TIME-FREQUENCY CONVOLUTIONAL NETWORKS FOR ROBUST SPEECH RECOGNITION

Vikramjit Mitra, Horacio Franco

Speech Technology and Research Laboratory, SRI International, Menlo Park, CA
{vikramjit.mitra, horacio.franco}@sri.com

ABSTRACT

Convolutional deep neural networks (CDNNs) have consistently shown more robustness to noise and background contamination than traditional deep neural networks (DNNs). For speech recognition, CDNNs apply their convolution filters across frequency, which helps to remove cross-spectral distortions and, to some extent, speaker-level variability stemming from vocal tract length differences. Convolution across time has not been considered with much enthusiasm within the speech technology community. This work presents a modified CDNN architecture that we call the time-frequency convolutional network (TFCNN), in which two parallel layers of convolution are performed on the input feature space: convolution across time and frequency, each using a different pooling layer. The feature maps obtained from the convolution layers are then combined and fed to a fully connected DNN. Our experimental analysis on noise-, channel-, and reverberation-corrupted databases shows that TFCNNs demonstrate reduced speech recognition error rates compared to CDNNs whether using baseline mel-filterbank features or noise-robust acoustic features.

Index Terms— *time-frequency convolution nets, deep convolution networks, robust features, robust speech recognition.*

1. INTRODUCTION

Deep learning techniques [1] are now integral to current automatic speech recognition (ASR) systems [2]. Deep learning has been used for feature representation [3], acoustic modeling [1], and language modeling [4]. Although the results from deep neural networks (DNNs) have always been encouraging, current research is focused on both improving the state-of-the-art and increasing scientific understanding of deep learning's strengths and weaknesses. Although DNNs have been observed to work highly reliably under matched conditions, they are susceptible to performance degradations under mismatched conditions [28]. Speech-signal degradations (such as reverberation, noise, and channel mismatch) can significantly reduce DNN recognition accuracy, revealing DNN's vulnerability [4, 5] to unseen conditions.

Noise, reverberation, and channel mismatches are the usual causes of speech data mismatches and, hence, are the common sources of performance degradation for ASR systems [4]. Robust features have been demonstrated to help in noisy [5] and reverberant [7-9] conditions, where such features provide an invariant representation of speech despite environmental distortions.

Recently, convolutional neural networks (CNNs) [10, 11] have been proposed and are often found to outperform fully connected DNN architectures [6, 9]. CNNs are also expected to be noise-robust [11], especially in the cases where noise/distortion is localized in the spectrum. Speaker-normalization techniques, such as vocal tract length normalization (VTLN) [12], are also found to have less impact on speech recognition accuracy for CDNNs as compared to for DNNs. With CDNNs, the localized convolution filters across frequency tend to normalize the spectral variations in speech arising from vocal tract length differences, enabling the CDNNs to learn speaker-invariant data representations. Recent results [6, 7, 8] also showed that CDNNs are more robust to noise and channel degradations than DNNs. Typically for speech recognition, a single layer of convolution filters is used on the input contextualized feature space to create multiple feature maps that, in turn, are fed to fully connected DNNs. However, in [10], adding multiple convolution layers (usually up to two) was shown to improve the performance of CDNN systems beyond their single-layer counterparts.

In [13, 14], convolution across time has been applied over windows of acoustic frames that overlap in time to learn classes such as phone, speaker, and gender. In 1980s, the notion of weight sharing over time was first introduced through the time-delay neural network (TDNN) [15]. Recent DNN/CDNN architectures use a hybrid topology, in which DNN/CDNNs produce subword unit posteriors, and a hidden Markov model (HMM) performs the final decoding. As the HMMs typically do model time variations well, time convolution is usually ignored in current CNN architectures. However, with environmental degradation such as reverberation, the introduced distortion typically corrupts time-scale information.

The environment where the speech sample is collected introduces reverberation, which is the effect of multiple reflections of the source sound from the ambient enclosure. The degree of reverberation is usually defined by the time (typically in seconds) required for the reflections of a direct sound to decay to 60 dB, which is denoted as the RT60 value of reverberation. The higher the RT60 values, the more distorted the reverberated speech sounds, and vice versa. Such multiple reflections or reverberation seriously degrade speech-signal quality, which, in turn, seriously degrades the performance of ASR systems. As reverberation introduces multiple delayed reflections of the source to itself, the distortion predominantly is hence spread across the time scale. A common approach to cope with such distortion has been to learn an inverse filter that models the room impulse response (RIR) and then to perform inverse filtering on the reverberated speech to mitigate the effects of reverberation [16].

In this work, we revisit the CNN architecture. We first revisit the two-layer convolution approach suggested in [10] (for simplicity, we call it the double-convolution neural network or DCNN) and evaluate its performance with respect to the conventional single-layer convolution network (CNN). We propose a modified convolution network in which two parallel

layers of convolution filters operate on the input feature space: one across time (time-convolution), and the other across frequency (frequency-convolution). The feature maps from these two convolution layers are fused and then fed to a fully connected deep neural network. We name this system the time-frequency convolution network or TFCNN. Finally, we present a third modified CNN, in which double convolution is performed over frequency (essentially replicating the DCNN), and a single parallel convolution layer is used for time. The final feature maps from these layers are merged and then fed to a deep neural net. We name this system the time-frequency double-convolution neural network (TFDCNN).

We use two datasets for comparing the performance of different acoustic models. First, we use the data distributed through the REVERB (REverberant Voice Enhancement and Recognition Benchmark) 2014 challenge [17] to train and evaluate our systems. Second, we use the noisy English continuous speech dataset Aurora4 [18], which contains speech data corrupted by different noise types at different signal-to-noise ratios (SNRs) and recorded by different microphone types. We evaluate our system using both baseline mel-filterbank energy features and some of the noise-robust acoustic features previously found to work well under noise and reverberation corruption.

2. DATASET

The REVERB 2014 challenge speech dataset [17] contains single-speaker utterances recorded with one-channel, two-channel, or eight-channel circular microphone arrays. The dataset includes a training set, a development set, and an evaluation set. The training set consists of the clean WSJCAM0 [19] dataset, which was convolved with room impulse responses (with reverberation times from 0.1 sec to 0.8 sec) and then corrupted with background noise. The evaluation and development data contain both real recordings (real data) and simulated data (sim data). The real data is borrowed from the MC-WSJ-AV corpus [20], which consists of utterances recorded in a noisy and reverberant room. For the sim data, reverberation effects were artificially introduced. For our experiments, we used the channel-1 training data to build our acoustic models, which contained altogether 7861 utterances (5699 unique utterances). The simulated dev set included 742 utterances in each of the far- and near-microphone conditions, almost equally spread in three room types (1, 2, and 3). The real dev set contained 179 utterances almost equally spread into far- and near-microphone conditions. The simulated evaluation set contained 1088 utterances in each of the far- and near-microphone conditions, each of which was split into three room conditions (1, 2, and 3). The real evaluation set contained 372 utterances split equally between far- and near-microphone conditions. For this dataset, no speaker information was used during acoustic model training and testing, and all processing was independent of the room impulse responses and the relative position of the speakers with respect to the recording device. We report our results in terms of word error rate (WER), using conditions identical to those of the baseline system distributed with the REVERB 2014 challenge data.

The Aurora4 [18] dataset was created from the standard 5K Wall Street Journal (WSJ0) database and includes 7180 training utterances of approximately 15 hours total duration, and 330 test utterances, each with an average duration of 7 seconds. It contains six additive noise versions with channel matched and mismatched conditions. The acoustic data (both training and test sets) comes with two different sampling rates (8 kHz and 16 kHz). Two different training conditions were specified: (1) clean training, which is the full SI-84 WSJ train set without any added noise; and (2) multi-condition training, with about half of the training data recorded by using one microphone, and the other half recorded by using a different microphone (hence incorporating two different channel conditions), with different types of added noise at different SNRs. The noise types are similar for the training and testing data sets. The Aurora4 test data includes 14 test sets from two different channel conditions and six different added noises (in addition to the clean condition). The SNR was randomly selected between 0 and 15 dB for different utterances. The six noise types used were (1) car; (2) babble; (3) restaurant; (4) street; (5) airport; and (6) train (set07), along with a clean condition. The evaluation set comprised 5K words in two different channel conditions. The original audio data for test conditions 1–7 was recorded with a Sennheiser microphone, while test conditions 8–14 were recorded by using a second microphone that was randomly selected from a set of 18 different microphones (more details in [18]). The different noise types were digitally added to the clean audio data to simulate noisy conditions. These 14 test sets mentioned above were typically grouped into four subsets: (A) clean; (B) matched-channel, noisy; (C) matched-channel, clean with channel distortion, and (D) noisy with channel distortion, which are usually referred to as test sets A, B, C, and D, respectively. A part of the clean training (893 out of 7139 utterances) and the matched-channel noisy training (2676 utterances), which were not used in the multi-conditioned training set of Aurora4, were used as a held-out cross-validation set that was used to track the cross-validation error during neural network training.

3. ACOUSTIC FEATURES

We used several different acoustic features to parameterize speech. We briefly outline the features explored in this section.

3.1 Damped Oscillator Coefficients (DOC)

DOCs use forced damped oscillators to model the hair cells found within the human ear [21]. DOC tracks the dynamics of the hair cell oscillations to auditory stimuli and uses that as the acoustic feature. In human auditory system, the hair cells detect the motion of incoming sound waves and excite the neurons of the auditory nerves, which then transduce the relevant information to the brain. For our DOC processing, a bank of gammatone filters that produces 40 bandlimited subband signals analyzed the incoming speech signal. The gammatone filters were equally spaced on the equivalent rectangular bandwidth (ERB) scale. The outputs of the gammatone filters were used as the forcing functions to an array of 40 damped oscillators, whose response was then used as the acoustic feature. We analyzed the damped oscillator response by using a Hamming window of 26 ms with a frame rate of 10 ms. The power signal from the damped oscillator response was computed and then root compressed using the 15^{th} root, resulting in the 40 dimensional features that comprised the DOC feature in our experiments.

3.2 Normalized Modulation Coefficients (NMC)

The NMC [22] feature captures and uses the amplitude modulation (AM) information from bandlimited speech signals. NMC is motivated by AMs of subband speech signals playing an important role in human speech perception and recognition [23]. NMCs are obtained by using the approach outlined in [22], with which the

features are generated from tracking the AM trajectories of subband speech signals in a time domain by using a Hamming window of 26 ms with a frame rate of 10 ms. For our processing, a time-domain gammatone filterbank with 40 channels equally spaced on the ERB scale was used to analyze the speech signal. A modified version of the Discrete Energy Separation algorithm (DESA) that produced instantaneous estimates of AM signals then processed the subband signals. The powers of the AM signals were then root compressed using the 15th root. The resulting 40-dimensional feature vector was used as the NMC feature in our experiments.

3.3 Modulation of Medium Duration Speech Amplitudes (MMeDuSA)

MMeDuSA [24] is similar in essence to the NMC features, as it tracks the subband AM signals of speech by using a medium-duration analysis window. On top of tracking the subband AM signals, MMeDuSA also tracks the overall summary modulation information. The summary modulation plays an important role in both tracking voiced speech and locating events, such as vowel prominence/stress, etc. Unlike NMCs, MMeDuSA does not use the DESA algorithm to track the AM signals, but instead directly uses the nonlinear Teager energy operator [25] to crudely estimate the AM signal from the bandlimited subband signals. For our processing, the MMeDuSA-generation pipeline used a time-domain gammatone filterbank with 40 channels equally spaced on the ERB scale. The MMeDuSA pipeline used a Hamming analysis window of ~51 ms with a 10 ms frame rate. The powers were root compressed, and the resultant information was used as the acoustic feature in our experiments. More details regarding MMeDuSA feature extraction can be obtained in [24].

4. TIME FREQUENCY CONVOLUTION

CNNs [11] have demonstrated lower WERs compared to DNNs for clean [10], noisy [6], and reverberated [8, 9] datasets. Traditional CNNs for speech recognition usually apply the convolution operation across frequency, providing the network with immunity to small spectral shifts, such as those introduced by speaker-specific vocal tract length differences. In cases such as reverberation, where delayed versions of reflection introduce temporal artifacts, convolution across time can be useful. Figure 1 shows block diagram of a network using two separate convolution layers, one operating across time, and the other operating across frequency.

To illustrate how time and frequency convolution followed by ax-pooling is performed, let us assume the input feature map can be represented by either feature vectors V or U where

$$V = [V_1, V_2, \dots V_f, \dots . V_F]$$
$$U = [U_1, U_2, \dots U_t, \dots . U_T]^{\mathrm{T}}$$

where, V_f representing the feature vector at frequency band f and U_t represents the feature vector at a time frame t. Note that for simplicity let us assume that these feature vectors only represents the spectral energies and their dynamic information (Δ and ΔΔ) is not used. For frequency convolution, let us assume that the layer has K bands with N activations. Following the representation in [11], the convolution layer activations after non-linear activation function operation can be represented as

$$h_{k,n} = \sigma\left(\sum_{b=1}^{B-1} w_{b,n} V_{b+k}^{T} + \beta_n\right) \quad (1)$$

where σ(.) is the output activation function; B is the band size for convolution operation on V; w and β represents the weight and bias terms of the convolution layer. Similarly for time convolution, let us assume the layer has L bands (operating on time frames) and M activations, in such case the convolution layer activations after non-linear activation function operation can be represented as

$$g_{l,m} = \sigma\left(\sum_{c=1}^{C-1} \omega_{c,m} U_{c+m} + \gamma_n\right) \quad (2)$$

where σ(.) is the output activation function; C is the frame-band size for convolution operation on U; ω and γ represents the weight and bias terms of the time convolution layer. Now, after the pooling layer the outputs of each of these layers can be represented as

$$p_{b,n} = \max_r\left(h_{b\times i+r,n}\right)$$
$$q_{c,l} = \max_s\left(g_{c\times j+s,l}\right) \quad (3)$$

where, r and s are the pooling size, i and j are the sub-sampling factor, b and c are the pooling band sizes for frequency and time convolution layers respectively. The output feature space is flattened to a vector and then concatenated and fed to the fully connected neural net. We have used 75 filters to perform time convolution, and 200 filters to perform frequency convolution. For time and frequency convolution, eight bands were used. A max-pooling over three samples was used for frequency convolution, while max-pooling over five samples was used for time convolution. The feature maps after both the convolution operations were concatenated and then fed to a fully connected neural net, which had 1024 nodes and four hidden layers. Figure 1 briefly outlines the TFCNN architecture. Instead of a 2-D convolution operation, we selected to perform two independent 1-D convolutions, because optimizing the parameter configuration when the two layers are treated independently is easier, and because it also enables using different input feature maps or context sizes.

Figure 2 shows the block diagram of a two-layered or double-convolution network (DCNN), where the first convolution layer operates on the input feature map. The max-pooled output of the first convolution layer is used as the input to the second convolution layer. This architecture is similar to the two-layered convolution network presented in [10]. Note that the convolution operation performed in this architecture is only across the frequency scale. The first convolution layer uses 128 filters with 8 bands and a pool size of 3. The second convolution layer uses 256 filters with 8 bands and a pool size of 3.

Figure 3 shows the block diagram of a time-frequency convolutional net using a double-layer convolution on frequency. This is a hybrid of the previous two CNNs (TFCNN and DCNN). We name the time-frequency double-convolution neural network (TFDCNN). It uses a three-hidden-layer, fully connected network after the convolution layers. The double-convolution layer has the same architecture as mentioned in the last paragraph.

5. RESULTS

We performed different sets of acoustic model training for the two datasets mentioned in Section 2. First, we present the results from using the REVERB 2014 dataset. For this dataset, we trained traditional CNNs systems by using the top-performing robust features (NMC and DOC) reported in [9]. Note that [9] showed that using the velocity coefficient (Δ) is helpful, hence the filterbank features with their Δ were used to train the acoustic model presented in this section. In order to generate the alignments

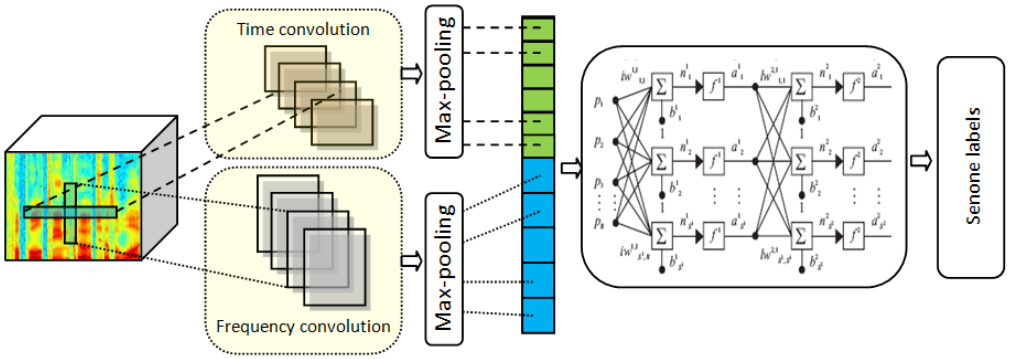

Figure 1. Block diagram showing time-frequency convolution neural nets (TFCNN). The top dotted block shows convolution filters working across time, and the bottom dotted block shows convolution filters working across frequency. The max-pooled outputs of these convolution filters are fed to a fully connected four-layered deep neural net.

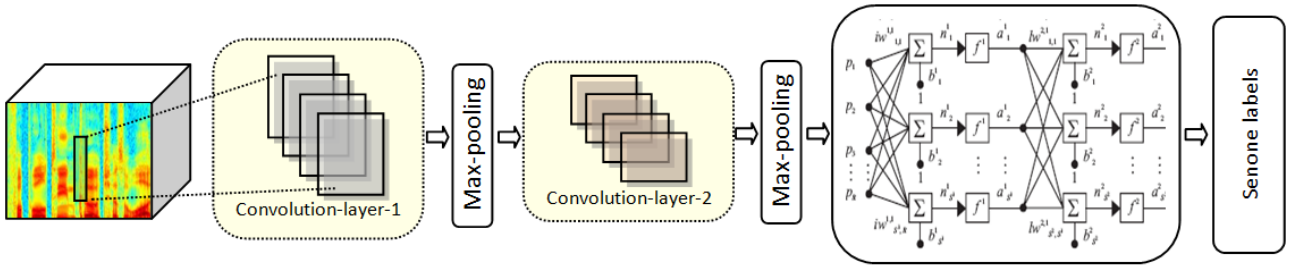

Figure 2. Block diagram showing double-layer convolution neural nets (DCNN). The feature maps from the first convolution layer after max-pooling are used as the input feature maps to the second convolution layer. The max-pooled outputs of the second convolution layer are fed to a fully connected three-layered deep neural net.

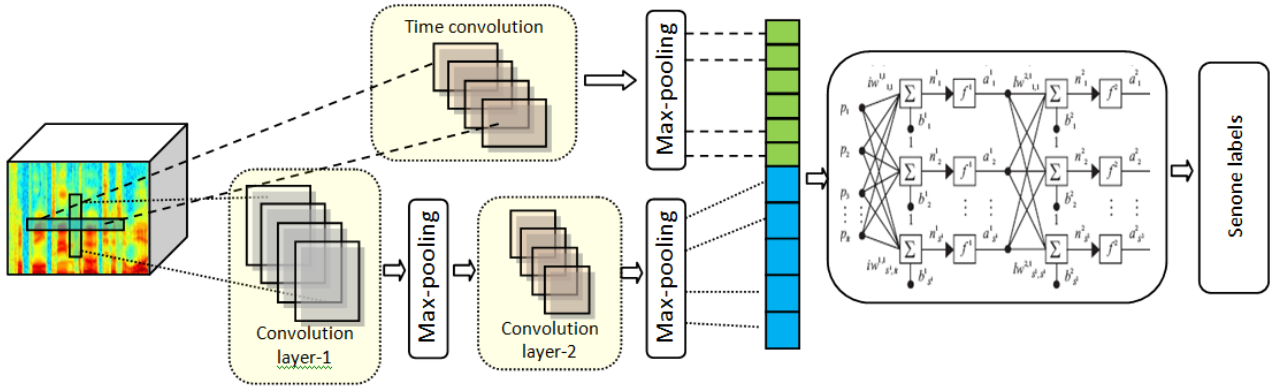

Figure 3. Block diagram showing TFDCNN, which performs two-layer convolution on frequency and one-layer convolution on time. The feature maps from the max-pooled output of each of the two parallel convolution branches are used as the input feature to a fully connected three-layered deep neural net.

necessary for training the CNN system, a GMM-HMM model was used to produce the senones' labels. Altogether, the GMM-HMM system produced 3276 senones. The input layer of the CNN systems was formed by using a context window of 15 frames (7 frames on either side of the current frame).

The CNN acoustic model was trained by using cross-entropy on the alignments from the GMM-HMM system. Two hundred convolutional filters of size 8 were used in the convolutional layer, and the pooling size was set to 3 without overlap. The subsequent fully connected network had four hidden layers, with 1024 nodes per hidden layer, and the output layer had 3276 nodes representing the senones. The networks were trained by using an initial four iterations with a constant learning rate of 0.008, followed by learning rate halving based on cross-validation error decrease.

Training stopped when no further significant reduction in cross-validation error was noted or when cross-validation error started to increase. Backpropagation was performed using stochastic gradient descent with a mini-batch of 256 training examples. In [9], it was demonstrated that for this dataset, the CNNs perform much better than the GMM-HMM system, offering more than 15% relative reduction in word error rates.

We also trained the TFCNN, DCNN, and TFDCNN architectures using the DOC and NMC features. For the TFCNN, the deep neural network had four hidden layers with 1024 neurons in each layer; whereas in the DCNN and TFDCNN, the fully connected network had three hidden layers with 1024 neurons in each layer. Table 1 and 2 present the WERs from the dev and eval sets of the REVERB 2014 challenge.

Table 1. WERs from the different systems using the dev. data of REVERB 2014 dataset.

		WER (sim. eval. data)			WER (real eval. data)		
		Far (avg.)	Near (avg.)	avg.	Far	Near	avg.
DOC	CNN	8.10	14.13	11.12	25.80	27.50	26.65
	TFCNN	**7.73**	**12.90**	**10.32**	**23.50**	**26.30**	**24.90**
	DCNN	8.03	13.77	10.90	24.30	27.70	26.00
	TFDCNN	8.33	13.53	10.93	25.90	27.90	26.90
NMC	CNN	7.73	13.50	10.62	24.90	28.80	26.85
	TFCNN	**7.63**	**12.87**	**10.25**	**23.90**	26.70	**25.30**
	DCNN	8.03	13.83	10.93	24.40	**26.20**	**25.30**
	TFDCNN	7.90	13.23	10.57	25.80	26.70	26.25

Table 2. WERs from the different systems using the eval. data of REVER-2014 dataset.

		WER (sim. eval. data)			WER (real eval. data)		
		Far (avg.)	Near (avg.)	avg.	Far	Near	avg.
DOC	CNN	8.60	13.13	10.87	31.70	32.40	32.05
	TFCNN	**7.90**	**12.67**	**10.28**	**29.00**	**30.00**	**29.50**
	DCNN	8.40	13.27	10.83	29.80	30.80	30.30
	TFDCNN	8.40	13.30	10.85	31.90	29.50	30.70
NMC	CNN	8.10	12.90	10.50	31.00	29.80	30.40
	TFCNN	**7.97**	**12.67**	**10.32**	30.10	30.00	30.05
	DCNN	8.80	13.60	11.20	29.20	28.70	28.95
	TFDCNN	8.50	13.43	10.97	**29.00**	**28.20**	**28.60**

Tables 1 and 2 show that TFCNN always demonstrate reduction in WERs compared to the CNN baseline. The improvements are more pronounced for the unseen real dev and eval data (the last three columns of Tables 1 and 2) than for the seen (simulated) dev and eval data. For DOC features, TFCNN provided a relative reduction of 6.5% and 7.9% in WER on the real dev and eval data compared to the CNN baseline. For NMC features, the relative reductions in WERs for real dev and eval data from TFCNN were 5.7% and 1.1%, respectively, compared to the CNN. Note that for almost all the conditions, DCNN and TFDCNN provided reduction in WERs, but overall TFCNNs performed better than the CNNs. Interestingly for the real eval data using NMC features, the TFDCNN performed the best, reducing the WER by 4.8% compared to the TFCNNs. From the results in Tables 1 and 2, we can conclude that time convolution is definitely beneficial for reverberated data and, hence, convincingly reduced WERs across all different datasets.

Next, we present ASR results from the Aurora-4 dataset. Similar to the last task, a GMM-HMM model was used to align the training data to produce senone labels for training the CNN systems. Altogether 3162 senones were used to train the systems. The input layer of the CNN systems was formed by using a context window of 15 frames (7 frames on either side of the current frame). The CNN acoustic model was trained using cross-entropy. The CNN model had a four-layer (with 1024 neurons) fully connected network; the convolution layer had 200 convolutional filters with 8 bands; and the pooling size was set to 3 without overlap. The networks were discriminatively trained by using an initial few iterations with a constant learning rate of 0.008, followed by learning rate halving based on cross-validation error decrease. Training stopped when no further significant reduction in cross-validation error was noted or when cross-validation error increased. Backpropagation was performed by using stochastic gradient descent with a mini-batch of 256 training examples. The TFCNN, DCNN, and TFDCNN systems had a three-hidden-layer fully connected network, each with 1024 neurons.

In [6], it was shown that for Aurora-4, the robust features provided lower WERs compared to the mel-filterbank (MFB) features, which is also found to be the case here. It was also observed in [6] that VTLN helped to reduce the WER slightly even for CNN systems, hence all the features used in the following experiments were VTLN transformed. Note that unlike the experiments reported earlier, Δ features were not used for the Aurora-4 task. Table 3 presents the WERs from the following features: MFB, DOC, MMeDuSA, and NMC, selected based on their performance cited in [6]. For decoding, we used the standard WSJ trigram language model distributed with the WSJ0 corpus.

Table 3. WER on multi-conditioned training task of Aurora-4 (16 kHz) from the different CNN architectures.

		A	B	C	D	avg.
MFB	CNN	3.50	6.17	6.60	15.72	10.10
	TFCNN	3.60	5.75	6.60	14.58	*9.44*
	DCNN	3.80	6.25	7.20	15.35	10.04
	TFDCNN	4.20	5.88	6.70	15.75	10.05
DOC	CNN	3.60	5.80	6.20	14.03	9.20
	TFCNN	3.70	5.82	6.50	**13.82**	***9.14***
	DCNN	4.00	6.08	6.40	14.67	9.64
	TFDCNN	3.60	6.05	6.40	14.62	9.57
NMC	CNN	3.20	5.77	5.70	14.32	9.24
	TFCNN	**3.10**	**5.63**	5.50	14.33	*9.17*
	DCNN	3.70	6.27	5.80	15.00	9.79
	TFDCNN	3.40	5.78	5.60	14.52	9.34
MMeDuSA	CNN	3.50	5.85	**5.40**	14.25	9.25
	TFCNN	3.30	5.98	**5.40**	13.88	***9.14***
	DCNN	3.60	6.28	5.90	14.62	9.64
	TFDCNN	3.50	6.13	5.50	14.20	9.36

Unlike the results in Tables 1 and 2, we did not see substantial reduction in WER from the new CNN architectures compared to the baseline CNN system. As before, TFCNNs did provide some reduction in WER, but that reduction is not significant. For DOC, NMC, and MMeDuSA, TFCNN provided an approximate 1% relative reduction of WER compared to the baseline CNN systems, but for MFBs that gain was much higher, roughly 6.5%. This may

indicate that the time domain filtering is more effective for the MFB features compared to the robust features DOC, NMC and MMeDuSA under noisy conditions which may be because the robust features already perform filtering across time during their feature extraction process to achieve their robustness, whereas the MFB has none. Finally it can be seen that for channel-mismatched condition (i.e., for test-set D) the TFCNNs always gave the lowest WER compared to the other networks.

Note that in Table 3, the training and testing conditions (i.e., noise, channel mismatch, etc.) are simulated, hence the acoustic variation is limited and is not completely unseen by the model. Whereas in the REVERB 2014 dataset, the real testing condition is much different than the simulated training condition, and, interestingly, the TFCNN model performed much better in the unseen condition than the seen condition. This result may indicate that under controlled train-test environments, the traditional CNN architecture is robust enough; however, for unseen conditions, its performance can be further improved by using the additional time-convolution layer.

6. CONCLUSION

In this work, we revisited the CNN architecture commonly used for acoustic modeling in automatic speech recognition. We demonstrated that using an additional time convolution layer improved ASR performance under reverberant condition. A separate study [26] using a much larger Fisher corpus distributed by IARPA through the ASpIRE speech recognition challenge [27] also showed similar performance gains.

For Aurora-4 noisy data conditions, the TFCNNs showed marginal improvements over the conventional CNNs. Note that the TFCNNs had fewer parameters than the CNNs, as they had one fewer 1024-neuron hidden layer, replaced by the extra convolution layer having 75 filters with 8 bands.

Future research should explore the new CNN architectures with larger datasets, such as Fisher, and explore their benefits compared to the standard CNN architecture.

7. ACKNOWLEDGMENT

This research was partially supported by NSF Grant # IIS-1162046 and BCS 1435831.

The authors would like to thank Andreas Kathol at SRI International for his help with the pronunciation dictionary.

This material is based upon work partly supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-15-C-0037. The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

8. REFERENCES

[1] A. Mohamed, G.E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. on ASLP*, vol. 20, no. 1, pp. 14 –22, 2012.

[2] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," *Proc. of Interspeech*, 2011.

[3] Z. Tuske, P., Golik, R., Schluter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for LVCSR," *Proc. of Interspeech*, 2014.

[4] E. Arısoy, T.N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep neural network language models," *Proc. of NAACL-HLT Workshop,* 2012.

[5] M. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition", Proc of ICASSP, 2013.

[6] V. Mitra, W. Wang, H. Franco, Y. Lei, C. Bartels, and M. Graciarena, "Evaluating robust features on deep neural networks for speech recognition in noisy and channel mismatched conditions," in *Proc. of Interspeech*, 2014.

[7] V. Mitra, W. Wang, Y. Lei, A. Kathol, G. Sivaraman, and C. Espy-Wilson, "Robust features and system fusion for reverberation-robust speech recognition," *in Proc. of REVERB Challenge*, 2014.

[8] V. Mitra, J. Van Hout, M. McLaren, W. Wang, M. Graciarena, D. Vergyri, and H. Franco, "Combating reverberation in large vocabulary continuous speech recognition," *Proc. of Interspeech,* 2015.

[9] V. Mitra, W. Wang, and H. Franco, "deep convolutional nets and robust features for reverberation-robust speech recognition," in *Proc. of SLT*, pp. 548–553, 2014.

[10] T. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural network for LVCSR," *Proc. of ICASSP*, 2013.

[11] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," *Proc. of ICASSP*, pp. 4277–4280, 2012.

[12] P. Zhan and A Waibel, "Vocal tract length normalization for LVCSR," in *Tech. Rep. CMU-LTI-97-150*. Carnegie Mellon University, 1997

[13] H. Lee, P. Pham, Y. Largman, and A. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," *Proc. of Adv. Neural Inf. Process. Syst*. 22, pp. 1096–1104, 2009.

[14] D. Hau and K. Chen, "Exploring hierarchical speech representations using a deep convolutional neural network," *Proc. of 11th UK Workshop Comput. Intell. (UKCI '11)*, 2011.

[15] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoust., Speech, Signal Process*., 38(3), pp. 328–339, 1989.

[16] K. Ohta and M. Yanagida, "Single channel blind dereverberation based on auto-correlation functions of frame-wise time sequences of frequency components," *Proc. of IWAENC*, pp. 1–4, 2006.

[17] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, E. Habets, R. Haeb-Umbach, V. Leutnant, A. Sehr, W. Kellermann, R. Maas, S. Gannot, and B. Raj, "The REVERB Challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," *Proc. of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2013.

[18] G. Hirsch, "Experimental framework for the performance evaluation of speech recognition front-ends on a large vocabulary task," *ETSI STQ-Aurora DSR Working Group*, June 4, 2001.

[19] T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals, “WSJCAM0: A British English speech corpus for large vocabulary continuous speech recognition,” *Proc. ICASSP*, pp. 81–84, 1995.

[20] M. Lincoln, I. McCowan, J. Vepa, and H.K. Maganti, “The multi-channel Wall Street Journal audio visual corpus (MC-WSJ-AV): Specification and initial experiments,” *Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding*, 2005.

[21] V. Mitra, H. Franco, and M. Graciarena, “Damped oscillator cepstral coefficients for robust speech recognition,” *Proc. of Interspeech*, pp. 886–890, 2013.

[22] V. Mitra, H. Franco, M. Graciarena, and A. Mandal, “Normalized amplitude modulation features for large vocabulary noise-robust speech recognition,” *Proc. of ICASSP*, pp. 4117–4120, 2012.

[23] R. Drullman, J. M. Festen, and R. Plomp, “Effect of reducing slow temporal modulations on speech reception”, *J. Acoust. Soc. of Am*., vol. 95, no. 5, pp. 2670–2680, 1994.

[24] V. Mitra, H. Franco, M. Graciarena, and D. Vergyri, “Medium duration modulation cepstral feature for robust speech recognition,” *Proc. of ICASSP*, Florence, 2014.

[25] H. Teager, “Some observations on oral air flow during phonation,” in *IEEE Trans. ASSP*, pp. 599–601, 1980.

[26] V. Mitra, J. Van Hout, W. Wang, M. Graciarena, M. McLaren, H. Franco, D. Vergyri, "Improving Robustness against Reverberation for Large-Vocabulary Continuous Speech Recognition," submitted to ASRU 2015.

[27] M. Harper, "The Automatic Speech recognition In Reverberant Environments (ASpIRE) Challenge," Proc. of ASRU, 2015.

[28] K. Yao, D. Yu, F. Seide, H. Su, L. Deng, Y Gong, "Adaptation Of Context-Dependent Deep Neural Networks For Automatic Speech Recognition," Proc. of SLT 2012.