INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

The JHU ASR System for VOiCES from a Distance Challenge 2019

Yiming Wang[1], *David Snyder*[1,2], *Hainan Xu*[1], *Vimal Manohar*[1,2], *Phani Sankar Nidadavolu*[1], *Daniel Povey*[1,2], *Sanjeev Khudanpur*[1,2]

[1]Center for Language and Speech Processing, Johns Hopkins University, USA
[2]Human Language Technology Center of Excellence, Johns Hopkins University, USA

{yiming.wang,hxu31,vmanoha1,snidada1,khudanpur}@jhu.edu, {david.ryan.snyder,dpovey}@gmail.com

Abstract

This paper describes the system developed by the JHU team for automatic speech recognition (ASR) of the VOiCES from a Distance Challenge 2019, focusing on single channel distant/far-field audio under noisy conditions. We participated in the Fixed Condition track, where the systems are only trained on an 80-hour subset of the Librispeech corpus provided by the organizer. The training data was first augmented with both background noises and simulated reverberation. We then trained factorized TDNN acoustic models that differed only in their use of i-vectors for adaptation. Both systems utilized RNN language models trained on original and reversed text for rescoring. We submitted three systems: the system using i-vectors with WER 19.4% on the development set, the system without i-vectors that achieved WER 19.0%, and the their lattice-level fusion with WER 17.8%. On the evaluation set, our best system achieves 23.9% WER.

Index Terms: far-field speech recognition, VOiCES Challenge 2019

1. Introduction

Far-field automatic speech recognition (ASR) under reverberant and noisy conditions is important in real-world applications of ASR. Speech signal containing additive noises and reverberations can degrade the ASR performance considerably [1, 2, 3]. This problem remains challenging due to ambiguity of audio signals, limited availability of training data with matched conditions with those of test data, etc.

The VOiCES from a Distance Challenge 2019 [4, 5] aims to benchmark and further improve state-of-the-art technologies in the area of speaker recognition and ASR for far-field speech. For the fixed condition of the ASR track, the training set consists of an 80-hour subset of the LibriSpeech corpus [6] from 427 different speakers, recorded with close microphones in quiet environments. The development and evaluation set consist of 19 hours and 20 hours, respectively, of distant recordings in environments representing different rooms, microphones, noise distractors, and loudspeaker angles [4, 5]. The training set has no overlaps in speakers with the development or evaluation set of the VOiCES corpus. Note that there are significant mismatches between the training and development/evaluation set in terms of the recording conditions, which makes this challenge even more difficult.

There has been much work on ASR dealing with robustness to reverberation [7, 8] and background noises [9, 10], including frond-end-based [11, 12, 13, 14] and back-end-based [15, 16] approaches. Among those, beamforming [17] is one of the most common approaches for dereverberation, but it is only applicable to multi-channel signals, which is not the case for the VOiCES Challenge. On the other hand, several feature-based and model-based adaptation techniques to the environment or speakers have also been proposed, such as maximum likelihood linear regression (MLLR) [18], vocal tract length normalization [19], mean-variance normalization [20], vector Taylor series [21] and i-vectors [22, 23].

This paper presents the JHU ASR system for the VOiCES Challenge in the fixed condition track. An overview of our submission systems is as follows. First, in order to increase the amount of training data and mitigate its domain mismatch with VOiCES test data, we augmented the training data with several additive background noises, speed perturbation [24] and simulated reverberation. Then we trained a factorized TDNN with skip connections [25] acoustic model with lattice-free maximum mutual information (LF-MMI) criterion [26]. For language modeling, 2 RNNLMs were trained on both forward and backward (reversed) text separately and combined together with the n-gram language model for lattice rescoring [27]. We trained two complete systems, one is with i-vectors [22] as its additional input, and the other is without i-vectors, and combined these two systems on lattice level as the third submitted system. Although i-vectors adaptation is generally not helpful for domain mismatched data as in this challenge, the improved performance after model combination implies its usefulness by leveraging complementary information from both the speaker-adapted system and the non-speaker-adapted system.

The remainder of this paper is organized as follows: Section 2 illustrates how we augment the training data. Section 3 presents the details of our acoustic model. Section 4 describes the RNNLMs used as well as the lattice rescoring methods. The details of our experiments and analysis are given in Section 5. We conclude in Section 6.

2. Data Augmentation

Given the fact that the training data has mismatched conditions with those of the VOiCES test data, we augment the training data with a number of datasets.

- *Babble*: a dataset consisting of audio files of 3 to 5 speakers from the microphone portion of Mixer 6 [28] that have been summed together to create babble noise.
- *Music*: the music files from the MUSAN corpus [29][1] that do not contain vocals.
- *Noise*: the noise files from the MUSAN corpus.

[1]http://www.openslr.org/resources/17

- *Reverb*: simulated RIRs[2] as described in [30].

To conform to the fixed condition of the challenge, the training data consists only of the Librispeech-80h dataset plus augmentations. We randomly apply additive noises from the "babble", "music" and "noise" datasets separately on each copy of the clean training data N times. For each training example, "babble" is added as background noises 3 to 7 times with SNRs ranging from 13 to 20; "music" is added as background noises once with SNRs ranging from 5 to 15; "noise" is added as foreground noises at the interval of 1 second with SNRs ranging from 0 to 15. Then reverberation is applied on top of them using the simulated RIRs with room sizes uniformly sampled from 1 meter to 30 meters. The above procedure leads to $3N$ times more augmented training data. The alignments for these augmented data are obtained from their clean counterparts. In addition, we apply 3-fold speed perturbation [24] to the clean training data. In total, these augmentations increase the amount of training data by $3N + 3$ times, where N is a hyper-parameter whose impact we will report in the experiment section.

3. Acoustic Modeling

We use factorized TDNN (F-TDNN) with skip connections [25] for acoustic modeling. Table 1 summarizes the layers. The F-TDNN reduces the number of parameters of the network by factorizing the weight matrix of each TDNN layer into the product of two low-rank matrices, the first of which is constrained to be semi-orthogonal. It is assumed that the semi-orthogonal constrain helps to ensure that we do not lose information when projecting from the high dimension to the low dimension.

The authors of the original paper found that, instead of factorizing the TDNN layer into a convolution times a feed-forward layer, it is better to factorize the layer into two convolutions with half the kernel size. For example, instead of using a kernel with context $(-3, 0, +3)$ in the first factor of the layer and 0 context in the second factor, it is better to use a kernel with context $(-3, 0)$ in the first factor and a kernel with context $(0, +3)$ in the second factor.

As in other architectures like ResNet [31], we incorporate skip connections. This means that for some layers, they receive input not only from the previous layer but also from other prior layers. This allows us to make the network deeper by alleviating the vanishing gradient problem. In the original paper [25], the prior layers were concatenated to the input of the current layer, and the skip connections were created between the low-rank interior layers of the F-TDNN. However, we found in our preliminary experiments that the following modifications are more helpful: 1) instead of individually specifying the skip connection, each F-TDNN layer always receives its immediate prior layers output as the skip connection, and 2) instead of concatenation, the prior layer is added to the input of the current layer after being scaled down with a constant (0.75 in our experiments). As shown in Table 1, we use 16 F-TDNN layers, with dimension 1536 and linear bottleneck layers of dimension 160. Note that BatchNorm applied after each ReLU is omitted in Table 1 for brevity. The acoustic model has about 20 million parameters.

For acoustic model training ,we use the LF-MMI objective function:

$$\mathcal{F}_{\text{LF-MMI}} = \sum_{n=1}^{N} \log \frac{P(\mathbf{O}_n|L_n)^{\kappa} P(L_n)}{\sum_L P(\mathbf{O}_n|L)^{\kappa} P(L)} \tag{1}$$

[2]http://www.openslr.org/resources/28

Table 1: *Factorized TDNN architecture for acoustic modeling in ASR systems.*

Layer	Layer Type	Context factor1	Context factor2	Skip conn. from layer	Size	Inner size
1	TDNN-ReLU	t			1536	
2	F-TDNN-ReLU	t-1, t	t, t+1	0 (input)	1536	160
3	F-TDNN-ReLU	t-1, t	t, t+1	1	1536	160
4	F-TDNN-ReLU	t-1, t	t, t+1	2	1536	160
5	F-TDNN-ReLU	t	t	3	1536	160
6	F-TDNN-ReLU	t-3, t	t, t+3	4	1536	160
7	F-TDNN-ReLU	t-3, t	t, t+3	5	1536	160
8	F-TDNN-ReLU	t-3, t	t, t+3	6	1536	160
9	F-TDNN-ReLU	t-3, t	t, t+3	7	1536	160
10	F-TDNN-ReLU	t-3, t	t, t+3	8	1536	160
11	F-TDNN-ReLU	t-3, t	t, t+3	9	1536	160
12	F-TDNN-ReLU	t-3, t	t, t+3	10	1536	160
13	F-TDNN-ReLU	t-3, t	t, t+3	11	1536	160
14	F-TDNN-ReLU	t-3, t	t, t+3	12	1536	160
15	F-TDNN-ReLU	t-3, t	t, t+3	13	1536	160
16	F-TDNN-ReLU	t-3, t	t, t+3	14	1536	160
17	F-TDNN-ReLU	t-3, t	t, t+3	15	1536	160
18	Linear				256	
19	Dense-ReLU-Linear				256	1536
20	Dense				N. targets	

where L_n is the phone sequence of the n-th utterance and $P(L)$ is the phone language model estimated from the previous HMM-GMM phone alignments.

4. Language Modeling

We follow Kaldi-RNNLM [32] to train TDNN-LSTM language models using the transcripts of Librispeech-80h. The LM is trained using the objective denoted in Equation (2).

$$\mathcal{J}_{\text{LM}} = z_l + 1 - \sum_i \exp z_i \tag{2}$$

where in the equation, z denotes the neural-network output before the last softmax operation, and l is the index of the correct word. This objective function is a linear approximation of the standard cross-entropy objective as denoted in Equation (3) but is a lower-bound of cross-entropy.

$$\mathcal{J}_{\text{CE}} = z_l - \log \sum_i \exp z_i \tag{3}$$

Using the new objective function allows the trained RNNLMs to self-normalize during inference and thus saves computational time. Also, the linearity of the objective allows a sampling-based method to be adopted during training which could significantly speed up training.

For representations of the words, letter n-gram features are combined with one-hot vectors in generating word-embeddings as described in [32]. We extract the most frequent words and letter n-grams from the training corpus, where each word/letter n-gram would be associated with its embedding. To generate the word-embedding for a word, we compute it as the sum of the embeddings of all the "features" it has. This allows us to utilize spelling information of a word in generating embeddings, and gives better embedding to words that appear relatively rare in the corpus.

To further improve the performance, on top of training a standard RNNLM on the training corpus, we also reverse the corpus text and train a "backward" RNNLM on that. When applying rescoring, the scores are averaged between the forward RNNLM scores and that from backward RNNLM (scored on reversed text).

We perform pruned lattice-rescoring [27] to combine the weights of the 2 RNNLMs with the original n-gram weights, where the weights for forward-RNNLM, backward-RNNLM, and the original n-gram are 0.3, 0.3, 0.4 respectively. Language models are also trained only on transcripts of the Librispeech-80h dataset.

5. Experiments

5.1. Baseline

For the baseline system, we follow the standard Kaldi [33] recipe[3] for HMM-GMM training on the original Librispeech-80h data up to the speaker adapted training with pronunciation and word-specific silence probabilities modeling [34]. The n-gram language model is trained on a vocabulary of size 200k with max-entropy criterion. Alignments and numerator lattices generated from the HMM-GMM model are used for neural network acoustic model training with the F-TDNN network and the LF-MMI criterion with cross-entropy regularization [26]. The features are 40 dimensional MFCCs extracted from the 16 kHz data. The baseline also incorporates i-vectors as an additional input features, as is standard in the Kaldi ASR recipes: the UBM uses a 512 component GMM and the i-vector extractor is trained on the clean Librispeech-80h data and then 100 dimensional embeddings are extracted. The results on the development/evaluation set are presented in Table 2.

Table 2: *WERs on Dev and Eval set with baseline models.*

System	Dev (%)	Eval (%)
GMM	79.5	87.0
F-TDNN	**68.3**	**78.6**

5.2. Data Augmentation

We investigate how different amount of augmented data affect the performance, and vary N (as described in Section 2) to increase the amount of augmented data by 6x, 12x and 18x, leading to a total training data of the size 480 hrs, 960 hrs and 1440 hrs respectively. The results are shown in Table 3, where we observe significant gain from 480 hrs to 960 hrs, but no further gains are attained when more augmentation data was added. We stick to the 960 hrs for later experiments.

Table 3: *WER on Dev set with different amount of data augmentation.*

Data Amount (hrs)	Dev (%)
480	24.5
960	**22.3**
1440	23.1

5.3. I-vectors Adaptation

Since there exists severe domain mismatch between VOiCES test data and Librispeech-80h data which i-vector extractor is trained on, it is possible that the extracted i-vectors from the development/evaluation data do not represent the speaker characters well. Therefore, we remove i-vectors from our previous system, while keep applying speaker-level cepstral mean subtraction (CMN) on training data and utterance-level CMN on development data[4]. We can see from Table 4 that removing i-vectors help improve the performance.

Table 4: *WERs on Dev set with/without i-vectors.*

	Dev (%)
with i-vectors	22.3
without i-vectors	**21.5**

5.4. RNNLM Rescoring

We utilize RNNLMs for 2nd pass rescoring. All hyper-parameters regarding RNNLMs are tuned based on the system without i-vectors. The network dimensions are tuned on WER scores on the dev data, and the numbers are shown in Table 5, where the WER numbers are based on lattice-rescoring with a language model weight of 0.5. We fix the LSTM dimension to be 1024 since it gives the best performance.

Table 5: *RNNLM hyper-parameter tuning.*

num-layers	LSTM-dim	Dev-Perplexity	Dev-WER (%)
2	512	298.3	20.0
2	1024	**291.2**	**19.6**
2	2048	300.1	19.8

For rescoring algorithm, we investigate both lattice-rescoring and n-best list rescoring to utilize RNNLMs. The comparison of the 2 methods is shown in Table 6. We see that in all settings, lattice-rescoring significantly outperforms n-best rescoring. We also notice there is significant speed difference in the experiments where lattice-rescoring experiments run much faster than n-best, so we use lattice-rescoring for all later experiments.

Table 6: *WER comparison between n-best and lattice rescoring on Dev set.*

rescoring method	Dev (%)
lattice	**19.6**
n-best (n=20)	20.7
n-best (n=50)	20.3

We perform lattice-rescoring to utilize the 2 RNNLMs trained on original and reversed texts. For the RNNLM trained on standard text, we follow [27] in performing lattice-rescoring on the generated lattices. The algorithm computes a heuristic score for each arc to be expanded in the output lattice, and prioritize "better" arcs and discard unpromising arcs. The heuristic score reflects how likely this arc is going to be part of the best path after rescoring, and considers both history and future information in the lattices.

After the rescoring is done, we reverse the lattice and perform another round of lattice rescoring with the RNNLM

[3]`https://github.com/kaldi-asr/kaldi/blob/master/egs/librispeech/s5/run.sh`

[4]Since batch-level processing is not allowed for evaluation, speaker-level CMN is not applied on development/evaluation data.

Table 7: *RNNLM weights and WERs on Dev set.*

RNNLM weights	Dev (%)
0.2	19.4
0.3	**19.0**
0.4	**19.0**
0.5	19.2

trained on reversed texts. After this procedure, the lattices are reversed again where we find the best-path. Table 7 reports the impact of RNNLM weights on WERs, where the weights reported are shared by both forward and backward RNNLM (For example, if weight = 0.3, it means the weights for both RNNLMs are 0.3, and the weight for the original n-gram is 0.4). We use weight = 0.3 for later experiments since it gives the best performance.

5.5. Model Fusion

We combine the two RNNLM rescored lattices from the above two systems (i.e. with and without i-vectors adaptation) into a single lattice to obtain a third system. It is achieved by removing the total cost of all paths (backward cost) from individual lattices and performing a union of the reweighted lattices, where the combination weights are 0.5 and 0.5 respectively. Then Minimum Bayes Risk decoding [35] is applied on top of the combined lattice. The improvement demonstrated in Table 8 implies complementarity of the two single systems.

Table 8: *WERs of the individual systems and their fusion on Dev and Eval set.*

System	Dev (%)	Eval (%)
F-TDNN_ivec	19.4	26.3
F-TDNN_noivec	19.0	25.3
fusion	**17.8**	**23.9**

5.6. Performance Analysis

We analyze the performance of our best system (the third row of Table 8) on the development/evaluation set, by breaking them down into different recording environments using the meta-data [36] provided by the organizer.

We note that there is a performance gap between WER on the development set (17.8%) and the evaluation set (23.9%). We also notice that the room-ids in the development set only consist of "rm1" or "rm2", while in the evaluation set there are additional rooms "rm3" and "rm4". So we break down the results into different rooms as shown in Table 9. The performance for "rm1" and "rm2" on the evaluation set is comparable to those on the development set, which indicates that there is not much difference in performance between the development and evaluation set for these two rooms. However, our system performs significantly worse for "rm3" and "rm4" on the evaluation set. According to the data description website [36], the size of "rm1" and "rm2" is 3.7 m $\times$ 2.7 m and 5.7 m $\times$ 4.0 m respectively, both of which are within the ranges of our simulated rooms described in Section 2. We do not have information regarding the room sizes of "rm3" and "rm4".

Next we break down the results according to different distractor types (in Table 10) and microphone locations and types (in Table 11). One of the observations is that the distractor "babb", microphone location "far" and microphone type "lav" make the performance worst among the others in their respective types.

Table 9: *Breakdown of the best system's performance according to room-ids.*

Room-id	Dev (%)	Eval (%)
rm1	18.0	18.9
rm2	17.7	18.3
rm3	—	31.8
rm4	—	26.7

Table 10: *Breakdown of the best system's performance according to distractor types.*

Distractor Type	Dev (%)	Eval (%)
none	14.6	24.8
musi	19.3	22.0
tele	18.0	19.0
babb	19.4	28.7

Table 11: *Breakdown of the best system's performance according to microphone locations and types.*

	Dev (%)	Eval (%)
Mic Loc		
clo	13.9	22.6
mid	—	24.4
far	20.9	25.8
beh	17.2	16.8
ceo	20.6	21.1
tbo	20.6	—
Mic Type		
stu	12.0	22.9
lav	19.0	24.7

6. Conclusions

In this paper we describe the JHU ASR system in the fixed condition for VOiCES from a Distance Challenge 2019. The features of our systems include data augmentation with various additive noises and simulated reverberations, factorized TDNN acoustic model with skip connections, RNNLMs trained on both forward and backward text, and model fusion from both the system with i-vectors and the system without i-vectors. The final fused system achieves WER 17.8% on the development set and 23.9% on the evaluation set. In particular, we also observe that due to domain mismatch, i-vectors trained on clean data does not help in the test condition for single systems.

7. Acknowledgements

This work was partially supported by the DARPA MATERIAL and LORELEI programs (FA8650-17-C-9115, HR0011-15-2-0024) and an unrestricted gift from Mobvoi Inc.

8. References

[1] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'chime' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. Interspeech 2018*, 2018, pp. 1561–1565.

[2] K. Kinoshita, M. Delcroix, S. Gannot, E. A. Habets, R. Haeb-Umbach, W. Kellermann, V. Leutnant, R. Maas, T. Nakatani, B. Raj *et al.*, "A summary of the reverb challenge: state-of-the-art and remaining challenges in reverberant speech processing research," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, p. 7, 2016.

[3] M. Harper, "The automatic speech recogition in reverberant environments (aspire) challenge," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 547–554.

[4] C. Richey, M. A. Barrios, Z. Armstrong, C. Bartels, H. Franco, M. Graciarena, A. Lawson, M. K. Nandwana, A. Stauffer, J. van Hout, P. Gamble, J. Hetherly, C. Stephenson, and K. Ni, "Voices obscured in complex environmental settings (voices) corpus," in *Proc. Interspeech 2018*, 2018, pp. 1566–1570.

[5] M. K. Nandwana, J. Van Hout, M. McLaren, C. Richey, A. Lawson, and M. A. Barrios, "The voices from a distance challenge 2019 evaluation plan," *arXiv preprint arXiv:1902.10828*, 2019.

[6] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[7] P. A. Naylor and N. D. Gaubitch, *Speech dereverberation*. Springer Science & Business Media, 2010.

[8] T. Yoshioka, A. Sehr, M. Delcroix, K. Kinoshita, R. Maas, T. Nakatani, and W. Kellermann, "Making machines understand us in reverberant rooms: Robustness against reverberation for automatic speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 114–126, 2012.

[9] T. Virtanen, R. Singh, and B. Raj, *Techniques for noise robustness in automatic speech recognition*. John Wiley & Sons, 2012.

[10] J. Li, L. Deng, Y. Gong, and R. 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.

[11] W. Li, L. Wang, F. Zhou, and Q. Liao, "Joint sparse representation based cepstral-domain dereverberation for distant-talking speech recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7117–7120.

[12] Y. Ueda, L. Wang, A. Kai, X. Xiao, E. S. Chng, and H. Li, "Single-channel dereverberation for distant-talking speech recognition by combining denoising autoencoder and temporal structure normalization," *Journal of Signal Processing Systems*, vol. 82, no. 2, pp. 151–161, 2016.

[13] M. Delcroix, T. Hikichi, and M. Miyoshi, "Precise dereverberation using multichannel linear prediction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 2, pp. 430–440, 2007.

[14] L. Wang, K. Odani, and A. Kai, "Dereverberation and denoising based on generalized spectral subtraction by multi-channel lms algorithm using a small-scale microphone array," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, p. 12, 2012.

[15] A. Sehr, R. Maas, and W. Kellermann, "Reverberation model-based decoding in the logmelspec domain for robust distant-talking speech recognition," *IEEE transactions on audio, speech, and language processing*, vol. 18, no. 7, pp. 1676–1691, 2010.

[16] H.-G. Hirsch and H. Finster, "A new approach for the adaptation of hmms to reverberation and background noise," *Speech Communication*, vol. 50, no. 3, pp. 244–263, 2008.

[17] X. Xiao, S. Watanabe, H. Erdogan, L. Lu, J. Hershey, M. L. Seltzer, G. Chen, Y. Zhang, M. Mandel, and D. Yu, "Deep beamforming networks for multi-channel speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5745–5749.

[18] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer speech & language*, vol. 9, no. 2, pp. 171–185, 1995.

[19] L. Lee and R. Rose, "A frequency warping approach to speaker normalization," *IEEE Transactions on speech and audio processing*, vol. 6, no. 1, pp. 49–60, 1998.

[20] F.-H. Liu, R. M. Stern, X. Huang, and A. Acero, "Efficient cepstral normalization for robust speech recognition," in *Proceedings of the workshop on Human Language Technology*. Association for Computational Linguistics, 1993, pp. 69–74.

[21] P. J. Moreno, B. Raj, and R. M. Stern, "A vector taylor series approach for environment-independent speech recognition," in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, vol. 2. IEEE, 1996, pp. 733–736.

[22] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 2013, pp. 55–59.

[23] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[24] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Interspeech 2015*, 2015, pp. 3586–3589.

[25] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Proc. Interspeech 2018*, 2018, pp. 3743–3747.

[26] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi," in *Interspeech 2016*, 2016, pp. 2751–2755.

[27] H. Xu, T. Chen, D. Gao, Y. Wang, K. Li, N. Goel, Y. Carmiel, D. Povey, and S. Khudanpur, "A pruned rnnlm lattice-rescoring algorithm for automatic speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5929–5933.

[28] L. D. Consortium, "Mixer 6 corpus specification v4.1,," 2013.

[29] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[30] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5220–5224.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[32] H. Xu, K. Li, Y. Wang, J. Wang, S. Kang, X. Chen, D. Povey, and S. Khudanpur, "Neural network language modeling with letter-based features and importance sampling," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 6109–6113.

[33] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU2011*. Waikoloa, HI, USA: IEEE, dec 2011, pp. 1–4.

[34] G. Chen, H. Xu, M. Wu, D. Povey, and S. Khudanpur, "Pronunciation and silence probability modeling for asr," in *Interspeech 2015*, 2015, pp. 533–3537.

[35] H. Xu, D. Povey, L. Mangu, and J. Zhu, "Minimum bayes risk decoding and system combination based on a recursion for edit distance," *Computer Speech & Language*, vol. 25, no. 4, pp. 802–828, 2011.

[36] "VOiCES corpus dataset description," https://voices18.github.io/Lab41-SRI-VOiCES_README/.