The IBM 2015 English Conversational Telephone Speech Recognition System

George Saon, Hong-Kwang J. Kuo, Steven Rennie and Michael Picheny

IBM T. J. Watson Research Center, Yorktown Heights, NY, 10598
gsaon@us.ibm.com

1. Introduction

Ever since [1] demonstrated the large accuracy gains from using deep neural network acoustic models versus Gaussian mixture models, the Switchboard corpus has become the de facto standard experimental testbed for reporting believable and, more importantly, reproducible results for LVCSR. We surmise that this is because it is the largest publicly available dataset (up to 2300 hours of training data) composed of truly conversational speech and because, in general, techniques which result in improvements on Switchboard tend to work well on both small and large vocabulary tasks. One can think of LDA/STC, VTLN, FMLLR and lattice-based model and feature-space discrimination training which were developed first on Switchboard and then became ubiquitous as prime examples of such techniques.

Since Switchboard is such a well-studied corpus, we thought we would take a step back and reflect on how far we have come in terms of speech recognition technology. To set the baseline, the human word error rate on this task is estimated to be around 4% [2]. Quoting [2] again, in 1995, “a high-performance HMM recognizer” achieved a 43% WER on Switchboard [3]. In 2000, Cambridge University achieved an at the time impressive error rate of 19.3% during the Hub5e DARPA evaluation [4] which they attributed to “careful engineering”. At the height of technological development for GMM-based systems, the winning IBM submission scored 15.2% WER during the 2004 DARPA EARS Rich Transcription evaluation [5] largely due to the Atila ASR toolkit [6] and lMPE [7]. Nowadays, deep neural networks have levelled the playing field and multiple sites can easily reach 12-14% WER using much simpler systems [8, 9, 10, 11, 12] as shown in Table 6.

To achieve an error rate of 8.0% on this task is not trivial. In our opinion, a successful recipe has to contain several ingredients. The first and most obvious one is to train larger acoustic and language models on more data. The second (a little less obvious) is to train neural nets that have diverse architectures and operate on different input representations so that we get accuracy gains from both feature and model combination. Third, extra “spice” such as networks with maxout nonlinearities and exponential and NN language models were also found to significantly lower the error rate of our system. Last but not least, it is our experience that having a strong GMM-HMM baseline system [6, 13] which provides high-quality alignments used for the various speaker adaptation techniques and for DNN cross-entropy training helps.

The paper is organized as follows: in section 2 we describe the processing steps that are common across all models, in section 3 we present a set of system improvements, and in section 4 we summarize our findings and ponder future opportunities for improvement.

2. General processing

Here we describe the common processing steps for all the models detailed in this paper. In particular, we discuss frontend processing, speaker adaptation and neural network training specifics which are largely similar to [14, 13].

2.1. Training and test data

The training data consists of 1975 hours of segmented audio from English telephone conversations between two strangers on a preassigned topic and is divided as follows: 262 hours from the Switchboard 1 data collection, 1698 hours from the Fisher data collection and 15 hours of CallHome audio. The test set is the Hub5 2000 evaluation set and contains two parts: 2.1 hours (21.4K words, 40 speakers) of Switchboard data and 1.6 hours (21.6K words, 40 speakers) of CallHome audio. The decoding vocabulary has 30.5K words and 32.9K pronunciations and all decodings were performed with a 4M 4-gram language model (and rescored with different LMs in subsection 3.4).

2.2. Feature extraction

Speech is coded into 25 ms frames, with a frame-shift of 10 ms. Each frame is represented by a feature vector of 13 VTL-warped perceptual linear prediction (PLP) cepstral coefficients which are mean and variance normalized per conversation side. Every 9 consecutive cepstral frames are spliced together and projected down to 40 dimensions using LDA. The range of this transformation is further diagonalized by means of a global semi-tied covariance transform. Next, the LDA features are transformed with one feature-space MLLR (FMLLR) transform per conversation side at both training and test time. Convolutional nets are trained on VTL-warped logmel features augmented with first and second temporal derivatives. The Mel filterbank has 40 filters and the input to the CNNs are blocks of 11 consecutive 40×3-dimensional frames (as described in [13]).

Abstract

We describe the latest improvements to the IBM English conversational telephone speech recognition system. Some of the techniques that were found beneficial are: maxout networks with annealed dropout rates; networks with a very large number of outputs trained on 2000 hours of data; joint modeling of partially unfolded recurrent neural networks and convolutional nets by combining the bottleneck and output layers and retraining the resulting model; and lastly, sophisticated language model rescoring with exponential and neural network LMs. These techniques result in an 8.0% word error rate on the Switchboard part of the Hub5-2000 evaluation test set which is 23% relative better than our previous best published result.

Index Terms: recurrent neural networks, convolutional neural networks, conversational speech recognition
In addition to VTLN and FMLLR, DNNs are adapted to the speaker by appending 100-dimensional i-vectors to every block of 11 FMLLR frames as described in [14]. The i-vectors are extracted using a universal background model given by a GMM with 2048 diagonal covariance mixture components which was trained with maximum likelihood on the speaker-adapted features. The i-vectors are extracted once per conversation side.

2.3. Neural network training
All models have sigmoid hidden layers and softmax output layers (except for the models from subsection 3.1) and are trained with 10-15 epochs of SGD on frame-randomized minibatches of 250 frames and a cross-entropy criterion. The targets correspond to the context-dependent HMM states obtained by aligning the audio with a GMM-HMM system with 300K Gaussians trained with maximum likelihood on the FMLLR features. The same alignments are mapped to the leaves of various phonetic decision trees which differ in phone context size (±2 or ±3) and number of leaves (16K, 32K and 64K). Prior to CE training, the networks are initialized with layerwise discriminative pretraining as suggested in [1]. Additionally, we applied 20-30 iterations of hessian-free sequence discriminative training (ST) by using the state-based minimum Bayes risk (MBR) objective function as described in [15]. The trained networks are used directly in a hybrid decoding scenario by subtracting the logarithm of the HMM state priors from the log of the DNN output scores.

3. System improvements
In this section we discuss specific improvements related to acoustic and language modeling. More concretely, we describe the following techniques: maxout models with annealed dropout (subsection 3.1); training DNNs, CNNs and RNNs with a very large number of outputs (subsection 3.2); improved joint training of convolutional and non-convolutional nets (subsection 3.3); and language model rescoring with exponential and neural network LMs (subsection 3.4).

3.1. Maxout networks with annealed dropout
Maxout networks [16] generalize rectified linear (ReLU, max[0, a]) units, employing non-linearities of the form:

\[ s_j = \max_{i \in C(j)} a_i \]

where the activations \( a_i = w_i^T x + b_i \) are based, as usual, on inner products, and the sets of activations \( \{ C(j) \} \) utilized by different hidden units are typically disjoint. Maxout networks are conditionally linear and so avoid the vanishing gradient problem, and are well suited for the dropout training procedure [17], which for a linear model, trains an exponentially sized model ensemble (2^D models for input dimension D), whose geometric average can be computed by simply renormalizing at test time.

Maxout networks for ASR have recently been investigated by several researchers, and found to produce significant gains when training data is limited [18], but negligible gains in our personal experience when the amount of training data exceeds approximately 100 hours. However, recently we showed that by annealing the dropout rate over the course of training, Maxout networks can produce substantial gains, even in big data scenarios [19]. The annealing procedure effectively initializes the ensemble of models being learned at a given iteration with an ensemble of models with lower mean and higher variance in the number of active units. This stochastic regularization procedure retains the benefits of the standard dropout training procedure (a strong exploration-phase; a preference for population-based predictions) without compromising the capacity of the network being learned.

Table 1 compares the performance of our annealed dropout Maxout networks (Maxout-AD) to corresponding sigmoid-based DNNs and CNNs from [13] learned using our standard training procedure, using only the SWB-1 training data (262 hours). The two convolutional layers of the Maxout-AD CNN and of the sigmoid CNN each have 512 feature map outputs. All Maxout networks utilize 2 filters per hidden unit, and the same number of layers and roughly the same number of parameters per layer as the sigmoid-based DNN/CNN counterparts. Parameter equalization is achieved by having a factor of \( \sqrt{2} \) more neurons per hidden layer for the maxout nets since the maxout operation reduces the number of outputs by a factor of 2. Note that ReLU networks, in our experience, perform on-par with sigmoid-based DNNs in this data regime. Maxout networks trained with AD (Maxout-AD), on the other hand, show a clear advantage over our traditional networks.

<table>
<thead>
<tr>
<th>Model</th>
<th>WER SWB (ST)</th>
<th>sigmoid</th>
<th>Maxout-AD</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN</td>
<td>11.9</td>
<td>11.0</td>
<td></td>
</tr>
<tr>
<td>CNN</td>
<td>11.8</td>
<td>11.2</td>
<td></td>
</tr>
<tr>
<td>DNN+CNN</td>
<td>10.3</td>
<td>10.0</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Word error rates of sigmoid vs. Maxout networks trained with annealed dropout (Maxout-AD) for ST CNNs, DNNs and score fusion on Hub5’00 SWB. Note that all networks are trained only on the SWB-1 data (262 hours).

3.2. Networks with very large output layers
When training on 2000 hours of data, we found it beneficial to increase the number of context-dependent HMM output targets to values that are far larger than commonly reported. To keep the computation and the number of parameters in check, we also had to use a bottleneck layer before the output layer [20] with typically 512 neurons. Back in the days when we were training GMM-based acoustic models, we did not notice accuracy improvements when using more than, say, 10000 HMM states [5]. We conjecture that this is because GMMs are a distributed model and require more data for each state to reliably estimate the mixture components, whereas the DNN output layer is shared between states. This allows DNNs to have a much richer target space. Additionally, we experimented with growing acoustic decision trees where the phonetic context is increased to heptaphones (±3 phones within words and ±2 phones across words). This was a distinct feature of our EARS RT’04 evaluation system which made a significant difference [5]. The effect of varying the number of outputs and phonetic context is shown in Table 2 for DNNs with 5 hidden layers (4 with 2048 units and 1 with 512 units) trained with 15 passes of cross-entropy on 2000 hours.

Based on these results, a compromise was struck by choosing models with 32K outputs and pentaphone acoustic context in all subsequent experiments. We have trained three types of models that differ in functionality and input features:

- Regular DNNs that operate on 11 spliced 40-dimensional FMLLR frames and 100-dimensional i-
Table 2: Comparison of word error rates for CE-trained DNNs with different number of outputs and phonetic context size on Hub5’00 SWB.

<table>
<thead>
<tr>
<th>Nb. outputs</th>
<th>Phonetic ctx.</th>
<th>WER SWB (CE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16000</td>
<td>±2</td>
<td>12.0</td>
</tr>
<tr>
<td>16000</td>
<td>±3</td>
<td>11.8</td>
</tr>
<tr>
<td>32000</td>
<td>±2</td>
<td>11.7</td>
</tr>
<tr>
<td>64000</td>
<td>±2</td>
<td>11.9</td>
</tr>
</tbody>
</table>

3.3. Improved joint training of recurrent and convolutional nets

In [13], we proposed a method for jointly modeling and training a CNN and a DNN. The crux of the method is to have the first layers be network specific (convolution and pooling for CNN operating on spectral features and input layer for DNN operating on PLP-based and i-vector features) and the remaining layers be shared. The outputs of the network-specific layers are merged into one common hidden layer followed by additional (common) hidden layers and one output layer. This graph structure for the joint network extends the standard linear sequence of layers for DNNs (or CNNs). By using this architecture, we reported a 12% relative gain on a Switchboard 300 hours setup over the best single model (from 11.8% for the CNN to 10.4% for the joint CNN/DNN). We also showed that performing score fusion of a CNN and a DNN trained separately achieves a similar WER of 10.5%. Hence, the main benefit of the joint model in [13] over the score fusion approach is the shared computation for the common hidden and output layers which is considerably faster than having to do two separate forward passes.

A different approach that we are advocating here is to initialize the joint model such that it is equivalent to the score fusion of the separate models. The reasoning behind this is that, after retraining, the objective function for sequence discriminative training can only improve (or, at worst, remain the same). For the case of log-linear score combination of multiple neural networks with the same number (and type) of outputs, this initialization is done by concatenating the individual weight matrices between the bottleneck and output layers and by dividing the resulting matrix by the number of models (assuming uniform weights). An example of a joint CNN/DNN model initialized in such a way is illustrated in Figure 1. For convenience, we have indicated the sizes of the weight matrices in the oval boxes and the dimensionality of the layers is attached to the arrows.

Table 3: Comparison of word error rates for CE and ST CNN, DNN, RNN and various score fusions on Hub5’00.

<table>
<thead>
<tr>
<th>Model</th>
<th>WER SWB</th>
<th>WER CH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CE</td>
<td>ST</td>
</tr>
<tr>
<td>CNN</td>
<td>12.6</td>
<td>10.4</td>
</tr>
<tr>
<td>DNN</td>
<td>11.7</td>
<td>10.3</td>
</tr>
<tr>
<td>RNN</td>
<td>11.5</td>
<td>9.9</td>
</tr>
<tr>
<td>DNN+CNN</td>
<td>11.3</td>
<td>9.6</td>
</tr>
<tr>
<td>RNN+CNN</td>
<td>11.2</td>
<td>9.4</td>
</tr>
<tr>
<td>DNN+RNN+CNN</td>
<td>11.1</td>
<td>9.4</td>
</tr>
</tbody>
</table>

We have experimented with jointly training the unfolded RNN and the CNN from subsection 3.2. Two experimental scenarios were considered. The first is where the joint model was initialized with the fusion of the cross-entropy trained RNN and CNN whereas the second uses ST models as the starting point. For both scenarios we generate numerator and denominator lattices with the initial joint model and optimize the lattice-based MBR loss using distributed hessian-free training [15]. In Table 4 we compare the WERs for several systems on the Hub5’00 test set (SWB and CallHome parts).

We observe that joint modeling and sequence discriminative retraining helps by 0.5% on the CallHome part and only 0.1% on SWB over score fusion of the ST models. Also, the
| Table 4: Comparison of word error rates for CE and sequence trained unfolded RNN and DNN with score fusion and joint modeling on Hub5’00. The WERs for the joint models are after sequence training.

<table>
<thead>
<tr>
<th>System</th>
<th>AM training data</th>
<th>SWB</th>
<th>CH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soltau et al. [13]</td>
<td>SWB+Fisher+CH</td>
<td>15.4</td>
<td>14.1</td>
</tr>
<tr>
<td>Hannun et al. [10]</td>
<td>SWB+Fisher</td>
<td>12.6</td>
<td>19.3</td>
</tr>
<tr>
<td>Maas et al. [12]</td>
<td>SWB</td>
<td>14.3</td>
<td>26.0</td>
</tr>
<tr>
<td>Maas et al. [12]</td>
<td>SWB+Fisher</td>
<td>15.0</td>
<td>23.0</td>
</tr>
<tr>
<td>Han et al. [18]</td>
<td>SWB</td>
<td>10.4</td>
<td>19.1</td>
</tr>
<tr>
<td>This system</td>
<td>SWB+Fisher+CH</td>
<td>8.0</td>
<td>14.1</td>
</tr>
</tbody>
</table>

Table 6: Comparison of word error rates on Hub5’00 (SWB and CH) for existing systems (* note that the 19.1% CallHome WER is not reported in [13]).

### 4. Discussion

We have presented a set of improvements to our English Switchboard system that lowered the error rate substantially compared to our previous best result [13]. In decreasing order of importance these are: rescoring with strong language models trained on diverse data sources; joint training of an RNN and a CNN with 32000 outputs on 2000 hours of audio and maxout networks with annealed dropout. We expect additional accuracy gains by training the maxout nets and larger CNNs with a 512/512 filter configuration on all the data.

Extrapolating from historical trends, we believe that human accuracy on this task can be reached within the next decade. We think that the way to get there will most likely involve an increase of several orders of magnitude in training data and the use of more sophisticated neural network architectures that tightly integrate multiple knowledge sources (acoustics, language, pragmatics, etc.).

### 5. Acknowledgment

The authors wish to thank the following present and former IBM colleagues: H. Soltau, D. Povey, S. Chen, A. Emami, V. Goel, B. Kingsbury, L. Mangu, B. Ramabhadran, T. Sainath and G. Zweig for significant contributions to the Switchboard system.

### 6. References


