INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Factored deep convolutional neural networks for noise robust speech recognition

Masakiyo Fujimoto
National Institute of Information and Communications Technology, Japan
masakiyo.fujimoto@nict.go.jp

Abstract

In this paper, we present a framework of a factored deep convolutional neural network (CNN) learning for noise robust automatic speech recognition (ASR). Deep CNN architecture, which has attracted great attention in various research areas, has also been successfully applied to ASR. However, to ensure noise robustness, since merely introducing deep CNN architecture into the acoustic modeling of ASR is insufficient, we introduce factored network architecture into deep CNN-based acoustic modeling. The proposed factored deep CNN framework factors out feature enhancement, delta parameter learning, and hidden Markov model state classification into three specific network blocks. By assigning specific roles to each block, the noise robustness of deep CNN-based acoustic models can be improved. With various comparative evaluations, we reveal that the proposed method successfully improves ASR accuracies in noise environments.

Index Terms: noise robust speech recognition, deep convolutional neural network, factored network, multi-channel input

1. Introduction

With the extensive use of speech applications on mobile devices, ensuring the noise robustness of automatic speech recognition (ASR) in daily environments is becoming more crucial. The simplest way to ensure noise robustness is speech enhancement during the front-end processing of ASR; however, in this approach, speech distortion sometimes degrades the ASR performance. This performance degradation is noteworthy in recent deep neural network (DNN)-based ASR frameworks. In particular, single-channel processing that includes traditional techniques [1, 2, 3], a denoising autoencoder (DAE) [4], and DNN-based ideal binary masking [5] often seriously degrade ASR performance, because most of these techniques have no distortionless framework. By contrast, as representative multi-channel processing, minimum variance distortionless response (MVDR) beamforming, which is designed with a distortionless framework, provides a positive improvement of ASR [6].

Instead of speech enhancement during front-end processing, various neural network-based acoustic modeling schemes have been proposed to ensure the noise robustness of ASR, for example, noise adaptive training (NAT) [7], noise-aware training [8], and DNN adaptation [9, 10]. Frameworks of precise acoustic modeling with various network architectures have also been proposed, including a convolutional neural network (CNN) [11, 12], a network in network (NIN) architecture [13, 14], and a long-short term memory (LSTM) recurrent neural network [15, 16].

In the recent progress of deep learning research, CNN and LSTM have become highly important tools for various pattern classification and regression tasks. In particular, deep CNN architecture has attracted great attention in various research areas [17, 18, 19, 20, 21, 22, 23, 24]. This architecture has also been introduced into ASR [25, 26]. On the other hand, LSTM has been applied to end-to-end mapping, e.g., connectionist temporal classification (CTC) [27]. A convolutional LSTM [28] has also been proposed to solve a spatiotemporal sequence forecasting problem.

Although the above deep CNN architecture is a powerful tool, we must address some considerations before introducing it into a noise robust ASR. With a noise robust ASR, the input speech is obviously corrupted by the interference noises. In this situation, merely learning deep CNN-based acoustic models with corrupted speech is insufficient, because its crucial discriminative characteristics might be seriously deteriorated. Therefore, additional network architectures are required to ensure noise robustness. As an answer to this problem, a factored network, which factors out some function blocks into separate layers in the network, has been proposed [29]. With it, we believe that the noise robustness of deep CNN-based acoustic models can be improved by building a network with specific roles for each layer.

Based on the above successful works, we propose a framework of factored deep CNN learning for noise robust ASR. In our proposed framework, we factor feature enhancement, delta parameter learning, and hidden Markov model (HMM) state classification into three network blocks. Here, the HMM state classification block is equivalent to the standard acoustic model used for ASR. In each network block, we build networks of multi-channel time-frequency filtering and time-domain filtering for the blocks of multi-channel feature enhancement and dynamic (delta and acceleration) feature parameter learning. As an HMM state classification block, we employed the architectures of a deep CNN-NIN [14] and a residual net (ResNet) [21] and evaluated our proposed method on the CHiME3 task [30]. The evaluation results reveal that the proposed method successfully improved ASR accuracies in noise environments.

2. Related work

In the research area of image recognition, various deep CNN architectures have been proposed, and excellent performances have been revealed. Alex net [17] and VGG net [18] stack 2-dimensional convolutional layers by changing their filter sizes depending on the network's depth. Inception net [19] stacks inception units. An inception unit consists of branch networks that have individual network architectures. The outputs of each branch are concatenated to one feature map. Currently, various inception network architectures have been reported in [20]. ResNet [21] introduces shortcut connection architecture to avoid the vanishing/exploding gradient problem. Therefore, this architecture makes it possible to stack more than 100 convolutional layers. The sequel work [22] discussed various ResNet unit architectures. On the other hand, super resolution CNN (SRCNN) [23] and deep convolutional generative adversarial network (DCGAN) [24] have also attracted attention in regres-

http://dx.doi.org/10.21437/Interspeech.2017-225

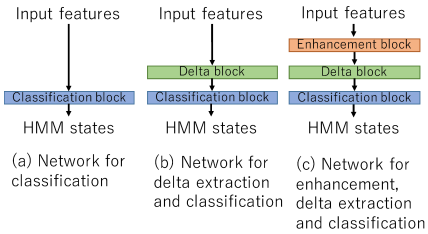

Figure 1: *Architecture of factored networks*

sion tasks of image processing.

Deep CNN also introduced a research area of ASR. Another work [14], which won the CHiME3 challenge, employed deep CNN-NIN architecture for acoustic modeling. ResNet was adopted for an ASR system of [25] that reported excellent performance. Another noteworthy work [26] proposed a novel end-to-end ASR system using residual convolutional LSTM.

As research works of factored networks, another work [29] factored out spatial and spectral filtering into separate layers and improved ASR accuracy by explicitly separating beamforming (spatial filtering) and frequency decomposition (spectral filtering) in a multi-channel raw waveform input system. Dynamic feature learning was proposed in [31], where dynamic feature parameters, i.e., delta and acceleration parameters, were extracted from static feature parameters by stacking time-domain filtering layers. A joint training approach, which concatenates the networks of front-end processing (speech/feature enhancement) and ASR, resembles a factored network. In this approach, each network is individually trained based on each optimization criterion in advance. After network concatenation, the parameters of the concatenated network are jointly optimized. In many cases, the joint training approach employs DAE [32, 33, 34], binary mask estimation [7, 35] or beamformer [36] for a network of front-end processing.

Inspired by these successful works, in this paper, we explore the factored deep CNN architectures.

3. Factored deep CNN architectures

In this section, we present our proposed factored network architecture. As shown in Fig. 1, we factored out feature enhancement, delta parameter learning, and HMM state classification into three specific network blocks. The classification block learns the HMM state classifier with input feature parameters and targets the HMM state labels. This block is equivalent to standard DNN-HMM-based acoustic modeling. The delta and enhancement blocks learn dynamic feature extraction with time-domain filtering and multi-channel feature enhancement with 2-dimensional time-frequency filtering, respectively.

3.1. Input feature parameters

As a common setup for acoustic modeling, the input feature parameters are the utterance-wise mean and variance normalized 40 log mel-filter bank (FBank) features, which are extracted using a Hamming window with a 25-ms frame length and a 10-ms frame shift. A context window with $\pm C_w$ frames is also applied to each utterance.

3.2. Classification block

The classification block is trained to output the posterior probabilities (softmax outputs) of 1,967 context dependent (CD) HMM states with frame-wise cross-entropy criterion.

As previously mentioned in [14], the architectures of a deep CNN and a NIN indicate a noticeable improvement of the noise levels in ASR. Therefore, we introduce this architecture into the classification block. For CNN-NIN training, we form $40 \times (2C_w + 1) \times C_{in}$ time-frequency feature maps for the CNN-NIN input, where C_{in} denotes the number of input feature maps. In the convolutional layer, zero padding is not applied to the edge of each feature map. After the convolutional layers, we added four fully connected (FC) layers. We use a rectified linear unit (ReLU) activation function [37] for each hidden layer. In addition, to avoid overfitting, we use batch normalization [38] and dropout for each hidden layer. Here dropout is only applied to the FC layers. The architecture of the CNN-NIN-based classification block is detailed in Table 1.

Table 1: *Details of CNN-NIN-based classification block. In convolutional layer, "$\boldsymbol{f} \times \boldsymbol{t} \times \boldsymbol{ch}$" represents $\boldsymbol{ch}$ output feature maps with a filter of $\boldsymbol{f}$ frequency bands and $\boldsymbol{t}$ time frames.*

Input	Input size	
	$40 \times (2C_w + 1) \times C_{in}$	
Convolutional layers	Filter size	Stride size
	$5 \times (2C_w + 1) \times 180$	1×1
	$1 \times 1 \times 180$	1×1
	Max pooling: size 2×1, stride 2×1	
	Filter size	Stride size
	$5 \times 1 \times 180$	1×1
	$1 \times 1 \times 180$	1×1
	Max pooling: size 2×1, stride 2×1	
	Filter size	Stride size
	$5 \times 1 \times 180$	1×1
FC layers $\times 4$	Unit size: 2,048	
Output	Output size: 1,967	

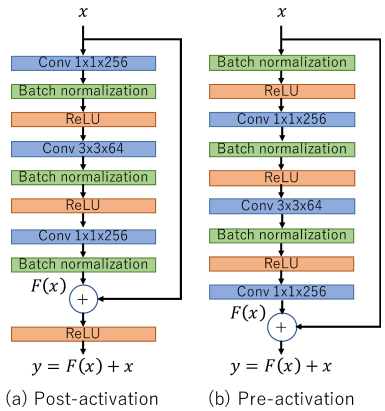

Figure 2: *Network architecture of bottleneck ResNet unit*

As a classification block, we also trained a very deep CNN with ResNet architecture. Fig. 2 illustrates typical ResNet units with a bottleneck architecture. In a ResNet unit, output feature map $\boldsymbol{y}$ is obtained by the shortcut connection of Fig. 2, which performs elemental-wise addition of input feature map $\boldsymbol{x}$ and a feature map obtained by forward propagation $F(\boldsymbol{x})$. The properties of the ResNet unit were previously detailed [21, 22]. In recent works on ResNet, various architectures of ResNet units have been proposed. In this work, we employed the pre-activation architecture [22] illustrated in Fig. 2(b) for the ResNet unit. Finally, a ResNet is built by stacking ResNet units. The architecture of the ResNet-based classification block used in this work is detailed in Table 2. In all the layers, zero padding is applied to the edge of each feature map.

3.3. Delta block

Dynamic feature learning with time-domain filtering, which we call a delta block, outperforms the deep CNN-NIN architecture [31]. The delta block's input is $40 \times (2(C_w + 2D_w) + 1) \times D_{in}$

Table 2: *Details of ResNet-based classification block*

Input	Input size	
	$40 \times (2C_w + 1) \times C_{in}$	
Convolutional layer	Filter size	Stride size
	$5 \times (2C_w + 1) \times 64$	1×1
ResNet units $\times 3$	Filter size	Stride size
	$1 \times 1 \times 256$	1×1
	$3 \times 3 \times 64$	1×1
	$1 \times 1 \times 256$	1×1
Max pooling	Size 2×2, stride 2×2	
ResNet units $\times 4$	Filter size	Stride size
	$1 \times 1 \times 512$	1×1
	$3 \times 3 \times 128$	1×1
	$1 \times 1 \times 512$	1×1
Max pooling	Size 2×2, stride 2×2	
ResNet units $\times 6$	Filter size	Stride size
	$1 \times 1 \times 1,024$	1×1
	$3 \times 3 \times 256$	1×1
	$1 \times 1 \times 1,024$	1×1
Max pooling	Size 2×2, stride 2×2	
ResNet units $\times 3$	Filter size	Stride size
	$1 \times 1 \times 2,048$	1×1
	$3 \times 3 \times 512$	1×1
	$1 \times 1 \times 2,048$	1×1
Global average pooling [13]		
Output	Output size: 1,967	

Table 3: *Details of delta block*

Input	Input size	
	$40 \times (2(C_w + 2D_w) + 1) \times D_{in}$	
Delta block	Filter size	Stride size
	$1 \times (2D_w + 1) \times D_{out}$	1×1
	$1 \times (2D_w + 1) \times D_{out}$	1×1

feature maps, which consists of only static FBanks. D_w and D_{in} denote the delta or the acceleration window length and the number of input feature maps. As indicated in Table 3, by stacking two time-domain convolutional layers, the output feature maps cover the same time-frequency span of the classification block as $40 \times (2C_w + 1)$, because zero padding is not applied to the edge of each feature map. In the table, D_{out} denotes the number of output feature maps.

3.4. Enhancement block

The enhancement block learns multi-channel feature enhancement with time-frequency filtering, i.e., a 2-dimensional CNN. In the proposed method, input feature parameters are given as M-channel FBank maps, and then 2-dimensional CNN-based time-frequency filtering is applied to each input channel. As shown in Fig. 3, we examined three types of filtering methods: (a) multi-channel filtering, (b) single-channel filtering, and (c) the integration of single- and multi-channel filtering.

First, as seen in Fig. 3(a), multi-channel filtering sums up the outputs of N convolution filters obtained by the whole channels. Although phase and spatial information are lost because FBank analysis is used, this method is expected to work in a way that resembles delay-and-sum beamformer.

Second, Fig. 3(b) illustrates individual single-channel filtering. Instead of Fig. 3(a), this method collects feature maps obtained from individual channels and forms output feature maps by concatenating whole filtered feature maps. Since this method propagates much information to succeeding networks, it resembles a type of data augmentation.

Finally, Fig. 3(c) integrates Figs. 3(a) and 3(b). This method realizes delay-and-sum beamformer like feature enhancement with many input feature maps, which are collected by the method in Fig. 3(b). We expect to obtain further improvement due to the synergy of these two methods.

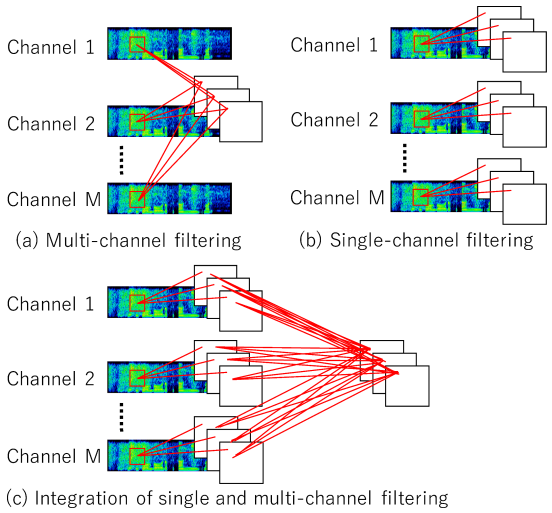

Figure 3: *Network architectures of enhancement block*

Table 4: *Details of each enhancement block*

Input	Input size	
	$40 \times (2(C_w + 2D_w) + 1) \times E_{in}$	
Enhancement block	Filter size	Stride size
(a) Multi-channel	$3 \times 3 \times N$	1×1
(b) Single-channel	$3 \times 3 \times (M \times N)$	1×1
(c) Integration of	$3 \times 3 \times (M \times N)$	1×1
(a) and (b)	$3 \times 3 \times N$	1×1

The architecture of each enhancement block is detailed in Table 4. The enhancement block's input is $40 \times (2(C_w + 2D_w)+1) \times E_{in}$ feature maps and only consists of static FBank features. E_{in} denotes the number of input feature maps, which equals M. For enhancement blocks (a), (b), and (c), the number of output channels E_{out} equals N, $M \times N$, and N, respectively. In all of the layers, zero padding is applied to the edge of each feature map.

4. Experiments

4.1. CHiME-3 corpus

We conducted ASR evaluations using the CHiME-3 task [30]. The CHiME-3 corpus consists of actual six-channel audio data collected in four different noise environments and additional simulated six-channel data. A tablet device equipped with six microphones was used for audio recording to simulate a situation where a user is talking on it in daily environments. The following are the noise environments of the CHiME-3 corpus: a form of public transportation (BUS), a cafeteria (CAF), a pedestrian area (PED), and a street junction (STR). The corpus includes only read speech, where the sentences to be read were taken from the WSJ0 corpus [39]. The training set consists of 1,600 real and 7,138 simulated (simu) utterances, which are equivalent to 18 hours of speech. With each bit of training data, the numbers of speakers were four and 83. The development (dev) and evaluation (eval) sets consist of 3,280 and 2,640 utterances, respectively, each containing equal quantities of real and simulated data. Both the real and simulated sets were spoken by four speakers. We used no audio data recorded by a second microphone, because it was located behind the tablet device.

4.2. Experimental setup

All the neural networks were trained using TensorFlow [40], and the evaluations (ASR decoding) with trained neural networks were conducted using the Kaldi toolkit [41]. We eval-

Table 5: *Numbers of feature maps for each block*

Network type	#input/output feature maps				
	E_{in}	E_{out}	D_{in}	D_{out}	C_{in}
C	—	—	—	—	3
D+C	—	—	1	16	16
E(a)+D+C	M	N	N	16	16
E(b)+D+C	M	$M \times N$	$M \times N$	16	16
E(c)+D+C	M	N	N	16	16
BFIT+D+C	—	—	1	16	16

uated the combination of each function block, i.e., the classification block (C), the delta block (D), and the enhancement blocks (E(a), E(b), and E(c)). We also compared our proposed method with the front-end speech enhancement method using "Beamformit" (BFIT) [42], which is used for the beamforming processing of the latest Kaldi CHiME3 recipe [43].

The target CD HMM state labels of the training and development sets were obtained using the latest Kaldi CHiME3 recipe. The parameters of each network were randomly initialized and optimized with a momentum stochastic gradient descent with a mini batch of 128 frames and an initial learning rate of 0.1. The keep probability of the dropout was set at 0.5. The context window length of the CNN-NIN-based classification block was set at $C_w = 5$. On the other hand, the ResNet-based classification block was set at $C_w = 8$. The delta or acceleration window length was set at $D_w = 2$.

Table 5 indicates the number of input/output feature maps for each factored block. The ASR evaluation with networks "C" or "D+C" was equivalent to a single-channel task. Thus, we used audio data recorded by a fifth microphone, which was the closest microphone to the speaker. The input feature map of "C" consisted of static FBanks and their delta and acceleration parameters, and the number of input feature maps was set at $C_{in} = 3$. For enhancement blocks "E(a)," "E(b)," and "E(c)," the numbers of input channels and filters were set at $M = 5$ and $N = 24$, respectively.

Language modeling also followed the latest Kaldi CHiME3 recipe. The ASR experiments were performed using fully composed trigram weighted finite state transducers with the above factored network-based acoustic models. The evaluation criterion was the word error rate (WER).

4.3. Experimental results

Table 6 shows the ASR results of the CNN-NIN-based classification block. The WERs obtained with "D+C" slightly outperformed those of "C." The improvements made by the delta block were small. However, it has an advantage: the input feature parameters became simple and easy to handle, because we can ignore the unstable dynamic feature parameters.

The results with enhancement blocks "E(a)+D+C," "E(b)+D+C," and "E(c)+D+C" showed significant more improvement than "C" and "D+C." In addition, "E(b)+D+C" and "E(c)+D+C" outperformed "BFIT+D+C" in average WERs of eval set. These results confirmed that the proposed time-frequency filtering architecture works effectively as a feature enhancement block. In particular, the results with "E(c)+D+C" confirmed our hypothesis mentioned in Section 3.4: the synergy of "E(a)" and "E(b)."

On the other hand, Table 7 shows the ASR results of the ResNet-based classification block. As seen in the table, we could obtain further improvement from results of the CNN-NIN-based classification block, and confirm that constant improvements are obtained by function blocks "D" and "E(a), (b), (c)". Here, in the learning of the ResNet-based classification block, the influence of overfitting was remarkably observed.

Table 6: *ASR results of CNN-NIN-based classification block in WER (%)*

Network type	dev			eval		
	simu	real	avg.	simu	real	avg.
C	11.82	12.60	12.21	13.76	20.57	17.17
D+C	11.85	12.26	12.06	13.55	20.60	17.08
E(a)+D+C	9.09	9.92	9.50	11.53	18.97	15.25
E(b)+D+C	9.12	10.09	9.60	11.48	17.96	14.72
E(c)+D+C	**8.88**	9.85	**9.37**	**11.31**	17.94	**14.63**
BFIT+D+C	10.63	**8.89**	9.76	15.26	**14.50**	14.88

Table 7: *ASR results of ResNet-based classification block in WER (%)*

Network type	dev			eval		
	simu	real	avg.	simu	real	avg.
C	11.04	10.86	10.95	11.94	19.02	15.48
D+C	10.97	11.10	11.03	11.98	18.45	15.20
E(a)+D+C	8.97	9.20	9.09	**10.89**	16.04	**13.47**
E(b)+D+C	**8.88**	9.06	**8.97**	11.01	16.02	13.52
E(c)+D+C	9.03	9.05	9.04	11.07	16.37	13.72
BFIT+D+C	10.14	**8.50**	9.32	15.15	**14.65**	14.90

Table 8: *ASR results of CNN-NIN-based classification block with noisy and Beamformit inputs in WER (%)*

Network type	dev			eval		
	simu	real	avg.	simu	real	avg.
BFIT+E(a)+D+C	8.56	8.21	8.39	12.64	14.38	13.51
BFIT+E(b)+D+C	9.07	**8.07**	8.57	13.01	**14.08**	13.55
BFIT+E(c)+D+C	**8.54**	8.20	**8.37**	**12.21**	14.44	**13.33**

Table 9: *ASR results of ResNet-based classification block with noisy and Beamformit inputs in WER (%)*

Network type	dev			eval		
	simu	real	avg.	simu	real	avg.
BFIT+E(a)+D+C	**8.62**	8.22	8.42	12.44	14.27	13.34
BFIT+E(b)+D+C	8.91	7.79	8.35	12.55	13.56	13.06
BFIT+E(c)+D+C	8.63	**7.68**	**8.15**	**11.78**	**13.33**	**12.56**

Therefore, to avoid overfitting, we plan to learn the ResNet-based classification block by reconsidering network configurations and parameter setup.

Finally, we carried out evaluations by combining the proposed methods and "BFIT," where we concatenated the FBanks of "BFIT" with those of the five-channel noisy speech as new input feature parameters. In that case, the number of input channels becomes $M = 6$. Tables 8 and 9 show the results of the proposed factored networks with new input feature parameters. The combination of the proposed methods and "BFIT" yielded further improvements. These results suggest that in the proposed enhancement block, feature map concatenation, which has different characteristics, works efficiently and improves ASR performance.

5. Conclusions

We proposed a framework of factored deep CNN learning for a noise robust ASR. In our framework, we factored out feature enhancement, delta parameter learning, and HMM state classification into three network blocks. We conducted various comparative evaluations, which compared each factored architecture, the deep CNN architectures, and the concatenation of the input feature maps. With these evaluations, we revealed that our proposed framework positively improved the ASR performance by carefully choosing network architecture and feature map combinations. In future research, we will introduce into our proposed framework various training techniques, for example, inception network and joint training.

6. References

[1] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Trans. on ASSP*, vol. 27, no. 2, pp. 113–120, April 1979.

[2] Y. Ephraim and D. Malah, "Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator," *IEEE Trans. on ASSP*, vol. 32, pp. 1109–1121, December 1984.

[3] P. J. Moreno, B. Raj, and R. M. Stern, "A vector Taylor series approach for environment-independent speech recognition," in *Proc. of ICASSP '96*, vol. II, May 1996, pp. 733–736.

[4] M. Fujimoto and T. Nakatani, "Feature enhancement based on generative-discriminative hybrid approach with GMMs and DNNs for noise robust speech recognition," in *Proc. of ICASSP '15*, April 2015, pp. 5019–5023.

[5] B. Li and K. C. Sim, "Improving robustness of deep neural networks via spectral masking for automatic speech recognition," in *Proc. of ASRU '13*, December 2013, pp. 279–284.

[6] T. Higuchi, N. Ito, T. Yoshioka, and T. Nakatani, "Robust MVDR beamforming using time-frequency masks for online/offline ASR in noise," in *Proc. of ICASSP '16*, March 2015, pp. 5210–5214.

[7] A. Narayanan and D. Wang, "Joint noise adaptive training for robust automatic speech recognition," in *Proc. of ICASSP '14*, May 2014, pp. 2523–2527.

[8] M. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proc. of ICASSP '13*, May 2013, pp. 7398–7402.

[9] H. Liao, "Speaker adaptation of context dependent deep neural networks," in *Proc. of ICASSP '13*, May 2013, pp. 7947–7951.

[10] J. Li, J. T. Huang, and Y. Gong, "Factorized adaptation for deep neural network," in *Proc. of ICASSP '14*, May 2014, pp. 5537–5541.

[11] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE Trans. on ASLP*, vol. 22, no. 10, pp. 1533–1545, October 2014.

[12] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," *Neural Networks*, vol. 64, pp. 39–48, 2015.

[13] M. Lin, Q. Chen, and S. Yan, "Network in network," arXiv:1312.4400v3, 2014.

[14] T. Yoshioka, N. Ito, M. Delcroix, A. Ogawa, K. Kinoshita, M. Fujimoto, C. Yu, W. J. Fabian, M. Espi, T. Higuchi, S. Araki, and T. Nakatani, "The NTT CHiME-3 system: Advances in speech enhancement and recognition for mobile multi-microphone devices," in *Proc. of ASRU '15*, December 2015, pp. 436–443.

[15] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. of Interspeech '14*, September 2014.

[16] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. of ICASSP '15*, April 2015, pp. 4580–4584.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. of NIPS '12*, December 2012, pp. 1097–1105.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv:1409.1556, 2014.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. of CVPR '15*, June 2015.

[20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," arXiv:1512.00567, 2015.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of CVPR '16*, June 2016, pp. 770–778.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. of ECCV '16*, October 2016, pp. 630–645.

[23] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. on PAMI*, vol. 38, no. 2, pp. 295–307, February 2015.

[24] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," arXiv:1511.06434, 2015.

[25] W. Xiong, J. Droppo, X. Huang, F. Seide, M. L. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "The Microsoft 2016 conversational speech recognition system," in *Proc. of ICASSP '17*, March 2017, pp. 5255–5259.

[26] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Proc. of ICASSP '17*, March 2017, pp. 4845–4849.

[27] A. Graves, S. Fernández, and F. Gomez, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. of ICML '06*, June 2006, pp. 369–376.

[28] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. of NIPS '15*, December 2015, pp. 802–810.

[29] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, and M. Bacchiani, "Factored spatial and spectral multichannel raw waveform CLDNNs," in *Proc. of ICASSP '16*, March 2016, pp. 5075–5079.

[30] "The 3rd CHiME speech separation and recognition challenge," http://spandh.dcs.shef.ac.uk/chime_challenge/.

[31] T. Yoshioka, K. Ohnishi, F. Fang, and T. Nakatani, "Noise robust speech recognition using recent developments in neural networks for computer vision," in *Proc. of ICASSP '16*, March 2016, pp. 5730–5734.

[32] M. Mimura, S. Sakai, and T. Kawahara, "Joint optimization of denoising autoencoder and DNN acoustic model based on multi-target learning for noisy speech recognition," in *Proc. of Interspeech '16*, September 2016, pp. 3803–3807.

[33] K. H. Lee, T. G. Kang, W. H. Kang, and N. S. Kim, "DNN-based feature enhancement using joint training framework for robust multichannel speech recognition," in *Proc. of Interspeech '16*, September 2016, pp. 3027–3031.

[34] T. Fukuda, O. Ichikawa, G. Kurata, R. Tachibana, S. Thomas, and B. Ramabhadran, "Effective joint training of denoising feature space transforms and neural network based acoustic models," in *Proc. of ICASSP '17*, March 2017, pp. 5190–5194.

[35] T. Higuchi, T. Yoshioka, and T. Nakatani, "Optimization of speech enhancement front-end with speech recognition-level criterion," in *Proc. of Interspeech '16*, September 2016, pp. 3808–3812.

[36] Z. Zhong Meng, S. Watanabe, J. R. Hershey, and H. Erdogan, "Deep long short-term memory adaptive beamforming networks for multichannel robust speech recognition," in *Proc. of ICASSP '17*, March 2017, pp. 271–275.

[37] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. of AISTATS '11*, April 2011, pp. 315–323.

[38] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of ICML '15*, July 2015, pp. 448–456.

[39] J. Garofolo, D. Graff, D. Paul, and D. Pallett, "CSR-I (WSJ0) complete," https://catalog.ldc.upenn.edu/LDC93S6A.

[40] "TensorFlow," https://www.tensorflow.org/.

[41] "Kaldi ASR tool-kit," http://kaldi-asr.org/.

[42] X. Anguera, C. Wooters, and J. Hernando, "Acoustic beamforming for speaker diarization of meetings," *IEEE Trans. on ASLP*, vol. 15, no. 7, pp. 2011–2023, September 2007.

[43] "Kaldi CHiME3 recipe with beamforming," https://github.com/kaldi-asr/kaldi/tree/master/egs/chime3.