ACOUSTIC MODELING FOR OVERLAPPING SPEECH RECOGNITION: JHU CHIME-5 CHALLENGE SYSTEM

Vimal Manohar[1,2], Szu-Jui Chen[1], Zhiqi Wang[1], Yusuke Fujita[3,1], Shinji Watanabe[1], Sanjeev Khudanpur[1,2]

[1]Center for Language and Speech Processing
[2]Human Language Technology Center Of Excellence
Johns Hopkins University, Baltimore, MD 21218
[3]Hitachi, Ltd. Research & Development Group, Kokubunji-shi, Tokyo, Japan
vimal.manohar91@gmail.com,{schen146,zwang132,shinjiw,khudanpur}@jhu.edu

ABSTRACT

This paper summarizes our acoustic modeling efforts in the Johns Hopkins University speech recognition system for the CHiME-5 challenge to recognize highly-overlapped dinner party speech recorded by multiple microphone arrays. We explore data augmentation approaches, neural network architectures, front-end speech dereverberation, beamforming and robust i-vector extraction with comparisons of our in-house implementations and publicly available tools. We finally achieved a word error rate of 69.4% on the development set, which is a 11.7% absolute improvement over the previous baseline of 81.1%, and release this improved baseline with refined techniques/tools as an advanced CHiME-5 recipe.

Index Terms— Robust speech recognition, acoustic modeling, Kaldi, CHiME-5 challenge

1. INTRODUCTION

Automatic speech recognition (ASR) has improved significantly in the area of conversational speech recognition. A lot of this can be attributed to larger datasets for training, new sequence training objectives like Connectionist Temporal Classification (CTC) [1] and Lattice-free MMI (LF-MMI) [2], and improved neural network architectures for acoustic modeling. Recently, super-human performance has been achieved on conversational telephone speech using publicly available datasets [3, 4, 5]. However, speech recognition in more difficult settings like reverberant, noisy, and overlap speech conditions, still lags behind, and is widely believed to be the next frontier in speech recognition. Interest in robust and distant speech recognition has increased greatly with new voice control applications in home environments, cars, etc.

The CHiME-5 challenge [6] focuses on the problem of distant microphone conversational speech recognition in the setting of everyday home environments. The speech data consists of real 4-people dinner party conversations recorded using linear array microphones, and annotated with speech start/end times, the speaker labels and speech transcription.

The challenge organizers developed the baseline system using state-of-the-art techniques including LF-MMI and time-delay neural network (TDNN) architecture [6]. However, the word error rate (WER) of the baseline system was 81.1% and did not reach a satisfied level. This huge number for WER implies two challenging problems in the CHiME-5. One challenging problem is that distant microphones are distributed in different home locations and speakers are not always facing to any microphones. Another big problem is spontaneous and overlapping nature of speech – speakers in the parties are all friends and instructed to behave naturally.

First, to tackle the distributed distant microphone setting, we explore data augmentation for improving robustness against a variety of room acoustics and directivity patterns of speech, which shows great success in noise robust ASR [7, 8, 9]. We also explore neural network architectures for capturing the variation introduced by data augmentation. In addition, the effectiveness of dereverberation techniques in different network architectures is evaluated. Second, to tackle spontaneous and overlapping nature of speech, a 2-stage decoding method is examined. This method extracts reliable single-speaker frames for i-vector extraction [10].

While we developed an advanced ASR system with other colleagues as the Hitachi/JHU CHiME-5 Challenge submission [11] during the challenge period, we used different implementations in this work. One of the most important contributions of this work is to provide our outcomes in a publicly available new Kaldi recipe[1] so that the researchers in this field can reproduce this new advanced baseline. We explore the implementation of above mentioned techniques using publicly available tools including Nara WPE [12] and RIR generator [13]. We include these public tools in our Kaldi recipe to build a single-system baseline not using any system combination.

This work was partially supported by NSF Grant No CRI-1513128 and IARPA MATERIAL award number FA8650-17-C-9115. Vimal Manohar was supported by Alexa Graduate Fellowship.

[1]https://github.com/kaldi-asr/kaldi/tree/master/egs/chime5/s5b

2. CHIME-5 CONVENTIONAL ASR SYSTEM

This section describes the CHiME-5 data and the baseline ASR system released by the challenge organizers [6].

2.1. CHiME-5 Data

The dataset consists of 20 dinner parties, each recorded with six Kinect[2] devices placed in different locations. Each Kinect device has a linear array of 4 sample-synchronized microphones. 16 of these parties form the training set, 2 of them are in the development set, and the rest are in the evaluation set. Since the same speech is recorded from multiple locations and channels, we randomly select some of the utterances for training. We refer to these subsets of audio based on the amount of utterances selected e.g. "100k", "400k". In addition, each speaker is also recorded using a set of worn binaural microphones to provide parallel (relatively) "clean" speech. We arbitrarily selected to use only the left channel from this. We refer to this set of audio as "worn" data.

2.2. HMM-GMM System

A HMM-GMM system is used as a seed system to get alignments for neural network training. We use the same approach as in [6] for training this system; however we use larger amounts of utterances from the array microphone data. While the baseline system used "worn + 100k" set for training, we get improved results using "worn + 400k" set. In this work, we use the same array synchronization and speech enhancement methods (only at test time), as well as the same lexicon and language models used in the original baseline system.

We also do data cleanup as in [6, 10] to remove irregularities in the utterances such as transcription errors, excessive silence and noise, and train a new HMM-GMM system using the "cleaned" version of "worn + 100k" or "worn + 400k" set.

2.3. Neural network acoustic model

The baseline system in [6] used a simple time-delay neural network (TDNN). We make several improvements to this acoustic model as described in the following section.

3. ACOUSTIC MODELING IMPROVEMENTS

3.1. Data Augmentation

Simulating reverberated speech from worn microphone recordings is beneficial to improving robustness. This simulation is performed by utilizing room impulse responses (RIR) and point-source noises. We examine two approaches to generate RIRs to obtain the simulated recording.

[2]Kinect is a registered trademark of Microsoft Corporation.

3.1.1. CHiME-5 data augmentation

In the first method, we use the image method proposed by Allen and Berkeley in [14] to generate the RIR for each recording. Here, we use approximated room size and the position of the microphones in reference to the provided floorplans. All rooms are considered as a cube. Then we randomly put a location for each speaker inside the room. The reverberation time is randomly sampled from a Gaussian distribution between 0 to 0.9 second. After getting all these configurations, we generate an RIR for each microphone.

Point-source noises are collected from the training data because we are not allowed to use external noise corpora in the CHiME-5 challenge. We simply extract all the non-speech parts from the original recordings of CHiME-5 training data based on the annotated start/end times, and split them into 20-second chunks as noise chunks.

Then, we randomly pick up 0 to 3 point-source noises assigned with a corresponding random position inside the room. Each noise comprises of randomly selected 20-second noise chunks to make the noise have the same length as the reverberated speech. The noise is convolved with the corresponding RIR generated to get reverberated noise. Finally, we mix it with the reverberated speech with a signal-to-noise ratio (SNR) randomly selected from 0 to 30dB, and normalize the audio with the same intensity as the original recording.

3.1.2. Kaldi data augmentation

Ko et. al [15] found that augmentation using a large number of synthetic RIRs can show improvements over using even real RIRs. The synthetic RIRs were generated using the RIR generator by Habets et. al [13]. Ko et. al used additive noises from the MUSAN [16] corpus. This approach is the standard in many Kaldi [17] recipes, and we refer to this as the "*Kaldi data augmentation*". In this paper, we compare this method with our CHiME-5 data augmentation described in Section 3.1.1. Since the CHiME-5 challenge does not allow using external noise corpus, we a variation of this method using the noises extracted from CHiME-5 data as described in Section 3.1.1. Because of the simplicity of the latter method, we include it in our new improved Kaldi recipe.

3.2. Robust i-vector extraction

We use a 2-stage decoding approach used in [18] to get i-vectors from reliable speech segments. We first perform a first-pass decode of the utterances using i-vectors extracted from the entire utterance. Only the regions corresponding to the real words that are recognized with a lattice posterior probability of 1.0 and a duration less than 1s are considered reliable for the purpose of i-vector extraction[3]. This will exclude all the regions of silence and filler words, and ideally re-

[3]We used the parameters that worked the best in [18] and did not tune them for this work.

gions of overlapping speech, which is expected to have higher confusability (and hence lower lattice posterior probability). A second-pass decoding is then done using i-vectors extracted using statistics obtained from only these reliable regions. This approach gives a consistent 1.5-2% absolute improvement of WER as shown in Section 4.3.

3.3. Neural Network Architectures

This section describes the various neural network architectures we explored for acoustic modeling – TDNN, TDNN + LSTM, TDNN-F and CNN + TDNN + LSTM. We use batch normalization [19] in all the TDNN and CNN layers, and per-frame dropout in the LSTM layers [20, 21] with scheduling of dropout factor as suggested in [21].

3.3.1. TDNN

TDNN is the architecture used in the baseline system. This consists of 8 time-delay neural network (TDNN) layers of 512 hidden units and an additional TDNN layer before the output of the same size but factorized with a bottleneck of size 320.

3.3.2. TDNN + LSTM

TDNN + LSTM architecture consists of 4 long-short term memory with projection (LSTMP) [22] layers of size 1024 and output and recurrent projection dimensions of 256. The LSTMP layers are interleaved with TDNN layers. There are two TDNN layers between a pair of LSTMP layers.

3.3.3. TDNN-F

TDNN-F architecture is a factored form of TDNN introduced in [23]. The basic factorization idea here is to factorize the weights matrix $\mathbf{W}$ (of size $m \times n$) of a TDNN layer as a product of two matrices: $\mathbf{W} = \mathbf{AB}$, where $\mathbf{A}$ is of size $m \times p$ and $\mathbf{B}$ is of size $p \times n$ such that $p << m, n$ and matrix $\mathbf{B}$ is constrained to be semi-orthogonal.

In our work, we use an architecture with 15 TDNN-F layers with a hidden dimension $n = 1536$ and a bottleneck dimension $p = 160$. Each layer also has a resnet style bypass-connection from the previous layer's output i.e. the previous layer's output (scaled by a factor of 0.66) is added to the current layer's output. The TDNN-F layers also use a "continous dropout" with scheduling as suggested in [23].

3.3.4. CNN + TDNN + LSTM

This architecture has in the beginning of the network two 2-D convolution layers with 3x3 kernels with 256 and 128 filters respectively. There is a sub-sampling along the frequency dimension by a factor of 2 after the first convolution. The convolution layers are followed by 3 LSTMP layers interleaved with two TDNN layers between each pair of LSTMP layers.

4. RESULTS

In this section, we report experimental results using the acoustic modeling improvements described in the previous section. We train our neural networks with LF-MMI objective using the Kaldi toolkit [17]. For training details, the reader is directed to [2]. The neural networks use 40-dim MFCC features as input. I-vectors are used for speaker adaptation [24, 25] with the i-vectors being extracted on top of PCA-reduced spliced-MFCC features. As in [2], we always apply 3x speed perturbation of data and volume perturbation by a random factor in $[0.8, 2.0]$ to improve robustness of the models.

4.1. Effect of training data

We compare using different amounts of training data and data augmentation methods in Table 1. Rows 1 and 2 in Table 1 show results when the ASR systems are trained using a mix of the "worn" data and array microphone data. In row 1, we use 100k utterances corresponding to $\sim$ 70 hours after cleanup and in row 2, we use 400k utterances corresponding to $\sim$ 160 hours after cleanup. Note that this is not an increase in the real amount of data, but just involves using the same speech samples recorded using different microphone channels and locations. Since, using 400k utterances gives a 3% absolute improvement, for the rest of the experiments we use these many utterances. Increasing the number of utterances to 800k did not result in any improvements in our experiments.

Row 3 in Table 1 shows the results using the same system as row 2, but applying dereverberation at test time. This gives a small improvement. In the rest of the experiments in this section, we apply dereverberation at test time. Row 4 shows results using *Kaldi data augmentation* described in Section 3.1.2 with additive noises from MUSAN [16] corpus. We create 2 copies of corrupted "worn" data and mix it with microphone data. The Row 5 uses CHiME-5 data augmentation described in Section 3.1.1 with RIRs synthesized based on CHiME-5 room conditions and noises extracted from the CHiME-5 training data. Row 6 uses *Kaldi data augmentation*, but using noises from CHiME-5 data. The results show that the latter method gives competing results. Because of its simplicity and use of open-source tools, we include this augmentation method in our new improved Kaldi recipe.

Table 1. WER(%) results for different training data

WPE	Training data	Data augmentation		WER(%)
		RIRs	Noises	
No	worn + 100k	-	-	81.3
No	worn + 400k	-	-	78.8
Nara	worn + 400k	-	-	78.2
Nara	worn + 400k	Ko [15]	MUSAN	75.4
Nara	worn + 400k	CHiME-5	CHiME-5	75.3
Nara	worn + 400k	Ko [15]	CHiME-5	**73.9**

4.2. Effect of speech dereverberation

In this section, we investigate the effect of applying weighted prediction error (WPE) [26] based dereverberation before beamforming at test time. We tried two different implementations of the WPE algorithm – NTT WPE [26][4], which has limited use based on NTT's strict licence, and Nara WPE [12][5], which is based on the MIT licence. The results are as shown in Table 2 for TDNN, TDNN + LSTM and TDNN-F systems. Systems with "Aug" as "Y" use *Kaldi data augmentation* with noises from CHiME-5. We see that applying dereverberation improves the results for all the systems. Since the Nara WPE implementation [12] has a more open license, we include it in our improved Kaldi recipe.

Table 2. WER(%) results comparing methods for dereverberation

System	Aug	No WPE	NTT	Nara
TDNN	N	78.8	78.4	78.2
TDNN	Y	74.8	74.4	73.9
TDNN + LSTM	Y	74.6	74.2	74.0
TDNN-F	Y	72.3	**71.7**	**71.7**

4.3. I-vector improvement using 2-stage decoding

In this section, we investigate the effect of using the 2-stage decoding described in Section 3.2 for the various neural network architectures described in Section 3.3. We train the networks using "worn + 400k" dataset, which is augmented using *Kaldi data augmentation method* with additive noises from CHiME-5 data. We also apply Nara WPE-based dereverberation at test time. From the results, we see that 2-stage decoding gives around 2% absolute improvement in WER in all the cases. We also find that the best network architecture is the TDNN-F, which we set as the new baseline in our updated Kaldi recipe.

Table 3. WER(%) results using 2-stage decoding

System	1-stage	2-stage
TDNN	73.9	71.9
TDNN + LSTM	74.0	72.3
CNN + TDNN + LSTM	71.9	70.6
TDNN-F	**71.7**	**69.4**

5. CONCLUSIONS

In this paper, we presented the acoustic modeling efforts at JHU for the CHiME-5 ASR system. We explored data augmentation approaches for improving robustness and found that the *Kaldi data augmentation* [15] method using a publicly available RIR generator is very effective. We used it with additive noises extracted from CHiME-5 training data to improve WER by 4% absolute. We got an additional 0.5% improvement using speech dereverberation implemented in the publicly available Nara WPE tool. We investigated a 2-stage decoding approach, where the first stage is used to select reliable single-speaker frames for extracting i-vectors. This approach gave us a consistent 2% improvement in WER. We explored various neural network architectures, and got another 2% improvement using the factorized form of TDNN (TDNN-F). Overall, we achieved a WER improvement of 11.7% over the baseline of 81.1% released by the CHiME-5 challenge organizers using an ASR system comprising of a single acoustic model, a single language model and a single front-end enhancement, and built using publicly available tools. We finally release this improved system as a new CHiME-5 Kaldi baseline recipe.

[4] http://www.kecl.ntt.co.jp/icl/signal/wpe/

[5] https://github.com/fgnt/nara_wpe

6. REFERENCES

[1] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*. ACM, 2006, pp. 369–376.

[2] Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur, "Purely Sequence-Trained Neural Networks for ASR Based on Lattice-Free MMI," in *Proc. INTERSPEECH*, 2016, pp. 2751–2755.

[3] Gakuto Kurata, Bhuvana Ramabhadran, George Saon, and Abhinav Sethy, "Language modeling with highway LSTM," in *Proc. ASRU*. IEEE, 2017, pp. 244–251.

[4] Kyu J Han, Akshay Chandrashekaran, Jungsuk Kim, and Ian Lane, "The CAPIO 2017 conversational speech recognition system," *arXiv preprint arXiv:1801.00059*, 2017.

[5] Wayne Xiong, Lingfeng Wu, Fil Alleva, Jasha Droppo, Xuedong Huang, and Andreas Stolcke, "The microsoft 2017 conversational speech recognition system," in *Proc. ICASSP*. IEEE, 2018, pp. 5934–5938.

[6] Jon Barker, Shinji Watanabe, Emmanuel Vincent, and Jan Trmal, "The fifth'chime'speech separation and recognition challenge: Dataset, task and baselines," *arXiv preprint arXiv:1803.10609*, 2018.

[7] Keisuke Kinoshita, Marc Delcroix, Takuya Yoshioka, Tomohiro Nakatani, Armin Sehr, Walter Kellermann, and Roland Maas, "The REVERB challenge: A

common evaluation framework for dereverberation and recognition of reverberant speech," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2013, pp. 1–4.

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

[9] Emmanuel Vincent, Shinji Watanabe, Aditya Arie Nugraha, Jon Barker, and Ricard Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech & Language*, vol. 46, pp. 535–557, 2017.

[10] Vijayaditya Peddinti, Vimal Manohar, Yiming Wang, Daniel Povey, and Sanjeev Khudanpur, "Far-field ASR without parallel data," in *Proc. INTERSPEECH*, 2016.

[11] Naoyuki Kanda, Rintaro Ikeshita, Shota Horiguchi, Yusuke Fujita, Kenji Nagamatsu, Xiaofei Wang, Vimal Manohar, Nelson Enrique Yalta Soplin, Matthew Maciejewski, Szu-Jui Chen, et al., "The Hitachi/JHU CHiME-5 system: Advances in speech recognition for everyday home environments using multiple microphone arrays," in *Speech Processing in Everyday Environments, The 5th International Workshop on*, 2018.

[12] Lukas Drude, Jahn Heymann, Christoph Boeddeker, and Reinhold Haeb-Umbach, "NARA-WPE: A Python package for weighted prediction error dereverberation in Numpy and Tensorflow for online and offline processing," in *13. ITG Fachtagung Sprachkommunikation (ITG 2018)*, Oct 2018.

[13] Emanuel AP Habets, "Room impulse response generator," *Technische Universiteit Eindhoven, Tech. Rep*, vol. 2, no. 2.4, pp. 1, 2006.

[14] J. Allen and D. Berkeley, "Image method for efficiently simulating small room acoustics," *Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, 1979.

[15] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L Seltzer, and Sanjeev Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 5220–5224.

[16] David Snyder, Guoguo Chen, and Daniel Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[17] 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 *Proc. ASRU*, 2011, number EPFL-CONF-192584.

[18] Vijayaditya Peddinti, Guoguo Chen, Vimal Manohar, Tom Ko, Daniel Povey, and Sanjeev Khudanpur, "Jhu aspire system: Robust lvcsr with tdnns, ivector adaptation and rnn-lms," in *Proc. ASRU*. IEEE, 2015, pp. 539–546.

[19] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[20] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting.," *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[21] Gaofeng Cheng, Vijayaditya Peddinti, Daniel Povey, Vimal Manohar, Sanjeev Khudanpur, and Yonghong Yan, "An exploration of dropout with lstms," in *Proc. Interspeech*, 2017.

[22] Haşim Sak, Andrew Senior, and Françoise Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[23] Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohamadi, and Sanjeev Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Proc. INTERSPEECH*, 2018.

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

[25] Martin Karafiat, Lukas Burget, Pavel Matejka, Ondrej Glembek, and Jan Cernocky, "iVector-based discriminative adaptation for automatic speech recognition," in *Proc. Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*. Dec. 2011, pp. 152–157, IEEE.

[26] Tomohiro Nakatani, Takuya Yoshioka, Keisuke Kinoshita, Masato Miyoshi, and Biing-Hwang Juang, "Speech dereverberation based on variance-normalized delayed linear prediction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 7, pp. 1717–1731, 2010.