The IBM BOLT Speech Transcription System

Samuel Thomas, George Saon, Hong-Kwang Kuo and Lidia Mangu

IBM T.J. Watson Research Center, Yorktown Heights, USA.

{sthomas,gsaon,hkuo,mangu}@us.ibm.com

Abstract

We describe the IBM automatic speech recognition (ASR) system for the DARPA Broad Operational Language Translation (BOLT) program. The system is used to transcribe conversational telephone speech (CTS) prior to machine translation for Phase 3 of the program's Activity A. The ASR system is a combination of novel sequence trained ensemble deep neural network acoustic models on speaker adapted features and convolutional neural network models on two kinds of spectro-temporal representations of speech, in conjunction with a variety of class, neural network and n-gram based language models. Acoustic and language models for the recognition system are built on transcribed audio released under the program and further optimized for the final machine translation task as well. The evaluation system has a word error rate of 32.7% on a 2 hour Egyptian Arabic development set for this task.

Index Terms: Automatic speech recognition, conversational telephone speech, deep neural networks, machine translation

1. Introduction

The DARPA BOLT program develops systems and technologies capable of translating multiple languages in different genres, processing information from the translated text and allowing bilingual communication using speech or text [1]. The machine translation evaluation of BOLT Activity A for Phase 3 was to test English translation of material from three different genres in two separate languages - Egyptian Arabic and Mandarin Chinese. Translation material for the three genres were chosen from:

1. Text from discussion forums in Egyptian Arabic and Mandarin Chinese,
2. Text from SMS and chat in the same two languages, and
3. Conversational telephone speech recorded in these same languages as well.

The automatic speech recognition systems described in this paper were developed to transcribe conversational telephone speech in Egyptian Arabic and Mandarin Chinese, before machine translation (MT) to English. Similar to earlier systems built under the DARPA GALE program [2], the final ASR system uses neural network acoustic models, language model rescoring and system combination. In this paper we describe the development of our system with experiments only on Egyptian Arabic although similar trends are observed for Mandarin Chinese as well.

In section 2 we describe the neural network acoustic models used in our system. After combination, outputs from these systems are decoded and rescored using the language models described in section 3. Section 4 describes final MT results on various development sets after optimizations for machine translation. The paper concludes with a discussion and future directions (section 6).

2. Acoustic Models

Although the Egyptian Arabic ASR system uses only neural network acoustic models, the training recipe for these models starts with building traditional HMM-GMM based acoustic models. The HMM-GMM models are used to produce context-dependent state alignments and speaker adapted features for the neural network training pipeline. The GMM models are trained on 13 dimensional PLP features estimated in 25 ms windows of speech every 10 ms. Cepstral features from 9 consecutive frames are then spliced after speaker based cepstral mean-variance and vocal tract length normalizations. An LDA transform is applied to reduce the final feature dimensionality to 40. The ML training of the GMM models is also interleaved with the estimation of a global semi-tied covariance (STC) transform. Feature-space MLLR (FMLLR) is finally applied to train speaker adapted models. The speaker adapted models have 100000 Gaussian components distributed over 8000 pentaphone context-dependent states.

We use close to 100 hours of transcribed audio released by LDC under the program for training these models. This includes data previously available as Egyptian Arabic LDC CALLHOME (LDC97S45/LDC2002S37) and LDC CALLFRIEND (LDC96S49), in addition to more recently collected data under the program. With short vowels and other diacritic markers typically not orthographically represented in Arabic texts, there are a number of choices for building pronunciation dictionaries [3]. These include - 1) Unvowelized graphemic dictionaries in which the short vowels and diacritics are ignored 2) Vowelized dictionaries which use the Buckwalter morphological analyzer [4] for generating possible vowelized pronunciations and 3) Vowelized dictionary which uses the output of a morphological analysis and disambiguation tool (MADA) [5]. For the system described in this paper we use unvowelized graphemic transcripts and dictionaries based on 45 graphemes.

2.1. Deep Neural Network Models

Our deep neural network acoustic models are fully connected multilayer perceptrons with several non-linear hidden layers that are discriminatively trained to estimate posterior probabilities of context-dependent states. Using the standard error back-

This work was partially funded by the DARPA BOLT program - contract No HR0011-12-C-0015. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, DARPA or the U.S. Government.

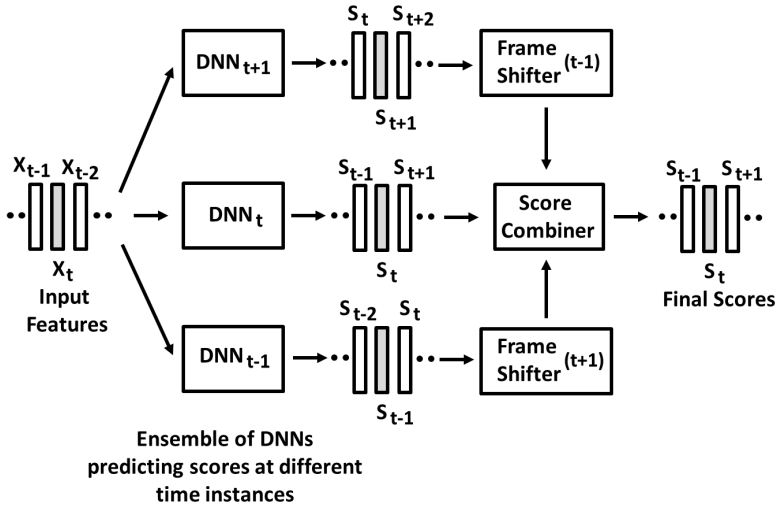

Figure 1: *Schematic of ensemble deep neural network models.*

propagation and cross-entropy objective function, the DNNs are trained on speaker adapted FMLLR features using alignments produced from the HMM-GMM acoustic model described earlier. The DNNs are pretrained by growing them layer-wise to 7 hidden layers, with one iteration over the training data, before being trained to convergence. Except for the softmax function at the output layer, each of the hidden layers with 1024 hidden units, uses the sigmoid activation function.

Similar to the GMMs, these neural networks are trained on close to 100 hours of data after the data is frame-randomized and split into mini-batches of 250 frames each. Starting with a step size of 5e-3, depending on improvements on a held-out set of about 5 hours, the step size is kept the same or halved if the relative improvement is less than 1%, as the training progresses. Cross-entropy training of the networks is followed by Hessian-free sequence training using a state-based minimum Bayes risk objective function described in [6].

2.2. Convolutional Neural Network Models

Convolutional neural networks (CNN) [7] are very similar to conventional DNNs - the difference between these models, being the additional CNN feature extracting layers. These layers generate features for succeeding layers instead of pre-processed features that are usually input to the DNNs. Each of the feature extracting layers consists of a pair of convolution and max pooling sub-parts. The convolution sub-part is a set of filters that are locally convolved with parts of the input to produce features that are further processed by a max pooling step. The max pooling operation involves picking the maximum from adjacent filter outputs. After passing through sigmoid nonlinearities, activations from lower layers are processed by subsequent feature extracting layers with more filters and down-sampling. The extracted features are finally received by fully connected DNN layers. Similar to DNNs, all the layers of the CNN are also trained using the standard back-propagation algorithm to minimize the cross entropy between the targets and the activations of the output layer.

We train CNN models on two spectro-temporal representations of speech. The first representation we use are 40 dimensional log-mel spectra augmented with Δ and $\Delta\Delta$s. The log-mel spectra are extracted by first applying mel scale integrators on power spectral estimates in short analysis windows (25 ms) of the signal followed by the log transform. Each frame of speech is also appended temporally with a fixed set of 11 frames. All of the 128 nodes in the first feature extracting layer are attached with 9 $\times$9 filters that are two dimensionally convolved with the input representations. The second feature extracting layer with 256 nodes has a similar set of 4 $\times$ 3 filters that process the non-linear activations after max pooling from the preceding layer. The non-linear outputs from the second feature extracting layer are then passed onto the following DNN layers.

Table 1: *WER (%) for DNN and CNN systems on the dev set.*

Acoustic Model	WER CE	WER ST
DNN	41.2	37.6
CNN (log-mel)	39.3	36.4
CNN (gammatone)	39.3	36.8

The second representation we use to train CNN models are 64 dimensional Gammatone auditory filter bank outputs [8] augmented with Δ and $\Delta\Delta$s. To extract these features, after pre-emphasis and Hanning windowing, the Fourier spectrum is filtered by a filter bank with 64 Gammatone filters. The spectrum is further post-processed by a cubed root compression and temporally smoothed using a second order ARMA filter [9]. The final Gammatone features are also mean and variance normalized on a per utterance basis. Similar to the CNN trained on the log-mel spectrum, we use 128 hidden nodes in the first convolutional layer with 9 $\times$ 9 filters and 256 hidden nodes in the second convolutional layer with 4 $\times$ 3 filters.

Both the CNN networks have 5 fully connected DNN layers with 1024 hidden units each and estimate posterior probabilities of 8000 context dependent phones. The CNN networks are trained on 100 hours of speech with the same training procedure described earlier for DNN networks. During discriminative pretraining the two convolutional layers are trained together while the remaining network is grown layer-by-layer.

As with the DNN models, cross-entropy training of the

CNNs is followed by Hessian-free sequence training. Table 1 shows the word error rate (WER%) performance of both models at the cross-entropy (WER CE) training level and after sequence training (WER ST) with 1-best decoding on a dev set of 1.5 hours of speech.

2.3. Ensemble Deep Neural Network Models

Inspired by [10], we experimented with training DNNs that predict HMM states at different time offsets and average the predictions. The authors propose to train a single DNN that predicts states at multiple times using multi-task learning. The score for the central HMM state at time t is obtained by averaging the predictions of the DNN at different times. When implementing this technique we found that, although we obtained good gains in frame classification accuracy, the word error did not improve.

We propose to train instead $2 * K + 1$ separate DNNs that predict states at individual times $t - K \ldots t \ldots t + K$ respectively, using the same input features as the baseline DNN (9×40 FMLLR frames). To get the scores for the states at time t, we average $2 * K + 1$ score matrices where the score matrix for the DNN corresponding to time offset k is shifted by k time steps $k \in \{-K, \ldots, K\}$. For example, the output matrix for the DNN that predicts s_{t-1} is shifted left by one frame; the output matrix for the DNN that predicts s_{t+1} is shifted right by one frame and so on (see Fig. 1).

All DNNs have the same topology: an input layer with 360 neurons, 7 hidden layers with 1024 sigmoid neurons and one output layer with 16000 softmax units. Training consists of 15 passes of minibatch SGD with a cross-entropy objective followed by sequence discriminative training. The DNN models also benefit by the increase in output targets from 8000 to 16000. In Table 2 we show the benefit of combining 5 DNNs (corresponding to $K = 2$) for both cross-entropy (CE) and sequence trained (ST) models.

Table 2: *Comparison of WER(%) for CE and ST single and combined DNNs on the dev set.*

K	WER CE	WER ST
0	40.0	37.0
2	39.0	36.4

Additionally, we combined the 5 DNNs with two convolutional nets that differ in input features (logmel and gammatone filterbanks) using tree array score combination [11]. The tree array score combination allows the summation of scores coming from models with different decision trees. This was necessary because the CNNs were built using a decision tree with 8000 leaves (as opposed to 16000 for the ensemble DNNs). In Table 3 we present results for both 1-best and consensus decoding using sequence trained models.

Table 3: *Comparison of WER(%) for ST combined DNNs and CNNs on the dev set.*

Fusion	WER 1-best	WER cons.
5DNN+CNN logmel	34.6	33.9
5DNN+CNN logmel+ CNN gamma	34.0	33.7

3. Language Models

For training language models, we use a collection of corpora provided by LDC, the transcripts for the acoustic model training data, various text corpora including LDC2012E54, LDC2012E16, LDC2012E21, LDC2012E04, LDC2012E16, and data used for machine translation. The total number of words (tokens) in all the corpora exceeds 700 million, but the most relevant part is the acoustic model training transcripts which has only 900K words. Many of the corpora contain text data which are not conversational and not Egyptian Arabic. We selected a vocabulary of 386K words, including 378K Arabic words and 8K English words. To build the baseline language model, for each corpus, we train a 4-gram model with modified Kneser-Ney smoothing [12] and then linearly interpolate the models with weights chosen to optimize perplexity on a held-out set (tune set), ending up with a single n-gram model containing 163M n-grams. In addition to the dev set which is used to report WER in earlier experiments we use a tune and test set also to validate our systems. The tune and test sets correspond to 1.5 and 3 hours of speech.

Even though the acoustic model training transcript is very small (900K words) compared with the entire set of text data (700M words), it is by far the most important component, having a weight of 0.79 in the final interpolated n-gram LM. The first two lines in Table 4 show that the n-gram LM trained on just the acoustic transcripts has a perplexity of 375 compared with 313 for the large n-gram LM trained on all the data.

Table 4: *Tune set perplexities for different language models.*

LM	Perplexity
n-gram (actrain transcripts, 900K words)	375
163M n-gram (all sources, 700M words)	313
n-gram + model M	289
n-gram + model M + NNLM	270

The 163M n-gram LM is used as the baseline LM for speech recognition experiments and to generate lattices for rescoring with better language models, such as model M, a class-based exponential model [13]. We trained a model M LM on each corpus and interpolated them together with the 163M n-gram LM, reducing the perplexity from 313 to 289 (8% relative) and the WER by 0.3-0.8% on various test sets as shown in Table 5.

Table 5: *WER(%) for different language models.*

LM	tune	dev	test
Baseline (n-gram LM)	32.5	33.7	36.2
n-gram + model M	32.1	33.0	35.9
n-gram + model M + NNLM	31.9	32.7	35.5

We also built a neural network language model (NNLM) [14, 15, 16] on the 900K word acoustic transcripts. It is a 6-gram model with 100 hidden units and with an output vocabulary limited to the 20K most frequent words. This was the optimal configuration found in [16]. The neural network LM was interpolated with model M and n-gram LMs and used for lattice rescoring. Compared with model M, the perplexity was reduced from 289 to 270 (7% relative) and the WER was further reduced by 0.3-0.4% on the different test sets.

In total, the combination of model M and NNLM improved the WER by 0.6-1.1%.

4. Machine Translation Experiments

The speech recognition outputs are automatically translated from Egyptian Arabic to English using a machine translation system developed for the program. The system is a syntax-based hierarchical [17] decoder that also allows for tree-to-string rules in addition to a hierarchical phrase-based (Hiero) grammar [18]. The acoustic model weight and presence of hesitation words/punctuation in the ASR transcript are optimized for the best MT outputs. Table 6 compares MT results on human transcripts versus MT on ASR output for the dev and test sets used earlier.

Table 6: *Word error rates for different language models.*

Dev set			
	BLEU	TER	(TER-BLUE)/2
MT on human transcript	25.99	57.18	15.60
MT on ASR output	17.14	67.84	25.35
Test set			
	BLEU	TER	(TER-BLUE)/2
MT on human transcript	24.23	59.86	17.82
MT on ASR output	16.93	70.11	26.59

An identical ASR and MT pipeline was also employed for automatic translation of Mandarin Chinese speech. We observe similar word error rates and machine translation scores on that task as well.

5. Conclusions

We have presented the IBM speech transcription system for the DARPA BOLT-A evaluation. For both Egyptian Arabic and Chinese Mandarin, we have used a set of DNN/CNN based acoustic models along with several language model variants. The performance of these systems on conversational telephone speech from non-English languages is much lower than results on similar tasks [19] based on English. The differences between human and automatic translations also show that significant improvements still need to be made in recognition of conversational speech in these settings.

6. Acknowledgements

The authors would like to thank Brian Kingsbury for his suggestions on building ASR acoustic models and Yaser Al-onaizan for his help with ASR-MT integration for this evaluation.

7. References

[1] NIST, "BOLT Activity A Machine Translation Evaluation Plan For Phase 3," http://www.nist.gov/itl/iad/mig/upload/BOLT_phase3_MT_evalplan_v4.pdf, [Online; accessed 2015-03-20].

[2] L. Mangu, H.-K. Kuo, S. Chu, B. Kingsbury, G. Saon, H. Soltau, and F. Biadsy, "The IBM 2011 GALE Arabic Speech Transcription System," in *IEEE ASRU*, 2011.

[3] H. Soltau, L. Mangu, and F. Biadsy, "From Modern Standard Arabic To Levantine ASR: Leveraging GALE For Dialects," in *IEEE ASRU*, 2011.

[4] T. Buckwalter, "LDC2004L02: Buckwalter Arabic Morphological Analyzer - Version 2.0," in *Linguistic Data Consortium*, 2004.

[5] N. Habash and O. Rambow, "Arabic Diacritization Through Full Morphological Tagging," in *NAACL HLT*, 2007.

[6] B. Kingsbury, T. Sainath, and H. Soltau, "Scalable Minimum Bayes Risk Training Of Deep Neural Network Acoustic Models Using Distributed Hessian-free Optimization," in *ISCA Interspeech*, 2012.

[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient Based Learning Applied To Document Recognition," *Proceedings of the IEEE*, 1998.

[8] M. S. et al., "An Efficient Implementation Of The Patterson-Holdsworth Auditory Filterbank," *Apple Computer, Perception Group, Tech. Rep*, 1993.

[9] S. Thomas, G. Saon, M. Van Segbroeck, and S. Narayanan, "Improvements To The IBM Speech Activity Detection System For The DARPA RATS program," in *IEEE ICASSP*, 2015.

[10] N. Jaitly, V. Vanhoucke, and G. Hinton, "Autoregressive Product Of Multi-frame Predictions Can Improve The Accuracy of Hybrid Models," in *ISCA Interspeech*, 2014.

[11] H. Soltau, G. Saon, and B. Kingsbury, "The IBM Attila Speech Recognition Toolkit," in *IEEE SLT*, 2010.

[12] S. Chen and J. Goodman, "An Empirical Study Of Smoothing Techniques For Language Modeling," *Computer Speech & Language*, 1999.

[13] S. Chen, "Shrinking Exponential Language Models," in *NAACL HLT*, 2009.

[14] Y. Bengio, R. Ducharme, and P. Vincent, "A Neural Probabilistic Language Model," in *Advances in Neural Information Processing Systems*, 2001.

[15] H. Schwenk, "Continuous Space Language Models," *Computer Speech & Language*, 2007.

[16] A. Emami and L. Mangu, "Empirical Study Of Neural Network Language Models For Arabic Speech Recognition," in *IEEE ASRU*, 2007.

[17] D. Chiang, "A Hierarchical Phrase-based Model For Statistical Machine Translation," in *ACL*, 2005.

[18] B. Zhao and Y. Al-onaizan, "Generalizing Local And Non-local Word-reordering Patterns For Syntax-based Machine Translation," in *EMNLP*, 2008.

[19] H. Soltau, G. Saon, and T. Sainath, "Joint Training Of Convolutional and Non-convolutional Nueral Networks," in *IEEE ICASSP*, 2014.