The Qdreamer Systems for Chime6 Challenge

Haoyuan Tang[1], *Huanliang Wang*[1], *Jiajun Wang*[1], *Li Zhang*[1], *JiaBin Xue*[2], *Zhi Li*[1]

[1]Qdreamer Research, Suzhou, JiangSu, P.R. China
[2]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, P.R. China

haoyuan.tang@qdreamer.com, huanliang.wang@qdreamer.com, jiajun.wang@qdreamer.com

Abstract

This paper presents our discription to the Chime-6 asr system. We experimented different ways to improve the performance of our asr system, including 1) training data augmentation via different version of enhanced training data. 2) state-level minimum bayes risk (sMBR) training. 3) acoustic model fusion. 4) system combination of different version of ehanced testing data using minimum bayes risk (MBR) decoding. 5) the forward and backward long short-term memory (LSTM) based language modeling. Experiments shows our best system in category A achieved 37.59 and 38.99 of word error rates (WERs) for development and evaluation set for track1 in category A.

Index Terms: speech recognition, human-computer interaction, computational paralinguistics

1. Background

This paper presents our experiment for Chime6 chanllege. We describe our effort to improve system performance for track1 in category A and B. Our system compose the following parts: 1). A front end including Source Activity Detector (SAD), weighted prediction error dereverberation (WPE), guided source separation (GSS) and Minimum Variance Distortionless Response (MVDR). 2). Acoustic modeling trained by lattice-free maximum mutual information (MMI) criterion and sMBR. 3). LSTM based language modeling trained on original and reversed text for rescoring.

2. Contributions

2.1. Front-end

For frontend processing, we using the baseline frontend system including SAD, SWPE, GSS and MVDR. To imporve the accuracy of SAD, besides the time annotations given by the organizers, we also take advantage of non silence alignments generated by acoustic model.

2.2. Training data augmentation

We applied multiple types of data augmentation to enlarge the data coverage. For worn microphone training data, we choose 2 type of channel selection to do front-end processing (x2). Then the signal is augmented with speed perturbation (x3), volume perturbation (x1), reverberation and noise perturbation (x2).

For multiple array data, we choose 5 types of channel selection to do front-end processing (x5). Then speed perturbation (x2) and volume perterbation (x1) is applied. The channel selection of worn microphone and multiple array training data is listed in tabel 1. 'L' represents the left channel of each worn microphone data is selected, while the 'R' stands for right channel. 'ch1+ch4' means the first and last channel of each multiple array data is selecte, while 'ch2+ch4' stands for the second and last channel, 'ch3+ch4' stands for the third and the last channel. 'all' stands for all 4 channels of each multiple array data is selected. 'ref-array' stands for only all channels of the reference are selected. For training data the reference array is manually set to be array ID of 'U02'. Finally, the training data is composed by the following parts.

- D1) Original worn microphone data.
- D2) Multiple array Data with 5 types of channel selection after frone-end processing along with its augmented data.
- D3) Worn microphone data with 2 types of channel selection along with its augmented data.

These procedure finally result in 940 hours of training data. We have investigated the impact of training data based on official TDNN-F structures. Table 2 shows the effect of data augmenation.

Table 1: *channel seletion of training data*

worn	**mutiple array**
	ch1+ch4
L	ch2+ch4
	ch3+ch4
L+R	all
	ch1
	ref-array

Table 2: *Comparison of acoustic models trained with different data*

Data	**dev**
baseline	51.73
D1+D2	46.48
D1+D2+D3	45.87

2.3. Acoustic models

In the back-end, we use 3 different kinds of acoustic models, all trained on LF-MMI criterion using kaldi toolkit. The asr system include TDNN-F (30 layers) nework, CNN-TDNN (11-layer CNN + 20-layer TDNN) trained and CNN-TDNN-LSTM. The model architecture of CNN-TDNN-LSTM model is shown in figure 1. TDNN-F network is trained with offical MFCC features and 100-dimension online ivector. CNN-TDNN is trained with 80-dimensional logmel-filterbank (LMFB) features and online ivector. CLDNN is trained with MFCC, LMFB and online ivector feautures. The 3 models are first trained

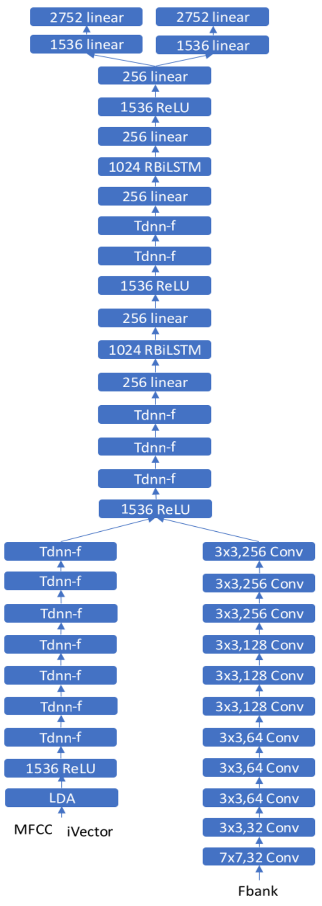

Figure 1: *architecture of CNN-TDNN-LSTM network.*

with full dataset (D1+D2+D3) with LF-MMI criterion, and further fine tuned with sMBR criterion on small training dataset (D1+D2). The 3 acoustic models show strong complementarity when fused together. The comparision of the performance of the 3 models is shown in table 3.

Table 3: *Comparison of network structures*

Model structure	**dev**
baseline	45.87
TDNN-F(30)	45.19
+sMBR	44.69
CNN-TDNN	44.48
+sMBR	44.05
CNN-TDNN-LSTM	44.97
+sMBR	44.06

2.4. decoding

As for development and evaluation data, we choose 1 types of channel selection (namely 'ch1-ch4') to do front-end processing. The enhanced signal is sent to the 3 acoustic models to caculate posterior respectively. We first ensemble the 3 acoustic models via state posterior averaging, and send the averaged posterior to the decoder. Then we do a second pass decoding by introduce non silence alignments generated by the ensemble model to refine activity in the front-end. This time, we choose 4 types of channel selection ('ch1+ch4', 'ch2+ch4', 'ch3+ch4', 'all') to do front-end processing to generate 4 enhanced signal. Again, we decode the 4 signal with model ensemble and get 4 decoding results for each enhanced signal. Finally we use MBR decoding method to combine the results of the 4 enhanced signal. The performance of model ensemble and MBR decoding is shown in tabel 4.

Table 4: *Performance of model ensemble and MBR decoding*

method	**dev**
posterior averaging	41.52
+alignment	39.71
+MBR decoding	37.59

2.5. Language models

We trained recurrent network for language models by using official original and reversed transcription of training data. We prepare two 2-layer LSTM models with forward and backward direction. In the rescoring stage, the language score of offical LM, the forward LSTM and backward LSTM is weighted with 0.4:0.3:0.3. The performance of our language model in rescoring is shown in table 6.

3. Experiment evaluation

Our final results is shown in Table 5 (category A without RNN-LM) and in Table 6 (category B with RNN-LM). Our best system in category A achived 37.59 of WERs, and 35.95 of WERS in category B for development set. In addition, our best system achived 38.99 of WERs in category A, and 37.45 WERs in category B for evaluation set.

Table 5: *WERS for category-A best system without RNN-LM*

Track	Session		DINING	KITCHEN	LIVING	Overall
track1	Dev	S02	41.42	43.35	34.76	
		S09	36.64	34.95	33.45	37.59
	Eval	S01	33.26	53.93	45.95	
		S21	31.74	45.25	31.82	38.99

Table 6: *WERS for category-B best system with RNN-LM*

Track	Session		DINING	KITCHEN	LIVING	Overall
track1	Dev	S02	39.59	41.83	33.52	
		S09	34.66	33.13	31.48	35.95
	Eval	S01	31.69	52.32	44.99	
		S21	30.00	43.52	30.23	37.45