 2000 Hub5 eval set. The FSH half of the 6.3h Spring 2003 NIST rich transcription set (RT03S) acts as the evaluation set.[1] The trigram language model was trained on the 2000h Fisher transcripts and interpolated with a written-text trigram. Test-set perplexity with the 58k lexicon is 84.

The right column of Table I shows the core accuracy improvement from using DNNs on the evaluation set: For single-pass speaker-independent recognition, the word-error rate (WER) is reduced from the discriminatively trained

[1]We learned recently that 36 of the 40 Hub5'00-SWB test speakers have (disjoint) data in the training set [21]. Table I eased our concerns by showing that the RT03S-FSH set, which has no speaker overlap, behaves very similarly.

TABLE II

Effect of modeling techniques for GMM on a shallow and a deep NN. Word-error rates in % for Hub5'00-SWB (relative change in parentheses).

modeling technique	GMM 40 mix	CD-DNN 1×2k	CD-DNN 7×2k
PLP baseline	28.7	24.1	17.0
+ HLDA 52→39 dim.	26.5 (-8%)	24.2 (+0%)	17.1 (+1%)
+ DT (GMM only)	23.6 (-11%)	-	-
+ VTLN from resp. model	21.5 (-9%)	22.5 (-7%)	16.8 (-2%)
+ {fMLLR,fDLR}×4	20.4 (-5%)	21.5 (-4%)	16.4 (-2%)
vs. VTLN from GMM	21.5 (-9%)	22.7 (-6%)	17.1 (-0%)
+ fMLLR×4 from GMM	20.4 (-5%)	21.1 (-7%)	16.3 (-5%)

HLDA-based GMM-HMMs' 27.4% to 18.5% for CD-DNN-HMMs (row labeled "sparsification")—a quite significant 33% relative gain.

The speaker-independent 309-hour CD-DNN-HMM system also marginally outperforms our best speaker-*adaptive* multi-pass system (18.6%, last row) which uses additional acoustic training data (the 2000h Fisher corpus), VTLN, MLLR, and ROVER. For comparison, results on more recent but similar test sets are around 17-18% [22] using 2000 hour training data.

The gains are not tied to Switchboard. In [3], we showed that much of the gain also carries over for four less well-matched test sets—the cell-phone SW portion of RT03S, two voice-mail sets, and wideband recordings of teleconferences. The relative improvements were 22–28% over HLDA/DT.

B. Modeling Techniques

1) Effects of DNN Modeling Techniques:

Table I also shows the system development for a CD-DNN-HMM with 7 hidden layers to illustrate the impact of several factors. Here, all DNNs are trained with DBN pretraining and have a softmax layer for 9304 tied states.[2]

Starting with a 1-hidden-layer MLP, we see that the use of context frames yields a 14% relative gain on the development data, nearly twice 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 (while keeping their number constant) shows the modeling power of intermediate layers: Errors are reduced by 24% relative. Note that even dramatic further increase of the hidden layer size would not even come close—a single-layer DNN with 16k nodes is only marginally better (22.1% [3]). This suggests that the deep models' additional factorization by means of intermediate layers is indeed critical.

The above networks were trained using state alignments generated from the GMM-HMM ML model. Updating the state alignment using the CD-DNN-HMM yields another 4% relative reduction. Lastly, moderate sparsification of the weight matrices that zeroes about two thirds of the weights yields another small improvement of 0.3 points. Overall, this model has seen about 20 sweeps through the training data, taking about 30 days of processing.

Up to this point, the model is speaker independent using single-pass decoding. Adding VTLN and unsupervised conversation-side fDLR speaker adaptation yields an additional 4% relative reduction for the dev set, and a total of 9% for the eval set. More on that in the next section.

2) Effects of GMM Modeling Techniques for DNN:

Table II shows the effect of three common GMM-HMM technologies, applied to the DNN. The 'PLP baseline' is a system as described in section IV-A but with HLDA and DT disabled. For the GMM, the three 'work horses' HLDA, DT, and VTLN together reduce errors by one quarter. HLDA and DT, accounting for an 18% reduction for the GMM, are 'innate' to the DNN and do not lead to explicit gains ([23] finds the same for LDA). Actually, HLDA causes a small loss.

While the DNN, with a WER of 17.0%, improves the 'naïve' ML-trained PLP baseline of 28.7% by a massive relative 41% (top row of Table II), we should meaningfully compare it against the GMM system with HLDA and DT applied (23.5%), since these are well-understood standard techniques. Hence, the DNN achieves 28% error reduction (or 32% if DNN state realignments and sparseness are used, cf. Table I). It does so without the hassle of estimating HLDA or generating word-lattice for discriminative training.

The center section of Table II considers multi-pass speaker adaptation. We applied VTLN in training (warping factors for DNN training copied from GMM-ML VTLN training), and testing (factors selected by conversation-side decoding likelihood using the model under test). To our great surprise, we found that VTLN does not quite carry over to the deep NN with 7 hidden layers. It achieves only a meager 2% relative gain (and even *none* for the final sparse model trained with DNN state alignments, Table I). On the eval set (Table I), the VTLN gain is 5%, a little more than half as that for the GMM. VTLN *is* more effective on the *shallow* 1-hidden layer NN, though (7% gain). Since the implementation is identical to the deep NN, we conclude that the deep NN is able to learn vocal-tract length invariant structures to a significant degree. On the eval set, about one third of the post-HLDA/DT gain obtained by the DNN is explained by this. We had not expected that.

For validation, we compare using VTLN warping factors chosen by the *GMM* system in DNN decoding (Table II, bottom). The VTLN gain reduces to 0 for the deep network—The DNN learns VTL invariance well enough to be sensitive to minor inconsistencies of warping factors.

fDLR (on top of VTLN, in test only; Table II center) further reduces WER by 7% rel. for the shallow network, but only by 2% for the deep NN. It falls short by 0.1 percentage points compared to directly using the GMM's fMLLR transform. For the eval set (Table I), the fDLR gain is 4%. This re-validates the fDLR criterion [17] in the context of a DNN, but raises the question whether BP is effective even through a deep stack of 8 model layers. It also shows that the DNN cannot learn speaker invariance to this degree.

C. Layers and Training Procedures

Figure 2 illustrates the effect of the number of hidden layers and different training procedures to the word error rate. First, increasing from 1 to 7 layers (2048 hidden nodes at each

[2]To our knowledge, [3] was the first to ever use such a large softmax layer for acoustic modeling in ASR.

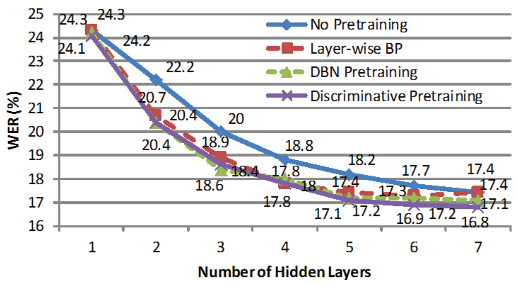

Fig. 2. *Comparing different pretraining techniques—DBN, discriminative, LBP and none—by word-error rate as a function of the number of layers. The baseline WER for the discriminatively trained GMM-HMM is 23.6%.*

layer), using DBN pretraining, reduces WER from 24.2% to 17.1%, where it saturates (further increasing to 9 and 11 layers[3] yields 17.0%).

In the Figure, "No Pretraining" refers to pure BP (section II-B1) with random initialization of all layers, while "DBN pretraining," "layer-wise BP," and "discriminative pretraining" use the pretraining methods (sections II-B2 and II-B3) to initialize the model parameters and then optimize parameters in all layers jointly with BP.

As expected, pure BP without pretraining performs worst. In fact, it is nearly 2 percentage points worse than the DBN pretraining for 2 and 3 hidden layers. However, when the number of layers increases, the difference continuously reduces and becomes only 0.3 points for 7 hidden layers.

Layer-wise BP (with iteration until full convergence at each layer) outperforms pure BP and gets close to DBN pretraining. This suggests that pretraining helps but is not critical for CD-DNN-HMMs to outperform GMM-HMMs. It also indicates that DBN pretraining is less effective for 5 and more layers.

Discriminative pretraining slightly outperforms DBN pretraining. The gap grows at 5 layers and reaches 0.3 percentage points with 7 hidden layers. Discriminative pretraining has the added benefit of using the same BP algorithm, and thus allows us to focus on improving just one single unified algorithm.

V. Conclusion

In this paper, we investigated the potential of CD-DNN-HMMs from a feature-engineering perspective. We looked at the DNN from a different angle by considering training it as joint optimization of a complex discriminative feature extractor and a log-linear model that exploits these environment- and speaker-invariant features. Our experiments on the conversational speech-to-text transcription task indicate that DNNs can subsume HLDA and much of the gain of VTLN, while still benefiting from unsupervised fMLLR-like speaker adaptation (fDLR) and from discriminative pretraining.

Our results suggest that shallow neural networks have much less effective feature extraction power than the deep neural networks. As a result, feature-engineering techniques such as VTLN, commonly used in GMM-HMMs, can bring further gains for shallow NNs but less so for deep NNs. On the other hand, the 30+% relative WER reduction of CD-DNN-HMM over CD-GMM-HMM with HLDA, DT, and VTLN indicates that the features automatically extracted by DNNs are far superior than those already captured in HLDA and VTLN.

The main challenge we are facing for making CD-DNN-HMMs mainstream is to improve efficiency of the training to scale up further to tens of thousands of hours of data.

[3]The 11-hidden-layer network has 66 million parameters, and is to our knowledge the largest artificial neural network for ASR to date.

VI. Acknowledgements

We thank Asela Gunawardana, Kit Thambiratnam, Jasha Droppo, Nikko Ström, Zhijie Yan, and Andreas Stolcke for their help with baselines and fruitful discussion. Special thanks 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.

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] 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).
[3] F. Seide, G. Li, and D. Yu, "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks," Interspeech, 2011.
[4] 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.
[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] G. Saon *et al.*, "Maximum Likelihood Discriminant Feature Spaces," ICASSP, 2000.
[8] P. Zhan *et al.*, "Vocal Tract Length Normalization for LVCSR," Technical Report CMU-LTI-97-150, Carnegie Mellon Univ., 1997.
[9] M. Gales, "Maximum Likelihood Linear Transforms for HMM-Based Speech Recognition," Computer Speech & Language, vol. 12, 1998.
[10] G. Hinton, S. Osindero, and Y. Teh, "A Fast Learning Algorithm for Deep Belief Nets", Neural Computation, vol. 18, pp. 1527–1554, 2006.
[11] F. Rosenblatt, "Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms", Spartan Books, Wash. DC, 1961.
[12] 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.
[13] L. Saul *et al.*, "Mean Field Theory for Sigmoid Belief Networks", Journal: Computing Research Repository–CORR, pp. 61-76, 1996.
[14] D. Yu and L. Deng, "Deep-Structured Hidden Conditional Random Fields for Phonetic Recognition", Proc. Interspeech, 2010.
[15] D. Rumelhart, G. Hinton, and R. Williams, "Learning Representations By Back-Propagating Errors," Nature, vol. 323, Oct. 1986.
[16] Y. Bengio *et al.*, "Greedy layer-wise training of deep networks," Advances in Neural Information Processing Systems 19, 2007.
[17] V. Abrash *et al.*, "Connectionist speaker normalization and adaptation," Proc. Eurospeech, 1995.
[18] G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines", Technical Report UTML TR 2010–003, Univ. of Toronto, 2010.
[19] A. Mohamed, G. Dahl, and G. Hinton, "Deep Belief Networks for Phone Recognition," in Proc. NIPS Workshop Deep Learning for Speech Recognition, 2009.
[20] J. Godfrey and E. Holliman, "Switchboard-1 Release 2," Linguistic Data Consortium, Philadelphia, 1997.
[21] J. Fiscus *et al.*, "2000 NIST Evaluation of Conversational Speech Recognition over the Telephone: English and Mandarin Performance Results," from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.27.5417.
[22] IEEE Trans. Audio, Speech, and Lang. Processing, Special Section on Rich Transcription, vol. 14, No. 5, pp. 1490-1608, 2006.
[23] A. Mohamed *et al.*, "Deep Belief Networks Using Discriminative Features for Phone Recognition," Proc. ICASSP, 2011.