AN ANALYSIS OF CONVOLUTIONAL NEURAL NETWORKS FOR SPEECH RECOGNITION

Jui-Ting Huang, Jinyu Li, and Yifan Gong

Microsoft Corporation, One Microsoft Way, Redmond, WA 98052

{jthuang;jinyli;ygong}@microsoft.com

Abstract

Despite the fact that several sites have reported the effectiveness of convolutional neural networks (CNNs) on some tasks, there is no deep analysis regarding why CNNs perform well and in which case we should see CNNs' advantage. In the light of this, this paper aims to provide some detailed analysis of CNNs. By visualizing the localized filters learned in the convolutional layer, we show that edge detectors in varying directions can be automatically learned. We then identify four domains we think CNNs can consistently provide advantages over fully-connected deep neural networks (DNNs): channel-mismatched training-test conditions, noise robustness, distant speech recognition, and low-footprint models. For distant speech recognition, a CNN trained on 1000 hours of Kinect distant speech data obtains relative 4% word error rate reduction (WERR) over a DNN of a similar size. To our knowledge, this is the largest corpus so far reported in the literature for CNNs to show its effectiveness. Lastly, we establish that the CNN structure combined with maxout units is the most effective model under small-sizing constraints for the purpose of deploying small-footprint models to devices. This setup gives relative 9.3% WERR from DNNs with sigmoid units.

Index Terms: Convolutional neural networks, DNN, low footprint models, maxout units

1. Introduction

Deep neural network (DNN) based acoustic models have been shown by many groups [1][2][3][4][5] to outperform the conventional Gaussian mixture model (GMM) on many automatic speech recognition (ASR) tasks. Recently, several sites have reported some successful results using deep convolutional neural networks (CNNs) as opposed to standard fully connected DNNs. By convolving along the frequency axis of the log-Mel spectrogram with the proposed limited-weight-sharing structure, [6] reported 8% relative reduction in phone error rate over DNNs on the TIMIT task and a similar gain on an 18-hour voice search task. Significant gains on Broadcast News and the Switchboard corpus are also reported in [7][8] by combining CNNs with speaker-adaptive features and sequence training techniques. Later there were several attempts to apply CNNs to different tasks, including low-resourced scenarios [9] and distant speech recognition [10].

There are two main properties of CNNs that can potentially improve speech recognition performance. First, pooling at a local frequency region makes CNNs more robust to slight formant shifts due to speaker or speaking style variation. Second, sparse local connections of the convolutional layer require far fewer parameters to extract low-level features which avoids over-fitting. However, both properties become less attractive compared to DNNs if an application has large amounts of training data, as a DNN of sufficient size can also learn to normalize variance in the feature space [11] given ample and diverse training data. Generally speaking, the gain of CNNs over DNNs decreases as the amount of training data increases. On the other hand, [12] reported that CNNs do not work well for semi-clean data in the Robust Automatic Transcription of Speech (RATS) program. Therefore, we are interested to find out in which case we should see CNNs' advantage. In other words, what are the tasks where DNNs cannot easily achieve their success by simply adding data?

We start with examining the spectro-temporal filters learned by CNN modeling to verify the structure design and to understand what kind of low-level features have been extracted. Then we use Aurora 4 to analyze CNNs' performance in different conditions. It is observed [12] that CNNs work better than DNNs for noisy channels in the RATS program. Following this direction, we aim to perform separate analysis for CNNs' ability to handle additive noise and channel mismatch, respectively. As CNNs are shown [13] to extract more invariant feature representation, we expect it has higher tolerance to highly distorted speech signals, e.g. distant-talking speech distorted by room reverberation and additive noise. On a 100-hour meeting room task, CNNs obtained relative 6-8% WERR over DNNs when trained on beamformed distant speech [10]. In this paper, we further show that the more distorted the distant speech signals are, the more robust CNNs are compared to DNNs. Here we work on a much larger-scale task, Xbox Kinect market search, where 1000 hours of audio data are used for acoustic model training.

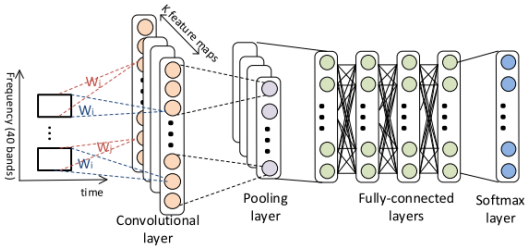
Another area we think CNNs could be good model candidates is for small-footprint applications on devices. A common way to fit DNN-based acoustic models on devices is to reduce the DNN model size by reducing the number of nodes in hidden layers and the number of senone (tied HMM states) targets in the output layer. Although the DNN model size is reduced, significant increase in word error rate is also observed [14]. We hypothesize that the weight-sharing structure of the convolutional layer and dimensionality reduction by max-pooling provides a good trade-off point between parameter reduction and ASR performance. On the other hand, we found that the choice of neuron types is also critical for maximizing the model capacity given the limited parameter budget. Maxout units [15] give the best WER among other units including sigmoid units and rectified linear units (RELU) [16][17]. Combining the CNN structure and maxout neuron types, we are able to reach the best performance for the deep network structure with small footprint.

In Section 2, we describe our CNN architecture and present the learned convolutional kernels for the speech task. We then analyze the performance of CNNs using the Aurora 4 task and the Kinect distant speech recognition task in Section 3. In Section 4, we present our study on small-footprint models. Finally, we summarize our study and conclude the paper in Section 5.

2. Convolutional Neural Networks for Speech

In this paper, all experiments are conducted under the context-dependent deep neural network hidden Markov model (CD-DNN-HMM) framework where a DNN or CNN is used to classify the acoustic input features (log-Mel filter banks in our case) into classes corresponding to the HMM states (tied states or senones, more specifically). We compute the HMM's state emission probability density function by dividing the state posterior derived from DNNs/CNNs by the prior probability of the considered state calculated from the training data.

Figure 1. CNN architecture for speech recognition



2.1. CNN architecture

As shown in Figure 1, a CNN consists of three types of layers, convolutional layers, pooling layers, and fully-connected layers. In a convolutional layer, each neuron takes inputs from a small rectangular section of the previous layer, multiplying those local inputs against the weight matrix W. The weight matrix, or the localized filter, will be replicated across the entire input space to detect a specific kind of local pattern. All neurons sharing the same weights compose a feature map. A complete convolutional layer is composed of many feature maps, generated with different localized filters, to extract multiple kinds of local patterns at every location. In our implementation for speech recognition, the input space is a 2-D plane with frequency and time axis. In particular, each dimension of the frequency axis corresponds to one of 40 log-Mel filter bank coefficients. We only apply convolution along the frequency axis, leaving HMMs to handle temporal variations because most recent works show that shift-invariance in frequency is more important than shift-invariance in time [6][7].

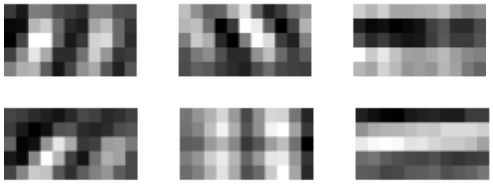
After each convolutional layer, there may be a pooling layer. The pooling layer similarly takes inputs from a local region of the previous convolutional layer and down-samples it to produce a single output from that region. One common pooling operator used for CNNs is max-pooling, which outputs the maximum value within each sub-region. By down-sampling, we not only reduce the computational complexity for the upper layer but also achieve a degree of robustness to slight position change of local patterns. There are also other pooling strategies, like l_p and stochastic pooling, which might improve generalization [8]. Since there have not been shown significant gains by l_p or stochastic pooling, we chose to adopt max-pooling as our pooling strategy in this paper. Finally, after one or more convolutional-pooling building blocks, fully-connected layers will take the output of all neurons from the previous layer and apply high-level reasoning on these "invariant" features.

While it is possible to stack multiple building blocks of convolutional and pooling layers, our experiments found that additional convolutional blocks does not result in further improvement. Therefore our architecture adopts one convolutional layer followed by one max-pooling layer and then four fully-connected layers.

2.2. Visualization of learned filters

Figure 2 shows some spectro-temporal filters learned by the convolutional layer using 60 hours of short message dictation training data. The vertical dimension represents 5 frequency bands, and the horizontal dimension represents 15 successive frames. These filters look like 2-D Gabor filters [18] in varying directions, including horizontal, vertical and diagonal filters. Different from the observation in [19] that diagonal filters are not present in the convolutional layer with random initialization, we found that a CNN is capable of learning different spetro-temporal filters including diagonal ones, which we think forms a powerful analyzer to detect different speech phenomena in a spectrogram [18].

Figure 2. Sampled spectro-temporal filters of size 5x11 learned by the convolutional layer



3. Invariance Property of CNNs

3.1. Noise robustness and channel mismatch

We first analyze the performance of CNNs with Aurora 4 [20], a noise-robust medium-vocabulary task. The clean-condition training set consists of 7138 utterances recorded with a Sennheiser microphone, corresponding to 14 hours of speech data. There are totally 14 evaluation sets. There is also a multi-condition training set, containing data recorded with the Sennheiser microphone and a secondary microphone, respectively, in clean condition or corrupted with 6 types of noise. Two clean evaluation sets (A and C) are recorded with the Sennheiser microphone and another secondary microphone, respectively. The remaining 12 subsets are divided into two groups (B and D), recorded with two types of microphone respectively and corrupted with 6 types of noise, same as the training set.

The baseline GMM-HMM system has 1206 senones, each with 16 Gaussians trained using maximum likelihood estimation criterion on the clean training set. The GMM-HMM system is used to align the training data to get the forced alignment for training the DNN-HMM system. Decoding is performed with the task-standard bigram language model. The baseline DNN is trained with 40-dimensional log-Mel filter-bank features and their first- and second-order derivative features. The input layer is formed from a context window of 11 frames. The DNN has 5 hidden layers with 2048 hidden units in each layer and the final soft-max output layer has 1206 units, corresponding to the senones of the HMM system. The network is initialized with pre-training and then fine-tuned using 25 iterations of back propagation. For the CNN, the bottom hidden layer is replaced with a convolutional layer followed by a max-pooling layer. The convolutional layer has 160 filters, each of which has size of 5 frequency bands and their first- and second-order derivative values, with 11 successive frames. The 160 feature maps are generated by

convolving each filter along the frequency axis. The following max-pooling layer outputs the maximum values over a non-overlapping window covering the outputs of every three frequency bands in each feature map, down-sampling the overall outputs of the convolutional layer to three times smaller. The outputs of the max-pooling layer are fed to the fully-connected layer structure. Due to the usage of small localized filters, the number of parameters used in the CNN (19M) is slightly less than the number in the DNN (21.9M). Both types of models are trained based on the cross-entropy criterion.

We first compare the performance of DNNs vs CNNs using clean training data in Table 1. Given similar amounts of training weights, CNNs consistently outperform DNNs across different test conditions regardless if the acoustic conditions are mismatched or not, producing 3.9% relative error reduction on average. This is consistent with our expectation that CNNs have better generalization ability than DNNs especially when the amounts of training data are limited. Among different test conditions, the largest relative gain (16.2% relative error reduction) is observed on Test Set C consisting of clean test data with microphone distortion. As those recordings are from various microphones mismatched to the one used in the clean training set, the improvement indicates that one of the biggest strengths of a CNN is its robustness to unseen/mismatched channel conditions. Note that utterance-level mean normalization is performed as part of data pre-processing stage (which is orthogonal to the zero-mean unit-variance normalization for neural network inputs), so we examine the effectiveness of CNNs after the basic channel compensation method is applied. We also see that the CNN provides 2.1% relative error reduction from the DNN for Test Set B where various types of noises are added. The gain is smaller than Test Set C, suggesting that CNNs are less effective in handling additive noise than channel distortion.

There are two possible factors in CNNs that might contribute to the robustness to environment/channel mismatch. One is that the weight-sharing structure of the frequency localized filters reduces the number of free parameters for the convolutional layer. This effective way of controlling model capacity could achieve better generalization, as opposed to adopting a fully-connected layer in the input layer. The other important factor is the use of max-pooling on top of the convolutional layer which makes CNNs more robust to translational variance in the input space. To be more specific, the channel/noise distortion affects the spectrum in some local filter banks. Due to the locality processing using frequency localized filters and the invariance processing via max-pooling, features processed by the convolution and pooling layers become more robust to distortion. Compared to DNNs which fully connect the inputs from all the filter banks, CNNs produce more invariant high-level features for the following network layers to perform the classification.

We are interested in knowing which factor is more important for the effectiveness of CNNs for Aurora-4 scenarios. Therefore we experimented with removing the max-pooling layer after the convolutional layer and directly connected the output of convolutional layer to the fully-connected layers. In other words, it is a CNN without any max-pooling. This model is denoted as CNN-NM in Table 1. Trained on the clean data, the WER is even worse than the clean DNN, as shown in the last row of the Table 1, which suggests the essential role of max-pooling in CNNs.

We also compare the performance of DNNs vs CNNs trained on multi-conditioned data in Table 2. The overall relative 2.9% error reduction by the CNN is slightly smaller than that achieved by the clean-trained models, showing that generalization to unseen data is the main advantage of CNNs. But the gain on Test set C, the channel mismatch case, is still significant (8.6% WRR) and remains largest among all test conditions.

Table 1. Word error rates of clean trained DNN and CNN models on Aurora 4 (Test set A-D)

	A	B	C	D	Avg
DNN	4.4	23.8	22.8	41.8	30.1
CNN	4.2	23.3	19.1	40.2	28.9
CNN-NM	4.6	25.3	22.4	43.3	31.3

Table 2. Word error rates of multi-style trained DNN and CNN models on Aurora 4 (Test set A-D)

	A	B	C	D	Avg
DNN	4.9	9.0	9.3	20.9	13.8
CNN	5.1	8.8	8.5	20.1	13.4

Table 3. Results of DNN and CNN models on Kinect beam-formed data recorded at different distance (1, 2, and 3 meter)

Distance	1m	2m	3m	Avg
#. Word counts	21629	22989	23099	
WER of DNN	13.79	16.06	16.00	15.31
WER of CNN	13.38	15.33	15.07	14.62
WERR(%)	2.97	4.55	5.81	4.51

3.2. Beamformed distant speech

We next examine the effectiveness of CNNs in the context of distant speech recognition. In this experiment, the training data are about 1000 hours of anonymized audio data recorded by the Kinect device and have been processed by beamforming. The test set consists of 18683 utterances recorded at 1, 2, or 3 meters away from the Kinect device. The DNN and CNN are trained using the same structures as described in Section 3.1, except that the output layer has 6700 nodes corresponding to senones derived from a decision tree trained with the GMM-HMM system. Also, the filter configuration used for the CNN is 8 frequency bands by 15 context frames. The recognition results are shown in Table 3.

We found that even with one thousand hour of training data, the improvement of the CNN over the DNN is remarkably significant (4.51% relative error rate reduction) for distance speech recognition. We believe this is the largest data set so far to report the effectiveness of CNNs. To our best knowledge, in the previous literature, the largest corpus that has been used in the study of CNNs for speech recognition is 400-hour Broadcast News [8].

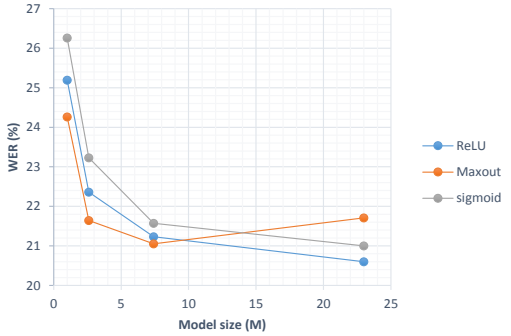
While one might suspect that for speech recognition, the advantage of a CNN over a DNN due to its better generalization to unseen data will soon disappear as long as there are large enough amounts of training data for a DNN to learn from, distant speech recognition provides an area for CNNs to show their robustness to distortion that cannot be easily achieved by DNNs. Distant-talking speech has more distortion than close-talking speech because the speech signals are increasingly degraded by additive noise and room reverberation as the distance between the speaker and the microphones grows. Even with beamforming and dereverberation techniques, there are some errors introduced by microphone array processing [21]. Another supporting evidence is that the gain of CNNs over DNNs increases as the distance between the speaker and microphone arrays increases,

which suggests that the more distorted the signals are, the more effective CNNs become. Note that since audio data recorded at different distances are different utterances, the absolute word error rates across distance are not directly comparable.

4. CNNs for Small-Footprint Models

In this section, we use a Microsoft internal Windows Phone short message dictation task to investigate the most effective model structure with small-footprint constraint. The transcribed training data has 60 hours of US-English audio. The test set is extracted from the live data of the Windows Phone task. Again, the input feature to CD-DNN-HMM system is a 40-dimension log-filter-bank feature with up to second-order derivatives. We augment the feature vectors with previous and next 5 frames (5-1-5). The system uses 6k senones, determined by the baseline CD-GMM-HMM system.

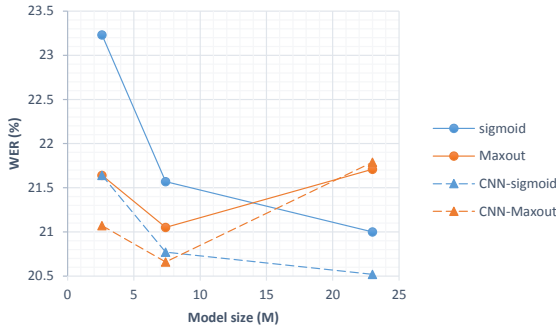
Figure 3. Comparison of sigmoid, ReLU and maxout units for DNN under different model sizes: 256x5, 512x5, 1024x5, 2048x5 (corresponding to 1M, 2.6M, 7.4M and 23M parameters)



First we investigated the behavior of different neurons under different sizing constraints by training DNNs of 5 hidden layers, with 256, 512, 1024, and 2048 neurons in each hidden layer, corresponding to 1M, 2.6M, 7.4M and 23M parameter, respectively. The neuron types include sigmoid, ReLU, and maxout units. For maxout units, the pooling size of 2 is used based on our experience. To make the number of trainable weights comparable to those from sigmoid and ReLU networks, the actual number of maxout units per layer is 180, 384, 724, 1536, respectively. Figure 3 summarizes the performance of different neuron types for different model sizes. We can see that maxout units perform best for the 256x5 and 512x5 structures but start to degrade for the 2048x5 structure. (KxH notation stands for H hidden layers of K neurons.) This indicates that maxout units have the greatest tendency to overfit among all neuron types. To avoid over fitting by adding dropout, we are able to boost the performance of maxout networks to 20.7%, outperforming all other units.

However, the interest of this paper is to find the most suitable neural network model under small-size constraint. To this end, maxout units appear to be the best option. We further suspect that CNNs could be a good structure given limited number of parameters. To verify this, we train CNNs using sigmoid and maxout units, respectively, for the hidden layers above the CNN layer, and compare their performance under different model sizes in Figure 4. For both units, CNNs consistently outperform DNNs for all model sizes, except for the 2048x5 maxout CNN as maxout units are worse than sigmoid units if without dropout. Among all model sizes, the gain of CNNs is most significant for the 512x5 structure (corresponding to 2.6M in Model Size), relative 6.8% WERR for sigmoid and 2.6% for maxout units, possibly because the smart way of allocating free parameters is more important for the case of limited parameter budget. Finally, for the 512x5 structure which is a common choice for device models [14][22], a CNN combined with maxout units achieves the lowest WER, yielding relative 9.3% WERR from DNNs with sigmoid units. In fact, a 512x5 CNN with maxout model has lower WER than a 2048x5 DNN with sigmoid model while the model size is 10 times smaller. A recent study has also reported the effectiveness of convolutional deep maxout networks on the TIMIT phone recognition task [23]. On top of this, our study explored its effectiveness for the large vocabulary task and established that it is the most effective model under small-sizing constraints for the purpose of deploying small-footprint models to devices.

Figure 4. Comparison of CNNs with sigmoid and maxout units under different model sizes: 512x5, 1024x5, 2048x5 (corresponding to 2.6M, 7.4M and 23M parameters)



5. Conclusions and Future Works

In this paper, we provided several detailed analyses of CNNs' capability and performance. We first showed that a CNN with random initialization can automatically learn various sets of edge detectors to locally extract low-level features. Our extensive experiments suggested that CNNs have an advantage over DNNs in the following four domains: channel-mismatched training-test conditions, noise robustness, distant speech recognition, and low-footprint models. One example is the 1000-hour Kinect distant speech recognition task, where a CNN established 4% relative improvement in WER over a DNN of similar size. For low-footprint models, we found the excellent learning ability of maxout neurons can be combined with a CNN structure to achieve the best performance for a small neural network. A CNN combined with maxout units yielded relative 9.3% WERR from a DNN with sigmoid units. In fact, a 512x5 CNN with maxout model has lower WER than a 2048x5 DNN with sigmoid model while the model size is 10 times smaller. Given the success of our separate work in distribution learning for small-size models [22], we are currently investigating if using convolutional maxout networks with distribution learning can bring further improvements over standard DNNs. Another future direction is to investigate the effectiveness of CNNs in the Recurrent Neural Network (RNN) framework, e.g. Long Short-Term Memory (LSTM) RNNs [24], which has emerged recently as a potentially better sequential model for acoustic modeling.

6. References

[1] D. Yu, L. Deng, and G. Dahl, "Roles of pretraining and fine-tuning in context-dependent DBN-HMMs for real-world speech recognition," in *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

[2] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A. Mohamed, "Making deep belief networks effective for large vocabulary continuous speech recognition," in *Proc. Workshop on Automatic Speech Recognition and Understanding*, pp. 30–35, 2011.

[3] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[4] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[5] L. Deng, J. Li, J. -T. Huang et al. "Recent advances in deep learning for speech research at Microsoft," in *Proc. ICASSP*, 2013.

[6] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, D. Yu, "Convolutional neural networks for speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing,* vol.22, no.1, pp.1533-1545, 2014.

[7] T.N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc IEEE ICASSP*, 2013.

[8] T.N. Sainath, B. Kingsbury, A. Mohamed, G.E. Dahl, G. Saon, H. Soltau, T. Beran, A.Y. Aravkin, and B. Ramabhadran, "Improvements to deep convolutional neural networks for LVCSR," in Proc *IEEE ASRU*, 2013.

[9] Y. Miao and F. Metze, "Improving language-universal feature extraction with deep maxout and convolutional neural networks," in *Proc. Interspeech*, 2014.

[10] P Swietojanski, A Ghoshal, and S Renals, "Convolutional neural networks for distant speech recognition," *IEEE Signal Process. Letters*, 2014,

[11] D. Yu, M. Seltzer, J. Li, J-T. Huang, F. Seide, "Feature learning in deep neural networks - studies on speech recognition tasks", *ICLR* 2013.

[12] H. Soltau, H-K. Kuo, L. Mangu, G. Saon, T. Beran, "Neural network acoustic models for the DARPA RATS program," in *Proc. Interspeech*, 2013.

[13] I. Goodfellow, H. Lee, Q. Le, A. Saxe, A. Ng, "Measuring invariances in deep networks," in *Proc. NIPS,* 2009.

[14] X. Lei, A. Senior, A., A. Gruenstein, and J. Sorensen, "Accurate and compact large vocabulary speech recognition on mobile devices," in *Proc. Interspeech*, 2013.

[15] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," in *Proc. ICML*, 2013.

[16] M.D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q.V. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, et al., "On rectified linear units for speech processing," in Proc. *ICASSP*, 2013.

[17] P. Swietojanski, J. Li, and J.-T. Huang, "Investigation of maxout networks for speech recognition," in *Proc. ICASSP*, 2014

[18] T. Ezzat, J. Bouvrie, and T. Poggio, "Spectro-temporal analysis of speech using 2-d Gabor filters," in *Proc. Interspeech*, 2007.

[19] S.-Y. Chang and N. Morgan, "Robust CNN-based speech recognition with Gabor filter kernels," in *Proc. Interspeech*, 2014.

[20] N. Parihar and J. Picone, "Aurora working group: DSR front end LVCSR evaluation AU/384/02," *Tech. Rep.,* Institute for Signal and Information Processing, Mississippi State Univ., 2002.

[21] K. Kumatani, T. Arakawa, K. Yamamoto, J. McDonough, B. Raj, R. Singh, and I. Tashev, "Microphone array processing for distant speech recognition: towards real-world deployment," *APSIPA Annual Summit and Conference*, 2012

[22] J. Li, R. Zhao, J.-T. Huang, and Y. Gong, "Learning small-size DNN with output-distribution-based criteria," in *Proc. Interspeech*, 2014.

[23] L. Toth, "Convolutional deep maxout networks for phone recognition," in *Proc. Interspeech*, 2014.

[24] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Interspeech*, 2014, pp. 338–342.