NOISE ROBUST SPEECH RECOGNITION ON AURORA4 BY HUMANS AND MACHINES

Yanmin Qian† *Tian Tan* *Hu Hu* *Qi Liu*

Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
{yanminqian@tencent.com}

ABSTRACT

Although great progress has been made in automatic speech recognition (ASR), significant performance degradation still exists in noisy environments. Based on our previous introduced very deep CNNs, this paper further integrates residual learning to evaluate very deep convolutional residual network (VDCRN) in noisy conditions, which shows more powerful robustness. Then, cluster adaptive training (CAT) is developed on the VDCRN to reduce the mismatch between the training and testing in noisy scenarios. Moreover, the advanced future-vector assisted LSTM-RNN LM is proposed to achieve a further gain. All the proposed approaches are evaluated on Aurora4 and show a significant improvement for each technology. The final system achieves 3.09% WER on Aurora4, which is approaching humans' performance on this task. This is a new milestone for noise-robust ASR on this benchmark.

Index Terms— robust speech recognition, very deep convolution residual network, cluster adaptive training, future-vector

1. INTRODUCTION

In recent years, significant progress has been observed in automatic speech recognition (ASR) due to the introduction of deep neural network based acoustic model [1, 2]. However, these systems still perform poorly in noisy environments [3], and previous research has revealed that acoustic mismatching between training and testing still leads to a great performance degradation even with the deep learning technologies [4, 5]. Thus noise robustness is still a key issue in allowing ASR systems to have a wider range of use cases.

Many technologies [6, 7, 8] have been proposed to handle the difficult problem of mismatch between training and testing in the noisy speech recognition scenario. Those methods can be grouped into two categories: feature enhancement on the front-end (denoising or dereverberation) [9, 10] and acoustic modeling with adaptation on the back-end [3]. In this paper, we focus on the technologies on the back-end, which can improve the robustness on ASR.

The convolutional neural network has shown lower word error rate (WER) than standard fully connected feed-forward DNN in speech recognition either with the shallow structure [11] or with the very deep structure [12, 13]. More recently in [14, 15], our previous works have shown that very deep CNN (VDCNN) particularly shows superior noise robustness than other models under the noisy scenarios. In this work, we integrate batch normalization [16] and residual learning [17] into our previous VDCNN to construct very deep convolutional residual network (VDCRN). We discover that VDCRN can further improve the model robustness and gain better performance. Then, adaptation techniques are developed based on VDCRN. Inspired by the success of our previous work on cluster adaptive training (CAT) for DNN [18], the similar idea is further extended to filters or feature maps in VDCRN, which utilizes the filter or feature map as the basis in adaptive training. CAT for VDCRN can reduce the mismatch between training and testing, and improve WER dramatically. Moreover, an advanced LSTM-RNN LM with the assistance from future-vector is proposed and implemented to further improve the system performance under noisy conditions. A comprehensive investigation and an in-depth analysis of all these technologies are performed. Experimental results on the noisy Aurora4 task show that very promising performance can be achieved with the proposed methods, even without using front-end denoising. Finally error analysis and performance comparison between humans and machines are performed on this noisy speech recognition task.

†Yanmin Qian is the corresponding author and now he is with Tencent AI Lab, Tencent, Bellevue, WA, USA.

2. ADAPTIVE VERY DEEP CONV-RESIDUAL NETWORK

2.1. Very deep convolution residual network

The main difference between CNN and DNN is the convolutional operation, which can extract and model the local information in a more effective mode, and we use $\otimes$ to represent it. A convolution layer consists of $\#outchannel \times \#inchannel$ filters. The i-th output feature map of layer l is given by

$$\mathbf{o}_i^l = \sigma(\sum_{j=1}^{N} W_{i,j}^l \otimes \mathbf{o}_j^{l-1} \oplus b_i^l) \quad (1)$$

where $\mathbf{o}_i^l$ and $\mathbf{o}_j^{l-1}$ are feature maps in the current layer l and previous layer $l-1$ respectively. $W_{i,j}^l$ is the filter between input feature map j and output feature map i at layer l. b_i^l is a bias applied to the whole feature map, and $\oplus$ indicates each element in the feature map plus the same scalar b_i^l. σ is the activation function, which is typically sigmoid or ReLU. N is the number of output feature maps. A pooling layer is a layer that performs down-sampling on the feature maps of the previous layer. In this work, max-pooling is used.

More recently VDCNNs, which have many convolutional layers, have been successfully used in ASR [12, 13, 19], and our previous work has shown that VDCNN can particularly get a huge improvement in noisy scenarios compared to DNN, CNN and even RNN [14, 15]. The structure of VDCNN is shown in the right part of Fig. 1 (a). It contains 5 blocks separated by the pooling operation, and each block contains two convolution layers and one pooling layer. The model configuration, such as $[3 \times 3, 64]$ indicates that the layer uses a 3×3 filter and the output contains 64 feature maps.

To better train the model with increased depth, batch normalization and residual learning are further incorporated to form a new model in this work, which is named very deep convolution residual network (VDCRN). It is mainly motivated by the great success of ResNet in image recognition [17] and telephone speech recognition

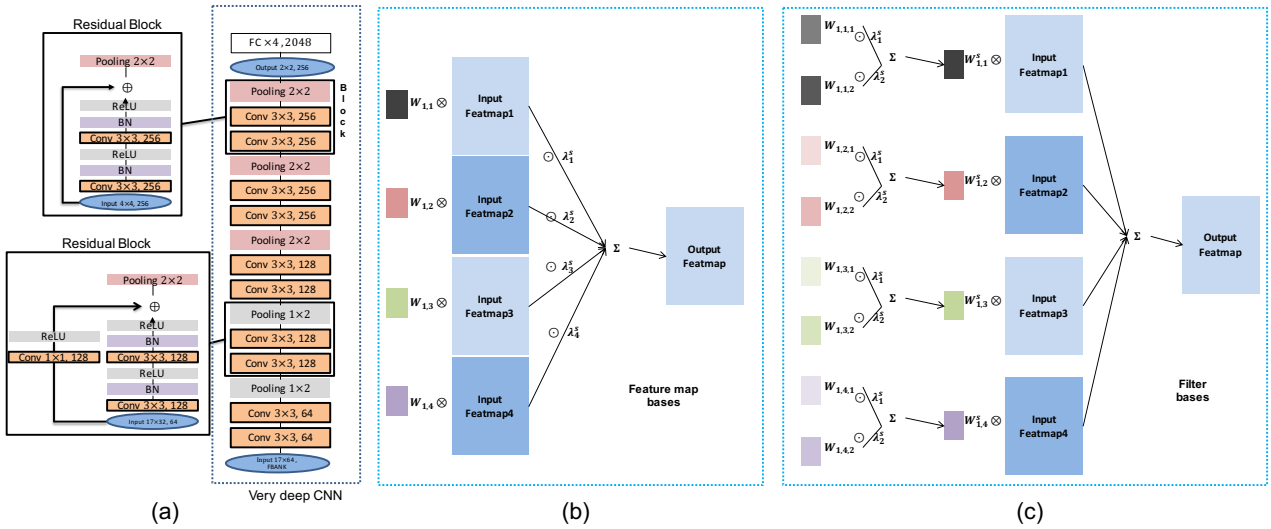

Fig. 1: (a) The structure of proposed VDCRN. (b) CAT-VDCRN with feature map bases (c) CAT-VDCRN with filter bases

[20, 21], and we want to see how it will perform in the noisy conditions. The designed very deep convolutional residual network is shown as Fig. 1 (a). Residual block is used to replace each block in previous VDCNN (two convolutional layer with one pooling layer in VDCNNs). Two kinds of res-block are designed, shown as the left part of Fig. 1 (a). Since the number of feature maps will double at the first, second and fourth block, a convolution layer with 1×1 filter is applied in the skip connection and the feature maps are increased to an equal number, which is shown as the bottom left in Fig. 1 (a). For the blocks with the same feature maps number as the previous block, e.g. the third and fifth block, the direct skip connection can be used in the res-block, which is shown in the top left in Fig. 1 (a).

2.2. Cluster adaptive training for VDCRN

Adaptive training is an effective method to reduce the mismatch between training and testing conditions, and cluster adaptive training (CAT) are designed for proposed VDCRN in this work. CAT was firstly proposed on GMM-HMM [22], then extended to DNN-HMM in our previous work [18, 23]. The basic idea is that multiple bases are first trained to construct the canonical parametric space, and then during adaptation an interpolation vector is estimated to combine the bases into the final adapted model.

The bases selection is the most important in CAT, which forms the canonical parametric space. Two kinds of bases can be used when applying CAT in the proposed VDCRN.

2.2.1. *Feature map bases*

At each convolution layer, an output feature map is generated by summing up over all input feature maps convoluted by its own filter as shown in Equation 1. The first method utilizes each input feature map as a basis and interpolate them with a speaker dependent interpolation vector. The output feature map is given by

$$\mathbf{o}_i^{sl} = \sigma(\sum_{j=1}^{N} \lambda_j^{sl}(W_{i,j}^l \otimes \mathbf{o}_j^{l-1}) \oplus b_i^l) \tag{2}$$

where $\mathbf{o}_i^{sl}, \mathbf{o}_j^{l-1}$ are i and j feature map in two consecutive layers. λ_j^{sl} is a scalar coefficient for the cluster j at layer l for speaker s. It is worth noting that in this case, the number of clusters is equal to the number of input channels (feature maps).

λ_j^{sl} can be extended to a matrix, which will be applied on the feature maps with the element-wise multiplication. The new speaker adapted output feature map is given by

$$\mathbf{o}_i^{sl} = \sigma(\sum_{j=1}^{N} \Lambda_j^{sl} \odot (W_{i,j}^l \otimes \mathbf{o}_j^{l-1}) \oplus b_i^l) \tag{3}$$

where Λ_j^{sl} is a matrix, and $\odot$ indicates element-wise multiplication. The structure is illustrated in the middle part of Fig. 1 (b), enclosed in the blue dotted line. There are four input feature maps and one output feature map, $W_{1,j}, 1 \leq j \leq 4$ is the filter and λ_j^s is the speaker dependent coefficient, it can be a scalar or matrix.

2.2.2. *Filter bases*

Another way is to use a filter basis rather than a single filter for each input/output feature map pair. The speaker adapted output feature map is given by

$$W_{i,j}^{sl} = \sum_{k=1}^{P} \lambda_k^{sl} W_{i,j,k}^l \tag{4}$$

$$\mathbf{o}_i^{sl} = \sigma(\sum_{j=1}^{N} W_{i,j}^{sl} \otimes \mathbf{o}_j^{l-1} \oplus b_i^l) \tag{5}$$

where $W_{i,j}^{sl}$ is a speaker dependent filter for layer l given by interpolating the filter bases using a speaker specific vector λ^{sl}, and $W_{i,j,k}^l$ is the kth element of filter bases. P is the number of filter bases.

In additional, the Equation 5 can be rewritten as Equation 6, and CAT can also be interpreted as splitting a conv-layer into P sublayers to represent different speaker or environment characteristics.

$$\mathbf{o}_i^{sl} = \sigma((\sum_{k=1}^{P} \lambda_k^{sl} \sum_{j=1}^{N} W_{i,j,k}^l \otimes \mathbf{o}_j^{l-1}) \oplus b_i^l) \tag{6}$$

Similarly, each interpolation scalar coefficient λ_k^{sl} can also be extended to a matrix, which will be applied on the feature maps using element-wise multiplication, and the formulation for adapted output feature map becomes

$$\mathbf{o}_i^{sl} = \sigma((\sum_{k=1}^{P} \mathbf{\Lambda}_k^{sl} \odot \sum_{j=1}^{N} W_{i,j,k}^l \otimes \mathbf{o}_j^{l-1}) \oplus b_i^l) \tag{7}$$

where $\odot$ denotes element-wise multiplication.

CAT with filter bases is illustrated in the right part of Fig. 1 (c), enclosed in the blue dotted line. $W_{1,j,1}, W_{1,j,2}, 1 \leq j \leq 4$ are the filter bases, and in this work we used two clusters for filter bases. λ_j^s is the speaker dependent coefficient, and it can be a scale or matrix. $W_{1,j}^s$ is a speaker adapted filter.

3. ADVANCED LANGUAGE MODELING

We introduced a new LSTM-RNN LM in this task, which is enhanced with the future vector, and it is named feature-vector LSTM-RNN LM (FV-LSTM RNN LM).

Traditional LSTM-RNN LM only predicts a single word x_{i+1} when given the history $\{x_1, ..., x_i\}$ [24]. However, LVCSR is a sequence-level task, so it should benefit from the sequence level knowledge if we can integrate the sequence information into LM. Future vector is one kind of sequence vectors [25, 26]. Thus we are inspired to incorporate the future vector into the normal LSTM-RNN LM, so that the new LM is able to predict long term future rather than only immediate word, which can finally contribute the system performance.

Three steps are designed to construct our new FV-LSTM RNN LM: (1) ground truth future vector: a reversed LSTM-RNN LM with reversed input sequence order is first trained, which means that this LM predicts the previous word with the given future. The outputs of the last hidden layer in this reversed LSTM-RNN LM is extracted as the ground truth future vector z_i. (2) predicted future vector: the above feature-vector z_i cannot be used directly due to that the future words in the real implementation are unknown. Accordingly an additional forward LSTM-RNN is trained to predict the future vector, and the mean squared error (MSE) criterion between the predicted future vector y_i and the ground truth future vector z_i is used to optimize this prediction model. (3) FV-LSTM RNN LM: after the prediction model training, predicted future vector y_i can be combined with original input x_i to train the FV-LSTM RNN LM.

This FV-LSTM RNN LM has been demonstrated to achieve better performance than normal LSTM-RNN LM in our recent work [27]. Moreover, it reveals that there is a huge complementarity between this new and conventional LSTM-RNN LMs. More details can be found in our recent work [27], and it is evaluated on the noisy ASR task in this work.

4. EXPERIMENTS

4.1. Experimental setup and baseline systems

The proposed approaches are implemented and compared on the standard Aurora 4 task, which has multiple additive noise conditions as well as channel mismatch. The Aurora 4 task is a medium vocabulary speech recognition task based on the Wall Street Journal (WSJ0) corpus [28]. It contains 16 kHz speech data in the presence of additive noises and linear convolutional channel distortions, which were introduced synthetically to clean speech from WSJ0. The multi-condition training set with 7138 utterances from 83 speakers includes a combination of clean speech and speech corrupted by one of six different noises at 10-20 dB SNR, and some data is from the primary Sennheiser microphone and some are from the secondary microphone. As for the training data, the test data is generated using the same types of noise and microphones, and these can be grouped into 4 subsets: clean, noisy, clean with channel distortion, and noisy with channel distortion, which will be referred to as A, B, C, and D, respectively.

Gaussian mixture model based hidden Markov models (GMM-HMMs) are first built with Kaldi [29] using the standard recipe, and consists of 3K clustered states trained using maximum likelihood estimation with the standard Kaldi MFCC-LDA-MLLT-FMLLR features. After the GMM-HMM training, a forced-alignment is performed to get the state level labels. All the neural networks were built using CNTK [30] in this work, and they were trained using cross-entropy criterion (CE) with stochastic gradient descent (SGD) based backpropagation (BP) algorithm. The task-standard WSJ0 bigram with 5K-word dictionary is used for decoding, and the standard testing pipelines in the Kaldi recipes are used for decoding and scoring. For better comparison, VDCNN model proposed in our previous work [14] is also built as the baseline, and listed as the first line of Table 1. Concluded as [14], VDCNN shows significantly better performance than DNN or LSTM-RNN, particularly for the noisy task (more details can be found in [14]).

4.2. Evaluation on adaptive very deep conv-residual network

4.2.1. Very deep convolutional residual network

VDCRN was evaluated first, and the results comparison on Aurora4 is shown as the second line in Table 1. It is observed that the application of res-block further significantly improves the performance in noisy scenarios, and the proposed VDCRN obtained another 8.0% relative improvement compared to the strong baseline VDCNN. Most gains are from the noisy subsets B, C and D, which further demonstrates the noise-robustness of VDCRN.

Systems	A	B	C	D	Avg.
VDCNN [14]	3.27	5.61	5.32	13.52	8.81
VDCRN	3.25	5.41	4.75	12.16	**8.10**

Table 1: WER (%) comparison of VDCNN / VDCRN on Aurora4.

4.2.2. CAT using different bases

Cluster adaptive training for VDCRN was then evaluated. As described in section 2.2, the bases can be defined in two modes, i.e. feature map bases and filter bases, and the speaker-dependent interpolation parameter λ^s can also be two ways, i.e. scalar or matrix. In this experiment, CAT was applied at the first convolution layer in the first block of VDCRN, and the related results are illustrated in Table 2. It is observed that using a scalar interpolation weight λ^s is useless, and in contrast the matrix based speaker-dependent interpolation parameter can get very large improvements within the CAT-VDCRN. Both bases types can give significant WER reduction with the appropriate interpolation weight λ^s, and the filter base seems slightly better. The system using filter bases with matrix interpolation weight λ^s reduces the WER from 8.10% to 6.01%, which is a relative 25.0% gain. We did the further development based on this system in the following experiments.

Systems	Base	λ^s	A	B	C	D	Avg.
VDCRN	-	-	3.25	5.41	4.75	12.16	8.10
+ CAT	fmap	scalar	3.40	5.80	5.68	12.14	8.34
		matrix	2.69	4.17	3.81	8.96	**6.09**
	filter	scalar	3.33	5.52	5.31	13.12	8.61
		matrix	2.65	4.06	3.38	8.95	**6.01**

Table 2: WER (%) Comparison of CAT with different bases.

4.2.3. CAT on different layers and residual blocks

The comparison of applying CAT on different convolutional layers of the first block is performed. In addition to only do the CAT on the first convolution layer, it is extended to both two layers in the first block. The top part of Table 3 shows that implementing CAT on both convolutional layers can get further small improvement compared to the single-layer CAT adaptation.

#Res-Block	#Layer	A	B	C	D	Avg.
B1	L1	2.65	4.06	3.38	8.95	6.01
	L1+L2	2.95	4.17	3.70	8.55	**5.92**
B2	L1+L2	2.95	4.31	3.70	9.06	6.21
B3	L1+L2	2.84	4.28	3.66	9.49	6.37

Table 3: WER (%) Comparison of applying CAT on different convolutional layers and residual blocks of VDCRN on Aurora4. L1 and L2 indicate the 1st and 2nd convolution layers in each block, and B1∼B3 indicate residual block 1∼block 3 in VDCRN.

As shown in the Fig. 1 (a), the VDCRN is separated by several residual blocks by the pooling operations. The CAT structure can be performed on different blocks. We did experiments with CAT-VDCRN on Block 1∼3, and both two convolutional layers in each block are applied with CAT. The comparison of different block positions is shown in the bottom part of Table 3. It indicates that the performance decreases when applying CAT on higher residual blocks, and the first block is the best position for cluster adaptive training. In summary, the second line of Table 3 is also the best configuration of the proposed CAT-VDCRN.

4.3. Evaluation on language model and system combination

In addition to CAT-VDCRN, LSTM-RNN with factor aware training was also built (FAT-LSTM) [31, 32]. The auxiliary feature was 100-dim i-vector and it was directly concatenated with acoustic feature as the inputs. The system combination using joint decoding is then implemented [15] (a weighted combination of state-level acoustic log likelihoods from individual models, denoted as $\otimes$), and results are listed in the top part of Table 4. It is interesting to find that although the performance gap within CAT-VDCRN and FAT-LSTM is huge (3.7% absolute), the complementarity is still obvious and the combination further obtains a significant gain.

Finally the LM rescoring is applied on the 100-best from the joint-decoding system. LM is evaluated from the task-standard WSJ0 bigram to 5-gram, conventional forward LSTM-RNN LM and the newly proposed FV-LSTM RNN LM (all LMs are trained with the same WSJ0 data), and the results are illustrated as the bottom part of Table 4. It shows that both the higher order ngram and RNN LMs can get another large gain upon the acoustic model advancement, and the new proposed FV-LSTM RNN LM can achieve further significant WER reduction when combined with the traditional LMs. Our best system obtained 3.09% WER on Aurora4, which is a huge progress compared to the previous work on this task. Particularly, the WERs on subset B and C are closer to that on A, and the WER on subset D is also reduced dramatically.

4.4. Error analysis within humans and machines

In this section, we want to do the error analysis and performance comparison between humans and machines for noisy speech transcription, similar as that for the telephone speech [20]. Three native English speakers are recruited to transcribe each noisy condition speech in Aurora4 for the first pass, and one additional speaker is asked to do the quality checking for the second pass. We want to see how humans perform on the corrupted noisy speech.

Systems	A	B	C	D	Avg.
CAT-VDCRN (I)	2.95	4.17	3.70	8.55	5.92
FAT-LSTM (II)	3.75	6.87	5.64	13.99	9.61
(I) $\otimes$ (II)	2.82	3.90	3.53	8.26	**5.67**
5-gram	1.70	2.46	2.00	6.60	4.15
5-gram + LSTM	1.55	2.03	1.89	5.60	3.51
5-gram + LSTM + FV-LSTM	1.10	1.63	1.59	5.13	**3.09**
Humans on Aurora4	1.58	1.92	1.61	2.88	2.28

Table 4: WER (%) Comparison of proposed systems on Aurora4

Table 5 show the top five types of sub/del/ins errors for both ASR and human transcripts on Aurora4. It reveals that the top error patterns seem very similar for the substitutions and deletions between humans and machines, and short function words are easier to be deleted for both humans and ASR. In contrast the insertion errors are not so correlated between ASR and humans, and both short function words and long notional words could be inserted by ASR systems on this noisy task. The overall error numbers broken down by three error types are shown as the last line of Table 5. We see that machines have slightly more errors on Sub and Ins, and in contrast deletion errors are much more than those of human (more than double). It shows that the corrupted speech is more prone to be mis-recognized as non-words, and the deletion error seems one main challenge for noise-robust speech recognition. This observation is different from the error analysis conclusion for telephone speech recognition [20].

Humans' performance on Aurora4 is listed as the last line of Table 4. Compared with our best system, we see that our ASR system is approaching humans' level on Aurora4, less than absolute 1.0% on averaged WER. More precisely, machines' accuracy has been comparable on condition B and C, which has either additive noise or channel distortion. In contrast the performance gap on condition D is still very large. It indicates that scenarios with both additive noise and channel distortion are much more complex and still very hard to be addressed, and it is also the main difficult research direction for noise-robust ASR in the future.

Sub	Del	Ins
ASR / Human	ASR / Human	ASR / Human
56: a-the/46: its-it's	92: the/43: the	14: education/19: the
28: ranged-range/33: a-the	55: in/14: in	14: era/14: and
20: isn't-is/23: adviser-advisor	49: it/10: of	14: convert/11: is
18: of-a/14: ranged-range	42: of/8: that	11: the/11: in
17: and-in/13: the-a	41: to/8: a	8: dollar/7: of
all:1453 / 1258	all:676 / 286	all:190 / 168

Table 5: Most common substitutions/deletions/insertions for ASR system and humans on Aurora4. The number indicates the times each error occurs. For substitutions, two words means the true word in the reference and the wrong word in the hypothesis.

5. CONCLUSION

In this paper, very deep convolutional residual network (VDCRN) with residual learning is first introduced, which shows better robustness in noisy speech recognition. Then cluster adaptive training is developed based on VDCRN, and it can obtain significant gains under all kinds of noisy conditions. Finally a more advanced LSTM-RNN LM with the assistance from future vector is proposed and implemented to further improve the system in noisy scenarios.

The new approaches are evaluated on Aurora4. The final system with the proposed methods achieves a WER of 3.09%, even without feature enhancement. To our knowledge this is the best published result on Aurora4, and it is even very close to humans' performance on this task.

6. REFERENCES

[1] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[2] Frank Seide, Gang Li, and Dong Yu, "Conversational speech transcription using context-dependent deep neural networks.," in *INTERSPEECH*, 2011, pp. 437–440.

[3] Yongqiang Wang and Mark JF Gales, "Speaker and noise factorization for robust speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 2149–2158, 2012.

[4] Yan Huang, Dong Yu, Chaojun Liu, and Yifan Gong, "A comparative analytic study on the gaussian mixture and context dependent deep neural network hidden markov models," in *INTERSPEECH*, 2014.

[5] Shuo-Yiin Chang and Steven Wegmann, "On the importance of modeling and robustness for deep neural network feature," in *ICASSP*. IEEE, 2015, pp. 4530–4534.

[6] Jinyu Li, Li Deng, Yifan Gong, and Reinhold Haeb-Umbach, "An overview of noise-robust automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 745–777, 2014.

[7] Yifan Gong, "Speech recognition in noisy environments: A survey," *Speech communication*, vol. 16, no. 3, pp. 261–291, 1995.

[8] LIU Shilin and Khe Chai Sim, "Joint adaptation and adaptive training of tvwr for robust automatic speech recognition," in *INTERSPEECH*, 2014, pp. 636–640.

[9] Dong Yu, Li Deng, Jasha Droppo, Jian Wu, Yifan Gong, and Alex Acero, "A minimum-mean-square-error noise reduction algorithm on mel-frequency cepstra for robust speech recognition," in *ICASSP*. IEEE, 2008, pp. 4041–4044.

[10] Takuya Yoshioka and Mark JF Gales, "Environmentally robust asr front-end for deep neural network acoustic models," *Computer Speech & Language*, vol. 31, no. 1, pp. 65–86, 2015.

[11] Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran, "Deep convolutional neural networks for lvcsr," in *ICASSP*. IEEE, 2013, pp. 8614–8618.

[12] Mengxiao Bi, Yanmin Qian, and Kai Yu, "Very deep convolutional neural networks for lvcsr," in *INTERSPEECH*, 2015, pp. 3259–3263.

[13] Tom Sercu, Christian Puhrsch, Brian Kingsbury, and Yann LeCun, "Very deep multilingual convolutional neural networks for lvcsr," in *ICASSP*. IEEE, 2016, pp. 4955–4959.

[14] Yanmin Qian, Mengxiao Bi, Tan Tian, and Kai Yu, "Very deep convolutional neural networks for noise robust speech recognition," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 12, pp. 2263–2276, 2016.

[15] Yanmin Qian and Philip C Woodland, "Very deep convolutional neural networks for robust speech recognition," in *SLT*, 2016.

[16] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[18] Tian Tan, Yanmin Qian, Maofan Yin, Yimeng Zhuang, and Kai Yu, "Cluster adaptive training for deep neural network," in *ICASSP*. IEEE, 2015, pp. 4325–4329.

[19] Dong Yu, Wayne Xiong, Jasha Droppo, Andreas Stolcke, Guoli Ye, Jinyu Li, and Geoffrey Zweig, "Deep convolutional neural networks with layer-wise context expansion and attention," in *INTERSPEECH*, 2016.

[20] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig, "Achieving human parity in conversational speech recognition," *arXiv preprint arXiv:1610.05256*, 2016.

[21] George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, Bergul Roomi, and Phil Hall, "English conversational telephone speech recognition by humans and machines," in *INTERSPEECH*, 2017.

[22] Mark JF Gales, "Cluster adaptive training for speech recognition.," in *ICSLP*, 1998, vol. 1998, pp. 1783–1786.

[23] Tian Tan, Yanmin Qian, and Kai Yu, "Cluster adaptive training for deep neural network based acoustic model," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 03, pp. 459–468, 2016.

[24] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur, "Recurrent neural network based language model," in *INTERSPEECH*, 2010, pp. 1045–1048.

[25] Quoc V. Le and Tomas Mikolov, "Distributed representations of sentences and documents," in *ICML*, 2014, pp. 1188–1196.

[26] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom, "A convolutional neural network for modelling sentences," in *ACL*, 2014, pp. 655–665.

[27] Qi Liu, Yanmin Qian, and Kai Yu, "Future vector enhanced lstm language model for lvcsr," in *ASRU*, Okinawa, Japan, December 2017.

[28] David Pearce and J Picone, "Aurora working group: Dsr front end lvcsr evaluation au/384/02," *Inst. for Signal & Inform. Process., Mississippi State Univ., Tech. Rep*, 2002.

[29] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., "The kaldi speech recognition toolkit," in *ASRU*. IEEE Signal Processing Society, 2011, number EPFL-CONF-192584.

[30] Dong Yu, Adam Eversole, Mike Seltzer, Kaisheng Yao, Zhiheng Huang, Brian Guenter, Oleksii Kuchaiev, Yu Zhang, Frank Seide, Huaming Wang, et al., "An introduction to computational networks and the computational network toolkit," *Microsoft Technical Report MSR-TR-2014–112*, 2014.

[31] Tian Tan, Yanmin Qian, Dong Yu, Souvik Kundu, Liang Lu, Khe Chai Sim, Xiong Xiao, and Yu Zhang, "Speaker-aware training of lstm-rnns for acoustic modeling," in *ICASSP*, Shanghai, China, March 2016, pp. 5280–5284.

[32] Qian Yanmin, Tan Tian, and Yu Dong, "Neural network based multi-factor aware joint training for robust speech recognition," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 12, pp. 2231–2240, 2016.