Recognition-Synthesis Based Non-Parallel Voice Conversion with Adversarial Learning

Jing-Xuan Zhang, Zhen-Hua Ling, Li-Rong Dai

National Engineering Laboratory for Speech and Language Information Processing,
University of Science and Technology of China, Hefei, P.R.China

nosisi@mail.ustc.edu.cn, {zhling, lrdai}@ustc.edu.cn

Abstract

This paper presents an adversarial learning method for recognition-synthesis based non-parallel voice conversion. A recognizer is used to transform acoustic features into linguistic representations while a synthesizer recovers output features from the recognizer outputs together with the speaker identity. By separating the speaker characteristics from the linguistic representations, voice conversion can be achieved by replacing the speaker identity with the target one. In our proposed method, a speaker adversarial loss is adopted in order to obtain speaker-independent linguistic representations using the recognizer. Furthermore, discriminators are introduced and a generative adversarial network (GAN) loss is used to prevent the predicted features from being over-smoothed. For training model parameters, a strategy of pre-training on a multi-speaker dataset and then fine-tuning on the source-target speaker pair is designed. Our method achieved higher similarity than the baseline model that obtained the best performance in Voice Conversion Challenge 2018.

Index Terms: voice conversion, recognition-synthesis, adversarial learning

1. Introduction

Voice conversion (VC) aims to modify a source utterance into an output utterance, which sounds as if it is uttered by a target speaker but keeps the linguistic contents unchanged [1, 2]. In recent years, neural networks, such as deep neural networks (DNN) [3, 4], recurrent neural networks (RNN) [5, 6] and sequence-to-sequence (seq2seq) networks [7–9], have been applied to build the acoustic models for voice conversion and achieved great success.

According to the characteristic of training data, VC methods can be roughly categorized into two classes, i.e. parallel VC and non-parallel VC [10]. In parallel VC, an acoustic model is trained with paired source-target acoustic frames or sequences. However, it's difficult to do so in non-parallel VC due to the lack of parallel training data. Many methods have been proposed for non-parallel VC and recognition-synthesis (Rec-Syn) is one of them [11–15]. At the conversion stage of this method, an automatic speech recognition (ASR) model is first employed to extract linguistic-related features, e.g. phonetic posteriorgrams (PPGs) [12] or bottleneck features [14], from the source speech. Then, a synthesis model is applied to predict the acoustic features of the target speaker. However, without explicitly disentangling linguistic and speaker representations, the outputs of the ASR model often contain the information of source speakers, which may harm the similarity of converted voice. Besides, the converted voice often suffers from the over-smoothing issue [16] because the mean square error (MSE) criterion is usually adopted for training the synthesis model.

To overcome these limitations, an adversarial learning method for Rec-Syn based non-parallel VC is presented in this paper. In our method, a recognizer is adopted for extracting linguistic representations and a synthesizer is adopted for predicting the converted acoustic features. When extracting linguistic representations, a speaker adversarial learning loss is employed besides the phoneme recognition loss, thus the linguistic representations are processed to be speaker-agnostic. Also, generative adversarial network (GAN) losses [17] are used in order to alleviate the over-smoothing effect. The WaveNet vocoder [18] is adopted for recovering the waveforms of converted voice. For training model parameters, an external multi-speaker dataset is first adopted for pre-training. Then, the model is adapted to the desired conversion pair by fine-tuning.

Experiments are conducted to compare our method with a Rec-Syn baseline, which achieved the best performance in Voice Conversion Challenge 2018 [14]. The experimental results showed that our proposed method obtained better performance, especially on the similarity of converted speech. Ablation studies were also carried out to demonstrate the effectiveness of several important components in our proposed model.

2. Related Work

Our method is similar to the auto-encoder (AE) based VC with speaker adversarial learning [19–22]. Polyak *et al.* [19] proposed a WaveNet based AE model for VC with a speaker confusion network. Chou *et al.* [20] employed an adversarial trained AE for VC and the voice quality is further improved by another residual generator and discriminator. In both our method and previous studies, the acoustic features are first transformed into speaker-independent representations , which are then decoded back into acoustic features. The main difference between our method and the AE-based VC is that our method utilizes text supervision for building the ASR module and extracting linguistic representations explicitly at training stage. Therefore, our method belongs to the category of Rec-Syn based VC rather than the AE-based one.

3. Proposed Method

3.1. Structure overview

Our model is consist of a recognizer **R** for transforming the acoustic features into linguistic representations, a phoneme

This work was partially funded by the National Natural Science Foundation of China (Grants No. 61871358 and U1613211), National Key R&D Program of China (2017YFB1002202) and the Key Science & Technology Project of Anhui Provinces (18030901016).

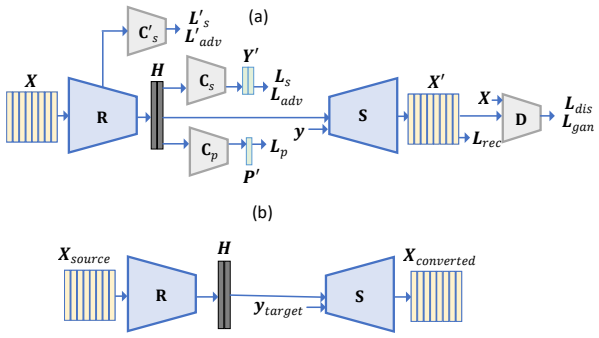


Figure 1: *(a) The diagram of our proposed method at training stage. (b) The conversion process of our proposed method.* $\boldsymbol{X}$*,* $\boldsymbol{H}$ *and* $\boldsymbol{y}$ *represent acoustic features, linguistic representations and speaker label respectively.*

classifier $\mathbf{C}_p$ for phoneme label classification, a speaker classifier $\mathbf{C}_s$ for eliminating speaker information, a synthesizer $\mathbf{S}$ for recovering acoustic features, and discriminators $\mathbf{D}$ for obtaining GAN losses. Figure 1 (a) depicts the overall structure of the proposed method at training stage. During conversion, $\mathbf{C}_p$, $\mathbf{C}_s$ and $\mathbf{D}$ are discarded as shown in Figure 1 (b). Details and training losses of these components are described in the following subsections.

3.2. Recognition process

Linguistic representations are extracted by the recognizer as $\boldsymbol{H} = \mathbf{R}(\boldsymbol{X})$, where $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_x}]$ and N_x are acoustic features and its frame number respectively. $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{N_h}]$ and N_h are linguistic representations and its frame number respectively. The recognizer is built with two-layer bi-directional LSTM interleaved with strided CNN. It decreases the sampling rate of input sequences by 4 times thus we have $N_h = N_x/4$.

With inputs of linguistic representations, the phoneme classifier predicts the sequence of phoneme labels as $\boldsymbol{P}' = \mathbf{C}_p(\boldsymbol{H})$, where $\boldsymbol{P}' = [\boldsymbol{p}'_1, \ldots, \boldsymbol{p}'_{N_p}]$ and N_p is the length of phoneme sequence. $\mathbf{C}_p$ is one-layer LSTM equipped with attention module [23] and auto-regressive connection. A cross-entropy loss is used as

$$L_p = \frac{1}{N_p}\Sigma_{n=1}^{N_p}\mathrm{CE}(\boldsymbol{p}_n, \boldsymbol{p}'_n). \tag{1}$$

The speaker classifier tries to infer the speaker identity from linguistic representations as $\boldsymbol{Y}' = \mathbf{C}_s(\boldsymbol{H})$ frame by frame, where each frame in $\boldsymbol{Y}' = [\boldsymbol{y}'_1, \ldots, \boldsymbol{y}'_{N_h}]$ is the probability distribution of the predicted speaker. It is built with a 3-layer CNN. A cross entropy loss of speaker classification is used for $\mathbf{C}_s$ as

$$L_s = \frac{1}{N_h}\Sigma_{n=1}^{N_h}\mathrm{CE}(\boldsymbol{y}, \boldsymbol{y}'_n), \tag{2}$$

where $\boldsymbol{y}$ represents the ground-truth speaker label encoded as one-hot vector. Meanwhile, the recognizer is trained adversarially to make $\boldsymbol{H}$ speaker-invariant. As suggested in previous studies of learning disentangled representations [24], a speaker adversarial loss is applied to the recognizer as

$$L_{adv} = \frac{1}{N_h}\Sigma_{n=1}^{N_h}\mathrm{MSE}(\frac{1}{|\boldsymbol{y}|}, \boldsymbol{y}'_n), \tag{3}$$

where $|\boldsymbol{y}|$ represents the number of speakers in the training dataset. Therefore, the loss penalizes the distance between prior and predicted distribution of speaker probabilities. To strengthen the adversarial training, a secondary speaker classifier $\mathbf{C}'_s$ is also applied to the outputs of the first LSTM layer in $\mathbf{R}$. And it's also trained with a classification loss L'_s and passes an adversarial loss $L_{adv'}$.

As indicated by Ocal *et al.* [21], the error rate of the optimal speaker classifier relates to an upper bound of mutual information $I(\boldsymbol{y}; \boldsymbol{H})$. In order to approximate the optimal classifier, the speaker classifiers are updated K times for each training step in our experiments.

3.3. Synthesis process

The synthesizer recovers acoustic features from the concatenation of linguistic representations and speaker label as $\boldsymbol{X}' = \mathbf{S}(\boldsymbol{H}, \boldsymbol{y})$, where $\boldsymbol{X}' = [\boldsymbol{x}'_1, \ldots, \boldsymbol{x}'_{N_x}]$. The linguistic and embedded speaker label are repeated to the length of acoustic features and then concatenated as the inputs of the synthesizer. The synthesizer architecture basically follows the decoder in Tacotron model [25, 26]. However, it is connected to the recognizer outputs frame-by-frame rather than utilizing an attention block. The predicted acoustic features are penalized by the MSE loss as

$$L_{rec} = \frac{1}{N_x}\Sigma_{n=1}^{N_x}\mathrm{MSE}(\boldsymbol{x}'_n, \boldsymbol{x}_n). \tag{4}$$

Simply applying the MSE criterion often leads to over-smoothed acoustic features. In order to generate more realistic acoustic features, GAN losses are further incorporated during model fine-tuning. The recognizer-synthesizer module is used as the generator, i.e. $\boldsymbol{X}' = \mathbf{S}(\mathbf{R}(\boldsymbol{X}), \boldsymbol{y})$. Speaker-dependent $\mathbf{D}$ is adopted to classify the natural or generated acoustic features for each speaker. The discriminators are based on 4-layer 1D-CNN followed by a mean pooling layer. Wasserstein GAN with gradient penalty (WGAN-GP) [27,28] is chosen as the objective function in order to stabilize the training process of GAN. The discriminators are trained with the loss as

$$L_{dis} = \mathbf{D}(\boldsymbol{X}') - \mathbf{D}(\boldsymbol{X}) + w_{gp} * (\| \nabla_{\hat{\boldsymbol{X}}}\mathbf{D}(\hat{\boldsymbol{X}}) \|_2 - 1)^2, \tag{5}$$

where w_{gp} represents the weighting factor of GP loss, and $\hat{\boldsymbol{X}}$ represents randomly sampled features by interpolating between $\boldsymbol{X}$ and $\boldsymbol{X}'$. The generator is trained with an adversarial loss

$$L_{gan} = -\mathbf{D}(\boldsymbol{X}'). \tag{6}$$

3.4. Training strategy

The training process of our proposed model includes pre-training on an external multi-speaker dataset and fine-tuning on the pair of source-target speakers. Such design aims to transfer the knowledge learned from large multi-speaker dataset to one pair of speakers. It is expected to increase the model's generalization ability especially when the training data of desired pair is insufficient. Despite that this paper concentrates on the conversion between a pair of two-speakers, our method can be readily extended to multiple speakers for many-to-many VC.

In summary, four kinds of losses are imposed during pre-training. They are the phoneme classification loss L_p, speaker classification losses L_s and $L_{s'}$, adversarial losses L_{adv} and $L_{adv'}$, and the reconstruction loss L_{rec}. L_{adv} and $L_{adv'}$ are scaled by w_{adv} and $w_{adv'}$ respectively. Then losses are added together for training the model. During fine-tuning, two additional speaker embeddings are initialized randomly while

Table 1: *Details of model configurations.*

R	Conv1D-k5s2c512-BN-ReLU-Dropout(0.2) $\rightarrow$ 1 layer BLSTM, 256 cells each direction $\rightarrow$ Conv1D-k5s2c512-BN-ReLU-Dropout(0.2) $\rightarrow$ 1 layer BLSTM, 256 cells each direction $\rightarrow$ $\boldsymbol{H}$
$\mathbf{C}_p$	one layer LSTM, 128 cells with attention
$\mathbf{C}_s$	Conv1D-k5s1c256-BN-LeakyReLU $\times 3$ $\rightarrow$ FC-99-Softmax
S	**Prenet**: FC-256-ReLU-Dropout(0.5) $\times 2$ **RNN**: 2 layer LSTM, 512 cells, 2 frames are predicted each RNN step **Postnet**: Conv1D-k5s1c256-BN-ReLU-Dropout(0.2) $\times 5$ $\rightarrow$ Conv1D-k5s1c80, with residual connection from the input to output
D	Conv1D-k5s2c256-LeakyReLU $\times 3$ $\rightarrow$ Conv1D-k5s2c1 $\rightarrow$ mean pooling

"FC" represents fully connected layer. "BN" represents batch normalization. "Conv1D-kksscc" represents 1-D convolution with kernel size k, stride s and channel c. "$\times N$" represents repeating the block for N times. Structure of **S** follows the decoder in the Tacotron model [25, 26].

Table 2: *MCDs and F_0 RMSEs on test set using training sets of different sizes. Lower is better.*

# of Utt.	rms-to-slt			
	VCC2018		Proposed	
	MCD (dB)	F_0 RMSE (Hz)	MCD (dB)	F_0 RMSE (Hz)
100	3.420	**14.573**	**3.323**	18.675
200	3.411	**15.100**	**3.252**	16.511
300	3.399	**14.207**	**3.246**	17.134
400	3.386	**14.784**	**3.246**	17.357
500	3.376	**15.042**	**3.213**	17.055
# of Utt.	slt-to-rms			
	VCC2018		Proposed	
	MCD (dB)	F_0 RMSE (Hz)	MCD (dB)	F_0 RMSE (Hz)
100	**3.218**	**16.226**	3.286	18.655
200	**3.200**	**15.956**	3.245	17.546
300	3.188	**15.455**	**3.175**	17.638
400	3.179	**15.595**	**3.173**	17.204
500	3.171	**15.771**	**3.147**	17.484

the rest parameters are loaded from the pre-trained model. In addition to the losses applied during pre-training, GAN losses L_{dis} and L_{gan} are further adopted. Here, L_{gan} is first scaled by w_{gan} then added to the total loss.

4. Experiments

4.1. Experimental conditions

One female speaker (slt) and one male speaker (rms) in the CMU ARCTIC dataset[1] were used as the pair of speakers for conversion in our experiments. For each speaker, the evaluation and test set both contained 66 utterances. The non-parallel training set for each speaker contained 500 utterances. Smaller training sets containing 100, 200, 300 and 400 utterances were also constructed by randomly selecting a subset of the 500 utterances for training. The multi-speaker VCTK dataset [29] was utilized for model pre-training. Altogether 99 speakers were selected from VCTK dataset. For each speaker, 10 and 20 utterances were used for validation and testing repsectively. The remaining utterances were used as training samples. The total duration of training samples was about 30 hours.

For acoustic features, 80-dimensional Mel-spectrograms were extracted every 10 ms and then scaled to logarithmic domain. Adam [30] optimizer was used with a learning rate of 0.001. The batch size was 32 and 8 at the pre-training and fine-tuning stage respectively. The weighting factors of adversarial losses were set as $w_{adv} = 100$, $w_{adv'} = 5$ and $w_{adv} = 1, w_{adv'} = 0.1$ during pre-training and fine-tuning respectively. K was set as 2. For the GAN loss, w_{gp} and w_{gan} were set as 10 and 0.05 respectively. After fine-tuning, the accuracy of the speaker classifier on the test sets of slt and rms was 72.2%. In comparison, it was 100.0 % without using adversarial losses. And the accuracy of phoneme classifier was 89.4%.

The details of our model structure are summarized in Table 1. The implementation of WaveNet vocoder followed our previous work [14]. Since this paper focuses on the acoustic models for VC, the same WaveNet vocoders trained with 500 utterances were used when varying the size of data for fine-tuning acoustic models.

We compared our proposed method with a Rec-Syn baseline [14] (i.e., VCC2018)[2]. In this method, bottleneck features were extracted by an ASR model trained on about 3000 hours of external speech data as linguistic descriptions and were used as the inputs of speaker-dependent synthesis models. This method achieved the best performance on the non-parallel VC task of Voice Conversion Challenge 2018.

4.2. Objective evaluation

For objective evaluation, F_0 and 25-dimensional MCCs features were extracted by STRAIGHT [31] from the reconstructed waveforms for evaluation. Then, Mel-cepstrum distortions (MCD) and root mean square error of F_0 (F_0 RMSE) on test set were reported in Table 2.

Compared with the VCC2018 baseline, our proposed method achieved lower MCD except in slt-to-rms conversion given 100 and 200 training utterances. However, for F_0 RMSE metric, the VCC2018 achieved better results compared to the proposed method. This results indicated the potential of further improving F_0 prediction in our proposed method. We should notice that VCC2018 method exploited a large amount of data (i.e., 3000 h) for training the ASR model. On the other hand, the proposed method was pre-trained on much smaller VCTK dataset (i.e., 30 h).

In order to analyze the effects of various strategies used in our model, ablation studies were further conducted. For investigating the effects of speaker adversarial training, we removed the losses of L_{adv} and $L_{adv'}$ (i.e., *"-adv"*). For investigating the effects of phoneme classification, the loss L_p was removed (i.e., *"-phone"*). For investigating the effects of pre-training, the model was initialized randomly before fine-tuning (i.e., *"-pretrain"*). For investigating the effects joint optimization, the recognizer and synthesizer were trained

[1] `http://festvox.org/cmu_arctic/index.html`

[2] Audio samples of our experiments are available at `https://jxzhanggg.github.io/advVC/`.

Table 3: *MCDs and F_0 RMSEs in ablation studies of proposed method. Lower is better.*

Methods	rms-to-slt		slt-to-rms	
	MCD (dB)	F_0 RMSE (Hz)	MCD (dB)	F_0 RMSE (Hz)
Proposed	**3.213**	17.055	**3.147**	**17.484**
-adv	3.967	29.140	3.683	22.929
-phone	3.781	22.232	3.753	20.038
-pretrain	4.228	27.177	3.911	44.790
-joint	3.267	17.223	3.214	17.550
-tunerec	3.444	**16.905**	3.411	18.968
-all	4.287	24.443	3.900	35.969

separately (i.e., *"-joint"*). An experiment was also conducted that fixed the recognizer and only adapted the synthesizer on the target speaker during fine-tuning (i.e., *"-tunerec"*). In analogy to the VCC2018 baseline, a conventional Rec-Syn model was built (i.e., *"-all"*) using the same training data and model structure as those of our proposed method. In this method, the recognizor was first trained with the phoneme classification loss for extracting linguistic features. Then, the synthesizer was pretrained and finetuned on the target speaker.

Table 3 summarizes the results of ablation studies. From the table, we can see that performance of the proposed method degraded without using either the speaker adversarial loss or the phoneme classification loss. When listening to the converted samples for further examination, it's found that the voice converted by *"-adv"* method suffered from low similarity while those converted by *"-phone"* method had low intelligibility. For the *"-pretrain"* method, objective errors increased drastically. And the converted voice was hardly intelligible. Objective errors slightly rose when using the *"-joint"* method. It indicated that training the recognizer and the synthesizer separately leaded to sub-optimal solutions. For the *"-tunerec"* method, the spectral distortion increased and the F_0 error was close to the proposed method. These results indicated fine-tuning the whole model on both source and target data improved the performance of the model. From the last row of the table, we can see the improvement of our proposed method over conventional Rec-Syn method was significant .

Figure 2 (a) and (c) show the Mel-spectrograms of one source utterance and its converted voice using our proposed method respectively. The converted Mel-spectrogram is similar to that of natural reference in Figure 2 (d). Comparing the Mel-spectrogram converted by our proposed method to that converted by the method without GAN loss in Figure 2 (b), we can see that the GAN loss helped to alleviate the over-smoothing problem and to enhance the format structures.

4.3. Subjective evaluation

The "*-all*" method in previous ablation study, the VCC2018 baseline and the proposed method were compared in subjective evaluations. For each experiment, at least thirteen listeners were involved. Samples were presented to them using headphones in random order. They were asked to give a 5-scale opinion score (5: excellent, 4: good, 3: fair, 2: poor, 1: bad) on both similarity and naturalness for each converted utterance. 20 utterances were selected randomly from the test set and two conversion directions (i.e., slt-to-rms and rms-to-slt) were evaluated for each method.

Table 4: *Mean opinion scores with 95% confidence intervals of different methods on test set. Higher is better.*

# of Utt.		*-all*	VCC2018	Proposed
100	Nat.	1.514 ± 0.091	**3.714** ± 0.130	3.628 ± 0.119
	Sim.	1.471 ± 0.086	3.764 ± 0.153	**3.850** ± 0.134
500	Nat.	1.493 ± 0.093	3.636 ± 0.132	**3.950** ± 0.101
	Sim.	1.457 ± 0.088	3.685 ± 0.154	**4.129** ± 0.120

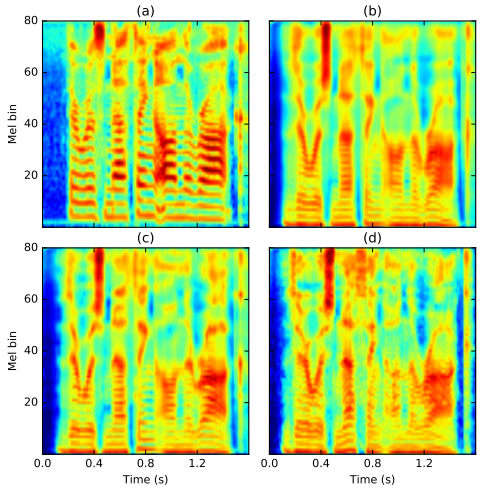


Figure 2: *Mel-spectrograms of (a) a source utterance, (b) the voice converted by our proposed method without GAN loss, (c) the voice converted by our proposed method and (d) the target utterance.*

From Table 4, we can see that the proposed method improved the naturalness and similarity of the *"-all"* method with a large margin. It indicated that our proposed method exploited training data more efficiently with adversarial learning. Our method outperformed the VCC2018 baseline given 500 training utterances of both speakers for fine-tuning, in terms of both naturalness and similarity. In the condition of using 100 training utterances, our method achieved higher similarity while lower naturalness than the VCC2018 method. Despite that our method could obtain better disentangled representations, the VCC2018 baseline learned more fine-grained linguistic descriptions by training on large external corpus. This is especially favorable when the training data of the conversion pair is scarce.

5. Conclusions

In this paper, a method for non-parallel voice conversion is proposed. Our model is based on the recognition-synthesis framework and a speaker classifier module is introduced for speaker adversarial learning. We also incorporate GAN losses for boosting the quality of converted voice. The model is first pre-trained on a multi-speaker dataset then fine-tuned on the desired conversion pair. Both objective and subjective evaluations proved the effectiveness of our method. Our future work will try to further improve the performance of our method by pre-training on larger datasets.

6. References

[1] D. G. Childers, B. Yegnanarayana, and K. Wu, "Voice conversion: Factors responsible for quality," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1985, pp. 748–751.

[2] D. G. Childers, K. Wu, D. M. Hicks, and B. Yegnanarayana, "Voice conversion," *Speech Communication*, vol. 8, no. 2, pp. 147–158, 1989.

[3] S. Desai, E. V. Raghavendra, B. Yegnanarayana, A. W. Black, and K. Prahallad, "Voice conversion using artificial neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 3893–3896.

[4] S. Desai, A. W. Black, B. Yegnanarayana, and K. Prahallad, "Spectral mapping using artificial neural networks for voice conversion," *IEEE Transactions on Audio Speech and Language Processing*, vol. 18, no. 5, pp. 954–964, 2010.

[5] L. Sun, S. Kang, K. Li, and H. Meng, "Voice conversion using deep bidirectional long short-term memory based recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4869–4873.

[6] T. Nakashika, T. Takiguchi, and Y. Ariki, "Voice conversion using RNN pre-trained by recurrent temporal restricted Boltzmann machines," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 580–587, 2015.

[7] J.-X. Zhang, Z.-H. Ling, L.-J. Liu, Y. Jiang, and L.-R. Dai, "Sequence-to-sequence acoustic modeling for voice conversion," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 3, pp. 631–644, 2019.

[8] K. Tanaka, H. Kameoka, T. Kaneko, and N. Hojo, "ATTS2S-VC: Sequence-to-sequence voice conversion with attention and context preservation mechanisms," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6805–6809.

[9] J.-X. Zhang, Z.-H. Ling, and L.-R. Dai, "Non-parallel sequence-to-sequence voice conversion with disentangled linguistic and speaker representations," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, no. 1, pp. 540–552, 2020.

[10] S. H. Mohammadi and A. Kain, "An overview of voice conversion systems," *Speech Communication*, vol. 88, pp. 65–82, 2017.

[11] Huadi Zheng, W. Cai, Tianyan Zhou, Shilei Zhang, and M. Li, "Text-independent voice conversion using deep neural network based phonetic level features," in *International Conference on Pattern Recognition (ICPR)*, Dec 2016, pp. 2872–2877.

[12] L. Sun, K. Li, H. Wang, S. Kang, and H. Meng, "Phonetic posteriorgrams for many-to-one voice conversion without parallel data training," in *2016 IEEE International Conference on Multimedia and Expo (ICME)*, 2016, pp. 1–6.

[13] H. Miyoshi, Y. Saito, S. Takamichi, and H. Saruwatari, "Voice conversion using sequence-to-sequence learning of context posterior probabilities," in *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2017, pp. 1268–1272.

[14] L.-J. Liu, Z.-H. Ling, Y. Jiang, M. Zhou, and L.-R. Dai, "WaveNet vocoder with limited training data for voice conversion," in *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2018, pp. 1983–1987.

[15] S. Liu, J. Zhong, L. Sun, X. Wu, X. Liu, and H. Meng, "Voice conversion across arbitrary speakers based on a single target-speaker utterance," in *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2018, pp. 496–500.

[16] C. M. Bishop, "Mixture density networks," *Technical Report NCRG/4228, Aston University, Birmingham, UK*, 1994.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courvilla, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[18] A. V. Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," in *9th ISCA Speech Synthesis Workshop (SSW9)*, 2016, pp. 125–125.

[19] A. Polyak and L. Wolf, "Attention-based WaveNet autoencoder for universal voice conversion," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6800–6804.

[20] J.-c. Chou, C.-c. Yeh, H.-y. Lee, and L.-s. Lee, "Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations," in *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2018, pp. 501–505.

[21] O. Ocal, O. H. Elibol, G. Keskin, C. Stephenson, A. Thomas, and K. Ramchandran, "Adversarially trained autoencoders for parallel-data-free voice conversion," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 2777–2780.

[22] K. Qian, Z. Jin, M. Hasegawa-Johnson, and G. J. Mysore, "F0-consistent many-to-many non-parallel voice conversion via conditional autoencoder," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6284–6288.

[23] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in Neural Information Processing Systems*, 2015, pp. 577–585.

[24] H. Zhou, Y. Liu, Z. Liu, P. Luo, and X. Wang, "Talking face generation by adversarially disentangled audio-visual representation," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2019, pp. 9299–9306.

[25] Y. Wang, R. J. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio *et al.*, "Tacotron: Towards end-to-end speech synthesis," in *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2017, pp. 4006–4010.

[26] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan *et al.*, "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.

[27] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017, pp. 214–223.

[28] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," in *International Conference on Neural Information Processing Systems*, 2017, pp. 5769–5779.

[29] C. Veaux, J. Yamagishi, K. MacDonald *et al.*, "CSTR VCTK corpus: English multi-speaker corpus for cstr voice cloning toolkit," *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2017.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations, ICLR*, 2015.

[31] H. Kawahara, I. Masuda-Katsuse, and A. D. Cheveigné, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency based F0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, no. 34, pp. 187–207, 1999.

Semi-supervised end-to-end ASR via teacher-student learning with conditional posterior distribution

Zi-qiang Zhang[1], *Yan Song*[1], *Jian-shu Zhang*[1], *Ian McLoughlin*[1,2], *Li-rong Dai*[1]

[1]National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei, China
[2] ICT cluster, Singapore Institute of Technology, Singapore.
zz12375@mail.ustc.edu.cn, {songy, ivm, lrdai}@ustc.edu.cn

Abstract

Encoder-decoder based methods have become popular for automatic speech recognition (ASR), thanks to their simplified processing stages and low reliance on prior knowledge. However, large amounts of acoustic data with paired transcriptions is generally required to train an effective encoder-decoder model, which is expensive, time-consuming to be collected and not always readily available. However unpaired speech data is abundant, hence several semi-supervised learning methods, such as teacher-student (T/S) learning and pseudo-labeling, have recently been proposed to utilize this potentially valuable resource. In this paper, a novel T/S learning with conditional posterior distribution for encoder-decoder based ASR is proposed. Specifically, the 1-best hypotheses and the conditional posterior distribution from the teacher are exploited to provide more effective supervision. Combined with model perturbation techniques, the proposed method reduces WER by 19.2% relatively on the LibriSpeech benchmark, compared with a system trained using only paired data. This outperforms previous reported 1-best hypothesis results on the same task.

Index Terms: semi-supervised learning, ASR, teacher-student learning

1. Introduction

In recent years, attention based encoder-decoder models have shown their ability in many sequence-to-sequence tasks, such as machine translation [1] and automatic speech recognition (ASR) [2, 3]. Despite promising performance, limited availability of paired training data (where both speech records and transcriptions are available) may degrade the effectiveness of encoder-decoder models. However, manual transcription is time-consuming and costly [4], especially for low-resource languages. And thus semi-supervised learning methods, which aim at leveraging large amounts of unpaired data to improve the model's performance, have drawn increasing interest from ASR researchers.

There are several semi-supervised learning based ASR methods in the literature, which will be detailed in Section 2. One typical method is based on teacher-student (T/S) learning [5]. This was first proposed for knowledge distillation where it aims to transfer information from a large highly regularized model into a compact one [5]. In [6, 7], sequence-level T/S learning has been successfully applied to model compression tasks. In the scenario of semi-supervised learning, T/S learning involves first generating supervision information on unlabeled speech from a trained teacher model, and then using this re-paired data to train a student model. The supervision information is usually in the form of an output distribution, making T/S learning different from pseudo-labeling [8], which aims to generate hard labels on unpaired data. Unfortunately, since deriving a sequence-level distribution is intractable due to the huge exploration space involved, it is not easy to apply T/S learning to sequence-to-sequence tasks such as ASR [6, 7]. In [9], 1-best hypothesis approximation based T/S learning was applied for rare word ASR, which is actually similar to pseudo-labeling [10]. Then in [6, 7], 1-best hypothesis and N-best hypotheses were adopted. From [11], it was found that N different 1-best hypotheses obtained from N's decoding with dropout configurations was beneficial. The key point of sequence-level T/S learning is to find an effective approximation of the sequence distribution produced by the teacher model.

In this work, a novel T/S learning method is proposed for encoder-decoder based semi-supervised ASR. Unlike 1-best or N-best hypotheses based T/S learning methods, a new type of supervision for training the student model is presented. Specifically, the teacher model is used to generate two kinds of information concerning label through beam search: 1) 1-best hypothesis just like the previous works [6, 9]; and 2) the sequence of conditional posterior distribution through the 1-best decoding path, which indicates the confidence of the produced symbols during decoding. In the remainder of this paper, the 1-best hypothesis and conditional posterior distribution are named hard- and soft-labels respectively. Exploiting these two levels of information may provide an effective approximation of sequence distribution, and help the student model to tolerate more uncertainty in the unpaired data.

In practice, the effectiveness of the supervision can reduce when the output distributions of the teacher and student models are very close, especially soon after the student model is initialized by the teacher. To address this issue, data augmentation [12] and random dropout are applied during student model training, aiming to introduce some perturbation in the student's output distribution. This idea is motivated by model-consistency loss [13, 14] in semi-supervised image classification, where data and/or model perturbation plays a key role.

We evaluate the proposed T/S learning method on the LibriSpeech [15] corpus, demonstrating a 19.2% relative word error rate (WER) reduction compared to a system using paired data solely. This result also outperforms the previous 1-best hypothesis based T/S learning method [6, 9, 10] under a like-for-like comparison.

2. Related work

Our approach is related to semi-supervised ASR methods which aim to leverage unpaired speech data. These methods can be divided into the following two categories. The first involves

reconstructing speech data, restricting them similar to the real input. It is usually implemented by chaining an ASR model to a reconstruction network, such as a text-to-speech (TTS) system [16, 17, 18], a text-to-encoder (TTE) model [19], or just the decoder of a TTS [20, 21].

The second category, known as pseudo-labeling [8] or self-training [10], involves first generating transcriptions of unpaired speech records using an existing ASR system, and then using the pseudo-paired data to train a new model. T/S learning actually resembles pseudo-labeling when using only 1-hypothesis approximation when applied to ASR task. Pseudo-labeling has been shown useful in training conventional ASR systems [22, 23] in early years. However very recently, an end-to-end model based self-training method with confidence data selection [10] and local prior matching [24] was studied, achieving state-of-the-art WER recovery rate [10] on the LibriSpeech semi-supervised benchmark. We note that a CTC based semi-supervised ASR system [25] also applied data augmentation and dropout to the student model, but that work was based on online pseudo-labeling with hard labels, and without beam search.

3. T/S learning for end-to-end ASR

3.1. End-to-end attention based ASR model

End-to-end ASR models are designed to directly map variable length speech features to a character or word sequence. State-of-the-art end-to-end ASR models usually consist of an encoder and a decoder equipped with an attention network [26, 27]. Input sequence $X = \{x_t|t = 1, \dots, T'\}$ is first converted to an hidden representation $H = \{h_t|h = 1, \dots, T\}$ by the encoder, where T' and T are the length of input and representation sequence respectively. The hidden representation is then fed into the decoder through an attention mechanism (Att). The decoder is usually a recurrent network (Rnn), outputting the current state s_i from the attention result c_i,

$$c_i = Att(s_{i-1}, H) \tag{1}$$

$$s_i = Rnn(s_{i-1}, y^*_{i-1}, c_i) \tag{2}$$

where y^*_{i-1} is the previous predicted symbol. The i-th output probability is obtained through the softmax of a linear transformation of s_i,

$$p(y_i|y^*_{1:i-1}, X) = Softmax(linear(s_i)) \tag{3}$$

For the next step, y^*_i is sampled from $p(y_i|y^*_{1:i-1}, X)$ to compute the state s_{i+1} in the subsequent iteration of eqn. 2.

In the training phase, a teacher-forcing technique is applied to replace the previous output symbol y^*_{i-1} with the target label $\overline{y}_{i-1}$, in which the probability of the whole sequence $\boldsymbol{y} = \{y_i|i = 1, \dots, L\}$ is

$$p(\boldsymbol{y}|X) = \prod_{i=1}^{L} p(y_i|\overline{y}_{1:i-1}, X) \tag{4}$$

then the cross-entropy (CE) loss is computed at the output of the network,

$$\begin{aligned} \mathcal{L}_{CE} &= -\frac{1}{L}\log\left(p(\boldsymbol{y} = \overline{\boldsymbol{y}}|X)\right) \\ &= -\frac{1}{L}\sum_{i=1}^{L}\log\left(p(y_i = \overline{y}_i|\overline{y}_{1:i-1}, X)\right) \end{aligned} \tag{5}$$

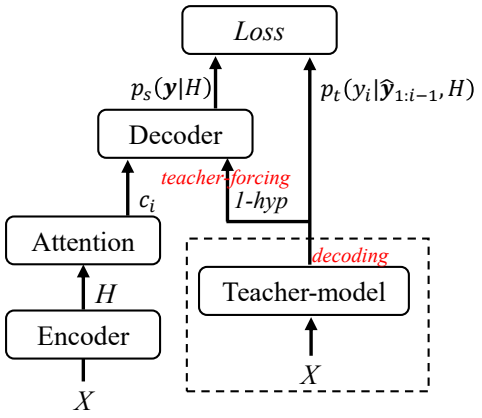


Figure 1: *Model architecture and training process. Two kinds of labels are generated by the teacher model to train the student model: 1-best hypotheses and conditional posterior distributions through the 1-best decoding path.*

Note that the target label sequence $\overline{\boldsymbol{y}} = \{\overline{y}_i|i = 1, \dots, L\}$ is deterministic, the CE loss is used to maximize the log probability of the corresponding symbol at every time-step i.

3.2. Proposed T/S learning via 1-best hypothesis and conditional posterior distribution

Our proposed T/S learning approach involves three stages. Fig. 1 provides a conceptual illustration.

In the first stage, a teacher model is trained with paired data by supervised learning described in Section 3.1. Then in the second stage, the teacher model is used to generate hard and soft labels for all unlabeled speech samples using a beam search algorithm. Fig. 2 illustrates an example of this process across two sequential time steps. At inference time-step i, k prefix-paths have already been sampled from time-step 1 to $i-1$ according to their decoding scores (log posterior probabilities). Based on each possible prefix-path $\hat{\boldsymbol{y}}_{1:i-1}$, the distribution of the next symbol $p_t(y_i|\hat{\boldsymbol{y}}_{1:i-1}, X)$ can be computed, where $y_i \in \mathcal{Z}$, and $\mathcal{Z}$ is the set of all possible output symbols. Subscript t indicates the teacher model.

Pruning is then executed on each distribution to keep k most likely elements, whose correlative symbols are sampled as the suffix, obtaining $k \times k$ paths (k red boxes in the second column in Fig. 2, where each box corresponds to k paths given one prefix). After adding the log probabilities of the chosen symbols to the decoding scores and comparing them, k best final paths are then selected from the $k \times k$ choices. When beam search proceeds along time, for each prefix $\hat{\boldsymbol{y}}_{1:i-1}$, we record $p_t(y_i|\hat{\boldsymbol{y}}_{1:i-1}, X)$ of all possible y_i, called conditional posterior distributions here.

After this procedure completes, we can easily pick the best possible path (1-best hypothesis), $\hat{\boldsymbol{y}} = \{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_L\}$, and the conditional posterior distribution sequence through the path,

$$\{p_t\left(y_i|\hat{\boldsymbol{y}}_{1:i-1}, X\right)|i = 1, \dots, L\} \tag{6}$$

where $\hat{\boldsymbol{y}}_{1:,i-1}$ is the prefix of $\hat{y}_i$.

The final stage is to train the student model. Following [6], we define the sequence-level T/S learning loss as,

$$\mathcal{L}_{T-S} = -\frac{1}{L}\sum_{\boldsymbol{y}\in\mathcal{Y}} p_t(\boldsymbol{y}|X)\log\left(p_s(\boldsymbol{y}|X)\right) \tag{7}$$

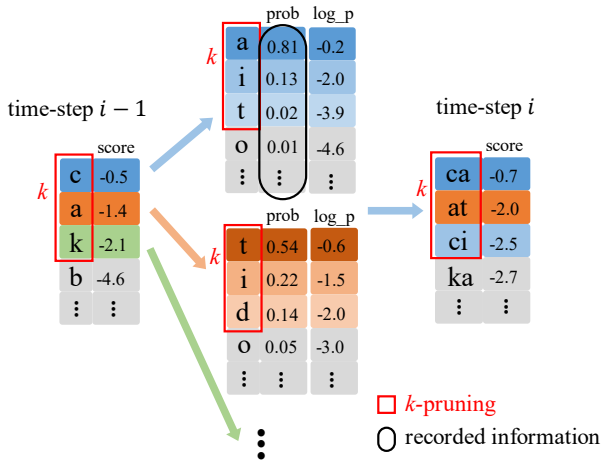


Figure 2: *An example of generating hard and soft labels. Assume the final 1-best hypothesis is **...ca...**, then the conditional posterior distribution recorded at time-step i should be the vector in the black ellipse.*

where subscript s indicates the student model. $\mathcal{Y}$ is the set of all possible paths, which is intractable. Using only 1-best hypothesis $\hat{\boldsymbol{y}}$ is an appropriate approximate of eqn. 7 [6, 9] so,

$$\begin{aligned}\mathcal{L}_{T-S} &\approx -\frac{1}{L}\log\left(p_s(\boldsymbol{y}=\hat{\boldsymbol{y}}|X)\right) \\ &= -\frac{1}{L}\sum_{i=1}^{L}\log\left(p_s(y_i=\hat{y}_i|\hat{y}_{1:i-1},X)\right)\end{aligned} \tag{8}$$

Note that eqn. 8 would be the same as eqn. 5, if we replaced $\hat{\boldsymbol{y}}$ with the true label $\overline{\boldsymbol{y}}$, maximizing the log probability of symbol $\hat{y}_i$ at every time-step. But here we suggest a further improvement. Specifically, the 1-best hypothesis is used for teacher forcing as usual, but at time-step i we do not maximize the exact probability of the chosen symbol $\hat{y}_i$, instead we minimize the cross entropy between the distribution of y_i and the conditional posterior distribution generated from the teacher,

$$\mathcal{L}_{T-S} = -\frac{1}{L}\sum_{i=1}^{L}\sum_{z\in\mathcal{Z}} p_t(z|\hat{\boldsymbol{y}}_{1:i-1},X)\log\left(p_s(z|\hat{\boldsymbol{y}}_{1:i-1},X)\right) \tag{9}$$

In the case that the student is initialized by the teacher, the output distributions of these two models are very similar, making eqn. 9 less effective. To address this issue, we consider introducing a perturbation to the student output distribution. In this way the supervision information from the teacher is generally more accurate than the output of the student, and is better able to guide the student's training. Several techniques to introduce random perturbation could be chosen at this point, and here we apply spectrum data augmentation [12] and dropout.

4. Experimental setup

In this section, we construct experiments to assess the basic aims of the proposed T/S learning method, namely; (1) that T/S learning can help improve the performance of ASR by leveraging additional unlabeled speech data; (2) that using conditional posterior distribution (eqn. 9) can further improve ASR performance.

4.1. Dataset

Our experiments were conducted on the Wall Street Journal (WSJ) [28] and LibriSpeech [15] corpora. We used different subsets as paired and unpaired training set, following the setting of recent works [17, 10]. Specifically, for WSJ the whole training set *si284* contains about 81 hours of speech records, including a 15-hour subset *si84*. The latter was used as paired data while the remainder of *si284* was used as unpaired data. For LibriSpeech, two clean speech subsets, *train_clean_100* (100 hours) and *train_clean_360* (360 hours), were used as paired data and unpaired data respectively. Validation sets for WSJ and LibriSpeech experiments are the same as [17, 10].

4.2. Model details

The encoder-decoder architectures of the student and teacher models were the same. 83 dimensional input features comprised 80 filter-bank and 3 pitch coefficients. For WSJ experiments, the encoder stacked 4 convolutional layers with 64, 64, 128, 128 channels, and 4 BiLSTM [29] layers each with 800 units, the decoder contained one LSTM layer with 800 units, connected to the encoder by location aware attention [30]. 2×2 pooling was performed after every two convolutional layers to downsample the input sequence. For LibriSpeech experiments, we adopted the same architecture as the LAS-medium model [31], in which the encoder stacked two 3×3 convolutional layers with a stride of 2, four BiLSTM layers with projection of 1024 units, and a decoder containing two LSTM layers with 512 units. Character level symbols were modeled in all of our experiments. Both character error rate (CER) and word error rate (WER) were used for evaluation. AdaDelta [32] was adopted to optimize the training, with the epsilon decay and early stop based on the character error rate over the validation set. Training batch size was set to 30. The decoding beam for WSJ experiments was 30, and for LibriSpeech was 10, both for generating labels on unlabeled data and reporting the performance of the test set. No language model (LM) fusion was used through all of our experiments and networks were implemented based on ESPnet [33].

4.3. Training procedure

A teacher model was first trained by standard supervised learning. Then we used the trained teacher model to generate 1-best hypotheses and conditional posterior distributions for the whole paired & unpaired set. For student model training, our proposed training loss (eqn. 9) was used, and eqn. 8 also performed for comparison. Student models started from random initialization for WSJ experiments, while they were initialized by the teacher model for LibriSpeech experiments.

4.4. Data selection & augmentation

We considered applying a data-selection mechanism to filter out some bad-transcribed data, which is also used in [23, 11, 10]. Specifically, we empirically filtered out 20% of the data according to their decoding scores. Initial experiments showed that it did not improve performance on the LibriSpeech experiments, hence data selection was not applied there.

When training the student model on LibriSpeech, we applied spectrum data augmentation [12] (SpecAugment), specifically, the LD policy in [12]. Dropout was also applied after every projection layer of encoders, with a drop probability of 0.2. Considering the influence of the inherent regularization effect of SpecAugment and dropout, we also trained the supervised baseline with these two techniques.

5. Results and analysis

In this section we present experiments on both WSJ and LibriSpeech. Baseline models trained by supervised learning with paired data are denoted Teacher (T) or Oracle (O), according to the data amount they used, with several variants of student system evaluated to identify separately the effect of difference proposed enhancements.

Table 1: *Performance of ASR models on WSJ eval92 via T/S learning. The result of each student model is the average of 5 runs with different random initialization.*

Model	Train. set	Train. label	Data Sel.	Test CER	Test WER
Teacher (T)	*si84*	transcript	-	9.8	26.4
Student I	*si284*	1-hyp	no	8.7	24.6
Student II	*si284*	1-hyp	yes	8.4	24.1
Student III	*si284*	1-hyp+prob	yes	8.2	23.0
Oracle (O)	*si284*	transcript	-	4.4	12.6

Table 1 presents T/S learning systems that have been trained and evaluated on the WSJ corpus, all without SpecAugment or dropout. We first note that the large performance difference between the T and O systems provides a clear indication of the benefit to the latter of the additional labeled data (around 3-5$\times$).

When we then train a straightforward T/S learning system [6, 9] (eqn. 8) with that additional unpaired data (Student I), we note a small improvement in WER (of around 6.8%) to 24.6%. Applying data selection (Student II), which means removing some bad samples which can bring errors when training, further reduces WER slightly to 24.1%. Now incorporating our proposed training loss (eqn. 9), in Student III, ASR performance improves more – reducing CER and WER to 8.2% and 23.0% ;relative improvements of 16.3% and 12.8% respectively.

We believe that such improvement is gained from replacing hard targets from the 1-best hypotheses with the conditional posterior distributions because it allows more information to be provided to the student model. This helps the student to generalize better, as well as handle occasional erroneous teacher assignments in a more advantageous way.

Table 2: *Results tested on LibriSpeech test_clean.*

Model	Train set	Specaug. Drop.	Train label	Test CER	Test WER
T1	LS-100	$\times$	transcript	7.0	16.4
T2	LS-100	✓	transcript	4.3	10.4
S1	LS-460	$\times$	1-hyp	6.2	14.9
S2	LS-460	$\times$	1-hyp+prob	6.0	14.4
S3	LS-460	✓	1-hyp	3.6	9.1
S4	LS-460	✓	1-hyp+prob	3.3	8.4
O1	LS-460	$\times$	transcript	3.3	8.5
O2	LS-460	✓	transcript	2.3	6.2

The results of the LibriSpeech experiments are now presented in Table 2 for several system variants. Again we see a very large performance difference between baseline teacher and oracle models (T1 and O1 respectively). Also as we saw for the WSJ results, the performance gain on the LibriSpeech corpus of using the proposed loss (eqn. 9), in system S2 compared to the previous method (S1), reduces CER and WER slightly, to 6.0% and 14.4% (which is a relative improvement over the teacher of 14.3% and 12.2% respectively).

Incorporating data augmentation and dropout clearly provides performance gains to all systems. However when this is used to introduce a perturbation to the student model (S4 vs. S2 and S3 vs. S1), ASR performance improves markedly. In fact, it is remarkable to note that the student model trained by our proposed method with SpecAugment and dropout (S4), can match the 3.3% CER performance of the baseline oracle model (without Specaug./Drop., O1).

If we consider these results in terms of relative WER reduction, SpecAugment and Dropout together improve the teacher by 36.6% (T1$\rightarrow$T2) and the oracle by 27.1% (O1$\rightarrow$O2), thanks to the inherent regularization effect. However the student is improved by 41.7% (S2$\rightarrow$S4). This greater degree of improvement is thanks to the perturbation effect.

Table 3: *Comparison of the literature of ASR systems trained with LibriSpeech 100h paired data + 360h unpaired data.*

Model	WER	WER $\downarrow$	WRR
Cycle ASR$\rightarrow$TTE [19]	21.5	14.7%	27.6%
Cycle ASR$\leftrightarrow$TTS [17]	17.5	16.7%	38.0%
Pseudo-labeling [24]	12.57	15.3%	33.2%
Pseudo-labeling (ours, S3)	9.1	12.5%	30.9%
Our proposed system	8.4	19.2%	47.6%

Table 3 further presents results reported in several recent works. All WERs reported here were obtained on *test_clean* without language model (LM) fusion. Note that S3 in Table 2 can be seen as a re-implementation of Pseudo-labeling, without well-designed data selection technique in [10]. The relative WER reduction (WER $\downarrow$) and WER recovery rate (defined in [10]) show that our proposed T/S learning surpasses the existing semi-supervised end-to-end ASR methods. It is worth noting at this point that pseudo label quality can be improved by incorporating a LM [10, 24]. For example, the WER in line 3 of Table 3 can be reduced from 12.57% to 9.62% when fusing with a strong LM to refine better hypotheses [10]. Although not demonstrated in this paper, integrating a LM would also be beneficial to our T/S learning method.

6. Conclusion

In this paper we have investigated T/S learning for semi-supervised end-to-end ASR on an encoder-decoder based model. Instead of generating 1-best hypotheses as targets of unlabeled speech, or using N-best hypotheses, we proposed a novel loss function that makes use of the conditional posterior distribution through a 1-best decoding path. Our proposed method incurs minimum modification to the standard supervised training procedure, and can be easily integrated into existing semi-supervised end-to-end frameworks. Experiments on WSJ and the LibriSpeech corpus demonstrate that our proposed method can outperform previous T/S leaning methods, and provides a new perspective for semi-supervised end-to-end ASR.

7. Acknowledgements

This work was supported in part by National Natural Science Foundation of China (Grant No. U1613211), National Key R&D Program of China (2017YFB1002202), the Leading Plan of CAS (XDC08010200), and the Key Science&Technology Project of Anhui Provinces (18030901016).

8. References

[1] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[2] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[3] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 4945–4949.

[4] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-supervised end-to-end speech recognition," in *Proc. Interspeech 2018*, 2018, pp. 2–6.

[5] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[6] Y. Kim and A. M. Rush, "Sequence-level knowledge distillation," *arXiv preprint arXiv:1606.07947*, 2016.

[7] R. M. Mun'im, N. Inoue, and K. Shinoda, "Sequence-level knowledge distillation for model compression of attention-based sequence-to-sequence speech recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6151–6155.

[8] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *ICML Workshop on Challenges in Representation Learning*, 2013.

[9] B. Li, T. N. Sainath, R. Pang, and Z. Wu, "Semi-supervised training for end-to-end models via weak distillation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2837–2841.

[10] J. Kahn, A. Lee, and A. Hannun, "Self-training for end-to-end speech recognition," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7084–7088.

[11] S. Dey, P. Motlicek, T. Bui, and F. Dernoncourt, "Exploiting semi-supervised training through a dropout regularization in end-to-end speech recognition," *Proc. Interspeech 2019*, pp. 734–738, 2019.

[12] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[13] A. Oliver, A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," in *Advances in Neural Information Processing Systems*, 2018, pp. 3235–3246.

[14] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *arXiv preprint arXiv:2001.07685*, 2020.

[15] 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.

[16] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while speaking: Speech chain by deep learning," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 301–308.

[17] M. K. Baskar, S. Watanabe, R. Astudillo, T. Hori, L. Burget, and J. Černockỳ, "Semi-supervised sequence-to-sequence asr using unpaired speech and text," *Proc. Interspeech 2019*, pp. 3790–3794, 2019.

[18] A. Tjandra, S. Sakti, and S. Nakamura, "End-to-end feedback loss in speech chain framework via straight-through estimator," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6281–6285.

[19] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. Le Roux, "Cycle-consistency training for end-to-end speech recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6271–6275.

[20] S. Karita, S. Watanabe, T. Iwata, M. Delcroix, A. Ogawa, and T. Nakatani, "Semi-supervised end-to-end speech recognition using text-to-speech and autoencoders," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6166–6170.

[21] Y. Ren, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Almost unsupervised text to speech and automatic speech recognition," *arXiv preprint arXiv:1905.06791*, 2019.

[22] K. Yu, M. Gales, L. Wang, and P. C. Woodland, "Unsupervised training and directed manual transcription for lvcsr," *Speech Communication*, vol. 52, no. 7-8, pp. 652–663, 2010.

[23] O. Kapralova, J. Alex, E. Weinstein, P. J. Moreno, and O. Siohan, "A big data approach to acoustic model training corpus selection," in *Fifteenth Annual Conference of the International Speech Communication Association (Interspeech)*, 2014.

[24] W.-N. Hsu, A. Lee, G. Synnaeve, and A. Hannun, "Semi-supervised speech recognition via local prior matching," *arXiv preprint arXiv:2002.10336*, 2020.

[25] Y. Chen, W. Wang, and C. Wang, "Semi-supervised asr by end-to-end self-training," *arXiv preprint arXiv:2001.09128*, 2020.

[26] M. K. Baskar, L. Burget, S. Watanabe, M. Karafiát, T. Hori, and J. H. Černockỳ, "Promising accurate prefix boosting for sequence-to-sequence ASR," in *ICASSP*. IEEE, 2019, pp. 5646–5650.

[27] S. Sabour, W. Chan, and M. Norouzi, "Optimal completion distillation for sequence learning," *arXiv preprint arXiv:1810.01398*, 2018.

[28] D. B. Paul and J. M. Baker, "The design for the wall street journal-based csr corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[30] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[31] K. Irie, R. Prabhavalkar, A. Kannan, A. Bruguier, D. Rybach, and P. Nguyen, "On the choice of modeling unit for sequence-to-sequence speech recognition," *Proc. Interspeech 2019*, pp. 3800–3804, 2019.

[32] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[33] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "Espnet: End-to-end speech processing toolkit," in *Proc. Interspeech 2018*, 2018, pp. 2207–2211.

A Noise-Aware Memory-Attention Network Architecture for Regression-Based Speech Enhancement

Yu-Xuan Wang[1], *Jun Du*[1], *Li Chai*[1], *Chin-Hui Lee*[2], *Jia Pan*[1]

[1]University of Science and Technology of China, Hefei, Anhui, P.R.China
[2]Georgia Institute of Technology, Atlanta, GA, USA

yxwang1@mail.ustc.edu.cn, jundu@ustc.edu.cn, cl122@mail.ustc.edu.cn,
chl@ece.gatech.edu, jiapan@iflytek.com

Abstract

We propose a novel noise-aware memory-attention network (NAMAN) for regression-based speech enhancement, aiming at improving quality of enhanced speech in unseen noise conditions. The NAMAN architecture consists of three parts, a main regression network, a memory block and an attention block. First, a long short-term memory recurrent neural network (LSTM-RNN) is adopted as the main network to well model the acoustic context of neighboring frames. Next, the memory block is built with an extensive set of noise feature vectors as the prior noise bases. Finally, the attention block serves as an auxiliary network to improve the noise awareness of the main network by encoding the dynamic noise information at frame level through additional features obtained by weighing the existing noise basis vectors in the memory block. Our experiments show that the proposed NAMAN framework is compact and outperforms the state-of-the-art dynamic noise-aware training approaches in low SNR conditions.

Index Terms: attention mechanism, memory block, noise-aware training, LSTM-RNN, speech enhancement

1. Introduction

Single-channel speech enhancement (SE) is a widely studied problem in signal processing which aims at enhancing noisy speech to improve speech quality and intelligibility [1]. Notable conventional algorithms include spectral subtraction [2, 3], Wiener filtering [4, 5], MMSE estimator [6, 7], and OM-LSA speech estimator [8]. In recent years, most supervised SE techniques have been based on deep neural network (DNN) architectures [9], which show strong regression capabilities of mapping from the input noisy log-power spectra (LPS) features to the target clean LPS features. Although DNN-based SE algorithms have achieved considerable success, more and more research efforts are made to further improve the speech enhancement performance.

On the one hand, due to the fully-connected structure, DNN cannot fully utilize the relationship between the neighbouring frames under long-term acoustic contexts even with the help of frame expansion [9, 10]. As an alternative, long short-term memory recurrent neural network (LSTM-RNN) makes a full use of the information between the current and the previous frames by adding the memory cells and a series of "gates" to determine the retention and deletion information of previous frames [11, 12]. LSTM-RNN also achieves better generalization at low signal-to-noise ratios (SNRs) than DNN [13, 14]. More recently, inspired by the success of attention models in various sequence-to-sequence learning tasks [15, 16, 17], an attention mechanism can also be added to LSTM-RNN [18] or bidirectional long short term memory (BLSTM) [19] for the SE task. It is proved to have a better generalization ability. Besides LSTM-RNN, other powerful structures, such as convolutional neural network (CNN) [20], convolutional-recurrent neural network (CRNN) [21], generative adversarial network (GAN) [22], have also been proposed.

On the other hand, it is noted that the DNN performance deteriorates when a mismatch exists between the training and testing sets [23]. Many noise types have been added to the training set to resolve this issue in [24], but it cannot always improve the speech quality. Noise-aware training (NAT) attains state-of-the-art noisy speech recognition results on the Aurora-4 task [25], and has been applied successfully to speech enhancement. Static noise aware training (SNAT) predicts the noise information, and appends the same information to each frame by assuming that the noise signal during the whole utterance is stationary [10]. However noise is changing greatly in most realistic environments. Accordingly dynamic noise aware training (DNAT) estimates the noise signal in a dynamic manner, and is able to deal with the non-stationary scenes [26]. Further improvements and deformations, such as post-processing, turning full-band features into sub-band features and interpolating SNAT & DNAT [27], are considered as DNAT extensions. Similarly, an SNR-aware model is adopted to predict SNR levels [28], and speaker-aware denoising autoencoder (SaDAE) predicts the speaker identities [29]. Other studies in [30] and [31] use the framework of denoising auto-encoders (DAE) to learn the transformation, but follow almost the same idea as DNAT.

In this paper, we propose a novel noise-aware memory-attention network (NAMAN) for single-channel speech enhancement. Unlike the way attention model embedded into the backbone of the neural network structure [18], we utilize the attention mechanism in a side branch, which is designed to learn the similarities between the current frame and the existing noise basis vectors in the static memory block instead of the previous frames. The clustered acoustic feature vectors, namely Mel-frequency cepstral coefficients (MFCCs) of noise signals, are extracted as the prior noise information and stored in the memory block. The dynamically predicted noise features are obtained by combining the weights learned from the attention mechanism and MFCCs in the memory block together, and are then attached to the noisy features during training. With the help of memory block, our noise-aware training is carried out jointly with the process of denoising. This one-stage model training design significantly simplifies the complicated two-stage design of DNAT [26]. Moreover, DNAT can achieve a good performance over DNN, but when it turns to LSTM-RNN, which has a more powerful modeling ability on the acoustic context of neighboring frames, the performance gain is less significant. The experimental results show that our NAMAN model can still maintain significant improvements under the LSTM-RNN setting.

2. Proposed Deep NAMAN Architecture

Figure 1 illustrates the NAMAN structure consisting of the main network, the memory block and the attention block. The key of the proposed framework is to generate predicted features incorporating the noise information embedded in the current noisy speech frame by a weighted combination of the noise basis vectors in the memory block. With the help of the attention mechanism and LSTM-RNN, the predicted noise vectors can provide useful information for speech enhancement. The details are elaborated in the following subsections.

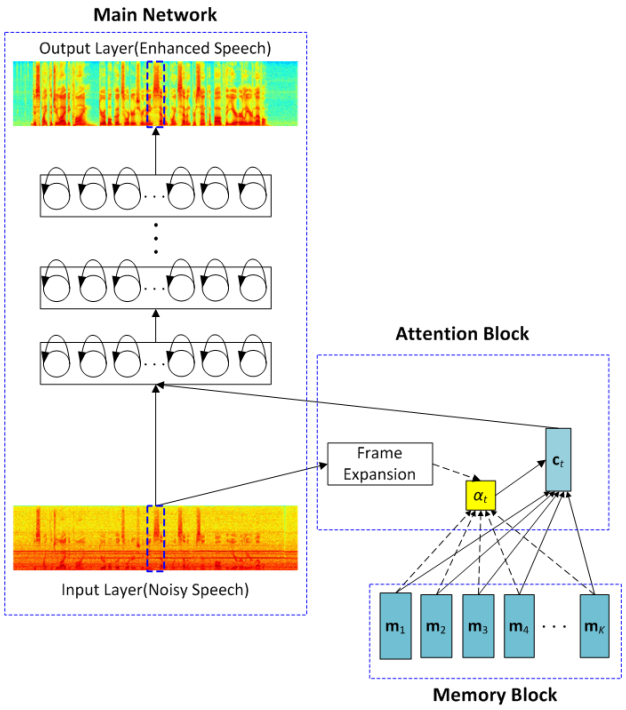


Figure 1: *The structure of NAMAN, including the main network, the attention block and the memory block.*

2.1. Main Speech Enhancement Regression Network

The main network has two effects on the whole framework: denoising and exchanging information with the attention block. Denoising aims to remove the noise from noisy speech to get the enhanced speech. On the other hand, the attention block needs the noise information to pick up the most relevant vectors from the memory block. With the layers increasing, the noise is removed gradually. Here, we append the output from the attention block as auxiliary noise information to the input features to be fed into the NAMAN input layer for subsequent processing.

Given a noisy utterance with T frames, the input noisy LPS features are represented by

$$X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}, \tag{1}$$

where $\mathbf{x}_t$ denotes the noisy feature vector at frame t. Here LSTM-RNN is adopted as the main network for its congenital advantage of sequence representation and temporal contexts acquisition. A detailed calculation in the LSTM-RNN cells is implemented as follows:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i), \tag{2}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f), \tag{3}$$

$$\mathbf{c}_t = \mathbf{f}_t \otimes \mathbf{c}_{t-1} + \mathbf{i}_t \otimes \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c), \tag{4}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o), \tag{5}$$

$$\mathbf{h}_t = \mathbf{o}_t \otimes \tanh(\mathbf{c}_t), \tag{6}$$

where **i**, **f**, **o** represent the "input gate", "forget gate" and "output gate", respectively. **c** is the cell activation vector, and **h** is the hidden vector. **W** and **b** stand for the weight matrices and bias vectors from the cell to gate. σ is the logistic sigmoid function, and $\otimes$ denotes element-wise multiplication. The corresponding outputs of the l-th hidden layer of the main network are:

$$H^l = \{\mathbf{h}_1^l, \mathbf{h}_2^l, ..., \mathbf{h}_T^l\}, \tag{7}$$

where $\mathbf{h}_t^l$ denotes the output of the l-th hidden layer at frame t.

2.2. Noise-Basis Memory Block

The memory block provides the prior noise information for the attention block. It consists of an extensive set of basis vectors, which contain rich noise information and can represent a new noise type by weighted combination. Moreover, the vectors are bound to be quite distinguishable from each other by its corresponding noise type. In view of the random and abrupt nature of noise signals, we adopt the frame-level MFCC features extracted from noise signals.

The procedure of memory block generation is illustrated in Algorithm 1. First, we need to collect different types of noise waveforms from different environments, such as fax machine noises, car idling, footsteps, paper rustling, rain, animal noises, etc. Second, we cut the noise waveforms into frames and extract the MFCC features. Next, based on those noise MFCC feature frames, we can cluster them to form a compact set of K distinguishable noise basis vectors, using a K-means algorithm with a cosine distance shown below:

$$d_{\cos}(\mathbf{n}_i, \mathbf{n}_j) = \frac{\mathbf{n}_i \cdot \mathbf{n}_j}{||\mathbf{n}_i|| \cdot ||\mathbf{n}_j||}, \tag{8}$$

where $d_{\cos}(\mathbf{n}_i, \mathbf{n}_j)$ is exactly the cosine distance between $\mathbf{n}_i$ and $\mathbf{n}_j$, and $\mathbf{n}_i$ stands for the i-th noise feature vector. Finally, the K cluster centers are stored as the memory block defined as:

$$\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_K\}, \tag{9}$$

where $\mathbf{m}_k$ is the k-th noise basis vector.

Algorithm 1 Procedure of Memory Block Generation.

Step1: Noise Sources Collection
collect different noise types as many as possible.
Step2: Feature Extraction
extract frame-level MFCC features from all noise waveforms.
Step3: Clustering
luster all the noise feature vectors into K clusters.
Step4: Memory Block Generation
form the memory block **M** with the K cluster centroids.

It is noted that the memory vectors are static, and should not be updated during either the training or testing step.

2.3. Memory-Aware Attention Block

The attention block is another important part of the whole architecture, it focuses on selecting the basis vectors from the memory block, which are the most relevant to the noise information

embedded in the current speech frame [15]. To gather accurate information for the attention model, not only placing the attention block close to the input, but also performing frame expansion on the input features as follows:

$$\mathbf{f}_t = [\mathbf{x}_{t-\tau}, ..., \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}, ..., \mathbf{x}_{t+\tau}], \tag{10}$$

where $\mathbf{f}_t$ stands for the feature vector after frame expanding at frame t, and τ controls how many history and future frames are involved. Frame expansion is just a simple step which can avoid overfitting and really contribute to collecting noise and responding to mutation of signals.

The attention model takes $\mathbf{f}_t$ and $\mathbf{m}_k$ as input and combines them to a vector with the learned weights. A small neural network is designed to learn the similarity scores between $\mathbf{f}_t$ and $\mathbf{m}_k$, which can be defined by the general formula [32]:

$$e_{t,k} = \mathbf{m}_k^\top \mathbf{W}_a \mathbf{f}_t, \tag{11}$$

where $e_{t,k}$ scores the similarity between $\mathbf{f}_t$ and $\mathbf{m}_k$. The matrix $\mathbf{W}_a$ contains the the attention model parameters. The attention value $\alpha_{t,k}$ is then calculated by $\mathbf{f}_t$ and $\mathbf{m}_k$ through a softmax operation, as shown in the dashed arrows of Figure 1:

$$\alpha_{t,k} = \frac{\exp(e_{t,k})}{\sum_{i=1}^{K} \exp(e_{t,i})}. \tag{12}$$

After normalization, the value $\alpha_{t,k}$ is regarded as a weight, and multiplied by $\mathbf{m}_k$, as shown in the solid arrows of Figure 1:

$$\mathbf{c}_t = \sum_{k=1}^{K} \alpha_{t,k}\, \mathbf{m}_k, \tag{13}$$

where $\mathbf{c}_t$ is the predicted noise vector by the attention model at frame t. So $\mathbf{c}_t$ is a weighted sum of all the basis vectors $\mathbf{m}_k$, and is then concatenated to the input vector:

$$\bar{\mathbf{x}}_t = [\mathbf{x}_t\ \mathbf{c}_t]^\top, \tag{14}$$

where the new vector $\bar{\mathbf{x}}_t$ is fed to the first hidden layer.

2.4. Training and Testing

With the outputs of final LSTM layer shown in Eq. (7), we adopt a linear layer on top of it to generate the outputs of the main network, namely, the LPS features of enhanced speech. Then the parameters of NAMAN are optimized with a minimum mean squared error (MMSE) criterion:

$$E = \frac{1}{T} \sum_{t=1}^{T} ||\hat{\mathbf{s}}_t - \mathbf{s}_t||_2^2, \tag{15}$$

where $\hat{\mathbf{s}}_t$ and $\mathbf{s}_t$ are the t-th LPS feature vectors of estimated and clean reference utterances, respectively.

In the training stage, the parameters in both the main network and the attention block are jointly optimized. In the testing or enhancement stage, the predicted noise vector concatenating to the input of the main network can be obtained by the attention mechanism for each frame to improve the performance of output enhanced speech.

3. Experiments and Result Analysis

3.1. Database

In order to improve the generalization capacity of unseen environments, 958 noise types including 100 noise types [33], 15 home-made noise types and 843 noise types from Free Sound part of the MUSAN corpus [34] were selected as the noise database for training. All 7138 utterances from the training set of WSJ0 corpus were corrupted with the above-mentioned 958 noise types at six levels of SNRs (-5dB, 0dB, 5dB, 10dB, 15dB and 20dB) to build a 36-hour multi-condition training set composed of pairs of clean and noisy speech utterances. Approximately 200 sentences randomly selected from the 36-hour data set were used as the cross-validation set. Similarly, the 330 utterances from the core test set of WSJ0 corpus were used to construct the test set for each combination of noise types and SNR levels (-5dB, 0dB and 5dB). As we only conducted the evaluation of mismatched non-stationary noise types in this study, three unseen noise types, namely Buccaneer1, Destroyer engine and HF channel, were adopted for testing, which were all collected from the NOISEX-92 corpus [35].

3.2. Experimental Setting

As for the front-end, all the speech waveforms were sampled at 16kHz, and the frame length was set to 512 samples with a frame shift of 256 samples. A short-time Fourier transform (STFT) was used to compute the spectra of each overlapping windowed frame. Thus, the 257-dimensional LPS features were produced to train the neural network. Both the input and the reference feature vectors were normalized by global mean and variance before feeding into the networks. In the memory block, the 12-dimensional MFCC features of the noise waveforms in the training set, with their first and second order derivatives, were extracted, and were clustered into 500 classes at the frame level by the K-means algorithm.

For the main network, on top of the input layer there were 2 stacked LSTM layers with projection, each hidden layer had 1024 memory cells and the output layer had 257 units. To make our predictions more accurate, we expanded the input to the attention block 3 frames forward and backward, respectively. All the networks were initialized with random weights. The learning rate for the fine-tuning was set to 0.1 for the first 6 epochs and declined at a rate of 90% after every 6 epochs. Original phase of noisy speech was adopted with the enhanced LPS for the waveform reconstruction.

In this experiment, two other noise-aware models, denoted as SNAT and DNAT, were used for performance comparison. SNAT and DNAT had the same network configurations as our model, i.e. 2 LSTM hidden layers with 1024 cells per layer, other model parameters were consistent with "SNAT" and "DNAT3" in [26]. We also provided the oracle experiment assuming the real noise spectrum was known on the test set as the upper bound, approximatively. The enhancement performance was assessed by using perceptual evaluation of speech quality (PESQ) [36] for measuring speech quality, short-time objective intelligibility (STOI) [37] for measuring speech intelligibility, and log-spectral distortion (LSD) (in dB) [38] for evaluating signal differences in the frequency domain.

3.3. Experimental Results

Table 1 lists the average PESQ, STOI and LSD performance comparison of different models on the test set. "Noisy" denotes noisy speech with no processing. "Mapping" represents

the original LSTM-based regression model using the direct mapping approach without noise-aware training, "SNAT" and "DNAT" refer to "SNAT" and "DNAT3" in [26], respectively. "NAMAN" denotes our proposed approach. "Oracle" means the real noise spectrum is known [26]. Three low SNRs (-5dB, 0dB, 5dB) are selected where the enhancement task is hard and necessary to carry out. Both the two improved models (SNAT, DNAT) outperform the direct mapping system (Mapping) on all the three measures and SNR levels, and severely underperform Oracle, which leaves a lot of room to further improve the performance. Besides, NAMAN performs much better than Mapping, achieving an average PESQ gain of 0.177 (from 2.093 to 2.27), an average STOI gain of 0.029 (from 0.77 to 0.799) and an average LSD decrease of 0.635 (from 4.394 to 3.759). NAMAN also yields better results than SNAT, which can not well handle the non-stationary noise types. What's more, even compared with the powerful model DNAT, NAMAN can also keep the consistent superiority, which is more obvious for low SNRs, PESQ improves from 1.683 to 1.828 with the gain of 0.145, STOI increases from 0.67 to 0.696 with the gain of 0.029 at SNR=-5dB.

Table 1: *Performance comparison on the test set at different SNRs of the three unseen noise environments, among: Noisy, Mapping, SNAT, DNAT, NAMAN and Oracle. Ave denotes the average of three SNRs (-5dB, 0dB and 5dB).*

SNR(dB)		-5	0	5	Ave
PESQ	Noisy	1.300	1.509	1.783	1.531
	Mapping	1.592	2.115	2.572	2.093
	SNAT	1.669	2.196	2.606	2.157
	DNAT	1.683	2.214	2.635	2.177
	NAMAN	1.828	2.304	2.677	**2.270**
	Oracle	2.295	2.681	2.985	2.654
STOI	Noisy	0.596	0.714	0.823	0.711
	Mapping	0.650	0.785	0.875	0.770
	SNAT	0.667	0.800	0.879	0.782
	DNAT	0.670	0.806	0.887	0.788
	NAMAN	0.696	0.814	0.887	**0.799**
	Oracle	0.814	0.879	0.924	0.872
LSD	Noisy	15.826	12.228	9.102	12.385
	Mapping	4.846	4.395	3.941	4.394
	SNAT	4.698	3.992	3.565	4.085
	DNAT	4.664	3.755	3.047	3.822
	NAMAN	4.554	3.691	3.031	**3.759**
	Oracle	3.666	3.320	2.985	3.324

Figure 2 shows an utterance example corrupted by Buccaneer1 noise at SNR=0dB. DNAT successfully removes most of the noise in noisy speech. NAMAN not only reconstructs more speech details compared with DNAT (shown in the dashed rectangular boxes), but also restores more information during high-frequency bands through the whole fragment (shown in the dashed oval boxes). Hence NAMAN can obtain higher scores, which estimates noise by attention mechanism, performs better in speech restoration and achieves less speech distortions.

Table 2 compares the run-time latency and the model size of different models. A set of 500 noisy test utterances are selected randomly and fed to the network to estimate the latency and model size which are normalized by the corresponding values of the Mapping model. From the last two rows we can observe that, for both latency and model size, NAMAN uses only about a half of those values in DNAT.

Table 2: *A comparison among Mapping, SNAT, DNAT and NAMAN. N_T and N_M are the run-time latency and model size, respectively, normalized by Mapping.*

	Mapping	SNAT	DNAT	NAMAN
N_T	1	1.06	2.06	1.03
N_M	1	1.08	2.08	1.03

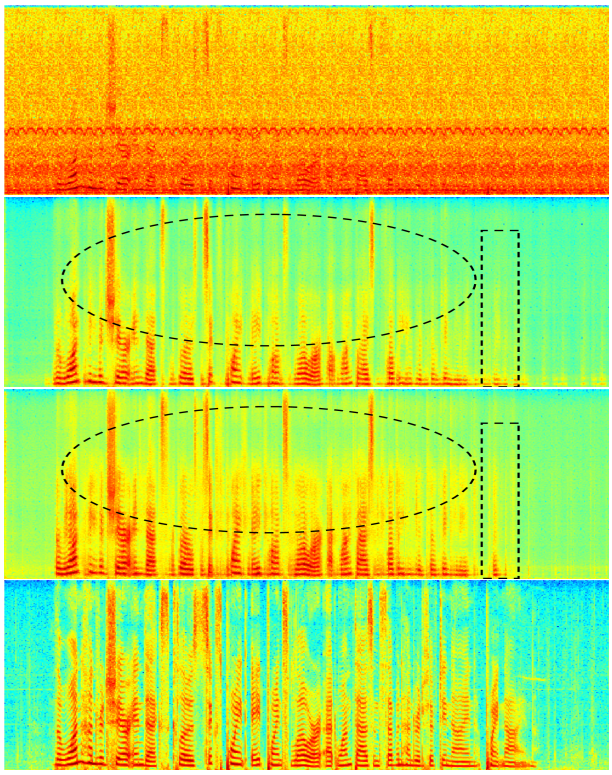


Figure 2: *Spectrograms of an utterance tested on* Buccaneer1 *noise at SNR=0 dB (from top to bottom): noisy speech, DNAT, NAMAN, clean speech.*

4. Conclusion

In this study, we have proposed a novel noise-aware memory-attention framework for regression-based speech enhancement. Compared with the two-stage DNAT model, NAMAN can predict noise information jointly with the denoising process. Experimental results show the proposed NAMAN approach consistently achieves better performances in low SNR conditions, in terms of PESQ, STOI and LSD, than those obtained with DNAT. Furthermore, NAMAN has the distinctive advantages of simple structures and better generalization ability on mismatched conditions. In future work, we plan to expand our model with SNR-aware and speaker-aware training, which may embody complementary capabilities for speech enhancement.

5. Acknowledgements

This work was supported in part by the National Key R&D Program of China under contract No. 2017YFB1002202, the National Natural Science Foundation of China under Grants No. 61671422 and U1613211, the Key Science and Technology Project of Anhui Province under Grant No. 17030901005. This work was also funded by Tencent.

6. References

[1] P. C. Loizou, *Speech enhancement: theory and practice*. CRC press, 2013.

[2] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 27, no. 2, pp. 113–120, 1979.

[3] K. Paliwal, K. Wójcicki, and B. Schwerin, "Single-channel speech enhancement using spectral subtraction in the short-time modulation domain," *Speech communication*, vol. 52, no. 5, pp. 450–475, 2010.

[4] J. Lim and A. Oppenheim, "All-pole modeling of degraded speech," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 3, pp. 197–210, 1978.

[5] J. S. Lim and A. V. Oppenheim, "Enhancement and bandwidth compression of noisy speech," *Proceedings of the IEEE*, vol. 67, no. 12, pp. 1586–1604, 1979.

[6] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE Transactions on acoustics, speech, and signal processing*, vol. 32, no. 6, pp. 1109–1121, 1984.

[7] Y. Ephraim and D. Malah, "Speech enhancement using a minimum mean-square error log-spectral amplitude estimator," *IEEE transactions on acoustics, speech, and signal processing*, vol. 33, no. 2, pp. 443–445, 1985.

[8] I. Cohen and B. Berdugo, "Speech enhancement for non-stationary noise environments," *Signal processing*, vol. 81, no. 11, pp. 2403–2418, 2001.

[9] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal processing letters*, vol. 21, no. 1, pp. 65–68, 2013.

[10] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "A regression approach to speech enhancement based on deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 1, pp. 7–19, 2014.

[11] D. Servan-Schreiber, A. Cleeremans, and J. L. McClelland, "Encoding sequential structure in simple recurrent networks," CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF PSYCHOLOGY, Tech. Rep., 1989.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] F. Weninger, F. Eyben, and B. Schuller, "Single-channel speech separation with memory-enhanced recurrent neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 3709–3713.

[14] F. Weninger, J. R. Hershey, J. Le Roux, and B. Schuller, "Discriminatively trained recurrent neural networks for single-channel speech separation," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2014, pp. 577–581.

[15] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[16] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 4945–4949.

[17] C. Shan, J. Zhang, Y. Wang, and L. Xie, "Attention-based end-to-end speech recognition on voice search," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4764–4768.

[18] X. Hao, C. Shan, Y. Xu, S. Sun, and L. Xie, "An attention-based neural network approach for single channel speech enhancement," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6895–6899.

[19] M. Ge, L. Wang, N. Li, H. Shi, J. Dang, and X. Li, "Environment-dependent attention-driven recurrent convolutional neural network for robust speech enhancement," *Proc. Interspeech 2019*, pp. 3153–3157, 2019.

[20] S. R. Park and J. Lee, "A fully convolutional neural network for speech enhancement," *arXiv preprint arXiv:1609.07132*, 2016.

[21] H. Zhao, S. Zarar, I. Tashev, and C.-H. Lee, "Convolutional-recurrent neural networks for speech enhancement," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2401–2405.

[22] S. Pascual, A. Bonafonte, and J. Serra, "Segan: Speech enhancement generative adversarial network," *arXiv preprint arXiv:1703.09452*, 2017.

[23] D. Liu, P. Smaragdis, and M. Kim, "Experiments on deep learning for speech denoising," in *Interspeech*, 2014, pp. 2685–2689.

[24] Y. Wang and D. Wang, "Towards scaling up classification-based speech separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 7, pp. 1381–1390, 2013.

[25] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 7398–7402.

[26] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "Dynamic noise aware training for speech enhancement based on deep neural networks," in *Interspeech*, 2014, pp. 2670–2674.

[27] Q. Wang, J. Du, L.-R. Dai, and C.-H. Lee, "Joint noise and mask aware training for dnn-based speech enhancement with sub-band features," in *2017 Hands-free Speech Communications and Microphone Arrays (HSCMA)*. IEEE, 2017, pp. 101–105.

[28] S.-W. Fu, Y. Tsao, and X. Lu, "Snr-aware convolutional neural network modeling for speech enhancement." in *Interspeech*, 2016, pp. 3768–3772.

[29] F.-K. Chuang, S.-S. Wang, J.-w. Hung, Y. Tsao, and S.-H. Fang, "Speaker-aware deep denoising autoencoder with embedded speaker identity for speech enhancement," *Proc. Interspeech 2019*, pp. 3173–3177, 2019.

[30] X. Lu, Y. Tsao, S. Matsuda, and C. Hori, "Speech enhancement based on deep denoising autoencoder." in *Interspeech*, 2013, pp. 436–440.

[31] B. Xia and C. Bao, "Speech enhancement with weighted denoising auto-encoder." in *Interspeech*, 2013, pp. 3444–3448.

[32] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[33] G. Hu, "100 nonspeech environmental sounds," *The Ohio State University, Department of Computer Science and Engineering*, 2004.

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

[35] A. Varga and H. J. Steeneken, "Assessment for automatic speech recognition: Ii. noisex-92: A database and an experiment to study the effect of additive noise on speech recognition systems," *Speech communication*, vol. 12, no. 3, pp. 247–251, 1993.

[36] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, vol. 2. IEEE, 2001, pp. 749–752.

[37] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.

[38] J. Du and Q. Huo, "A speech enhancement approach using piecewise linear approximation of an explicit model of environmental distortions," in *Interspeech*, 2008, pp. 569–572.

A Space-and-Speaker-Aware Iterative Mask Estimation Approach to Multi-channel Speech Recognition in the CHiME-6 Challenge

Yan-Hui Tu[1], *Jun Du*[1], *Lei Sun*[1], *Feng Ma*[1], *Jia Pan*[1], *Chin-Hui Lee*[2]

[1]University of Science and Technology of China, Hefei, Anhui, P. R. China
[2]Georgia Institute of Technology, Atlanta, Georgia, USA

{jundu, tuyanhui}@ustc.edu.cn

Abstract

We propose a space-and-speaker-aware iterative mask estimation (SSA-IME) approach to improving complex angular central Gaussian distributions (cACGMM) based beamforming in an iterative manner by leveraging upon the complementary information obtained from SSA-based regression. First, a mask calculated by beamformed speech features is proposed to enhance the estimation accuracy of the ideal ratio mask from noisy speech. Second, the outputs of cACGMM-beamformed speech with given time annotation as initial values are used to extract the log-power spectral and inter-phase difference features of different speakers serving as inputs to estimate the regression-based SSA model. Finally, in decoding, the mask estimated by the SSA model is also used to iteratively refine cACGMM-based masks, yielding enhanced multi-array speech. Tested on the recent CHiME-6 Challenge Track 1 tasks, the proposed SSA-IME framework significantly and consistently outperforms state-of-the-art approaches, and achieves the lowest word error rates for both Track 1 speech recognition tasks.

Index Terms: speech recognition, CHiME-6 Challenge, multi-channel speech enhancement, SSA-IME

1. Introduction

Automatic speech recognition (ASR) in distant-talking scenarios based on the use of microphone arrays has become an important part of everyday life with the emergence of speech-enabled applications on multi-microphone portable devices due to its convenience and flexibility [1]. Many limited tasks were first investigated, such as the TIdigits [2], the TIMIT [3], the Wall Street Journal (WSJ) [4] and LibriSpeech [5] corpora, which do not consider noisy or reverberant conditions. The CHiME (1-4) [6, 7, 8] series were also launched to investigate the effects of background noise in far-field cases, focusing on solving ASR problems in real-world applications. To improve ASR robustness, multi-channel speech enhancement was usually adopted as front-end system. The representative algorithms in this category include multi-channel Wiener filtering [9], blind source separation methods [10, 11, 12, 13], and beamforming methods [14, 15, 16]. Beamforming became a popular approach in the CHiME-3 Challenge [17]. In CHiME-4 Challenge, the best system proposed an approach combining the conventional multi-channel speech enhancement and deep learning methods [18] to improve multi-channel speech recognition.

Recently, the CHiME-5 Challenge provides the first large-scale corpus of real multi-talker conversational speech recorded via commercially available microphone arrays in multiple realistic homes [19]. It essentially congregates a large number of acoustic problems that may exist in real life, which poses a great challenge to existing ASR systems, especially for front-end processing with noisy, reverberant, and overlapping speech. In this challenge, the best system [20] proposed a speaker-dependent speech separation framework, exploiting advantages of both deep learning based and conventional preprocessing techniques. In the latest CHiME-6 Challenge [21], the data set for Track 1 is generated from the CHiME-5 data with array synchronization. The word error rates (WERs) of the worn microphone and multi-array data in the official baseline report are 41.21% and 51.76%, respectively, fully illustrating the difficulty of and issues confronted with the CHiME-6 ASR tasks.

In this paper, we propose a novel space-and-speaker-aware iterative mask estimation (SSA-IME) approach to multi-channel speech recognition in the CHiME-6 Challenge. It aims to improve the complex angular central Gaussian distributions (cACGMM)-based beamforming approach in an iterative manner by leveraging upon the complementary information obtained from space-and-speaker-aware (SSA)-based regression model. Although cACGMM has been recently demonstrated to be quite effective for multi-channel, ASR in operational scenarios, the corresponding mask estimation, however, is not always accurate in multi-talker environments due to the lack of prior or context information. To train this model, we construct a simulated dataset based on the official real multi-channel training data. First, to avoid the impact of noise on accuracy of the ideal ratio mask, the beamformed mask calculated by beamformed features is proposed. Second, The log-power spectral (LPS) and inter-phase difference (IPD) features of different speakers as the input of the proposed SSA model are extracted from the beamformed outputs of cACGMM-based beamforming with time annotation as initial values. These features contain rich space and speaker information which can make the regression model distinguish the different speakers by itself from multi-channel noisy data without any prior information. Finally, the mask estimated by SSA model is also used to refine cACGMM-based mask estimation, yielding an ASR performance improvement. Tested on the recently launched CHiME-6 Challenge Track1 tasks (multiple-array speech recognition), the proposed SSA-IME approach significantly and consistently outperforms the GSS approach [22]. Furthermore, the SSA-IME approach plays a key role in the ensemble system that achieves the best performance in the CHiME-6 Challenge Track 1 tasks.

2. The SSA-IME Framework

The overall SSA-IME framework is shown in Fig. 1. The SSA model is trained using the concatenated features which contain the space and speaker information. To reduce the impact of noise on the accuracy of ideal ratio mask [23], the learning target of the SSA model is calculated by beamformed signals.

The decoding process of SSA-IME is divided into four successive steps, namely, beamforming initialization, SSA-based signal statistics estimation, beamforming, and recogni-

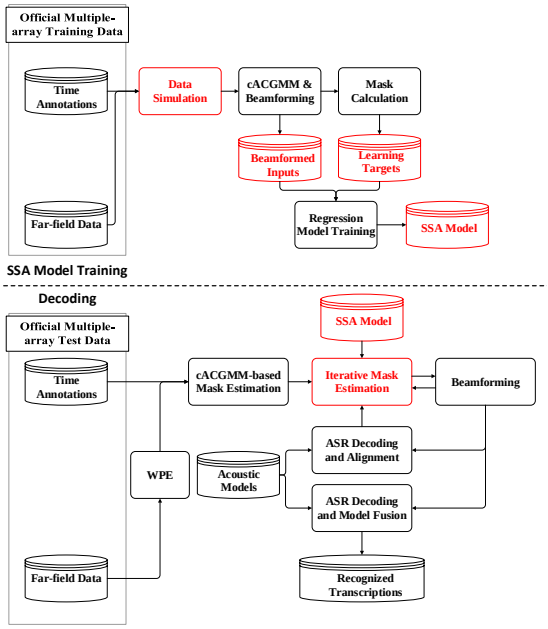


Figure 1: *An illustration of SSA-IME framework.*

tion. First, beamformed speech is initialized and a T-F mask of test speech is obtained by cACGMM-based beamforming [24] using time annotation as initial prior values. Then, the mask estimated by our SSA model is used to improve the initial mask where the SSA model uses the features of the initial beamformed speech. And the ASR-based voice activity detection (VAD) information from the segmentation results of a recognizer with beamformed speech [18] also can be used to improve the initial mask. Next, the improved mask is used as the initial values of the cACGMM-based approach to generate the estimated mask which steers the beamforming, thereby obtaining the beamformed speech for ASR.

2.1. Multi-channel beamforming

At the beginning, we use a weighted prediction error (WPE) [25] algorithm on the multi-channel signals of the reference array, which is commonly used as a dereverberation preprocessor. We use generalized eigenvalue (GEV) beamformer which aims to maximize the signal-to-noise power ratio in the output [26]. Using the information provided by SSA model, a cACGMM is adopted to better estimate the cross-power density matrices in the GEV beamformer, while avoiding the speaker permutation problem.

The goal of a GEV beamformer is to find a linear vector of filter coefficients $\boldsymbol{W}_{\text{GEV}}(f) \in \mathbb{R}^{M\times 1}$ to maximize the signal-to-noise power ratio in each frequency bin [26]:

$$\boldsymbol{W}_{\text{GEV}}(f) = EV\{\boldsymbol{R}_{\boldsymbol{nn}}^{-1}(f)\boldsymbol{R}_{\boldsymbol{ss}}(f)\} \tag{1}$$

where f is the frequency bin index and $EV\{\}$ denotes the eigenvector corresponding to the largest eigenvalue. $\boldsymbol{R}_{\boldsymbol{ss}}(f) \in \mathbb{R}^{M\times M}$ and $\boldsymbol{R}_{\boldsymbol{nn}}(f) \in \mathbb{R}^{M\times M}$ are the cross-power density matrices of the speech and noise terms, respectively. The above cost function has the same form as the Rayleigh coefficient.

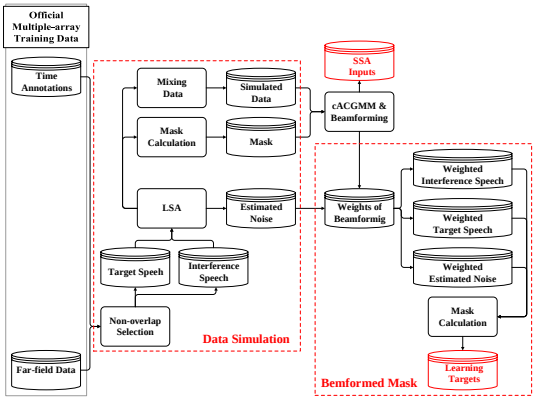


Figure 2: *Training data generation for building SSA models.*

The cross-power density matrices can be defined as:

$$\boldsymbol{R}_{\boldsymbol{vv}}(f) = \sum_{t=1}^{T} M_v(t,f)\boldsymbol{X}(t,f)\boldsymbol{X}^H(t,f) \tag{2}$$

where f is the frequency bin index and t is the frame index. $\boldsymbol{X}(t,f) \in \mathbb{C}^{M\times 1}$ is the observed signal from M microphones of the reference array. v can represent the speech of different speakers or noise class, and $M_v(t,f)$ denotes the probabilities of v in the time-frequency bin (t,f).

Finally, the estimate for the source signal is achieved as:

$$\hat{S}(t,f) = \boldsymbol{W}_{\text{GEV}}^H(f)\boldsymbol{X}(t,f). \tag{3}$$

Obviously, the key of the GEV beamformer is the estimation of time-frequency masks $M_v(t,f)$.

2.2. Training data generation for SSA model

In this section, we will give a detailed description on the training data generation of the SSA model as shown in Fig. 2. Based on the speech analysis, the most challenging part of CHiME-6 is about the dialogue style. Unlike reading speech, the complexity of conversational and spontaneous speech greatly increases the difficulty of a speech recognition system. For instance, casual pronunciation and frequent overlapping speech severely decrease the discriminating ability of acoustic models.

First, to investigate the speech overlapping problem, we excluded non-speech regions and aligned the time stamps of all speakers to locate the overlapped speech regions. The non-overlapped speech of each speaker is obtained by removing the overlapped speech regions from the aligned time stamps of each speaker. According to the introduction of the CHiME-6 dataset, there are a fixed number of four speakers in each session. Therefore, the non-overlapped speech of the four speakers in each session is used for generating training data. And STFT features of these mixed speech are denoted as $\boldsymbol{X}^{\text{T}_1}(t,f), \boldsymbol{X}^{\text{T}_2}(t,f), \boldsymbol{X}^{\text{T}_3}$and $\boldsymbol{X}^{\text{T}_4}(t,f) \in \mathbb{C}^{M\times 1}$, respectively. Note that the four speakers in one session are in turn considered as target speakers. Because the speech is directly selected from the far-field data, it contains much background noise. To reduce the noise influence on data simulation, we first perform single-channel noise estimation and suppression based on Log-Spectral Amplitude Estimator (LSA) [27]. We can obtain the estimated noise and enhanced speech of each speaker as $\hat{\boldsymbol{N}}_{\text{LSA}}^{\text{T}_i}(t,f) \in \mathbb{C}^{M\times 1}$ and $\hat{\boldsymbol{S}}_{\text{LSA}}^{\text{T}_i}(t,f) \in \mathbb{C}^{M\times 1}$, respectively, to calculate:

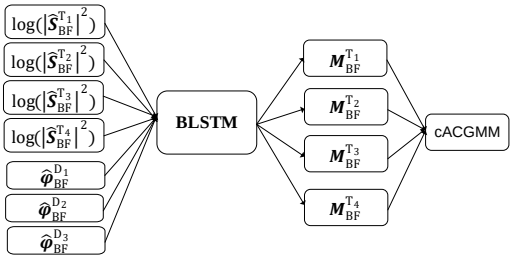


Figure 3: *An illustration of SSA model training.*

$$M_{\mathrm{LSA}}^{\mathrm{T}_i}(t,f) = \frac{\left\|\hat{\boldsymbol{S}}_{\mathrm{LSA}}^{\mathrm{T}_i}(t,f)\right\|_2^2}{\sum_{j=1}^{4}\left\|\hat{\boldsymbol{S}}_{\mathrm{LSA}}^{\mathrm{T}_j}(t,f)\right\|_2^2 + \sum_{j=1}^{4}\left\|\hat{\boldsymbol{N}}_{\mathrm{LSA}}^{\mathrm{T}_j}(t,f)\right\|_2^2} \quad (4)$$

where $\|\cdot\|_2$ denotes the L2 norm of a vector. $M_{\mathrm{LSA}}^{\mathrm{T}_i}$ denotes the mask calculated by LSA-based speech. To generate the simulated data, the enhanced target speech and interference speech are linearly added.

Then, because the enhanced speech, $\hat{\boldsymbol{S}}_{\mathrm{LSA}}^{\mathrm{T}_i}$, is obtained by conventional single-channel speech enhancement, it also contains non-linear residue noises. Accordingly the mask, $M_{\mathrm{LSA}}^{\mathrm{T}_i}$, can not accurately present the speech presence probability, but it can provide more elaborate information at time-frequency bin level comparing to the time annotation at frame level. The mask is just used as the initial value for cACGMM and the outputs of cACGMM-based beamforming are used for SSA model training. And the noise estimated by LSA is directly adopted as the initial value. According to the Eqs. (1) and (2), different mask estimations, $M_{\mathrm{LSA}}^{\mathrm{T}_i}(t,f)$, will result in different beamforming weights, $\boldsymbol{W}_{\mathrm{GEV}}^{\mathrm{T}_i}(f)$, which not only suppress noises but also provide space and speaker information. The beamformed features of each target can be obtained as:

$$\begin{aligned}
\hat{S}_{\mathrm{BF}}^{\mathrm{T}_i}(t,f) &= (\boldsymbol{W}_{\mathrm{GEV}}^{\mathrm{T}_i}(f))^H \hat{\boldsymbol{S}}_{\mathrm{LSA}}^{\mathrm{T}_i}(t,f) \\
\hat{S}_{\mathrm{BF}}^{\mathrm{T}_{ij}}(t,f) &= (\boldsymbol{W}_{\mathrm{GEV}}^{\mathrm{T}_i}(f))^H \hat{\boldsymbol{S}}_{\mathrm{LSA}}^{\mathrm{T}_j}(t,f) \\
\hat{N}_{\mathrm{BF}}^{\mathrm{T}_{ij}}(t,f) &= (\boldsymbol{W}_{\mathrm{GEV}}^{\mathrm{T}_i}(f))^H \hat{\boldsymbol{N}}_{\mathrm{LSA}}^{\mathrm{T}_j}(t,f)
\end{aligned} \quad (5)$$

where $\hat{S}_{\mathrm{BF}}^{\mathrm{T}_i}(t,f)$, $\hat{S}_{\mathrm{BF}}^{\mathrm{T}_{ij}}(t,f)$ and $\hat{N}_{\mathrm{BF}}^{\mathrm{T}_{ij}}(t,f)$ are weighted target speech, weighted interference speech and weighted estimated noise. Finally the learning target of each speaker can be computed as:

$$M_{\mathrm{BF}}^{\mathrm{T}_i}(t,f) = \frac{\left|\hat{S}_{\mathrm{BF}}^{\mathrm{T}_i}(t,f)\right|^2}{\sum_{j=1}^{4}\left|\hat{S}_{\mathrm{BF}}^{\mathrm{T}_{ij}}(t,f)\right|^2 + \sum_{j=1}^{4}\left|\hat{N}_{\mathrm{BF}}^{\mathrm{T}_{ij}}(t,f)\right|^2} \quad (6)$$

2.3. SSA model training

In this section, we will describe the training process of the SSA model in detail. To improve the mask estimation accuracy, a neural-network-based mask estimator learned from a multi-feature concatenation data set is proposed. The beamformed STFT features, $\hat{\boldsymbol{S}}_{\mathrm{BF}}^{\mathrm{T}_i}$, are composed of the elements in Eq. (6). Unlike conventional regression model for mask estimation, the beamformed features of four speakers are used together as the input of the BLSTM-based regression model as shown in Fig. 3. Specifically, $\log(|\hat{\boldsymbol{S}}_{\mathrm{BF}}^{\mathrm{T}_i}|^2)$ $(i = 1, 2, 3, 4)$ denotes the log-power spectral (LPS) features of four speakers on a whole utterance. And $\hat{\boldsymbol{\varphi}}_{\mathrm{BF}}^{\mathrm{D}_j}$ $(j = 1, 2, 3)$ denotes the inter-phase difference (IPD) between a target speaker and three other interfering speakers on a whole utterance, which contains the spatial information between different speakers. Based on the above introduction, the BLSTM-based regression model can learn both space and speaker information at the same time. Therefore, we defined this regression model as space-and-speaker-aware (SSA) model which is also a speaker-independent speech separation model.

To train the BLSTM-based SSA model, the learning targets generated in Section 2.2 are used because they are calculated by beamformed features which are more reliable than the conventional masks. The optimization function of the BLSTM-based model is defined as:

$$E_{\mathrm{SSA}} = \sum_{i=1}^{4}\sum_{t,f}\left(\hat{M}_{\mathrm{SSA}}^{\mathrm{T}_i}(t,f) - M_{\mathrm{BF}}^{\mathrm{T}_i}(t,f)\right)^2 \quad (7)$$

where $\hat{M}_{\mathrm{SSA}}^{\mathrm{T}_i}(t,f)$ and $M_{\mathrm{BF}}^{\mathrm{T}_i}(t,f)$ are the BLSTM estimated mask and the reference mask, respectively. By using E_{SSA}, the model can not only distinguish four speaker as much as possible by taking advantage of the space and speaker information but also yield robust and refined masks. After training, the one single SSA model of all four speakers can be generated.

3. Experiments

3.1. Data corpus

The latest CHiME-6 Challenge provides the first large-scale corpus of real multi-talker conversational speech recorded via commercially available microphone arrays in multiple realistic homes [28]. Speech material is elicited using a dinner party scenario with efforts taken to capture data that is representative of natural conversational speech. The parties have been made using multiple 4-channel microphone arrays and have been fully transcribed. This corpus essentially congregates a large number of acoustic problems that may exist in real life, which poses a great challenge to existing ASR systems, especially for the front-end processing in the case of noise, reverberation, overlapping speech. The CHiME-6 Challenge contains two tracks, namely Track 1 for multiple-array speech recognition and Track 2 for multiple-array diarization and recognition. Here, we focus on Track 1 where annotations can be used to recognize a given test utterance.

3.2. Implementation detail

For front-end configurations, speech waveform is sampled at 16 kHz, and the corresponding frame length is set to 1024 samples (or 64 msec) with a frame shift of 256 samples. The STFT analysis is used to compute the DFT of each overlapping windowed frame. To train the SSA model, the four reference masks were concatenated to the size of 513×4 as the learning targets. Four beamformed LPS features and three IPD features were concatenated to the size of 513×7 as the input. PyTorch was used for neural network training [29]. The learning rate for the first 3 epochs was initialized as 0.01 and then decreased by 90% after each epoch, and the number of epochs was 10. For beamforming, we stack all arrays into one big array according to [30]. The channel selection [31] and online beamforming [32] are also adopted. The CHiME-6 Challenge training set was used as

our training data. BLSTM with 2 hidden layers and 1024 cells for each layer was employed as mask estimator.

For the back-end configurations, the baseline ASR recognition system is trained on the speech recognition toolkit Kaldi [33]. For factorized time delay neural network (TDNN-F) acoustic model training, backstitch optimization method is used. The decoding is based on 3-gram language models with explicit pronunciation and silence probability modeling.

3.3. Results and analysis

Table 1: *WER (%) comparison of official BeamformIt, GSS-based approach and our SSA-IME based approach for multi-channel speech enhancement using baseline ASR system on the development and evaluation set.*

Enhancement	Dataset	DINING	KITCHEN	LIVING	AVG
BeamformIt	Dev	68.54	74.11	65.74	69.48
	Eval	53.69	67.55	64.15	61.19
GSS	Dev	50.61	50.13	45.30	48.34
	Eval	42.18	58.13	48.49	48.89
SSA-IME	Dev	48.35	46.05	42.56	45.23
	Eval	39.36	54.45	45.33	45.71

In Table 1 we show a WER (%) comparison of official BeamformIt, GSS-based and our SSA-IME based approach for multi-channel speech enhancement using the baseline ASR system on the development and evaluation sets.

First, "BeamformIt" [34] and "GSS" [22] are two baseline multi-channel speech enhancements, respectively. "GSS" used a spatial mixture model initiated with time annotations and the ASR-based VAD information from the segmentation results of a recognizer, while "BeamformIt" is a conventional multi-channel beamforming without any prior information. Comparing the two methods, we could find that the "GSS" significantly outperformed the "BeamformIt", e.g., the AVG WERs were significantly reduced from 69.48% to 48.34% and from 61.19% to 48.89% on development and evaluation sets, respectively. Based on the above results, it indicates that the speaker prior information is very important to improve the performance of multi-channel speech enhancement.

Second, "SSA-IME" denotes the proposed method which estimated the mask in an iterative manner from different pieces of complementary information sources, such as, the mask estimated by SSA model and the ASR-based VAD information from the segmentation results of a recognizer, yielding absolute WER reductions of 3.11% and 3.18% over GSS approach on development and evaluation sets, respectively. The proposed SSA-IME framework significantly and consistently outperforms the state-of-the-art GSS approach, and achieves the lowest ASR word error rates for both Track 1A and Track 1B.

To better understand the effectiveness of the proposed SSA-IME approach, an utterance of Speaker P05 selected from Session 02 was illustrated in Fig. 4. In the top panel, the boundaries from different speakers shown with the red areas indicating the target speaker P05 and the blue area denoting the interfering Speaker P07. The spectrograms of speech recorded with channel-1 and worn microphones are plotted in Figs. 4 (b) and (c) respectively. Compared with the spectrogram after BeamformIt shown in Fig. 4(d), speech processed by GSS shown in Fig. 4(e) removed most of the interferences. Though it also retains some residual noises, it shows that GSS greatly improves

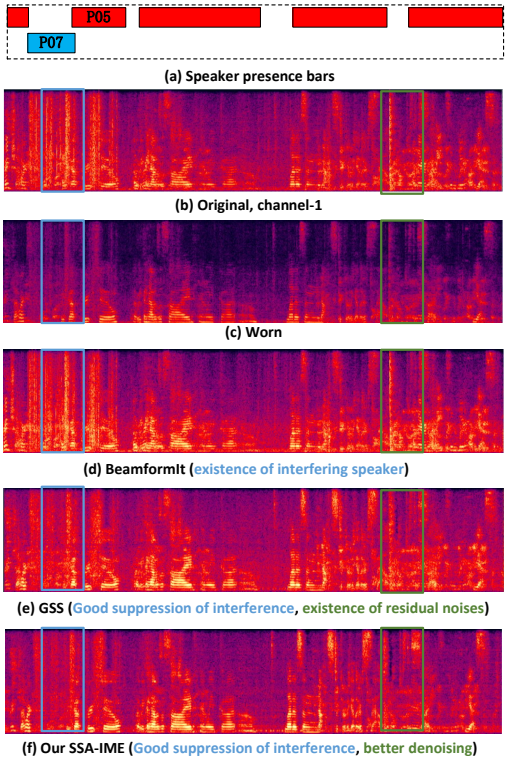


Figure 4: *Spectrogram comparison of an utterance of speaker P05 from Session 02. (d) and (e) are the spectrograms from BeamformIt and GSS methods, respectively. The spectrogram after our final multi-channel beamforming is plotted in (f).*

the speech intelligibility. In Fig. 4(f), the proposed SSA-IME method cannot only removes the interfering speaker well but also have better denoising effect than GSS, yielding a better recognition performance.

4. Summary

In this paper, we have proposed an effective SSA-IME speech preprocessing framework to accurately estimate speech masks in an iterative manner from different pieces of complementary information sources with comprehensive and promising results on a state-of-the-art ASR challenge corpus. By using multi-feature concatenation, the SSA model not only makes a full use of the space and speaker information but also distinguishes different speakers from multi-channel noisy data. In the future, we can improve SSA-IME further by leveraging upon better spatial beamforming approaches, better deep learning architectures for mask estimation, and more informative feedback from the ASR systems. Finally, our back-end acoustic modeling effort, a key to our overall Track 1 ASR system, is described in another companion paper submitted to the same conference.

5. Acknowledgements

This work was supported in part by the National Key R&D Program of China under contract No. 2017YFB1002202, the National Natural Science Foundation of China under Grant Nos. 61671422 and U1613211, the Key Science and Technology Project of Anhui Province under Grant No. 17030901005. This work was also funded by Huawei Noah's Ark Lab.

6. References

[1] L. Deng, G. Hinton, and B. Kingsbury, "New types of deep neural network learning for speech recognition and related applications: an overview," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 8599–8603.

[2] R. Leonard, "A database for speaker-independent digit recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 1984 IEEE International Conference on*, vol. 9, pp. 328–331.

[3] J. Garofolo, "Getting started with the DARPA TIMIT CD-ROM: an acoustic phonetic continuous speech database," 1988.

[4] D. Paul and J. Baker, "The design for the Wall Street Journal-based CSR corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

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

[6] J. Barker, E. Vincent, N. Ma, H. Christensen, and P. Green, "The PASCAL CHiME speech separation and recognition challenge," *Computer Speech & Language*, vol. 27, no. 3, pp. 621–633, 2013.

[7] E. Vincent, J. Barker, S. Watanabe, J. Le R., F. Nesta, and M. Matassoni, "The second 'CHiME' speech separation and recognition challenge: datasets, tasks and baselines," in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*. IEEE, 2013, pp. 126–130.

[8] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' speech separation and recognition challenge: dataset, task and baselines," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*, pp. 504–511.

[9] B. Cornelis, M. Moonen, and J. Wouters, "Performance analysis of multichannel Wiener filter-based noise reduction in hearing aids under second order statistics estimation errors," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 5, pp. 1368–1381, 2011.

[10] P. Comon, "Independent component analysis, a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[11] N. Ono, "Stable and fast update rules for independent vector analysis based on auxiliary function technique," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2011 IEEE Workshop on*, pp. 189–192.

[12] H. Buchner, R. Aichner, and W. Kellermann, "A generalization of blind source separation algorithms for convolutive mixtures based on second-order statistics," *IEEE transactions on speech and audio processing*, vol. 13, no. 1, pp. 120–134, 2005.

[13] L. Wang, H. Ding, and F. Yin, "A region-growing permutation alignment approach in frequency-domain blind source separation of speech mixtures," *IEEE transactions on audio, speech, and language processing*, vol. 19, no. 3, pp. 549–557, 2011.

[14] H. Cox, R. Zeskind, and M. Owen, "Robust adaptive beamforming," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 10, pp. 1365–1376, 1987.

[15] R. Talmon, I. Cohen, and S. Gannot, "Convolutive transfer function generalized sidelobe canceler," *IEEE transactions on audio, speech, and language processing*, vol. 17, no. 7, pp. 1420–1434, 2009.

[16] M. Souden, J. Benesty, and S. Affes, "A study of the LCMV and MVDR noise reduction filters," *IEEE Transactions on Signal Processing*, vol. 58, no. 9, pp. 4925–4935, 2010.

[17] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third ¡®chime¡¯ speech separation and recognition challenge: Dataset, task and baselines," in *Proc. IEEE Automat. Speech Recognition and Understanding Workshop.(ASRU)*, 2015.

[18] Y. Tu, J. Du, L. Sun, F. Ma, H. Wang, J. Chen, and C. Lee, "An iterative mask estimation approach to deep learning based multi-channel speech recognition," *Speech Communication*, vol. 106, pp. 31–43, 2019.

[19] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: dataset, task and baselines," *arXiv preprint arXiv:1803.10609*, 2018.

[20] L. Sun, J. Du, T. Gao, Y. Fang, F. Ma, and C. Lee, "A speaker-dependent approach to separation of far-field multi-talker microphone array speech for front-end processing in the chime-5 challenge," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 4, pp. 827–840, 2019.

[21] S. Watanabe, M. Mandel, J. Barker, and E. Vincent, "Overview of the 6th chime challenge," in *CHiME6 Workshop*, 2020.

[22] C. Boeddeker, J. Heitkaemper, J. Schmalenstroeer, L. Drude, J. Heymann, and R. Haeb-Umbach, "Front-End Processing for the CHiME-5 Dinner Party Scenario," in *CHiME5 Workshop*, 2018.

[23] C. Hummersone, T. Stokes, and T. Brookes, "On the ideal ratio mask as the goal of computational auditory scene analysis," *Blind Source Separation*, pp. 349–368, 2014.

[24] N. Ito, S. Araki, and T. Nakatani, "Complex angular central gaussian mixture model for directional statistics in mask-based microphone array signal processing," *european signal processing conference*, pp. 1153–1157, 2016.

[25] L. Drude, J. Heymann, C. Boeddeker, and R. Haeb-Umbach, "Nara-wpe: A python package for weighted prediction error dereverberation in numpy and tensorflow for online and offline processing," in *ITG 2018, Oldenburg, Germany*, 2018.

[26] E. Warsitz and M. R. Haebumbach, "Blind acoustic beamforming based on generalized eigenvalue decomposition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 5, pp. 1529–1539, 2007.

[27] Y. Ephraim and D. Malah, "Speech enhancement using a minimum mean-square error log-spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, no. 2, pp. 443–445, 1985.

[28] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'chime' speech separation and recognition challenge: Dataset, task and baselines," *INTERSPEECH 2018 – 19th Annual Conference of the International Speech Communication Association*, pp. 1561–1565, 2018.

[29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[30] N. Kanda, C. Boeddeker, J. Heitkaemper, Y. Fujita, S. Horiguchi, and R. Haebumbach, "Guided source separation meets a strong asr backend: Hitachi/paderborn university joint investigation for dinner party asr," *conference of the international speech communication association*, 2019.

[31] K. Wojcicki and P. C. Loizou, "Channel selection in the modulation domain for improved speech intelligibility in noise," *Journal of the Acoustical Society of America*, vol. 131, no. 4, pp. 2904–2913, 2012.

[32] T. Higuchi, N. Ito, T. Yoshioka, and T. Nakatani, "Robust mvdr beamforming using time-frequency masks for online/offline asr in noise," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016.

[33] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society.

[34] X. Anguera, C. Wooters, and J. Hernando, "Acoustic beamforming for speaker diarization of meetings," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2011–2022, 2007.

Using Speech Enhancement Preprocessing for Speech Emotion Recognition in Realistic Noisy Conditions

Hengshun Zhou[1], *Jun Du*[2], *Yan-Hui Tu*[2], *Chin-Hui Lee*[3]

[1]School of Data Science, University of Science and technology of China, Hefei, Anhui, P. R. China
[2]University of Science and technology of China, Hefei, Anhui, P. R. China
[3]Georgia Institute of Technology, Atlanta, GA. USA

zhhs@mail.ustc.edu.cn, jundu@ustc.edu.cn, tuyanhui@ustc.edu.cn, chl@ece.gatech.edu

Abstract

In this study, we investigate the effects of deep learning (DL)-based speech enhancement (SE) on speech emotion recognition (SER) in realistic environments. First, we use emotion speech data to train regression-based speech enhancement models which is shown to be beneficial to noisy speech emotion recognition. Next, to improve the model generalization capability of the regression model, an LSTM architecture with a design of hidden layers via simply densely-connected progressive learning, is adopted for the enhancement model. Finally, a post-processor utilizing an improved speech presence probability to estimate masks from the above proposed LSTM structure is shown to further improves recognition accuracies. Experiments results on the IEMOCAP and CHEAVD 2.0 corpora demonstrate that the proposed framework can yield consistent and significant improvements over the systems using unprocessed noisy speech.

Index Terms: speech emotion recognition, speech enhancement, realistic environments, multiple-target learning, LSTM

1. Introduction

Speech emotion recognition, as an important part of human-computer interaction, has been widely investigated [1, 2, 3]. However, the systems that are trained with clean speech often suffer from a huge performance degradation when tested in a noisy environment due to the mismatch between the train and test conditions [4, 5, 6]. Unfortunately, noise pollution is an indiscernible part of our daily life, caused often by various human activities and other background noise. Therefore, in real world applications, speech enhancement (SE) is a necessary module for emotion recognition system.

Despite recent advances in the field of speech emotion recognition [7, 8, 9], recognizing emotions in noisy environments remains an open research topic [10, 11, 12]. The primary concern of SE for emotion recognition is to remove noise effectively and preserve emotional information in noisy speech. Huang et al. [13] have studied the influence of white Gaussian noise on speaker's emotional states based on Gaussian mixture model (GMM), a typical emotion recognition system. By using algorithm based on spectral substraction and masking properties, they showed that the SE algorithms constantly improved the performance of emotion recognition system under various signal-to-noise ratios (SNRs). In [14], noise robust feature selection with k nearest neighbor (KNN) was found to be beneficial to emotion recogniztion in noisy speech. A front-end voice activity detector (VAD) based unsupervised method to select the frames with a relatively better SNR in the spoken utterances was proposed and shown to be effective in [15]. In [16], effects of different feature types and optimization techniques with different noises or microphone positions for automatic speech emotion recognition have been explored. Authors in [6] compared various front-end techniques for their efficacy in emotion recognition. In terms of the intelligibility of expressive speech in noise, researchers in [17] suggested that the intelligibility of emotion speech in noise was simply related to its audibility as conditioned by the effect that the expression of emotion has on its spectral profile. In [18], an interesting research investigated the performance of two enhancement methods in terms of perceptual quality as well as their impacts on emotion recognition. Furthermore, it demonstrated that quality measures can be an important indicator of enhancement model performance towards emotion recognition.

Although the aforementioned studies have shown the benefit of applying denoising algorithms to noisy speech, there are few studies on emotion recognition in realistic noisy environments. The most important reason may be that there are complex environmental noises and interferences to deal with. Researchers in [19] studied how a scalable deep learning (DL) architecture can be trained to enhance audio signals in a large number of unseen environments and benefitf common emotion recognition pipelines in terms of noise robustness. However the tested noisy data in [19] is still simulated.

In this paper, we investigated deep learning based speech enhancement framework for speech emotion recognition (SER). Specifically, the ideal ratio mask (IRM) estimated by the trained a long short-term memory (LSTM) model was first used for SE. We also find that the SE model trained with emotional corpus could achieve a higher accuracy for SER. To improve the model generalization capability of the regression model, an LSTM architecture with a design of hidden layers via simply densely-connected progressive learning, is adopted for the enhancement model. The proposed architecture further improves the performance of emotion recognition. Finally, considering the complexity of the realistic environment, the proposed improved speech presence probability (ISPP) based post-processing algorithm combined with deep learning by incorporating the estimated progressive ratio mask (PRM) obtained from the progressive learning structure further improves the noise robustness. Synthesized training data pairs generated from the WSJ0 [20] and IEMOCAP databases [21] were used to train SE models. Evaluated on the IEMOCAP and CHEAVD 2.0 databases [22], adopting emotional speech corpus (IEMOCAP) is crucial to SER performance rather than using non-emotional corpus (WSJ0) for both simulated and realistic noisy speech data. Moreover, the progressive learning network combined with ISPP post-processing can yield significant improves for SER on CHEAVD dataset recorded in realistic noisy conditions.

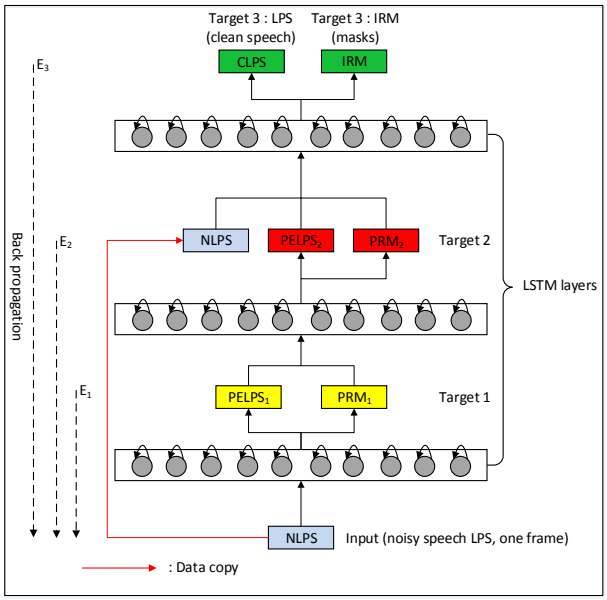


Figure 1: *Architecture of speech enhancement preprocessor.*

2. Speech Enhancement Preprocessing

In daily situations, the speech polluted by background noises may severely destroy the SER performance. Hence, a reliable SE system which can not only effectively suppress the background noise but also retain the emotional information is the key to improve the SER performance. In this study, three strategies used to improve SER performance are investigated.

2.1. Training data for speech enhancement

For automatic speech recognition (ASR), in order to ensure the effectiveness of system, a large number of data must be collected to cover all acoustic changes in speech recognition applications, such as speakers, background noises, different effects of microphone and communication channels, even different recognition tasks, etc. Training data is also important for speech emotion recognition. In [23], generalization involving the target persons speech samples and prior knowledge about their emotional content are investigated. In [24], the authors proposed an adversarial learning framework to alleviate the culture influence on emotion recognition. The effect of gender bias in speech emotion recognition has also been studied in [25]. The preprocessor for speech emotion recognition, which is different from the traditional SE, needs to remove the noise without destroying the emotion clues of speech as much as possible. Inspired by this, we found that when the SE system trained with more matching data (corpus with emotional speech), it is helpful to improve the performance of SER from noisy speech compared to using non-emotional speech corpus. It is verified for the settings of both simulated and realistic noisy data.

2.2. A novel progressive multi-target architecture

Deep neural networks (DNNs) and recurrent neural networks (RNNs) have been widely used in speech enhancement for a long time [26]. However, the conventional RNN can not hold information for a long period and the optimization of RNN parameters via the back propagation through time (BPTT) faces the problem of the vanishing and exploding gradients [27]. The problems can be well alleviated by the invention of LSTM which introduces the concepts of memory cell and a series of gates to dynamically control the information flow. As shown in Figure 1, all LSTM layers consist of memory cells.

The LSTM-based densely connected progressive learning was proposed by [28] and proved to be effective for speech enhancement. To improve the generalization capability of LSTM architecture, a design of hidden layers via densely connected progressive learning and output layer via multiple-target learning is presented (denoted as LSTM-PL-MTL), as illustrated in Figure 1. The log-power spectra (LPS) features are adopted for network inputs and outputs. The input is noisy LPS (NLPS) features and final output is clean LPS (CLPS) and IRM.

A series of progressive ratio masks (PRMs) are concatenated with the progressively enhanced LPS (PELPS) features together as the learning targets. PRM, to perform a trade-off between noise reduction and speech distortion, is defined as:

$$M_{\mathrm{PRM}}(t,f) = \frac{S(t,f) + N_{\mathrm{T}}(t,f)}{S(t,f) + N_{\mathrm{I}}(t,f)} \tag{1}$$

where $S(t,f)$ represents the power spectrum of the speech signal at the time-frequency (T-F) unit (t,f), $N_{\mathrm{T}}(t,f)$ and $N_{\mathrm{I}}(t,f)$ represent the power spectrum of the noise signals in one PRM target and input noise signals at the T-F unit (t,f) respectively. All the target layers are designed to learn intermediate speech with higher SNRs or clean speech. The multi-task error between the output of target layer k and its ground-truth label is

$$\begin{aligned} E_{\mathrm{MTL}}(k) = \sum_{t,f} [& (\hat{x}_{\mathrm{PELPS}}(k,t,f) - x_{\mathrm{PELPS}}(k,t,f))^2 \\ & + (\hat{M}_{\mathrm{PRM}}(k,t,f) - M_{\mathrm{PRM}}(k,t,f))^2] \end{aligned} \tag{2}$$

where $\hat{x}_{\mathrm{PELPS}}(k,t,f)$ and $x_{\mathrm{PELPS}}(k,t,f)$ are predicted and ground-truth PELPS features of the k^{th} target layer, while $\hat{M}_{\mathrm{PRM}}(k,t,f)$ and $M_{\mathrm{PRM}}(k,t,f)$ are predicted and ground-truth PRM features of the k^{th} target layer. Both $\hat{x}_{\mathrm{PELPS}}(k,t,f)$ and $\hat{M}_{\mathrm{PRM}}(k,t,f)$ are nonlinear functions of PELPS and PRM in preceding target layers. $x_{\mathrm{PELPS}}(k,t,f)$ and $M_{\mathrm{PRM}}(k,t,f)$ can be easily calculated with a predefined SNR gain of target layer k. Please note that PELPS and PRM of target layer 3 correspond to clean LPS (CLPS) features and IRM, respectively. The errors of all target layers are computed in the mean squared error (MSE) sense, and added together to optimize the trainable parameters. In our LSTM-PL-MTL, the dimension of both LPS and PRM (IRM) feature vectors is 257, single frame is used for input, the number of LSTM memory cells in each layer is 1024, and we use 3 target layers (with 10dB SNR gain for both target layer 1 and 2).

2.3. Speech post-processing with ISPP

One advantage of the method based on progressive learning is that there are multiple estimated targets that can be obtained from the network. The different targets can provide rich information for post-processing. Meanwhile, a post-processing approach, the improved speech presence probability (ISPP) combining conventional and deep learning techniques [29, 30] by incorporating the estimated PRM obtained from the proposed structure was employed. By incorporating neural network based mask estimation $\hat{M}_{\mathrm{PRM}}(t,f)$ to define an intermediate item

$$\hat{G}(t,f) = \delta \hat{M}_{\mathrm{PRM}}(t,f) + (1-\delta) G_{\mathrm{ISPP}}(t,f) \tag{3}$$

where $G_{\text{ISPP}}(t, f)$ denotes ISPP-based gain function at T-F unit (t, f) and δ is a weighting factor empirically set to 0.5 in our experiments, see [29] for details.

3. Experiments and Result Analysis

3.1. Databases

7138 utterances of WSJ0 corpus [20] (about 12 hours of reading style speech) from 83 speakers were used to train LSTM-IRM model, denoted as SI-84 training set.

Interactive emotional dyadic motion capture database (IEMOCAP) corpus [21], one of the widely used standard emotional databases on speech emotion recognition, comprises five sessions, each of which includes labeled emotional speech utterances from recordings of dialogs between two actors. There is no actor overlapping between these sessions.

Chinese natural audio-visual emotion database (CHEAVD) 2.0 [22] was collected by capturing clips from films and TV programs and used for multimodal emotion challenge (MEC) 2017 [31]. These clips are not captured in the controlled studio environment, so there might contain background noises, which are very close to real world scenarios. Each speech utterance has one label among eight emotion categories. The SNR distribution was investigated in [32].

3.2. Implementation details

To train SE models, WSJ0 corpus and IEMOCAP that do not include the same speaker in SER system (about 9 hours) are corrupted with CHiME-4 noise at four SNR levels (-5dB, 0dB, 5dB and 10dB) to build a 36-hour training set respectively, consisting of pairs of clean and noisy utterances. For SER system, we conducted experiments on IEMOCAP in mismatched scenarios, i.e clean-training and noisy-testing. We randomly picked out a session (session 3 was used here) and only added noise to the test set, see [33] for details. Four noise types (BUS, CAF, PED and STR) [34] in CHiME-4 challenge were selected as the noise database for simulation. We investigated the performance of our algorithm at SNR levels ranging from -5dB to 15dB, with an interval of 5dB and used the speech utterances from four emotion categories, i.e., happy, sad, angry and neutral.

MEC 2017 is a more challenging task and the labels of the test set are not available. We randomly selected 700 utterances from the training set with a total of 4917 utterances as the validation set and the rest as the new training set, and the validation set of the competition as the new test set for experiments. Attention based fully convolution network [33] is used as SER system for both IEMOCAP and CHEAVD tasks. Please note that the test set are recorded in realistic noisy conditions. Therefore, there are high mismatches between SE model and SER system, such as speaking style and types of background noise. These mismatches make SER system quite challenging for our proposed enhancement approach.

For front-end configurations, we used pytorch to train the SE network. Each stage consists of 6 epochs and 5 stages are used. The learning rate for the first stage was initialized as 0.25 and then decreased by 20% after each stage. The batch size is 8. For the back-end configurations, the SER systems were trained on TensorFlow, referring to [33] for specific parameter.

3.3. Results on simulated test data using IEMOCAP

Deep learning-based IRM estimation was first used for SE. Under clean conditions, we trained the SER system and achieved an accuracy of 71.90% on the test set. Our results are presented in Table 1. 'Noisy' denotes unprocessed noisy speech. 'IRM-WSJ0' and 'IRM-IEM-2' represent LSTM-IRM model trained by WSJ0 and IEMOCAP respectively, where '2' means the number of hidden layers used in LSTM is 2.

Table 1: *The accuracy (%) comparison of using IRM estimation with different hidden layers (with the corresponding 71.90% for clean speech).*

Enhancement	-5dB	0dB	5dB	10dB	15dB
Noisy	48.54	50.00	52.19	55.47	59.12
IRM-WSJ0	47.81	50.37	56.20	60.95	62.04
IRM-IEM-2	52.92	56.57	61.68	65.33	66.79
IRM-IEM-3	47.81	52.19	59.85	63.87	65.33
IRM-IEM-4	46.35	48.91	57.66	63.50	64.60

Our first observation is that the accuracy decreases to a certain extent as CHiME-4 noise is added to the test set in IEMOCAP, 48.54% at -5dB and 59.12% at 15dB. By using the LSTM-IRM enhancement model trained on WSJ0, the SER systems achieve better performance when using enhanced audio compared to using noisy audio in most cases. The only exception is when the test speech is under -5dB. When the enhanced model trained on the data set of IEMOCAP using the same network structure, the performance of SER has a comprehensive improvement. When the number of hidden layers in LSTM is increased, the performance of the SER system decreases. The reason might be that the deeper structures with limited training data lead to the overfitting problem and emotional information is destroyed, which is also the difficulty of SE for SER.

Table 2: *The accuracy (%) comparison of different targets by using LSTM-PL-MTL (with 71.90% for clean speech).*

Enhancement	-5dB	0dB	5dB	10dB	15dB
Noisy	48.54	50.00	52.19	55.47	59.12
PL-MTL [35]	51.46	56.20	59.85	63.50	67.15
T1-LPS	49.27	54.02	60.58	63.87	67.15
T1-PRM	47.08	48.18	54.02	63.50	64.60
T2-LPS	38.32	47.08	54.02	56.20	57.66
T2-PRM	47.81	48.91	56.57	63.87	66.06
T3-LPS	40.88	48.91	50.73	52.92	55.11
T3-IRM	54.38	57.30	62.77	67.15	68.25

To improve the SE model generalization for SER system, we further investigated the SE model structure based on progressive learning which has been successfully applied to speaker diarization in quite challenging realistic environments [36]. As shown in Table 2, we used the structure in [35]. Interestingly, we find that the best performance can be obtained when decoding with Target 3 IRM. However, the original LSTM-PL-MTL model in [35] underperforms LSTM-IRM model (IRM-IEM-2) in Table 1 for most SNR cases. This might be explained as that the dense connections in LSTM-PL-MTL result in very high dimensional intermediate target layers and the overfitting problem under the setting of limited training data.

Nevertheless, with our simplified architecture as shown in Figure 1, almost all dense connections in original LSTM-PL-MTL model [35] are removed with only one connection from the input layer to the final intermediate target layer. From Ta-

ble 2, we can observe the results with the Target 3 IRM (T3-IRM) in our proposed LSTM-PL-MTL perform better than the system in [35] and IRM-IEM-2, across all SNR levels. Among all the learning targets (T1-LPS, T1-PRM, T2-LPS, T2-PRM, T3-LPS, T3-IRM) in our LSTM-PL-MTL model, T3-IRM achieved the best results. The new architecture helps in all cases and the performance gaps between the highest SNRs (10dB and 15dB) and clean condition are small. In the case of low SNR, there will be more residual noises after enhancement, leading to the poor performance of SER. We compared the enhanced speech spectrograms and observed that more distortions destroying the emotion information appeared at low SNRs. These could explain why performance is still far from clean audio even after being enhanced.

3.4. Results on realistic test using CHEAVD

To verify the effectiveness of our proposed SE approach in more realistic conditions, we conducted the experiments on CHEAVD dataset recorded in realistic noisy conditions, detailed results are presented in Table 3 and 4.

Table 3: *The accuracy (%) comparison of using different speech enhancement methods in real situations.*

Noisy	1000h SE [35]	IRM-WSJ0	IRM-IEM
41.58	41.30	40.88	**42.01**

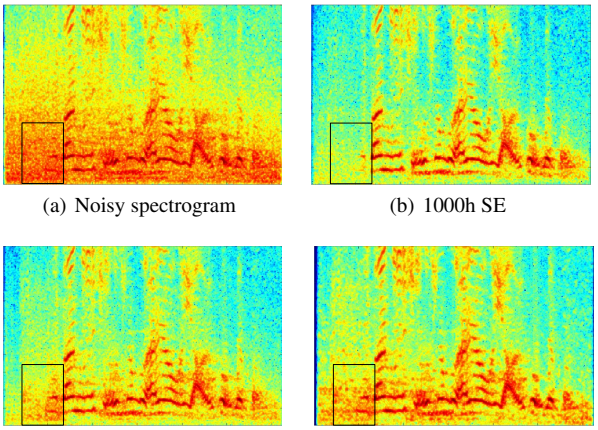
In Table 3, a general SE model trained in corpus of about 1000 hours was first used for comparison [35], and it was found that the performance of SER decreased slightly. Second, IRM-WSJ0 model still degraded the SER performance. The reason may be the high mismatch of speech styles (emotional vs. non-emotional). When IRM-IEM was used, the performance of SER not only exceeded that of 1000h enhanced model, but also exceeds that of unprocessed speech. This is consistent with our observation in the simulation data set.

Table 4: *The accuracy (%) comparison of using ISPP post-processing. "Fusion" means that the score fusion of SER systems with enhanced speech obtained from T1-PRM and corresponding ISPP post-processing.*

T1-PRM-ISPP	T2-PRM-ISPP	T3-IRM-ISPP	Fusion
43.00	42.29	42.72	**44.13**

Considering no significant performance improvements in Table 3, we add to two LSTM layers for each target learning in LSTM-PL-MTL and use the ISPP post-processing in Section 2.3. The results are shown in Table 4 and remarkable improvements of SER performance could be achieved by using the proposed LSTM-PL-MTL structure and post-processing. When using post-processing based on T1-PRM, 43.00% accuracy can be obtained. By score fusion of SER systems with enhanced speech from T1-PRM and its post-processing, the best accuracy of 44.13% is achieved, which yields an absolute 2.55% improvement over the unprocessed noisy speech. This also shows the effectiveness of proposed method.

Finally, to illustrate why the proposed speech preprocessing can help emotion recognition. Figure 2 gives an utterance example from the real test set of CHEAVD 2.0. Figure 2(a) plots the spectrogram of the unprocessed noisy utterance. The girl's



(a) Noisy spectrogram (b) 1000h SE

(c) LSTM-IRM (d) LSTM-PL-MTL + ISPP

Figure 2: *The comparison of SE results of different approaches for an utterance from the real test set of CHEAVD 2.0.*

sad voice was concealed to some extent by the environmental noise, and wrongly classified as "neutral". By using the trained enhancement model, a lot of background noise was removed, and it was also correctly classified as "sad". But it also brings some non-linear distortions to speech, as shown in the spectrogram in the black rectangle of Figure 2(b). A listening inspection on enhanced speech showed that for SE model trained with non-emotional corpus, in addition to removing the noise, sounds like slight "ha ha" and sighs were also destroyed or removed to some extent. Considering that emotion recognition is sensitive to these changes resulting in performance degradations, training SE models using emotional speech data can help recovering these key speech emotion cues as shown in Figure 2(c) and Figure 2(d). Moreover, our proposed LSTM-PL-MTL with T1-PRM and ISPP post-processing made the better tradeoff between noise reduction and speech emotion preservation over LSTM-IRM method.

4. Conclusion

In this paper, we study the effects of speech enhancement as a preprocessor on speech emotion recognition in challenging noisy environments. We first find that speech enhancement models trained with emotion speech is more effective than non-emotion speech. We also observe that important cues, such as low-energy signs and laughters, are often masked by noises and distorted by some enhancement models. We propose training SE models with emotion speech corpora to achieve a higher accuracy for speech emotion recognition. We also present a novel LSTM-PL-MTL architecture with ISPP-based post-processing that proves to be effective in enhancing speech for emotion recognition, achieving a considerable performance improvement over unprocessed noisy speech.

5. Acknowledgement

This work was supported in part by the National Key R&D Program of China under contract No. 2017YFB1002202, the National Natural Science Foundation of China under Grants No. 61671422 and U1613211, the Key Science and Technology Project of Anhui Province under Grant No. 17030901005. This work was also funded by Huawei Noah's Ark Lab.

6. References

[1] X. Wu, S. Liu, Y. Cao, X. Li, and H. M. Meng, "Speech emotion recognition using capsule networks," *IEEE ICASSP2019*, 2019.

[2] Y. Li, T. Zhao, and T. Kawahara, "Improved end-to-end speech emotion recognition using self attention mechanism and multitask learning," *Interspeech 2019*, 2019.

[3] S. Sahu, R. Gupta, and C. Espy-Wilson, "On enhancing speech emotion recognition using generative adversarial networks," *Interspeech 2018*, Sep 2018. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1883

[4] F. Chenchah and Z. Lachiri, "Speech emotion recognition in noisy environment," *International Conference on Advanced Technologies for Signal & Image Processing*, 2016.

[5] M. Pandharipande, R. Chakraborty, A. Panda, and S. K. Kopparapu, "Robust front-end processing for emotion recognition in noisy speech," *International Symposium on Chinese Spoken Language Processing*, 2018.

[6] R. Chakraborty, A. Panda, M. Pandharipande, S. Joshi, and S. K. Kopparapu, "Front-end feature compensation and denoising for noise robust speech emotion recognition," *Interspeech*, 2019.

[7] S. Yoon, S. Byun, S. Dey, and K. Jung, "Speech emotion recognition using multi-hop attention mechanism," *IEEE ICASSP 2019*, 2019.

[8] G. Paraskevopoulos, E. Tzinis, N. Ellinas, T. Giannakopoulos, and A. Potamianos, "Unsupervised low-rank representations for speech emotion recognition," *Interspeech 2019*, 2019.

[9] B. Wang, M. Liakata, H. Ni, T. Lyons, and K. Saunders, "A path signature approach for speech emotion recognition," *Interspeech 2019*, 2019.

[10] F. Weninger, B. Schuller, A. Batliner, S. Steidl, and D. Seppi, "Recognition of nonprototypical emotions in reverberated and noisy speech by nonnegative matrix factorization," *Journal on Advances in Signal Processing*, vol. 2011, no. 1, pp. 1–16, 2011.

[11] R. Xia and Y. Liu, "Using denoising autoencoder for emotion recognition," *INTERSPEECH*, 2013.

[12] E. Parada-Cabaleiro, A. Baird, A. Batliner, N. Cummins, and B. Schuller, "The perception of emotions in noisified nonsense speech," *Interspeech 2017*, 2017.

[13] C. Huang, G. Chen, H. Yu, Y. Bao, and L. Zhao, "Speech emotion recognition under white noise," *Archives of Acoustics*, vol. 38, no. 4, pp. 457–463, 2013.

[14] T. L. Pao, W. Y. Liao, Y. T. Chen, J. H. Yeh, and C. S. Chien, "Comparison of several classifiers for emotion recognition from noisy mandarin speech," *International Conference on Intelligent Information Hiding & Multimedia Signal Processing*, 2007.

[15] M. Pandharipande, R. Chakraborty, A. Panda, and S. K. Kopparapu, "An unsupervised frame selection technique for robust emotion recognition in noisy speech," *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018.

[16] B. Schuller, D. Seppi, A. Batliner, A. Maier, and S. Steidl, "Towards more reality in the recognition of emotional speech," *IEEE International Conference on Acoustics*, 2007.

[17] C. Davis, C. S. Chong, and J. Kim, "The effect of spectral profile on the intelligibility of emotional speech in noise," *Interspeech*, 2017.

[18] A. R. Avila, M. J. Alam, D. D. O'Shaughnessy, and T. H. Falk, "Investigating speech enhancement and perceptual quality for speech emotion recognition," *INTERSPEECH*, 2018.

[19] A. Triantafyllopoulos, G. Keren, J. Wagner, I. Steiner, and B. W. Schuller, "Towards robust speech emotion recognition using deep residual networks for speech enhancement," *Interspeech 2019*, 2019.

[20] J. Garofalo, D. Graff, D. Paul, and D. Pallett, "Csr-i (wsj0) complete," *Linguistic Data Consortium, Philadelphia*, 2007.

[21] C. Busso, M. Bulut, C. C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan, "Iemocap: interactive emotional dyadic motion capture database," *Language Resources & Evaluation*, vol. 42, no. 4, pp. 335–359, 2008.

[22] Y. Li, J. Tao, L. Chao, W. Bao, and Y. Liu, "Cheavd: a chinese natural emotional audiovisual database," vol. 8, no. 6, pp. 913–924, 2017.

[23] L. Chao, J. Tao, M. Yang, and Y. Li, "Improving generation performance of speech emotion recognition by denoising autoencoders," *The 9th International Symposium on Chinese Spoken Language Processing*, pp. 341–344, 2014.

[24] J. Liang, S. Chen, J. Zhao, Q. Jin, H. Liu, and L. Lu, "Cross-culture multimodal emotion recognition with adversarial learning," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4000–4004, 2019.

[25] C. Gorrostieta, R. Lotfian, K. Taylor, R. Brutti, and J. Kane, "Gender de-biasing in speech emotion recognition," *Interspeech*, 2019.

[26] Y. Xu, J. Du, L. Dai, and C. Lee, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal Processing Letters*, vol. 21, no. 1, pp. 65–68, 2014.

[27] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, 2013.

[28] Y.-H. Tu, J. Du, T. Gao, and C.-H. Lee, "A multi-target snr-progressive learning approach to regression based speech enhancement," *Audio, Speech, and Language Processing (accepted), IEEE/ACM Transactions on*, 2020.

[29] Y. Tu, I. Tashev, S. Zarar, and C. Lee, "A hybrid approach to combining conventional and deep learning techniques for single-channel speech enhancement and recognition," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2531–2535, 2018.

[30] Y.-H. Tu, J. Du, and C.-H. Lee, "Speech enhancement based on teacher-student deep learning using improved speech presence probability for noise-robust speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. P-P, no. 99, pp. 1–1, 2019.

[31] Y. Li, J. Tao, B. Schuller, S. Shan, D. Jiang, and J. Jia, "Mec 2017: Multimodal emotion recognition challenge," *2018 First Asian Conference on Affective Computing and Intelligent Interaction (ACII Asia)*, pp. 1–5, 2018.

[32] F. Tao, G. Liu, and Q. Zhao, "An ensemble framework of voice-based emotion recognition system for films and TV programs," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pp. 6209–6213, 2018. [Online]. Available: https://doi.org/10.1109/ICASSP.2018.8461617

[33] Y. Zhang, J. Du, Z. Wang, J. Zhang, and Y. Tu, "Attention based fully convolutional network for speech emotion recognition," *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2018, Honolulu, HI, USA, November 12-15, 2018*, pp. 1771–1775, 2018. [Online]. Available: https://doi.org/10.23919/APSIPA.2018.8659587

[34] E. Vincent, S. Watanabe, A. A. Nugraha, J. Barker, and R. Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech & Language*, vol. 46, pp. 535–557, 2017.

[35] L. Sun, J. Du, X. Zhang, T. Gao, X. Fang, and C. Lee, "Progressive multi-target network based speech enhancement with snr-preselection for robust speaker diarization," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7099–7103, 2020.

[36] L. Sun, J. Du, T. Gao, Y. Lu, Y. Tsao, C. Lee, and N. Ryant, "A novel lstm-based speech preprocessor for speaker diarization in realistic mismatch conditions," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5234–5238, 2018.

Adaptive Speaker Normalization for CTC-Based Speech Recognition

Fenglin Ding, Wu Guo, Bin Gu, Zhenhua Ling, Jun Du

National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China, Hefei, China
{flding,bin2801}@mail.ustc.edu.cn, {guowu,zhling,jundu}@ustc.edu.cn

Abstract

In this paper, we propose a new speaker normalization technique for acoustic model adaptation in connectionist temporal classification (CTC)-based automatic speech recognition. In the proposed method, for the inputs of a hidden layer, the mean and variance of each activation are first estimated at the speaker level. Then, we normalize each speaker representation independently by making them follow a standard normal distribution. Furthermore, we propose using an auxiliary network to dynamically generate the scaling and shifting parameters of speaker normalization, and an attention mechanism is introduced to improve performance. The experiments are conducted on the public Chinese dataset AISHELL-1. Our proposed methods present high effectiveness in adapting the CTC model, achieving up to 17.5% character error rate improvement over the speaker-independent (SI) model.

Index Terms: speaker normalization, speech recognition, connectionist temporal classification

1. Introduction

With the widespread use of deep learning in automatic speech recognition (ASR), recognition accuracy has been greatly improved over the past several years [1, 2]. However, the performance of deep neural network (DNN)-based ASR will still deteriorate under the mismatches between training and test conditions, which are caused by the different characteristics of acoustic variability, such as speakers, channels and environmental noise. In ASR, speaker normalization (SN) techniques are used to minimize the mismatch between the training and testing conditions due to speaker variability. Typical normalization techniques transform the model to match the testing condition or the inputs to match the model.

Speaker normalization techniques for DNNs can be categorized into two broad approaches: adaptation and adaptive training. Speaker adaptation methods address speaker variability by estimating speaker-dependent (SD) parameters from a trained speaker-independent (SI) model on additional adaptation data. Speaker adaptive training attempts to address the speaker mismatch during training on the fly.

For the adaptation method, a straightforward idea is to retrain all SI model parameters. To avoid overfitting, regularization approaches such as L2 regularization using weight decay [3], Kullback-Leibler divergence (KLD) [4] and adversarial multitask learning (MTL) [5] were proposed. There are also many approaches in which only small subsets of the network parameters are adapted [6, 7, 8]. Recently, adaptation schemes using parameterized hidden activation functions have been widely explored [9, 10, 11] and have achieved good improvements.

In adaptive training, a traditional technique is to transform the acoustic features to a normalized space, and then the adapted features are used to train DNN models. Typical methods include MLLR transforms and the feature-space variant (fMLLR) [12, 13]. Another effective method is to provide the network with auxiliary features that characterize speaker information such as i-vector [14, 15, 16] and speaker code [17, 18]. In addition, cluster adaptive training (CAT) has been applied for speaker normalization [19, 20].

Despite the great success of these methods in hybrid systems, there has been limited investigation in speaker normalization for the end-to-end (E2E) ASR. In [21], two regularization-based approaches were shown to be effective for connectionist temporal classification (CTC) [22]-based E2E ASR. In [23], several conventional adaptation methods were integrated to adapt the attention-based encoder-decoder (AED) model.

In this paper, we propose a novel speaker normalization technique for CTC-based ASR. The CTC models take all utterances as input and produce a sequence of activations. These allow us to make use of better context modeling capabilities and statistical information of hidden activations for a speaker. Additionally, inspired by the idea of batch normalization (BN) [24], we propose to normalize each speaker representation independently by making each activation follow the standard normal distribution. The mean and variance of each activation are estimated at the speaker level. Then, a pair of scaling and shifting parameters are introduced to transform the normalized value, which are learned along with the original model parameters. Furthermore, motivated by dynamic layer normalization (DLN) [25] and attentive batch normalization (ABN) [26], we also use an auxiliary network with an attention mechanism to dynamically generate the normalization parameters, which we call adaptive speaker normalization (ASN). However, unlike DLN and ABN, we propose to generate the parameters at the batch level and at the speaker level to fulfill speaker adaptation. We evaluated the proposed algorithms on the AISHELL-1 corpus [27], an open-source Mandarin ASR task. Experimental results show that the proposed methods present high effectiveness in adapting the CTC model, achieving up to 17.5% character error rate (CER) improvement over the speaker-independent (SI) model.

2. Relation to prior work

Well-known normalization techniques for reducing the train-test mismatch include the application of input normalization, such as the mean normalization (MN) [28] and mean and variance normalization (MVN) [29]. MN assumes that the data mean is invariant, and MVN uses the stronger assumption that the mean and variance of data are invariant, so standardizing the mean and/or variance removes irrelevant information [30]. In deep learning, un-normalized features with greater variance dominate the DNN learning process, so scaling the inputs is a standard procedure that can improve DNN performance.

Similar normalization techniques can be found in DNN

training. Batch normalization (BN)[24] and layer normalization (LN) [31] are two well-known methods for normalizing the activations of the hidden layers. BN was originally designed to alleviate the issue of internal covariate shifting, a common problem in DNN training. BN addresses the problem by normalizing each dimension of activations in a mini-batch by making it follow a standard normal distribution. LN has the same idea as batch normalization, but the difference is that LN normalizes each node of a neural layer, which is independent of the size of each batch.

However, BN or LN is a DNN training technique that is not targeted at speaker adaptation. There has still been limited investigation in speaker adaptation using similar normalization techniques. In [32], researchers used the auxiliary network to learn speaker-specific information and then performed normalization at the speaker level. Although the method achieves a lower word error rate (WER) than the unadapted models, it only uses the mean information of activations and performs normalization at a specific layer. The novelty of our proposed methods lies in the following aspects: first, we use the idea of BN to perform normalization for activations at the speaker level, which is layer-wise and makes use of the mean and variance information of activations. Furthermore, we introduce an attention mechanism to the auxiliary network and use it to dynamically generate the normalization parameters. Finally, we investigate our approaches in connectionist temporal classification (CTC)-based end-to-end speech recognition and demonstrate competitive performance in the speaker-adapted scenario.

3. Proposed methods

We first introduce the modified speaker normalization (SN) method in speech recognition. Moreover, its application in Bi-LSTM is discussed. In the following sections, the details of the proposed speaker-level and batch-level adaptive speaker normalization (ASN) are discussed.

3.1. Speaker normalization

For a neural layer with p-dimensional input feature $x_s = \{x_s^{(1)}, ..., x_s^{(i)}, ..., x_s^{(p)}\}$, where s means the feature belongs to a certain speaker s, the proposed speaker normalization for each dimension is defined as:

$$\hat{x}_s^{(i)} = \frac{x_s^{(i)} - \mathrm{E}[x_s^{(i)}]}{\sqrt{\mathrm{Var}[x_s^{(i)}]}} \tag{1}$$

where the expectation $\mathrm{E}[x_s^{(i)}]$ and variance $\mathrm{Var}[x_s^{(i)}]$ are computed over all training samples belonging to speaker s.

However, in most instances, training DNNs uses stochastic optimization. Parameter updates are on a mini-batch basis. It is impractical to use the whole set to normalize activations. Therefore, we make the simplification as BN that each mini-batch produces estimates of the mean and variance in each activation of speaker s. Eq. (1) is rewritten as:

$$\hat{x}_s^{(i)} = \frac{x_s^{(i)} - \mu_s}{\sqrt{\sigma_s^2 + \varepsilon}} \tag{2}$$

where ε is a small positive constant to prevent numerical instability, and the mini-batch speaker mean μ_s and variance σ_s are given by

$$\mu_s = \frac{1}{\sum_k \mathbf{1}[s_k = s]} \sum_k \mathbf{1}[s_k = s] x_k \tag{3}$$

and

$$\sigma_s^2 = \frac{1}{\sum_k \mathbf{1}[s_k = s]} \sum_k \mathbf{1}[s_k = s](x_k - \mu_s)^2 \tag{4}$$

where s_k denotes the speaker label of the k^th sample in the mini-batch and $\mathbf{1}[.]$ is the indicator function that evaluates to 1 when its argument holds.

However, according to BN, simply normalizing each input of a layer may change what the layer can represent. To account for this, we also introduce additional learnable parameters γ and β, which respectively scale and shift the normalized activation to enhance the representational power of the layer, leading to a layer of the form:

$$y_s^{(i)} = \gamma^{(i)} \hat{x}_s^{(i)} + \beta^{(i)} \tag{5}$$

where γ and β are parameters to be trained along with the original model parameters. By setting $\gamma^{(i)}$ to σ_s and $\beta^{(i)}$ to μ_s, the network can recover the original layer representation.

Note that SN normalizes the activations at the speaker level, which can be considered a subset of BN. When all samples of a mini-batch belong to the same speaker, SN is equal to the standard BN. However, the proposed SN may solve the drawbacks of BN to some extent. According to [33], the effectiveness of BN diminishes when the training mini-batches are small or do not consist of independent samples. For small mini-batches, the estimates of the mean and variance become less accurate. These inaccuracies are compounded with depth and reduce the quality of the resulting models. In SN, however, we estimate the mean and variance of each speaker instead of the entire training set. This allows us to make more accurate estimates from activations in smaller batches. In addition, similar to the definition of BN, SN also requires that the samples have the assumption of independent and identical distribution (i.i.d.). However, the connectionist temporal classification (CTC) [22] criterion has the same assumptions of i.i.d., which makes the proposed SN better match the scenario of CTC-based ASR.

For a standard feedforward layer in a neural network, speaker normalization can be applied easily before an arbitrary activation function as in BN. However, we are more concerned with its application in the long short-term memory (LSTM) model since LSTM is widely used to model the temporal information of acoustic features in speech recognition, especially in CTC-based speech recognition. However, according to the investigations in BN, it is quite challenging to perform normalization in such recurrent neural networks due to their complicated framework. In this paper, we apply speaker normalization to the input-to-hidden transitions of LSTMs as the researchers did in [34]. This has proven to be effective in our experiments. For a bidirectional LSTM, speaker normalization is equally applied to the forward and backward LSTM.

3.2. Adaptive speaker normalization

The scaling and shifting parameters of SN can be computed at the speaker level and batch level, respectively. For the speaker-level strategy, normalizing activations of each speaker corresponds with specific scaling and shifting parameters. For the batch-level strategy, one pair of scaling and shifting parameters are generated for all mini-batch samples.

3.2.1. Speaker level

Assume that $\mathbf{h}_t^{l-1}$ denotes the p-dimensional hidden activation of the $l-1^th$ layer at time step t. For the normalization pa-

rameter generation network, a nonlinear transformation is first applied to the normalized hidden activation as:

$$\mathbf{g}_t^{l-1} = \tanh(\mathbf{W}_g \mathbf{h}_t^{l-1} + \mathbf{b}_g) \tag{6}$$

where $\mathbf{W}_g$ is a $d_g \times p$ weight matrix and d_g is set to be less than p. This nonlinear transformation is designed to project the hidden activation to a low dimensional space. In this way, the computational cost of the auxiliary network can be greatly reduced.

We use the weighted summation of all frames belonging to speaker s to generate the normalization parameters for that speaker. The mean of all the elements in $\mathbf{g}_t^{l-1}$ can measure the importance of the t^{th} frame-level representation. With the softmax function, the attention weight for each frame of the speaker s can be calculated as:

$$\alpha_{s,t} = \frac{\exp(\text{mean}(\mathbf{g}_{s,t}^{l-1}))}{\sum_\tau \mathbf{1}[s_\tau = s] \exp(\text{mean}(\mathbf{g}_\tau^{l-1}))} \tag{7}$$

where s_τ denotes the speaker label of the τ^{th} frame in the mini-batch and $\mathbf{1}[.]$ is the indicator function that evaluates to 1 when its argument holds.

The context vector of speaker s can be easily computed with $\alpha_{s,t}$ serving as the combination weights.

$$\mathbf{c}_s = \sum_t \alpha_{s,t} \mathbf{1}[s_t = s] \mathbf{g}_t^{l-1} \tag{8}$$

Finally, the scaling and shifting parameters for speaker s are generated as a linear transformation of the context vector:

$$\gamma_s^l = \mathbf{W}_\gamma^l \mathbf{c}_s + \mathbf{b}_\gamma^l \tag{9}$$

$$\beta_s^l = \mathbf{W}_\beta^l \mathbf{c}_s + \mathbf{b}_\beta^l \tag{10}$$

Assume $\hat{\mathbf{h}}_{s,t}^{l-1}$ denotes the activation normalized by the mean and variance of speaker s. The final speaker normalized activation is given by

$$\tilde{\mathbf{h}}_{s,t}^l = \hat{\mathbf{h}}_{s,t}^{l-1} \odot \gamma_s^l + \beta_s^l \tag{11}$$

where $\odot$ denotes the elementwise product.

3.2.2. Batch level

In batch-level speaker normalization, all activations of all speakers in the mini-batch share one pair of scaling and shifting parameters indiscriminately as with the standard format. The difference is that the parameters are dynamically generated by the activations.

A straightforward idea is to use the weighted mean of all activated frames to generate the parameters. Then, Eq. (7) and Eq. (8) are rewritten as:

$$\alpha_t = \frac{\exp(\text{mean}(\mathbf{g}_t^{l-1}))}{\sum_\tau \exp(\text{mean}(\mathbf{g}_\tau^{l-1}))} \tag{12}$$

$$\mathbf{c} = \sum_t \alpha_t \mathbf{g}_t^{l-1} \tag{13}$$

where the symbols denote the same meaning as above. Then, the context vector is used to generate the scaling and shifting parameters as Eq. (9-10).

To take advantage of the discriminative information among different speakers, we further propose speaker interclass attention to combine the activations of different speakers. After the context vectors of all speakers are computed in Eq. (8). The attention weight for each speaker context vector can be calculated as:

$$\alpha_s = \frac{\exp(\text{mean}(\mathbf{c}_s))}{\sum_m \exp(\text{mean}(\mathbf{c}_m))} \tag{14}$$

where m denotes the m^{th} speaker in the mini-batch.

Then, the weighted mean of all speaker context vectors is formed with α_s serving as the combination weights:

$$\mathbf{u} = \sum_s \alpha_s \mathbf{c}_s \tag{15}$$

Finally, the weighted mean is used to generate the scaling and shifting parameters as Eq. (9-10).

4. Experiments

4.1. Dataset

We evaluated our proposed methods on an open-source Mandarin speech corpus AISHELL-1 [27]. All speech files are sampled at 16 K Hz with 16 bits. We trained our models on the training set which contains 150 hours of speech (120,098 utterances) recorded by 340 speakers. The development set contains 20 hours of speech (14,326 utterances) recorded by 40 speakers was used for early-stopping. And the test set contains 10 hours of speech (7,176 utterances) recorded by 20 speakers was used for the final evaluation. The speakers of the training set, development set, and test set are not overlapped.

4.2. SI system

We used connectionist temporal classification (CTC)-based speech recognition systems in our experiments. The input acoustic feature was 108-dimensional filter-bank features (36 filter-bank features, delta coefficients, and delta-delta coefficients) with mean and variance normalization. All neural acoustic models in the experiments had three bidirectional LSTM hidden layers with 512 LSTM cells. To improve recognition performance and training efficiency, we appended a convolutional neural network (CNN) before the LSTM layers. For the SI model, the bottom two layers were 2D convolution layers with output channels of 64 and 256. Each convolution layer was followed by a max-pooling layer with a stride of 2 in the time dimension for finally downsampling utterances to a quarter of the original length. We used a dropout rate of 0.3 for the LSTM layers to avoid overfitting.

For the output of the Mandarin acoustic model, according to the statistical information of the transcripts, we collected 4,294 Chinese characters in the training and development sets. With the special symbol blank involved, 4,295 modeling units were used for the output inference. Additionally, to further improve the performance of the SI model, the trigram language model, which was trained by using the transcription of the training set, was used in the decoding procedure.

4.3. Network training setups

The CTC-based acoustic model used the whole utterance as input, while utterances varied in length. Therefore, we sorted all the utterances of the training set in descending order by length. For the input features of the network, each utterance was represented as a sequence of frames. We set a maximum number of frames of $f_{\max}$ to control the batch size. The number of utterances included in each mini-batch was $f_{\max}/l_{\max}$, where/means rounding operation, and $l_{\max}$ denotes the length

Table 1: *The CERs (%) of SI and SN models under different learning rates. "–" denotes that the model did not converge.*

Learning rate	0.0001	0.0002	0.0004
SI	9.96	-	-
SN	9.44	9.04	8.89

of the longest utterance of the mini-batch. All utterances whose lengths were less than $l_{\max}$ were unified by a zero-padding operation. Therefore, the size of each mini-batch was variable but did not exceed $f_{\max}$.

PyTorch toolkits [35] were used in our model training process. All the model parameters were randomly initialized and updated by Adam [36]. The network was trained to minimize the CTC loss function with an initial learning rate of 0.0001. The development set was used for learning rate scheduling and early stopping. We started to halve the learning rate when the relative improvement fell below 0.004, and the training ended if the relative improvement was lower than 0.0005.

4.4. Results of SN

The standard speaker normalization described in section 3.1 was first applied to the input of all LSTM layers. We investigated the effect of normalization on the learning rate of network training, where $f_{\max}$ was set to 5,000.

Table 1 shows the character error rate (CER) of different acoustic models with and without speaker normalization under different initial learning rates. It can be seen that with a learning rate greater than 0.0001, the model without speaker normalization did not converge. The model with speaker normalization obtained a 5.2% reduction in CER under a small learning rate of 0.0001. In the case of a larger learning rate, the speaker normalized model significantly outperformed the SI model, achieving a CER of 8.89% and up to 10.7% relative improvement.

According to the analysis of Table 1, we found that standard speaker normalization enables higher learning rates and makes the model perform better. Speaker normalization makes model training more resilient to the parameter scale. Normally, large learning rates may increase the scale of layer parameters, which then amplify the gradient during backpropagation and lead to model explosion. However, with speaker normalization, backpropagation through a layer is unaffected by the scale of its parameters.

We further explored the influence of batch size during model training. Since SN uses similar ideas as BN, we compared the proposed SN with the BN algorithm. The initial learning rates for the SN and BN models were set to 0.0002. As shown in Table 2, in the case of a smaller batch size, BN greatly deteriorated the ASR performance. This is an inherent flaw of BN, as mentioned in [39]. For the proposed SN, a smaller batch size had little impact on the model performance, resulting in a comparable performance with a favorable batch size of 5,000. This shows that SN can allow the model to be trained at a smaller batch size without significantly reducing performance, thereby adapting to scenarios with sparse data.

Table 2: *The CERs (%) of the SN and BN models with different training batch sizes.*

Batch size($f_{\max}$)	2000	5000	8000
SN	10.97	9.71	9.90
BN	9.01	9.04	9.38

4.5. Results of ASN

In the ASN model, the size of the nonlinear transformation in the auxiliary network, i.e., d_g, was set to 256 to speed up the training. The hidden activation of the previous layer was used to generate the scaling and shifting parameters for the current layer. We also used dropout for the auxiliary network to improve performance. Note that the main network was more adaptable to smaller learning rates due to the influence of the auxiliary network. Therefore, the learning rate was set to 0.0001 for the ASN models, and $f_{\max}$ was set to 5000.

Table 3: *The CERs (%) of SI, SN and different ASN models*

Model	Model Size(M)	CER(%)
SI	85.82	9.96
SN	85.84	8.89
ASN-S	92.75	**8.22**
ASN-B1	92.75	8.51
ASN-B2	92.75	8.39

Table 3 summarizes the experimental results of the SI, SN and ASN models. ASN-S indicates that the scaling and shifting parameters in SN were generated at the speaker level. ASN-B1 and ASN-B2 denote the parameters that were generated by using the weighted mean of all activated frames and all speaker context vectors, respectively. As shown in Table 3, all ASN models outperformed the SN models. In batch-level ASN, since the proposed speaker interclass attention utilized the discriminative information among different speakers, ASN-B1 performed better than ASN-B2. In speaker-level ASN, since specific scaling and shifting parameters were generated for each speaker to provide more discriminative information, ASN-S further outperformed ASN-B. Finally, ASN-S achieved the best CER of 8.22%, resulting in 17.5% relative improvement over the SI model.

5. Conclusions

In this work, we propose a novel speaker normalization technique for neural acoustic model adaptation in CTC-based ASR. Unlike previous work, we use the idea of BN to normalize hidden activations at the speaker level. The method performs a layer-wise normalization for hidden activations and utilizes the mean and variance information of each speaker. Experimental results show that the proposed SN enables the model to be trained with a higher learning rate, resulting in a better performance. Additionally, SN makes model training more resilient to the batch size, which makes it possible to use it in different scenarios. Furthermore, based on SN, we propose ASN, in which the scaling and shifting parameters are dynamically generated by using an auxiliary network with an attention mechanism. We generate the parameters at the speaker level and batch level. The two strategies both outperform the standard SN, finally achieving up to a 17.5% relative reduction in CER.

6. Acknowledgements

This work was partially funded by the National Key Research and Development Program of China (Grant No. 2016YFB100 1303) and the National Natural Science Foundation of China (Grant No. U1836219).

7. References

[1] G. Hinton, L. Deng, D. Yu *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, 2012.

[2] D. Yu and J. Li, "Recent progresses in deep learning based acoustic models," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 3, pp. 396–409, 2017.

[3] L. Hank, "Speaker adaptation of context dependent deep neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7947–7951.

[4] D. Yu, K. Yao, H. Su *et al.*, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7893–7897.

[5] Z. Meng, J. Li, and Y. Gong, "Adversarial speaker adaptation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5721–5725.

[6] K. Yao, D. Yu, F. Seide *et al.*, "Adaptation of context-dependent deep neural networks for automatic speech recognition," in *2012 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2012, pp. 366–369.

[7] S. M. Siniscalchi, J. Li, and C.-H. Lee, "Hermitian polynomial for speaker adaptation of connectionist speech recognition systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2152–2161, 2013.

[8] L. Samarakoon and K. C. Sim, "Factorized hidden layer adaptation for deep neural network based acoustic modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 12, pp. 2241–2250, 2016.

[9] P. Swietojanski, J. Li, and S. Renals, "Learning hidden unit contributions for unsupervised acoustic model adaptation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1450–1463, 2016.

[10] P. Swietojanski and S. Renals, "Differentiable pooling for unsupervised speaker adaptation," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4305–4309.

[11] C. Zhang and P. C. Woodland, "Dnn speaker adaptation using parameterised sigmoid and relu hidden activation functions," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5300–5304.

[12] A.-r. Mohamed, T. N. Sainath, G. E. Dahl *et al.*, "Deep belief networks using discriminative features for phone recognition." in *ICASSP*, 2011, pp. 5060–5063.

[13] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, 2011, pp. 24–29.

[14] 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.

[15] Y. Miao, H. Zhang, and F. Metze, "Towards speaker adaptive training of deep neural network acoustic models," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[16] A. Senior and I. Lopez-Moreno, "Improving dnn speaker independence with i-vector inputs," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 225–229.

[17] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7942–7946.

[18] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, "Direct adaptation of hybrid dnn/hmm model for fast speaker adaptation in lvcsr based on speaker code," in *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2014, pp. 6339–6343.

[19] T. Tan, Y. Qian, M. Yin *et al.*, "Cluster adaptive training for deep neural network," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4325–4329.

[20] C. Wu and M. J. Gales, "Multi-basis adaptive neural network for rapid adaptation in speech recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4315–4319.

[21] J. Li, R. Zhao, Z. Chen *et al.*, "Developing far-field speaker system via teacher-student learning," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5699–5703.

[22] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[23] F. Weninger, J. Andrés-Ferrer, X. Li, and P. Zhan, "Listen, attend, spell and adapt: Speaker adapted sequence-to-sequence asr," *arXiv preprint arXiv:1907.04916*, 2019.

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

[25] T. Kim, I. Song, and Y. Bengio, "Dynamic layer normalization for adaptive neural acoustic modeling in speech recognition," *arXiv preprint arXiv:1707.06065*, 2017.

[26] F. Ding, W. Guo, L. Dai, and J. Du, "Attentive batch normalization for lstm-based acoustic modeling of speech recognition," *arXiv preprint arXiv:2001.00129*, 2020.

[27] H. Bu, J. Du, X. Na *et al.*, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. IEEE, 2017, pp. 1–5.

[28] 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.

[29] O. Viikki and K. Laurila, "Cepstral domain segmental feature vector normalization for noise robust speech recognition," *Speech Communication*, vol. 25, no. 1-3, pp. 133–147, 1998.

[30] Y. Obuchi and R. M. Stern, "Normalization of time-derivative parameters using histogram equalization," in *Eighth European Conference on Speech Communication and Technology*, 2003.

[31] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[32] L. Sarı, S. Thomas, and M. A. Hasegawa-Johnson, "Learning speaker aware offsets for speaker adaptation of neural networks," *Proc. Interspeech 2019*, pp. 769–773, 2019.

[33] S. Ioffe, "Batch renormalization: Towards reducing minibatch dependence in batch-normalized models," in *Advances in neural information processing systems*, 2017, pp. 1945–1953.

[34] C. Laurent, G. Pereyra, P. Brakel *et al.*, "Batch normalized recurrent neural networks," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2657–2661.

[35] A. Paszke, S. Gross, S. Chintala *et al.*, "Automatic differentiation in pytorch," 2017.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

An Adaptive X-vector Model for Text-independent Speaker Verification

Bin Gu, Wu Guo, Fenglin Ding, Zhenhua Ling, Jun Du

National Engineering Laboratory for Speech and Language Information Processing,
University of Science and Technology of China, Hefei, China

{bin2801,flding}@mail.ustc.edu.cn, {guowu,zhling,jundu}@ustc.edu.cn

Abstract

In this paper, adaptive mechanisms are applied in deep neural network (DNN) training for x-vector-based text-independent speaker verification. First, adaptive convolutional neural networks (ACNNs) are employed in frame-level embedding layers, where the parameters of the convolution filters are adjusted based on the input features. Compared with conventional CNNs, ACNNs have more flexibility in capturing speaker information. Moreover, we replace conventional batch normalization (BN) with adaptive batch normalization (ABN). By dynamically generating the scaling and shifting parameters in BN, ABN adapts models to the acoustic variability arising from various factors such as channel and environmental noises. Finally, we incorporate these two methods to further improve performance. Experiments are carried out on the speaker in the wild (SITW) and VOiCES databases. The results demonstrate that the proposed methods significantly outperform the original x-vector approach.

Index Terms: Speaker verification; Adaptive convolution; Adaptive batch normalization; Attention mechanism

1. Introduction

Speaker verification (SV) is a task to verify a person's claimed identity from speech signals. During the last decade, the i-vector [1] algorithm combined with a probabilistic linear discriminant analysis (PLDA) [2] used for similarity scoring has become a dominant approach for SV.

This paradigm has been improved by incorporating a deep neural network (DNN) to extract speaker representations, which are named the x-vector [3] or d-vector [4] in the SV field. In most of these DNN-based systems, several frame-level layers are stacked to deal with a local short span of acoustic features to obtain more effective high-level representations. These layers can be modeled by a time-delay neural network (TDNN) [5, 3], convolutional neural network (CNN) [6] or long short-term memory network (LSTM) [7, 8]. Then, a pooling layer maps all frames of the input utterance into a fixed-dimensionality vector, and speaker embedding is generated from the following stacked fully connected layers. Average pooling, max pooling [9] and statistics pooling [5] are widely used in pooling layers. Some researchers have also employed the attention mechanism [10] and gating mechanism [11, 12] in the pooling layer. By providing different frame weights, these methods can capture more expressive speaker characteristics. Such DNN embedding systems have become the current state-of-the-art systems in most public benchmarks.

Speech signals are easily corrupted by various factors, such as emotions, channels, and environmental noises. How to extract robust speaker embeddings is one of the principal interests of SV. The data augmentation technique [3, 13, 14] is the most straightforward way to solve this problem. The systems can achieve better performance by constructing additional training samples using expert knowledge or extra data sources. Another choice is applying the adversarial training strategy in the speaker characteristics modeling process. Through weakening the ability to discriminate the environment types, SNRs [15] or other relative information [16, 17] in a speech, the speaker embedding extractor generates more robust speaker representations.

Recently, an adaptive convolution neural network (ACNN) has been shown to be useful for natural language processing (NLP) tasks [18, 19] and computer vision (CV) tasks [20]. Unlike traditional convolutions that use the same set of filters regardless of different inputs, adaptive convolution employs adaptively generated convolutional filters that are conditioned on inputs. Similar to these works, dynamic layer normalization (DLN) [21] and adaptive batch normalization (ABN) [22] are proposed for adaptive neural acoustic modeling in speech recognition. The parameters in the normalization layer are substituted with learned functions, the outputs from which are then used as normalization parameters. In these studies, an adaptive mechanism gives stronger flexibility to networks and allows networks to utilize the information inputs contained.

In this paper, we investigate the abovementioned adaptive learning methods for robust embedding extraction. More specifically, the ACNN and ABN are employed in the frame-level layers for extracting more expressive feature representations. In addition, we incorporate these two methods into an x-vector network to further improve performance. To the best of our knowledge, this study is the first to employ input-aware methods to extract robust speaker embeddings. We evaluate our experiments on the SITW and VOiCES datasets. The experimental results show that the two methods can both achieve better performance than the original x-vector approach, and the appropriate integration of the methods can further improve performance.

The remainder of this paper is organized as follows. Section 2 gives an introduction to our x-vector baseline. Section 3 describes the proposed input-aware model in detail. Then, the experimental setup, the results and the analysis are presented in section 4. Finally, the conclusions are given in section 5.

2. Baseline Network Architecture

The network architecture of our x-vector baseline system is the same as that described in [3]. As depicted in Figure 1, the x-vector baseline consists of three time-delay frame-level layers, two more frame-level layers without time delay, a pooling layer that converts the variable-length frame-level representations into a single fixed-length vector and, finally, two utterance-level layers followed by the output layer.

As we know, the TDNNs in the frame-level layers could be implemented as 1-D convolutional neural networks (1-D CNNs), where the filters slide along the time axis. The mean

and standard deviation of the final frame-level output vectors are calculated and then concatenated together for the pooling layer. All activations in the network are rectified linear units (ReLUs). Batch normalization is used on the activations from all layers except the output layer. The output layer computes an affine transform of its input and then transforms the outputs using softmax. The network is trained to predict the correct speaker labels with cross entropy (CE) loss. Once the DNN is trained, the speaker embeddings are extracted from the layer right after pooling.

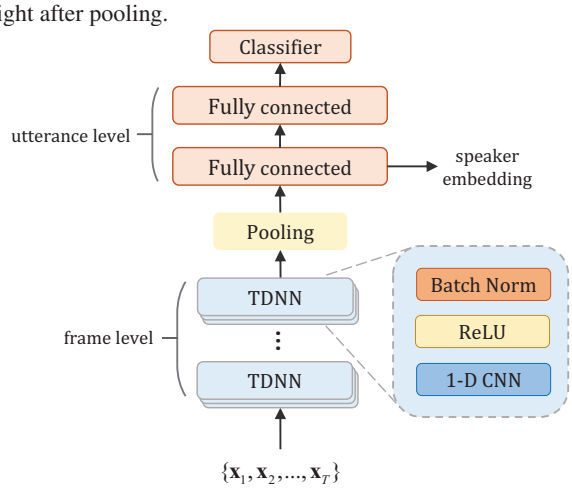


Figure 1: *Baseline network architecture.*

3. Adaptive X-vector Model

This section introduces the proposed ACNN and ABN. We explain how the parameters are generated and applied to the existing framework.

3.1. Adaptive convolution neural network

Figure 2 schematically shows the overall architecture of our ACNN. Attentive statistics pooling is used to encapsulate the variable size input into a fixed size context vector at first, and this vector adjusts the convolution parameters by determining the weights of the component filters and biases. The final convolutional parameters are linear regressions of these components.

Suppose $\mathbf{h}_t^l$ is the hidden representation in the l^{th} layer. The attention mechanism is applied first. The value vectors $\mathbf{e}_t$ and the attention weights α_t are calculated as follows:

$$\begin{aligned} \mathbf{e}_t &= \mathbf{h}_t^l * \mathbf{W}_e + \mathbf{b}_e \\ \alpha_t &= \mathbf{v}^T \tanh(\mathbf{h}_t^l * \mathbf{W}_\alpha + \mathbf{b}_\alpha) \end{aligned} \tag{1}$$

where $\mathbf{W}_e$ and $\mathbf{W}_\alpha$ are the convolution parameters, while $\mathbf{b}_e$ and $\mathbf{b}_\alpha$ are the bias parameters. $\mathbf{v}$ is a vector that converts the hidden vector to a scalar value. Then, we generate a context vector by leveraging the statistical information inherent in the weighted value vectors $\mathbf{e}_t$.

$$\begin{aligned} \boldsymbol{\mu} &= \sum_t \alpha_t \mathbf{e}_t \\ \boldsymbol{\sigma} &= \sqrt{\sum_t \alpha_t \mathbf{e}_t \odot \mathbf{e}_t - \boldsymbol{\mu} \odot \boldsymbol{\mu}} \\ \mathbf{c}_{acnn} &= [\boldsymbol{\mu}, \boldsymbol{\sigma}] \end{aligned} \tag{2}$$

Finally, we concatenate $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ as the context vector $\mathbf{c}_{acnn}$ to generate the convolutional parameters by a linear combination of components from a parameter pool.

$$\begin{aligned} \boldsymbol{\beta} &= [\beta_1, ..., \beta_N] = \mathbf{W}_\beta \mathbf{c}_{acnn} + \mathbf{b}_\beta \\ \tilde{\mathbf{W}} &= \sum_{i=1}^{N} \beta_i \mathbf{W}_i \\ \tilde{\mathbf{b}} &= \sum_{i=1}^{N} \beta_i \mathbf{b}_i \end{aligned} \tag{3}$$

where $\mathbf{W}_\beta$ and $\mathbf{b}_\beta$ are trainable parameters to generate the weight vector $\boldsymbol{\beta}$ and the parameters $\{\mathbf{W}_i\}_{i=1...N}$ and $\{\mathbf{b}_i\}_{i=1...N}$ of the component filters can also be trained through a typical backpropagation algorithm.

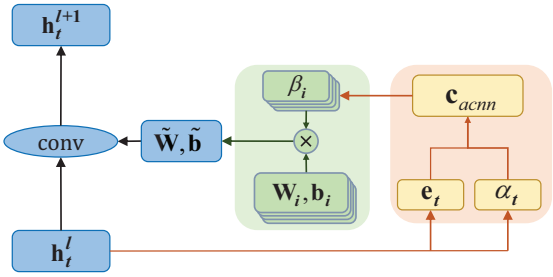


Figure 2: *Structure of the proposed ACNN.*

Once we obtain the convolutional filter $\tilde{\mathbf{W}}$ and bias $\tilde{\mathbf{b}}$, a conventional convolution operation is applied to the inputs as follows:

$$\mathbf{h}_t^{l+1} = f(\mathbf{h}_t^l * \tilde{\mathbf{W}} + \tilde{\mathbf{b}}) \tag{4}$$

where f is a nonlinear function and is usually composed of an activation function and batch normalization.

3.2. Adaptive batch normalization

The main difference between the ABN and BN is that the scaling and shifting parameters are dynamically generated for different inputs in the ABN while they are fixed for all the inputs in the BN in the testing procedure. The main procedure of the ABN is similar to that of the ACNN. First, the weighted context vector $\mathbf{c}_{abn}$ can be calculated as follows:

$$\begin{aligned} \mathbf{e}_t &= \tanh(\mathbf{W}_e * \mathbf{h}_t^l + \mathbf{b}_e) \\ \alpha_t &= \frac{\exp(\text{mean}(\mathbf{e}_t))}{\sum_i \exp(\text{mean}(\mathbf{e}_i))} \\ \mathbf{c}_{abn} &= \sum_t \alpha_t \mathbf{e}_t \end{aligned} \tag{5}$$

where $\mathbf{W}_e$ and $\mathbf{b}_e$ are trainable parameters. The mean of all the elements in $\mathbf{e}_t$, which is the nonlinear transformed low-dimension vector of input $\mathbf{h}_t^l$, is used to measure the importance of each frame, and the weighted sum of $\mathbf{e}_t$ is used as the context vector $\mathbf{c}_{abn}$. Then, the scaling $\boldsymbol{\gamma}$ and shifting parameters are generated from $\boldsymbol{\beta}$.

$$\begin{aligned} \boldsymbol{\gamma} &= \mathbf{W}_\gamma \mathbf{c}_{abn} + \mathbf{b}_\gamma \\ \boldsymbol{\beta} &= \mathbf{W}_\beta \mathbf{c}_{abn} + \mathbf{b}_\beta \end{aligned} \tag{6}$$

where $\mathbf{W}_\gamma$, $\mathbf{W}_\beta$,$\mathbf{b}_\beta$ and $\mathbf{b}_\gamma$ are all trainable parameters. Finally, standard batch normalization [23] is employed with the generated parameters.

4. Experiments and Discussion

4.1. Data set and evaluation metrics

All experiments are conducted on the SITW and VOiCES datasets. For the SITW dataset [24], there are two standard datasets for testing: dev. core and eval. core. We use both sets to conduct the experiments. The VoxCeleb database [25], including the VoxCelebb1 and VoxCeleb2, is used for training. Since a few speakers are included in both the SITW and VoxCeleb datasets, these speakers are removed from the training dataset.

The VOiCES dataset for the speaker verification task is described in the "VOiCES from a Distance Challenge 2019" [26]. The development dataset contains 15,904 noisy and far-field speech segments from 196 speakers. The evaluation set consists of 11,392 distant recordings from different microphone types and different rooms, both of which could be more challenging than those featured in the development set.

Due to the background noise, reverberation, laughter and acoustic artifacts contained in speech data, the data augmentation techniques described in [3], including adding additive noise and reverberation data, are applied to improve the robustness of the system. Because there is a possibility that there will be an overlap between the MUSAN [27], which is a publicly available augmentation dataset, and the VOiCES dataset, babble noise is not created for augmentation. In summary, there are a total of 2,236,567 recordings from 7185 speakers for training, including approximately 1,000,000 randomly selected augmented recordings. Note that the training data for VOiCES are consistent with those for the SITW dataset.

The results are reported in terms of three metrics: the equal error rate (EER), the minimum of the normalized detection cost function (minDCF) and the actual detection cost function (actDCF). The minDCF has two settings: one with a prior target probability P_{tar} set to 0.01 (DCF(10^{-2})) and the other with a P_{tar} set to 0.001 (DCF(10^{-3})).

4.2. Features

We select 30-dimensional MFCC features containing delta and delta-delta coefficients as the input acoustic features. Each MFCC feature is extracted from the speech signal with a 25 ms window and a 10 ms frame shift. Each feature is mean-normalized over a 3 s sliding window, and energy-based VAD is employed to filter out non-speech frames. The acoustic features are randomly cropped to lengths of 2-4 s, and 128 utterances with the same duration are grouped into a mini-batch. Data processing is implemented with the Kaldi toolkit [28].

4.3. Model configuration

All neural networks are implemented using the TensorFlow toolkit [29]. The network is optimized using the Adam optimizer, and the learning rate gradually decreases from 1e-3 to 1e-4. If not specified, all of the setups are the same as the baseline system. Other configurations of each system are listed as follows:

x-vector:This is a deep embedding learning baseline system. Only the fifth hidden layer has 1536 nodes, while the other layers have 512 nodes. The kernel sizes of the first five layers are 5, 3, 3, 1 and 1, while the dilation rates are set to 1, 2, 3, 1 and 1 respectively. The same type of L2 weight decay and batch normalization as described in [30] are used in the baseline system to prevent overfitting.

ACNN:In this system, the ACNN is only applied in the fourth frame-level layer. Such a setup can achieve satisfactory results with a minimum increase in parameters. The hidden dimensions of both $\mathbf{W}_e$ and $\mathbf{W}_\alpha$ in Eq. (1) are set to 256. The number of component filters N in Eq. (3) is chosen to be 4.

ABN:All frame-level layers employ the ABN in this system. The hidden dimension of $\mathbf{W}_e$ in Eq. (5) is set to 256. Note that the utterance-level layers use the conventional BN, and other setups are exactly the same as the baseline system.

ACNN&ABN:Both the ACNN and ABN are employed in this system. The ACNN is employed in the fourth layer, while the ABN is used in the remaining frame-level layers. The setup is consistent with the abovementioned **ACNN** and **ABN** systems.

Fusion:The complementarity between the above two different adaptive learning methods at the score level is also investigated here. We only report the results using the score fusion of the **ACNN** and **ABN** with equal weights

4.4. PLDA Backend

The DNN embeddings are centered using the training set and are projected to a low-dimensional space using LDA at first. The dimensions of the x-vectors are reduced to 100 for both datasets. After length normalization, we select the longest 200,000 recordings from the training set to train the PLDA backend. The backend classifier is implemented with the Kaldi toolkit.

4.5. Results and analysis

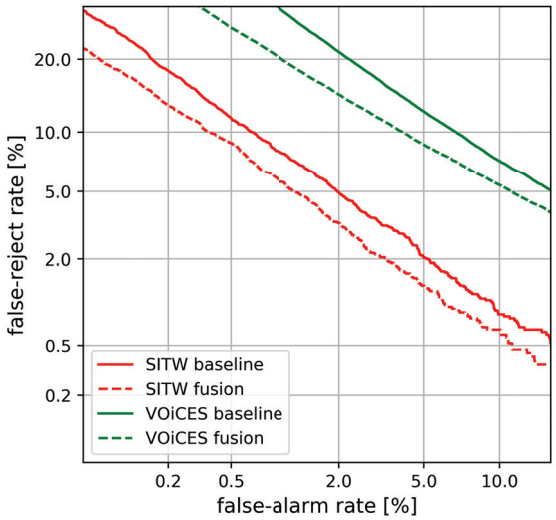


Figure 3: *DET curve comparison for the evaluation set of SITW and VOiCES.*

Table 1 presents the results of different systems on the SITW and VOiCES datasets. It can be observed that the system applying either the ACNN or ABN outperforms the x-vector baseline system. On the SITW dataset, these two systems can both improve the baseline by approximately 10% $\sim$ 19% for all evaluation metrics. For the VOiCES dataset, the ACNN achieves at most 21% relative improvements over the baseline in terms of minDCF and actDCF, while the ABN achieves at most 17% relative improvements in terms of EER. The ACNN&ABN system can obtain some further performance improvement over the ACNN and ABN systems, especially on the SITW dataset.

Table 1: *Results of different systems on the SITW and VOiCES datasets. DCF2, DCF3 and aDCF refer to DCF(10^{-2}), DCF(10^{-3}) and actDCF, respectively. Impr denotes the relative improvement of the best results with respect to the baseline system.*

Systems	SITW						VOiCES					
	Dev			Eval			Dev			Eval		
	EER	DCF2	DCF3	EER	DCF2	DCF3	EER	DCF2	aDCF	EER	DCF2	aDCF
x-vector	2.88	0.2956	0.4752	3.28	0.3063	0.4974	3.44	0.3952	0.4925	8.34	0.6203	0.7299
ACNN	2.54	0.2389	0.4126	2.73	0.2824	0.4430	3.25	0.3346	0.3663	7.57	0.5553	0.5800
ABN	2.35	0.2444	0.4110	2.90	0.2765	0.4380	2.87	0.3206	0.4060	7.47	0.5709	0.6167
ACNN&ABN	2.35	0.2317	**0.3693**	**2.54**	0.2687	0.4233	2.73	0.3263	0.4291	7.12	0.5676	0.6090
Fusion	**2.12**	**0.2264**	0.3768	2.60	**0.2650**	**0.4106**	**2.71**	**0.2841**	**0.3060**	**7.03**	**0.5114**	**0.5182**
Impr.(%)	26	23	21	21	13	17	21	28	38	16	18	29

Table 2: *Comparison results of the proposed ACNN system using different setups. Except for the parameter N, the rest of the setup is consistent with the ACNN system described in section 4.3.*

Systems	SITW						VOiCES					
	Dev			Eval			Dev			Eval		
	EER	DCF2	DCF3	EER	DCF2	DCF3	EER	DCF2	aDCF	EER	DCF2	aDCF
ACNN(N=2)	**2.35**	**0.2379**	**0.3995**	2.82	0.2871	0.4457	3.25	0.3557	0.4083	7.87	0.5677	0.6144
ACNN(N=4)	2.54	0.2389	0.4126	**2.73**	**0.2824**	**0.4430**	3.25	**0.3346**	**0.3663**	**7.57**	**0.5553**	**0.5800**
ACNN(N=6)	2.54	0.2595	0.4060	3.01	0.2828	0.4562	**2.99**	0.3752	0.4764	7.95	0.6158	0.7067
ACNN(N=8)	2.58	0.2547	0.4234	2.82	0.2855	0.4558	3.24	0.3698	0.4321	7.74	0.6126	0.6847

Among all of the above systems, the fused system achieves the best performance especially in terms of the actDCF, which improves over the baseline by nearly 38% and 29% on the development set and the evaluation set of VOiCES, respectively. Figure.3 depicts the detection error trade-off (DET) curves of the baseline and the fusion systems, and obvious improvements can be observed.

Note that we only employ the ACNN in the fourth frame-level layer for two reasons. In the ACNN, the number of parameters is usually several times highter than the number of parameters in the CNN because each component filter is the same size as that in the CNN. The fourth frame-level layer has the minimum convolution kernel size and hidden dimension in our systems. Applying the ACNN in such a layer only causes an approximate 9% increase in parameters, while applying it in any other layer causes at least a 27% increase in parameters with respect to the baseline. On the other hand, the three bottom frame-level layers model long-term temporal dependencies with time delay and the frame-level feature representations are not high-level enough to reflect all kinds of information in high-dimensional abstract space.

The hyperparameter N in Eq. (3) controls the number of component filters. The experimental results with different N values are listed in Table 2. There is a larger gap between the development set and evaluation set with N=2. This means that too few component filters cannot guarantee the generalizability of the model. Generally the system with N=4 achieves the best performance. Therefore, N=4 is set in our experiments.

To test the effectiveness of the proposed ACNN and ABN in different speech environments, we select clean utterances and degraded utterances from the SITW dataset that are naturally degraded with noise, compression and reverberations. The trials were divided into 4 groups according to the speech quality and marked as "clean", "noise", "codec" and "reverb" respectively. The results in Table 3 and 4 show that both the ABN and ACNN achieve significant improvements under all conditions. This comparison demonstrates that the proposed algorithms are robust to environment types and speech quality.

Table 3: *Comparison results of the SITW development set under different conditions.*

Systems	Clean	Noise	Codec	Reverb
x-vector	3.23	2.68	3.09	2.44
ACNN	3.23	2.17	2.59	2.37
ABN	**1.61**	**1.92**	2.59	2.08
ACNN&ABN	2.42	2.22	**2.45**	**1.94**

Table 4: *Comparison results of the SITW evaluation set under different conditions.*

Systems	Clean	Noise	Codec	Reverb
x-vector	4.82	2.83	3.82	2.80
ACNN	4.42	2.37	**2.96**	2.47
ABN	**4.02**	2.62	3.09	2.47
ACNN&ABN	**4.02**	**2.23**	3.36	**2.32**

5. Conclusions

In this study, we employ adaptive mechanisms in the DNN embedding system to adaptively utilize the information of inputs. More specifically, the ACNN is introduced into the frame-level layers where the output representations are carefully modulated by adaptively estimating the convolution filters and biases. Such a mechanism helps to obtain more expressive features. Furthermore, the batch normalization layer is enhanced by dynamically generating the shifting and scaling parameters. The experimental results demonstrate that the adaptive mechanics outperform the conventional x-vector baseline. The proposed two methods have obvious complementarity with each other especially at the score level.

6. Acknowledgements

This work was partially funded by the National Natural Science Foundation of China (Grant No. U1836219) and the National Key Research and Development Program of China (Grant No. 2016YFB100 1303).

7. References

[1] 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, 2010.

[2] P. Kenny, "Bayesian speaker verification with heavy-tailed priors."

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[4] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4052–4056.

[5] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification." in *Interspeech*, 2017, pp. 999–1003.

[6] Z. Gao, Y. Song, I. V. McLoughlin, W. Guo, and L.-R. Dai, "An improved deep embedding learning method for short duration speaker verification," 2018.

[7] C.-P. Chen, S.-Y. Zhang, C.-T. Yeh, J.-C. Wang, T. Wang, and C.-L. Huang, "Speaker characterization using tdnn-lstm based speaker embedding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6211–6215.

[8] T. Liu, M. Madhavi, R. K. Das, and H. Li, "A unified framework for speaker and utterance verification," *Proc. Interspeech 2019*, pp. 4320–4324, 2019.

[9] S. Novoselov, A. Shulipa, I. Kremnev, A. Kozlov, and V. Shchemelinin, "On deep speaker embeddings for text-independent speaker recognition," *arXiv preprint arXiv:1804.10080*, 2018.

[10] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," *arXiv preprint arXiv:1803.10963*, 2018.

[11] L. You, W. Guo, L. Dai, and J. Du, "Deep neural network embeddings with gating mechanisms for text-independent speaker verification," *arXiv preprint arXiv:1903.12092*, 2019.

[12] Y. Jiang, Y. Song, I. McLoughlin, Z. Gao, and L. Dai, "An effective deep embedding learning architecture for speaker verification," *Proc. Interspeech 2019*, pp. 4040–4044, 2019.

[13] Z. Wu, S. Wang, Y. Qian, and K. Yu, "Data augmentation using variational autoencoder for embedding based speaker verification," *Proc. Interspeech 2019*, pp. 1163–1167, 2019.

[14] Y. Zhu, T. Ko, and B. Mak, "Mixup learning strategies for text-independent speaker verification," *Proc. Interspeech 2019*, pp. 4345–4349, 2019.

[15] Z. Meng, Y. Zhao, J. Li, and Y. Gong, "Adversarial speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6216–6220.

[16] X. Fang, L. Zou, J. Li, L. Sun, and Z.-H. Ling, "Channel adversarial training for cross-channel text-independent speaker recognition."

[17] J. Zhou, T. Jiang, L. Li, Q. Hong, Z. Wang, and B. Xia, "Training multi-task adversarial network for extracting noise-robust speaker embedding," pp. 6196–6200, 2019.

[18] D. Shen, M. R. Min, Y. Li, and L. Carin, "Learning context-sensitive convolutional filters for text processing," *arXiv preprint arXiv:1709.08294*, 2017.

[19] B.-J. Choi, J.-H. Park, and S. Lee, "Adaptive convolution for text classification," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 2475–2485.

[20] D. Kang, D. Dhar, and A. Chan, "Incorporating side information by adaptive convolution," in *Advances in Neural Information Processing Systems*, 2017, pp. 3867–3877.

[21] T. Kim, I. Song, and Y. Bengio, "Dynamic layer normalization for adaptive neural acoustic modeling in speech recognition," *arXiv preprint arXiv:1707.06065*, 2017.

[22] F. Ding, W. Guo, L. Dai, and J. Du, "Attentive batch normalization for lstm-based acoustic modeling of speech recognition," *arXiv preprint arXiv:2001.00129*, 2020.

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

[24] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The speakers in the wild (sitw) speaker recognition database." in *Interspeech*, 2016, pp. 818–822.

[25] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[26] 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.

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

[28] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[29] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[30] H. Zeinali, L. Burget, J. Rohdin, T. Stafylakis, and J. H. Cernocky, "How to improve your speaker embeddings extractor in generic toolkits," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6141–6145.

Unsupervised Regularization-Based Adaptive Training for Speech Recognition

Fenglin Ding, Wu Guo, Bin Gu, Zhenhua Ling, Jun Du

National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China, Hefei, China
{flding,bin2801}@mail.ustc.edu.cn, {guowu,zhling,jundu}@ustc.edu.cn

Abstract

In this paper, we propose two novel regularization-based speaker adaptive training approaches for connectionist temporal classification (CTC) based speech recognition. The first method is center loss (CL) regularization, which is used to penalize the distances between the embeddings of different speakers and the only center. The second method is speaker variance loss (SVL) regularization in which we directly minimize the speaker interclass variance during model training. Both methods achieve the purpose of training an adaptive model on the fly by adding regularization terms to the training loss function. Our experiment on the AISHELL-1 Mandarin recognition task shows that both methods are effective at adapting the CTC model without requiring any specific fine-tuning or additional complexity, achieving character error rate improvements of up to 8.1% and 8.6% over the speaker independent (SI) model, respectively.
Index Terms: speaker adaptive training, regularization, speech recognition, connectionist temporal classification

1. Introduction

Mismatches between training and testing conditions are a common problem in modern pattern recognition systems. It is particularly critical in perceptual sequence learning tasks such as automatic speech recognition (ASR) and speech emotion recognition (SER). For example, the performance of deep neural network (DNN) based ASR [1, 2] systems experience mismatches between training and testing conditions, which are caused by the different characteristics of acoustic variability such as speakers, channels and environmental noises. Adaptation techniques to transform a model to match the testing condition or augment the inputs to match a model have been investigated. In ASR, speaker adaptation (SA) techniques are used to minimize the mismatch between the training and testing conditions due to the speaker variability.

Speaker adaptation techniques for DNN based ASR can be categorized into two broad approaches: feature space and model space adaptation. In feature space adaptation, the traditional technique is to transform the acoustic features to a normalized space and then the adapted features are used to train the acoustic model. The maximum likelihood linear regression (MLLR) and its feature-space variant (fMLLR) [3, 4] are two of the most widely used methods. For a deep neural network (DNN) based acoustic model, another effective method is to provide the network with auxiliary features that characterize speaker information to perform adaptation such as the i-vector [5, 6, 7, 8] and speaker code [9, 10]. In model space adaptation, speaker dependent (SD) parameters are estimated from a trained speaker independent (SI) model using additional adaptation data. The DNN Adaptation techniques can also be categorized into two broad approaches: regularized adaptation and subspace or subset adaptation. For model adaption, a straightforward idea is to retrain all the SI model parameters. To avoid overfitting, regularization approaches such as L2 regularization using a weight decay [11], the Kullback-Leibler divergence (KLD) [12] and adversarial multitask learning (MTL) [13] have been proposed. There are also many approaches that have been proposed in which small subsets of the network parameters are adapted [14, 15, 16].Linear transformations, which augment the SI network with certain speaker-specific linear layer(s), including linear input network (LIN) [17, 18], linear hidden network (LHN) [19] and linear output network (LOH) [18], were investigated. Furthermore, parameterized hidden activation functions have also been widely explored [20, 21, 22] and have achieved good performances.

Recently, researchers began training adaptive models on the fly instead of estimating the adaptive parameters from a well-trained SI model [23, 24]. In such approaches, SD auxiliary networks are adopted to improve adaptive training and are jointly optimized with the main network. These methods greatly simplify model adaptation by using only one-pass training and not requiring additional adaptation data.

Although the methods using SD auxiliary networks [23, 24] make adaptive training easier, they usually add an extra burden to the acoustic model. On the other hand, the regularization-based adaptation techniques in [11, 12, 13] do not require additional processing. Inspired by the work mentioned above, we integrate the regularization approaches into adaptive training and propose two novel regularization-based speaker adaptive training methods. The first method is center loss (CL) [25] regularization, where the center loss is used to penalize the distances between the embeddings of different speakers and the only center of all speaker classes. For the second method, we propose a novel regular loss function called the speaker variance loss (SVL). We directly minimize the speaker interclass variance during model training by using SVL regularization.

The essential idea of both proposed methods is to adapt the speaker variability by encouraging speaker interclass compactness, which measures the degree of mismatches. Both methods achieve the purpose of training an adaptive model on the fly by adding regularization terms to the training loss function. More importantly, they hardly add any complexity to the model: the CL only increases the number of parameters of one vector while the SVL does not increase any number of parameters. Considering that there is limited work on speaker adaptive training for the connectionist temporal classification (CTC) [26] model, we applied the proposed methods to CTC-based ASR in this paper. The experiments are conducted on the public Chinese dataset AISHELL-1 [27]. The experimental results show that, both methods are effective at speaker adaptation without requiring any specific fine-tuning or additional complexity, achieving up to 8.1% and 8.6% character error rate improvements over the speaker independent (SI) model, respectively.

The rest of this paper is organized as follows. Section 2

gives a brief description of the related work. We introduce the adaptive training approaches we proposed in Section 3. Section 4 shows our experimental setup and other details, including the experimental results. Finally, the discussion and conclusion are presented in Section 5.

2. Relation to prior work

The center loss (CL) [25] was first proposed to learn discriminative features for Face Recognition (FR) tasks. CL encourages intraclass compactness by penalizing the distances between the features of samples and their centers. To avoid the deeply learned features and centers degrading to zeros, researchers adopted the joint supervision of the softmax loss and CL to train neural networks. Via joint supervision, the CL pulls the features in the same class closer to their class center, and the softmax cross-entropy loss separates the features from different categories. It has performed remarkably on various benchmark datasets for face recognition. Since the CL enjoys the same requirement as the softmax loss and needs no complex recombination of the training samples, it can be easily extended to other tasks. Variants of these methods have also been successfully adopted in Speaker Recognition (SR) tasks [28, 29], automatic speech recognition (ASR) [30] tasks and speech emotion recognition (SER) [31] tasks.

Intuitively, the center loss function pulls the deep features of the same class to their corresponding centers. The purpose of speaker adaptive training in ASR is to normalize the speaker variability between the training and testing conditions. In other words, we want the deep features to contain as little speaker information as possible. The center loss must be tailored for the adaptive training; therefore, we use the center loss in this work to penalize the distances between different embeddings of speakers and the only center of all speaker classes. By minimizing such a center loss, different speaker categories approach the same center to achieve the purpose of normalizing speaker variability.

The purpose of adaptive training is to reduce the interspeaker variability of speech. In addition to the proposed center loss function, we further propose a novel loss function called the speaker variance loss (SVL), which directly minimizes the speaker interclass variance. Similar to the center loss, we use the SVL as a regulation term and adopt the joint supervision of the conventional CTC loss and the proposed SVL in model training. In this way, speaker interclass variance can be normalized as much as possible on the premise of ensuring the accurate classification of acoustic features.

Different from previous regularization-based speaker adaptation approaches [11, 12, 13], our proposed methods train the adaptive model on the fly by adding regularization terms to the training loss function. Therefore, we do not need any additional adaptation data to fine-tune the model parameters. This simplifies adaptive training while introducing little additional model complexity.

3. Proposed methods

An illustration of the proposed regularization-based speaker adaptation approach is shown in figure 1. We directly add the regular loss while training the acoustic model instead of using it to prevent overfitting when estimating speaker dependent (SD) parameters. The regularization encourages the speaker interclass compactness while the CTC loss encourages the separability of features. Consequently, the joint supervision of these two loss function minimizes the speaker variability while keeping the features of different classes separable. More details are discussed as follows.

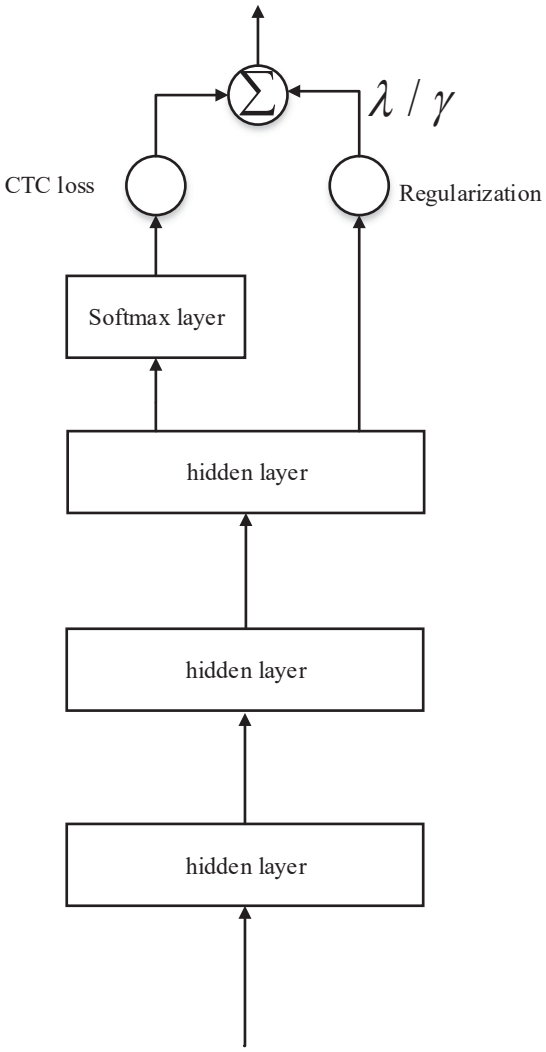


Figure 1: *Illustration of the proposed unsupervised regularization-based speaker adaptive training approach. The scalar λ or γ is used for balancing the CTC loss and the regular loss.*

3.1. Center Loss for adaptive training

Assuming that the training set contains a total of k speakers, we define the center loss function for adaptive training as follows:

$$\mathcal{L}_{\mathrm{c}} = \sum_{i=1}^{k} ||\mathbf{S}_i - \mathbf{C}||_2^2 \tag{1}$$

where $\mathbf{S}_i$ denotes the deep features of speaker i, $\mathbf{C}$ denotes the center of all speaker classes.

When we minimize the center loss, different speaker categories approach the same center, which is beneficial for the final sequence classification. As mentioned in [22], the center should be updated as the deep features change. In other words, we need to use the entire training set in each iteration, which is

inefficient and even impractical. Therefore, we make the necessary modification. Instead of updating the center with respect to the entire training set, we perform the update based on the mini-batch. Under this modification, the k in eq. (1) is redefined as the number of speaker classes within a mini-batch.

Then, the representation of speaker i, $\mathbf{S}_i$, can be easily calculated as follows:

$$\mathbf{S}_i = \frac{1}{\sum_t \mathbf{1}[s_t = i]} \sum_t \mathbf{1}[s_t = i]\mathbf{h}_t \tag{2}$$

where s_t denotes the speaker label of the t^{th} sample in the mini-batch, and $\mathbf{1}[.]$ is the indicator function that evaluates to 1 when its argument holds. $\mathbf{h}_t$ denotes the hidden activation of the second last layer.

We adopt the joint supervision of the classification loss and center loss in order to minimize the speaker variability as much as possible while retaining accurate sequence classification. The formulation is given as follows:

$$\mathcal{L}{=}\mathcal{L}_{\rm ctc} + \lambda\mathcal{L}_c \tag{3}$$

where the scalar λ is used for balancing the two loss functions. $\mathcal{L}_{\rm ctc}$ is the CTC loss function given by the following:

$$\mathcal{L}_{ctc} = -\sum_{(\mathbf{x},\mathbf{z})} \ln(\mathrm{p}(\mathbf{z}|\mathbf{x})) \tag{4}$$

where $(\mathbf{x}, \mathbf{z})$ are the training data pairs.

3.2. Speaker Variance Loss for adaptive training

According to the intuition behind the center loss, we propose a novel speaker variance loss (SVL) for speaker adaptation, which directly minimizes the speaker interclass variance. The formula is given as follows:

$$\mathcal{L}_{\rm sv} = ||\mathrm{var}(S_1, ..., S_i, ...S_k)||_2^2 \tag{5}$$

where $\mathrm{var}(S_1, ..., S_i, ...S_k)$ denotes the interclass variance of the k speakers. We then take into account the modification to update the mini-batch. The interclass variance can be calculated as follows:

$$\sigma_s^2 = \frac{1}{k}\sum_i (S_i - \mu_s)^2 \tag{6}$$

where μ_s is the mean of the k speaker classes. It is given by the following:

$$\mu_s = \frac{1}{k}\sum_i S_i \tag{7}$$

Then, the joint supervision of the CTC loss and SVL can be given as follows:

$$\mathcal{L}{=}\mathcal{L}_{\rm ctc} + \gamma\mathcal{L}_{sv} \tag{8}$$

where γ is the balance factor.

In fact, the SVL effectively characterizes the interclass variations of speakers. Compared with the modified center loss, the SVL targets more directly on the learning objective of the speaker interclass compactness, which is very beneficial for reducing speaker variability. More importantly, the SVL does not introduce any learnable parameters. In this way, it may avoid the local optimization caused by the random initialization of parameters to a certain extent.

Note that both proposed loss functions only take the hidden activation and do not need a complex recombination of the training samples. Therefore, their application for neural networks is more flexible. For example, the hidden activation used to calculate the speaker representation can be taken from any certain layer. In addition, each layer can add the regular loss direction to achieve layer-wise speaker adaptation. These will be discussed in detail in the experimental part.

4. Experiments

4.1. Dataset

We evaluate the proposed methods on an open-source Mandarin speech corpus AISHELL-1 [23]. All the speech files are sampled at 16K Hz with 16 bits. AISHELL-1 has 7,176 utterances from 20 speakers for evaluation (10 hours). We use 120,098 utterances from 340 speakers (150 hours) as the training set and 14,326 utterances from 40 speakers (20 hours) as the development set. The speakers of the training, development and test sets do not overlap.

4.2. Model setup

The PyTorch toolkit [32] is used in our model training process. All the model parameters are randomly initialized and updated by Adam [33]. The acoustic feature is 108-dimensional filter-bank features (36 filter-bank features, delta coefficients, and delta-delta coefficients) with mean and variance normalization. According to the statistical information of the transcripts, there are 4294 Chinese characters in the training set. Along with the added blanks, 4295 modeling units are used in the grapheme-based CTC system. The trigram language model is used in the decoding procedure.

The network is trained to minimize the CTC loss function with an initial learning rate of 0.0001. The development set is used for learning rate scheduling and early stopping. We start to halve the learning rate when the relative improvement falls below 0.004, and the training ends if the relative improvement is lower than 0.0005, which is usually approximately 13 epochs.

4.3. Network architecture

The acoustic modeling adopts a combination of CNNs and LSTM based RNNs for good performance as well as high efficiency. For this baseline, the bottom two layers are 2D convolution layers with 64 and 256 output channels. Each convolution layer is followed by a max-pooling layer with a stride of 2 in the time dimension to finally down sample an utterance to a quarter of its original length. After the CNN layers, there are three LSTM layers, each of which is a bidirectional LSTM layer with 512 units. We also use a dropout rate of 0.3 for the LSTM layers to avoid overfitting.

4.4. Results

We first investigate the sensitiveness of the balance factor of the standard CL regularization and SVL regularization in which only the last LSTM layer is adapted.

Table 1 shows the character error rate (CER) of the adaptive model with CL regularization and SVL regularization under different hyper parameters λ and γ. As shown in the table, at first, as the balance factor increases, the CER gradually decreases; and then when the balance factor continues to increase, the CER gradually increases. It is speculated that when the interclass variance is excessively penalized, some features become indistinguishable, which is not conducive to sequence classification. Finally, with only the third layer punished, the CL and SVL adaptive models achieve CER reductions of 5.6% and 5.8% reduction over SI model, respectively.

Table 1: *The CERs (%) of the CL and SVL adaptive models under different balance factors.*

CL		SVL	
λ	CER(%)	γ	CER(%)
0	9.96	0	9.96
0.005	9.88	1	9.91
0.01	9.69	10	9.64
0.1	**9.40**	25	**9.38**
1	9.68	30	9.43
5	9.92	50	9.80

Table 2: *The CERs (%) of the CL and SVL models with different adapted LSTM layers.*

Adapted LSTM layer	CL	SVL
SI	9.96	9.96
1	9.55	9.55
2	9.46	9.49
3	9.40	9.38
2,3	9.31	9.30
1,2,3	**9.15**	**9.10**

In the following experiments, we investigate how many and which hidden layers should be used in adaptive training. The combinations of different adaptation layers are investigated. The results are summarized in Table 2. Note that each layer corresponds to a separate center for multilayer adaptation in the CL adaptive model. For each adaptive model, the balance factor has been adjusted to achieve the best performance. It can be seen that the highest layer is most important for adaptation. If only one LSTM layer is adapted, the higher layer (3rd layer) can achieve better performance than the lower layer (1st layer). Furthermore, the CER steadily decreases as the number of adapted layers increases. When all three LSTM layers are used for adaptive training, the proposed CL and SVL adaptive models achieve CER reductions of 8.1% and 8.6% over the SI model, respectively.

5. Conclusions

In this work, we propose two novel regularization-based speaker adaptation approaches, center loss (CL) regularization and speaker variance loss (SVL) regularization. The idea of the proposed methods is to reduce the speaker variability by encouraging speaker interclass compactness, which measures the degree of mismatches. Different from previous work, both methods train an adaptive model on the fly by adding regularization terms to the training loss function. Moreover, they hardly add any complexity to the acoustic model: the CL only increases the number of parameters of one vector while the SVL does not increase any number of parameters. The experimental results show that both methods are effective at adapting the CTC model, achieving CER improvements of up to 8.1% and 8.6% over the SI model, respectively.

In the following work, we will investigate how to achieve more effective speaker representation for calculating the regular loss. Attention mechanisms and other schemes may be introduced to enhance the hidden activation used to extract speaker embeddings.

6. Acknowledgements

This work was partially funded by the National Key Research and Development Program of China (Grant No. 2016YFB100 1303) and the National Natural Science Foundation of China (Grant No. U1836219).

7. References

[1] G. Hinton, L. Deng, D. Yu *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, 2012.

[2] D. Yu and J. Li, "Recent progresses in deep learning based acoustic models," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 3, pp. 396–409, 2017.

[3] A.-r. Mohamed, T. N. Sainath, G. E. Dahl *et al.*, "Deep belief networks using discriminative features for phone recognition." in *ICASSP*, 2011, pp. 5060–5063.

[4] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, 2011, pp. 24–29.

[5] 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.

[6] Y. Miao, H. Zhang, and F. Metze, "Towards speaker adaptive training of deep neural network acoustic models," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[7] A. Senior and I. Lopez-Moreno, "Improving dnn speaker independence with i-vector inputs," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 225–229.

[8] P. Karanasou, Y. Wang, M. J. Gales, and P. C. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[9] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7942–7946.

[10] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, "Direct adaptation of hybrid dnn/hmm model for fast speaker adaptation in lvcsr based on speaker code," in *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2014, pp. 6339–6343.

[11] L. Hank, "Speaker adaptation of context dependent deep neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7947–7951.

[12] D. Yu, K. Yao, H. Su *et al.*, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7893–7897.

[13] Z. Meng, J. Li, and Y. Gong, "Adversarial speaker adaptation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5721–5725.

[14] K. Yao, D. Yu, F. Seide *et al.*, "Adaptation of context-dependent deep neural networks for automatic speech recognition," in *2012 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2012, pp. 366–369.

[15] S. M. Siniscalchi, J. Li, and C.-H. Lee, "Hermitian polynomial for speaker adaptation of connectionist speech recognition systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2152–2161, 2013.

[16] L. Samarakoon and K. C. Sim, "Factorized hidden layer adaptation for deep neural network based acoustic modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 12, pp. 2241–2250, 2016.

[17] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, "Speaker-adaptation for hybrid hmm-ann continuous speech recognition system," 1995.

[18] B. Li and K. C. Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid nn/hmm systems," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[19] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. De Mori, "Linear hidden transformations for adaptation of hybrid ann/hmm models," *Speech Communication*, vol. 49, no. 10-11, pp. 827–835, 2007.

[20] P. Swietojanski, J. Li, and S. Renals, "Learning hidden unit contributions for unsupervised acoustic model adaptation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1450–1463, 2016.

[21] P. Swietojanski and S. Renals, "Differentiable pooling for unsupervised speaker adaptation," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4305–4309.

[22] C. Zhang and P. C. Woodland, "Dnn speaker adaptation using parameterised sigmoid and relu hidden activation functions," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5300–5304.

[23] F. Ding, W. Guo, L. Dai, and J. Du, "Attention-based gated scaling adaptative acoustic model for ctc-based speech recognition," *arXiv preprint arXiv:1912.13307*, 2019.

[24] L. Sarı, S. Thomas, and M. A. Hasegawa-Johnson, "Learning speaker aware offsets for speaker adaptation of neural networks," *Proc. Interspeech 2019*, pp. 769–773, 2019.

[25] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European conference on computer vision*. Springer, 2016, pp. 499–515.

[26] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[27] H. Bu, J. Du, X. Na *et al.*, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. IEEE, 2017, pp. 1–5.

[28] S. Yadav and A. Rai, "Learning discriminative features for speaker identification and verification." in *Interspeech*, 2018, pp. 2237–2241.

[29] N. Li, D. Tuo, D. Su, Z. Li, D. Yu, and A. Tencent, "Deep discriminative embeddings for duration robust speaker verification." in *Interspeech*, 2018, pp. 2262–2266.

[30] J. Wang, D. Su, J. Chen, S. Feng, D. Ma, N. Li, and D. Yu, "Learning discriminative features in sequence training without requiring framewise labelled data," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5696–5700.

[31] D. Dai, Z. Wu, R. Li, X. Wu, J. Jia, and H. Meng, "Learning discriminative features from spectrograms using center loss for speech emotion recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 7405–7409.

[32] A. Paszke, S. Gross, S. Chintala *et al.*, "Automatic differentiation in pytorch," 2017.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Knowledge-and-Data-Driven Amplitude Spectrum Prediction for Hierarchical Neural Vocoders

Yang Ai, Zhen-Hua Ling

National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China, Hefei, P.R.China

ay8067@mail.ustc.edu.cn, zhling@ustc.edu.cn

Abstract

In our previous work, we have proposed a neural vocoder called HiNet which recovers speech waveforms by predicting amplitude and phase spectra hierarchically from input acoustic features. In HiNet, the amplitude spectrum predictor (ASP) predicts log amplitude spectra (LAS) from input acoustic features. This paper proposes a novel knowledge-and-data-driven ASP (KDD-ASP) to improve the conventional one. First, acoustic features (i.e., F0 and mel-cepstra) pass through a knowledge-driven LAS recovery module to obtain approximate LAS (ALAS). This module is designed based on the combination of STFT and source-filter theory, in which the source part and the filter part are designed based on input F0 and mel-cepstra, respectively. Then, the recovered ALAS are processed by a data-driven LAS refinement module which consists of multiple trainable convolutional layers to get the final LAS. Experimental results show that the HiNet vocoder using KDD-ASP can achieve higher quality of synthetic speech than that using conventional ASP and the WaveRNN vocoder on a text-to-speech (TTS) task.

Index Terms: neural vocoder, log amplitude spectrum, source-filter, TTS

1. Introduction

Nowadays, statistical parametric speech synthesis (SPSS) has become a popular text-to-speech (TTS) approach thanks to its flexibility and high quality. Both acoustic models which predict acoustic features (e.g., mel-cepstra and F0) from texts and vocoders [1] which reconstruct speech waveforms from predicted acoustic features are essential in SPSS. Early SPSS systems preferred to adopt conventional vocoders, such as STRAIGHT [2] and WORLD [3] as their vocoders. These vocoders are designed based on the source-filter model of speech production [4] and have some limitations, such as the loss of phase information and spectral details.

Recently, some autoregressive neural generative models such as WaveNet [5], SampleRNN [6] and WaveRNN [7] have been proposed and achieved good performance on generating raw audio signals. Their variants such as knowledge-distilling-based models (e.g., parallel WaveNet [8] and ClariNet [9]) and flow-based models (e.g., WaveGlow [10]) were also proposed to further improve the performance and generation efficiency. Based on these waveform generation models, neural vocoders have been developed [11, 12, 13, 14, 15, 16], which reconstruct speech waveforms from various acoustic features for SPSS, voice conversion [17, 18], bandwidth extension [19], etc. Although these neural vocoders outperformed the conventional ones significantly, they still have some limitations. The autoregressive neural vocoders have low generation efficiency due to their point-by-point generation process. For knowledge-distilling-based vocoders and flow-based vocoders, it is difficult to train them due to their complicated training process and high complexity of model structures respectively.

Subsequently, some improved neural vocoders, such as glottal neural vocoder [20,21], LPCNet [22], and neural source-filter (NSF) vocoder [23,24,25,26], have been further proposed. These vocoders combine speech production mechanisms with neural networks and have also demonstrated impressive performance. In our previous work [27], we proposed a neural vocoder named HiNet, which consists of an amplitude spectrum predictor (ASP) and a phase spectrum predictor (PSP). HiNet produces speech waveforms by first predicting amplitude spectra from input acoustic features using ASP and then predicting phase spectra from amplitude spectra using PSP. The outputs of ASP and PSP are combined to recover speech waveforms by short-time Fourier synthesis (STFS). Besides, generative adversarial networks (GANs) [28] are also introduced into ASP and PSP to further improve their performance. Experimental results show that the proposed HiNet vocoder can generate waveforms with high quality and high efficiency.

In this paper, we propose a novel knowledge-and-data-driven ASP (KDD-ASP) to replace the conventional one in a HiNet vocoder. The aim of KDD-ASP is to integrate speech production and analysis knowledge into data-driven LAS prediction, expecting to improve the accuracy and generalization ability of ASP, especially when predicted acoustic features are used as input. KDD-ASP consists of a knowledge-driven LAS recovery module and a data-driven LAS refinement module. The first module is designed based on the combination of STFT and the source-filter theory of speech production, and generates approximate LAS (ALAS) from input acoustic features (i.e., F0 and mel-cepstra). We assume that the speech signal is produced via a source-filter process [4]. The source excitation signal and the filter are designed according to the input F0 and mel-cepstra respectively. Then, ALAS can be calculated by imitating the process of STFT which includes truncation, windowing and FFT. All operations are performed in the frequency domain. The second module predicts the final LAS from ALAS. This module consists of multiple trainable convolutional layers and is trained in a data-driven way. Experimental results confirm that the HiNet vocoder using KDD-ASP can achieve higher quality of synthetic speech than that using conventional ASP and the WaveRNN vocoder on a TTS task.

This paper is organized as follows. In Section 2, we briefly review the HiNet vocoder [27]. In Section 3, we describe the details of our proposed KDD-ASP. Section 4 reports our experimental results. Conclusions are given in Section 5.

This work was partially funded by the National Key R&D Program of China under Grant 2019YFF0303001 and the National Nature Science Foundation of China under Grant 61871358 and U1613211.

2. HiNet vocoder

HiNet [27] is a novel neural vocoder which recovers speech waveforms by predicting amplitude and phase spectra hierarchically from input acoustic features. Conventional neural vocoders usually employ single neural networks to generate speech waveforms directly. In contrast, the HiNet vocoder consists of an amplitude spectrum predictor (ASP) and a phase spectrum predictor (PSP). ASP uses acoustic features as input and predicts frame-level log amplitude spectra (LAS). Then PSP uses the predicted LAS and F0 as input and recovers the phase spectra. Finally, the outputs of ASP and PSP are combined to recover speech waveforms by short-time Fourier synthesis (STFS).

In our implement, ASP is a simple non-autoregressive DNN containing multiple feed-forward (FF) layers. It concatenates the acoustic features at current and previous frames as input to predict the LAS at current frame. At the training stage, the target LAS are extracted from natural waveforms by STFT. A GAN criterion is adopted to build ASP. The DNN model is used as the generator of GAN and its discriminator consists of multiple convolutional layers which operate along the frequency axis of the input LAS. A Wasserstein GAN [29] loss is combined with the mean square error (MSE) between the predicted LAS and natural ones to train the generator.

PSP is constructed by concatenating a neural waveform generator with a phase spectrum extractor. The neural waveform generator is built by adapting the NSF vocoder [23] from three aspects, 1) using LAS as the input, 2) pre-calculating the initial phase of the sine-based excitation signal for each voiced segment at the training stage and 3) adopting a combined loss function including MSE on amplitude spectra, waveform loss and correlation loss. GAN is also introduced into PSP. Here, the neural waveform generator of PSP is used as the generator of GAN and its discriminator is similar with that of ASP except that its input features are waveforms instead of LAS.

3. Knowledge-and-Data-Driven ASP

This paper proposes a novel knowledge-and-data-driven ASP (KDD-ASP) to replace the conventional one in a HiNet vocoder. The KDD-ASP is constructed by concatenating an LAS recovery module which refers to the knowledge of source-filtering speech production with an LAS refinement module which is trained using a corpus in a data-driven way as shown in Fig. 1.

3.1. Knowledge-driven LAS recovery module

The equation for extracting LAS directly from a signal $\boldsymbol{s}$ by STFT can be written as follows,

$$\boldsymbol{LAS}_n = \log|\mathcal{F}(\boldsymbol{s}_n \odot \boldsymbol{w})|, \tag{1}$$

where $\boldsymbol{s}_n = [s_{n,1}, \ldots, s_{n,L}]^\top$ and $\boldsymbol{LAS}_n = [LAS_{n,1} \ldots, LAS_{n,K}]^\top$ are the framed signal of $\boldsymbol{s}$ and the LAS at the n-th frame respectively, and $\boldsymbol{w} = [w_1, \ldots, w_L]^\top$ denotes the Hanning window for short-time analysis. L is the frame number. $K = \frac{FN}{2} + 1$ represents the number of frequency bins and FN is the FFT point number. $\odot$ and $\mathcal{F}$ represent element-wise product and FFT, respectively.

Inspired by this process, the knowledge-driven LAS recovery module constructs approximate LAS (ALAS) from F0 and mel-cepstra based on the frequency-domain representation of Eq. (1). We assume that the speech signal at the n-th frame

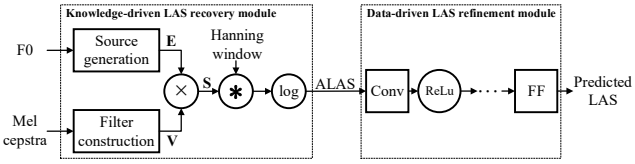


Figure 1: *Model structure of KDD-ASP. Here,* $\times$, $*$ *and* log *denote element-wise product, convolutional and log operation respectively,* FF *and* Conv *represent feed-forward and convolutional layers respectively and* ReLu *means rectified linear units.*

$\boldsymbol{s}_n$ is obtained by the convolution between a source excitation signal $\boldsymbol{e}_n$ and a filter impulse response $\boldsymbol{v}_n$. In frequency domain, this process can be represented as

$$\boldsymbol{S}_n = \boldsymbol{E}_n \odot \boldsymbol{V}_n, \tag{2}$$

where $\boldsymbol{S}_n = [S_{n,1}, \ldots, S_{n,K}]^\top$, $\boldsymbol{E}_n = [E_{n,1}, \ldots, E_{n,K}]^\top$ and $\boldsymbol{V}_n = [V_{n,1}, \ldots, V_{n,K}]^\top$ are the Fourier transform of $\boldsymbol{s}_n$, $\boldsymbol{e}_n$ and $\boldsymbol{v}_n$ respectively.

Let f_n denote the F0 value of the n-th frame when it is voiced and $f_n = 0$ when the frame is unvoiced. For voiced frames ($f_n > 0$), $\boldsymbol{E}_n$ is produced as a pulse train with equal frequency interval $K_0 = Round(\frac{f_n}{F_s} \cdot FN)$, which corresponds to constructing all the harmonics below the Nyquist frequency, where F_s is the sampling rate. For unvoiced frames ($f_n = 0$), we set $E_n \equiv 1$, meaning that the excitation signal is a Gaussian white noise. The equation for producing $\boldsymbol{E}_n$ based on F0 values can be written as

$$E_{n,k} = \begin{cases} 1, & f_n > 0, k = i \cdot K_0 \\ 0, & f_n > 0, k \neq i \cdot K_0 \text{ or } f_n = 0 \end{cases}, \tag{3}$$

where $i = 1, 2, \ldots, \lceil \frac{K}{K_0} \rceil$.

$\boldsymbol{V}_n$ is calculated by transforming mel-cepstra to amplitude spectra [30]. The mel-cepstral coefficients at the n-th frame (with energy as the first order) are first padded with zeros to form a K-dimensional vector $\boldsymbol{m}_n = [m_{n,1}, \ldots, m_{n,K}]^\top$. Then, the cepstral coefficients $c_{n,k}, k = 1, \ldots, K$ are calculated by the following iterative formulas

$$c_{n,k}(i) = \begin{cases} m_{n,i} - \alpha \cdot c_{n,1}(i+1), & k=1 \\ (1-\alpha^2) \cdot c_{n,1}(i+1) - \alpha \cdot c_{n,2}(i+1), & k=2 \\ c_{n,k-1}(i+1) - \alpha \cdot [c_{n,k}(i+1) - c_{n,k-1}(i)], & k>2 \end{cases}, \tag{4}$$

where i iterates from K to 1 with the initial value $c_{n,k}(K+1) = 0, k = 1, \ldots, K$. α is the mel-frequency warping coefficient, which is 0.42 for $F_s = 16000$. After the iteration, we can obtain the cepstra vector $\boldsymbol{c}_n = [c_{n,1}(1), \ldots, c_{n,K}(1)]^\top$, which is further transformed to the amplitude spectra $\boldsymbol{V}_n$ by

$$\boldsymbol{V}_n = \exp[\mathcal{F}(\boldsymbol{c}_n)]. \tag{5}$$

Finally, ALAS can be calculated as

$$\boldsymbol{ALAS}_n = \log|\boldsymbol{S}_n * \boldsymbol{W}|, \tag{6}$$

where $\boldsymbol{ALAS}_n = [ALAS_{n,1} \ldots, ALAS_{n,K}]^\top$ is the n-th frame ALAS and $\boldsymbol{W} = [W_1, \ldots, W_K]^\top$ is the Fourier transform of the analysis window $\boldsymbol{w}$. The operation $*$ represents convolution. It is worth mentioning that the elements in the vectors of $\boldsymbol{S}_n$ and $\boldsymbol{W}$ should be rearranged by complementing their mirror-symmetric parts and shifting the zero-frequency component to the center before convolution.

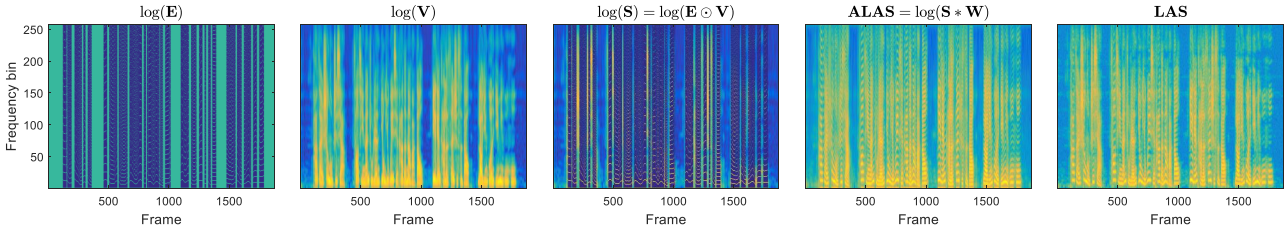


Figure 2: *The visualization of* $\log(\boldsymbol{E}_n)$, $\log(\boldsymbol{V}_n)$, $\log(\boldsymbol{S}_n)$, $\boldsymbol{ALAS}_n$ *and* $\boldsymbol{LAS}_n$ *for an example utterance. Here, the input F0 and mel-cepstra are natural ones.*

3.2. Data-driven LAS refinement module

The data-driven LAS refinement module converts ALAS to final LAS by a trainable neural network. In our implement, this module adopts the ASP model in Section 2 but has two structural improvements. First, convolutional layers are used instead of FF layers in the generator and the input is the ALAS at current frame instead of the concatenated ones as shown in Fig. 1. Second, another discriminator which operates along with the time axis of the input LAS is added[1].

4. Experiments

4.1. Experimental conditions

A Chinese speech synthesis corpus with 13334 utterances (∼20 hours) was used in our experiments. The speaker was a female and the waveforms had 16 kHz sampling rate with 16 bits resolution. The training, validation and test sets contained 13134, 100 and 100 utterances, respectively. The natural acoustic features were extracted with a frame length and shift of 25 ms and 5 ms respectively. The acoustic features at each frame were 43-dimensional including 40-dimensional mel-cepstra, an energy, an F0 and a V/UV flag. For SPSS, a bidirectional LSTM-RNN acoustic model [31] having 2 hidden layers with 1024 units per layer (512 forward units and 512 backward units) was trained as the acoustic model, which predicted acoustic features from 566-dimensional linguistic features. The output of the acoustic model was 127-dimensional including 43-dimensional acoustic features together with their delta and acceleration counterparts (the V/UV flag had no dynamic components). Then, the predicted acoustic features were generated from the output by maximum likelihood parameter generation (MLPG) [32] considering global variance (GV) [33]. Since this paper focuses on vocoders, natural durations obtained by HMM-based forced alignment were used at synthesis time.

Three vocoders were compared in our experiments[2]. The descriptions of these vocoders are as follows.

1) ***WaveRNN*** A 16-bit WaveRNN-based neural vocoder using acoustic features as input. This vocoder was implemented by ourselves and the efficiency optimization strategies [7] were not adopted here. Its structure was the same as ***WaveRNN*** in our previous work [27] which performed better than the 16-bit WaveNet vocoder using open source implementation[3]. The waveform samples were quantized to discrete values by 16-bit linear quantization and the model had one hidden layer with 1024 nodes where 512 nodes for coarse outputs and another 512 nodes for fine outputs. Models were trained and evaluated on a single Nvidia 1080Ti GPU using TensorFlow [34].

2) ***HiNet*** A HiNet vocoder using conventional ASP. The structure of ASP is the same with that of the data-driven LAS refinement module introduced in Section 3.2. When extracting natural LAS, the frame length and frame shift of STFT were 20ms (i.e., $L = 320$) and 5ms respectively and FFT point number was 512 (i.e., $K = 257$). There were 3 convolutional layers with 2048 nodes per layer (filter width=7), and a 257-dimensional linear output layer which predicted the LAS. For each training step, ASP used 128 frames of acoustic features as input and outputted corresponding 128 frames of LAS. GANs were also used in ASP. Discriminator #1 operated along with the frequency axis and consisted of 6 convolutional layers (filter width=9, stride size=2) and their channels were 16, 32, 64, 128 and 256 respectively. The resulting dimensions per layer, being it frequency bins × channels, were 257×1, 129×16, 65×32, 33×64, 17×128 and 9×256. Finally, two FF layers with 256 and 9 nodes respectively were used to map the 9×256 convolutional results into a value for loss calculation. Discriminator #2 operated along with the time axis and consisted of 4 convolutional layers (filter width=9, stride size=2) and their channels were 64, 128, 256 and 512 respectively. The resulting dimensions per layer, being it frequency bins × channels, were 128×257, 64×64, 32×128, 16×256 and 8×512. Finally, two FF layers with 512 and 8 nodes respectively were used to map the 8×512 convolutional results into a value for loss calculation. Remaining settings of ASP and all the settings of PSP are the same as the ***HiNet-S-GAN*** vocoder in our previous work [27]. ASP and PSP models were both trained and evaluated on a single Nvidia 1080Ti GPU using TensorFlow framework [34].

3) ***HiNet-KDD*** A HiNet vocoder using the KDD-ASP proposed in this paper. For KDD-ASP, the knowledge-driven LAS generation module adopted the same settings with that of extracting natural LAS (i.e., $L = 320$ and $K = 257$) and the settings of the data-driven LAS refinement module were the same as the ASP of ***HiNet***. The settings of PSP and the implementation conditions were all the same as that of ***HiNet***. Fig. 2 shows the visualization of $\boldsymbol{E}_n$, $\boldsymbol{V}_n$, $\boldsymbol{S}_n$, $\boldsymbol{ALAS}_n$ and $\boldsymbol{LAS}_n$ for all frames in an example utterance. We can see that the recovered ALAS is close to the reference LAS with analogous harmonic and formant structures, meaning that the input and output of the data-driven LAS refinement module are similar, expecting to facilitate the model learning and to improve the performance of predicting amplitude spectra.

4.2. Objective evaluation

We first compared the performance of these three vocoders using objective evaluations. Five objective metrics used in our

[1]Discriminators are not shown in Fig. 1 for simplification.

[2]Examples of generated speech can be found at `http://home.ustc.edu.cn/~ay8067/Interspeech2020/demo.html`.

[3]`https://github.com/r9y9/wavenet_vocoder`.

Table 1: *Objective evaluation results of* **WaveRNN**, **HiNet** *and* **HiNet-KDD** *on the test set. "AS" stands for analysis-synthesis task and "TTS" stands for TTS task.*

		WaveRNN	***HiNet***	***HiNet-KDD***
AS	SNR(dB)	4.6631	**5.2587**	5.0152
	LAS-RMSE(dB)	4.9623	**4.2602**	4.5659
	MCD-V(dB)	1.0702	**0.7686**	0.8583
	F0-RMSE(cent)	13.2365	9.3345	**9.0960**
	V/UV error(%)	4.2515	2.0116	**2.0041**
TTS	MCD-V(dB)	1.0702	1.0939	**0.9488**
	F0-RMSE(cent)	12.4645	7.0877	**6.4970**
	V/UV error(%)	3.5247	**1.7983**	2.0194

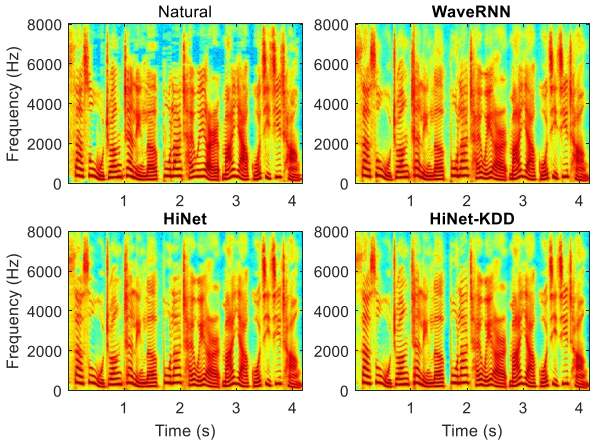


Figure 3: *The spectrograms of natural speech and the speech generated by* **WaveRNN**, **HiNet** *and* **HiNet-KDD** *on TTS task for an example sentence in the test set.*

previous work [27] were adopted here, including signal-to-noise ratio (SNR), root MSE (RMSE) of LAS (denoted by LAS-RMSE), mel-cepstrum distortion for voiced frames (denoted by MCD-V), MSE of F0 (denoted by F0-RMSE) and V/UV error. For the analysis-synthesis (AS) task, the references are natural waveforms or the acoustic features extracted from natural waveforms. For the TTS task, the references are the mel-cepstra and F0 predicted by the acoustic model and only MCD-V, F0-RMSE and V/UV error were adopted since the calculation of SNR and LAS-RMSE relied on natural speech waveforms.

The objective results on the test set are listed in Table 1. It is obvious that both ***HiNet*** and ***HiNet-KDD*** outperformed ***WaveRNN*** on most metrics for both AS and TTS tasks. By comparing ***HiNet*** and ***HiNet-KDD***, we can find that ***HiNet-KDD*** performed better on F0-RMSE than ***HiNet*** for both AS and TTS tasks, which indicated that ***HiNet-KDD*** is better at restoring harmonics for voiced frames. Considering the SNR, LAS-RMSE and MCD-V for AS task, ***HiNet-KDD*** was not as good as ***HiNet***. However, for TTS task, ***HiNet-KDD*** achieved better MCD-V than ***HiNet***. This advantage can be attributed to that using ALAS as the input to train the ASP model improves its generalization ability when dealing with unseen acoustic features. We also draw the spectrograms extracted from natural waveforms and from the waveforms generated by these three vocoders on TTS task in Fig. 3. We can see that ***HiNet-KDD*** can restore more clear harmonics (e.g., 0.7∼1.0s and 1.7∼2.0s) especially in the high-frequency band than the other two vocoders.

Table 2: *Average preference scores (%) on naturalness among different vocoders, where N/P stands for "no preference" and p denotes the p-value of a t-test between two vocoders. "AS" stands for analysis-synthesis task and "TTS" stands for TTS task.*

	WaveRNN	***HiNet***	***HiNet-KDD***	*N/P*	p
AS	2.73	**72.73**	–	24.54	< 0.01
	–	21.36	15.45	63.19	0.15
TTS	16.82	**57.27**	–	25.91	< 0.01
	10.91	–	**66.82**	22.27	< 0.01
	–	14.55	**53.64**	31.81	< 0.01

4.3. Subjective evaluation

Five groups of ABX preference tests were conducted to compare the subjective performance of different vocoders. In each subjective test, 20 utterances generated by two comparative vocoders were randomly selected from the test set. Each pair of generated speech were evaluated in random order. 11 Chinese native speakers were asked to judge which utterance in each pair had better naturalness or there was no preference. The p-value of a t-test was also calculated to measure the significance of the difference between two comparative vocoders.

The subjective results are shown in Table 2. We can see that ***HiNet*** outperformed ***WaveRNN*** very significantly ($p < 0.01$) on both AS and TTS tasks. However, the preference difference between these two vocoders became weaker on TTS task than on AS task. Comparing ***HiNet*** with ***HiNet-KDD***, we can see that there was no significant difference ($p > 0.05$) between these two vocoders on AS task but ***HiNet-KDD*** outperformed ***HiNet*** significantly ($p < 0.01$) on TTS task. We also conducted a group of ABX test between ***WaveRNN*** and ***HiNet-KDD*** for TTS task and ***HiNet-KDD*** also outperformed ***HiNet*** significantly ($p < 0.01$). Besides, the preference score difference between ***HiNet-KDD*** and ***WaveRNN*** was larger than that between ***HiNet*** and ***WaveRNN***. These results all indicated that using KDD-ASP in HiNet vocoder was helpful for improving the quality of reconstructed speech waveforms when the input acoustic features were predicted for TTS.

5. Conclusion

In this paper, we have proposed a novel knowledge-and-data-driven amplitude spectrum predictor (KDD-ASP) to replace the conventional one in HiNet, a hierarchical neural vocoder. KDD-ASP consists of a knowledge-driven LAS recovery module and a data-driven LAS refinement module. The first module is designed based on the combination of STFT and source-filter theories in order to convert F0 and mel-cepstra into approximate log amplitude spectra (ALAS). The input F0 values are used to produce the source signal and the filter part is calculated from mel-cepstra. The second module is a convolutional neural network which adopts GANs and predicts the final LAS from input ALAS. Experimental results show that the HiNet vocoder using KDD-ASP can achieve higher quality of synthetic speech than the HiNet vocoder using conventional ASP and the WaveRNN vocoder on a TTS task. To explore other knowledge-driven methods for ASP and further improve the performance of phase spectrum prediction will be the tasks of our future research.

6. References

[1] H. Dudley, "The vocoder," *Bell Labs Record*, vol. 18, no. 4, pp. 122–126, 1939.

[2] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne, "Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based f0 extraction: Possible role of a repetitive structure in sounds," *Speech communication*, vol. 27, no. 3, pp. 187–207, 1999.

[3] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Transactions on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[4] F. Gunnar, *The acoustic theory of speech production*. The Hague, The Netherlands: Mouton, 1960.

[5] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," in *9th ISCA Speech Synthesis Workshop*, 2016, pp. 125–125.

[6] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *Proc. ICLR*, 2017.

[7] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *Proc. ICML*, 2018, pp. 2410–2419.

[8] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg *et al.*, "Parallel WaveNet: Fast high-fidelity speech synthesis," in *Proc. ICML*, 2018, pp. 3918–3926.

[9] W. Ping, K. Peng, and J. Chen, "ClariNet: Parallel wave generation in end-to-end text-to-speech," in *Proc. ICLR*, 2019.

[10] R. Prenger, R. Valle, and B. Catanzaro, "WaveGlow: A flow-based generative network for speech synthesis," in *Proc. ICASSP*, 2019, pp. 3617–3621.

[11] A. Tamamori, T. Hayashi, K. Kobayashi, K. Takeda, and T. Toda, "Speaker-dependent WaveNet vocoder," in *Proc. Interspeech*, 2017, pp. 1118–1122.

[12] T. Hayashi, A. Tamamori, K. Kobayashi, K. Takeda, and T. Toda, "An investigation of multi-speaker training for WaveNet vocoder," in *Proc. ASRU*, 2017, pp. 712–718.

[13] N. Adiga, V. Tsiaras, and Y. Stylianou, "On the use of WaveNet as a statistical vocoder," in *Proc. ICASSP*, 2018, pp. 5674–5678.

[14] Y. Ai, H.-C. Wu, and Z.-H. Ling, "SampleRNN-based neural vocoder for statistical parametric speech synthesis," in *Proc. ICASSP*, 2018, pp. 5659–5663.

[15] Y. Ai, J.-X. Zhang, L. Chen, and Z.-H. Ling, "DNN-based spectral enhancement for neural waveform generators with low-bit quantization," in *Proc. ICASSP*, 2019, pp. 7025–7029.

[16] J. Lorenzo-Trueba, T. Drugman, J. Latorre, T. Merritt, B. Putrycz, R. Barra-Chicote, A. Moinet, and V. Aggarwal, "Towards achieving robust universal neural vocoding," in *Proc. Interspeech*, 2019, pp. 181–185.

[17] L.-J. Liu, Z.-H. Ling, Y. Jiang, M. Zhou, and L.-R. Dai, "WaveNet vocoder with limited training data for voice conversion," in *Proc. Interspeech*, 2018, pp. 1983–1987.

[18] K. Kobayashi, T. Hayashi, A. Tamamori, and T. Toda, "Statistical voice conversion with WaveNet-based waveform generation," in *Proc. Interspeech*, 2017, pp. 1138–1142.

[19] Z.-H. Ling, Y. Ai, Y. Gu, and L.-R. Dai, "Waveform modeling and generation using hierarchical recurrent neural networks for speech bandwidth extension," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 5, pp. 883–894, 2018.

[20] Y. Cui, X. Wang, L. He, and F. K. Soong, "A new glottal neural vocoder for speech synthesis," in *Proc. Interspeech*, 2018, pp. 2017–2021.

[21] L. Juvela, V. Tsiaras, B. Bollepalli, M. Airaksinen, J. Yamagishi, and P. Alku, "Speaker-independent raw waveform model for glottal excitation," in *Proc. Interspeech*, 2018, pp. 2012–2016.

[22] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," in *Proc. ICASSP*, 2019, pp. 5891–5895.

[23] X. Wang, S. Takaki, and J. Yamagishi, "Neural source-filter-based waveform model for statistical parametric speech synthesis," in *Proc. ICASSP*, 2019, pp. 5916–5920.

[24] X. Wang and J. Yamagishi, "Neural harmonic-plus-noise waveform model with trainable maximum voice frequency for text-to-speech synthesis," in *Proc. SSW*, 2019, pp. 1–6.

[25] X. Wang, S. Takaki, and J. Yamagishi, "Neural source-filter waveform models for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 402–415, 2019.

[26] Y. Zhao, X. Wang, L. Juvela, and J. Yamagishi, "Transferring neural speech waveform synthesizers to musical instrument sounds generation," in *Proc. ICASSP*, 2020, pp. 6269–6273.

[27] Y. Ai and Z.-H. Ling, "A neural vocoder with hierarchical generation of amplitude and phase spectra for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 839–851, 2020.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[29] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.

[30] K. Tokuda, T. Kobayashi, T. Masuko, and S. Imai, "Mel-generalized cepstral analysis-a unified approach to speech spectral estimation," in *Proc. ICSLP*, 1994, pp. 1043–1046.

[31] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *Proc. Interspeech*, 2014, pp. 1964–1968.

[32] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Proc. ICASSP*, vol. 3, 2000, pp. 1315–1318.

[33] T. Toda and K. Tokuda, "A speech parameter generation algorithm considering global variance for HMM-based speech synthesis," *IEICE Transactions on Information and Systems*, vol. 90, no. 5, pp. 816–824, 2007.

[34] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," 2015. [Online]. Available: http://download.tensorflow.org/paper/whitepaper2015.pdf

Reverberation Modeling for Source-Filter-based Neural Vocoder

Yang Ai[1]*, Xin Wang*[2]*, Junichi Yamagishi*[2,3]*, Zhen-Hua Ling*[1]

[1]NELSLIP, University of Science and Technology of China, Hefei, P.R.China
[2]National Institute of Informatics, Japan, [3]CSTR, University of Edinburgh, UK

ay8067@mail.ustc.edu.cn, wangxin@nii.ac.jp, jyamagis@nii.ac.jp, zhling@ustc.edu.cn

Abstract

This paper presents a reverberation module for source-filter-based neural vocoders that improves the performance of reverberant effect modeling. This module uses the output waveform of neural vocoders as an input and produces a reverberant waveform by convolving the input with a room impulse response (RIR). We propose two approaches to parameterizing and estimating the RIR. The first approach assumes a global time-invariant (GTI) RIR and directly learns the values of the RIR on a training dataset. The second approach assumes an utterance-level time-variant (UTV) RIR, which is invariant within one utterance but varies across utterances, and uses another neural network to predict the RIR values. We add the proposed reverberation module to the phase spectrum predictor (PSP) of a HiNet vocoder and jointly train the model. Experimental results demonstrate that the proposed module was helpful for modeling the reverberation effect and improving the perceived quality of generated reverberant speech. The UTV-RIR was shown to be more robust than the GTI-RIR to unknown reverberation conditions and achieved a perceptually better reverberation effect.

Index Terms: reverberation, room impulse response, source-filter-based model, neural vocoder

1. Introduction

Recently several neural autoregressive models such as WaveNet [1], SampleRNN [2], and WaveRNN [3], have been proposed for raw audio generation. Their variants, such as knowledge-distilling-based models (e.g., parallel WaveNet [4] and ClariNet [5]) and flow-based models (e.g., WaveGlow [6]), were then proposed to further improve the performance and efficiency. These models can be used as *neural vocoders* [7, 8, 9, 10, 11, 12] wherein speech waveforms can be reconstructed from acoustic features for various tasks [13, 14, 15]. It was confirmed that these neural vocoders outperform vocoders using classical signal processing techniques. However, some limitations still exist — either a low generation speed, tricky training process, or complicated model structure.

Motivated by the limitations, new types of neural vocoders, such as the glottal neural vocoder [16, 17] and LPCNet [18], have been further proposed by combining speech production mechanisms with neural networks, and their performance is impressive. However, all of the above models operate under the autoregressive assumption and are slow in either waveform generation or training. Previously, we proposed non-autoregressive neural source-filter (NSF) [19] and HiNet vocoders [20, 21]. The NSF vocoder uses dilated convolutions to transform a sine-based source signal into an output waveform, following the idea of the source-filter speech production model [22].The HiNet vocoder is composed of an amplitude spectrum predictor (ASP) and a phase spectrum predictor (PSP), where the PSP is built by using the NSF vocoder for better phase recovery. The outputs of the ASP and PSP are combined to recover speech waveforms via short-time Fourier synthesis (STFS). Experimental results show that the NSF and HiNet vocoders can generate waveforms with high quality and high efficiency for speech [20, 21] and musical instrument sounds [23] recorded in acoustically isolated studios.

However, unlike the ideal data for speech or music synthesis, audio signals captured for real-life applications typically contain room reverberation. The reverberation poses a challenge to non-autoregressive neural vocoders, and the quality of synthesized speech usually degrades. Recently, Engel *et al.* tried to introduce a reverberation module with a trainable room impulse response (RIR) into a sinusoidal vocoder [24]. Their model successfully learned room reverberation effects on a solo violin dataset under a signal reverberation condition. However, learning the reverberation effects in multiple acoustic environments and applying the model for unseen acoustic environments have not yet been investigated, and neither has the model been evaluated on a reverberant speech dataset.

As an initial step towards robust reverberation modeling for speech data, this paper proposes a trainable reverberation module for neural vocoders. This module uses the output waveform of neural vocoders as an input and outputs a reverberant waveform by convolving the input with a RIR. We design two types of neural RIR estimators. One estimates the global time-invariant (GTI) RIR, which is invariant among a whole dataset and is regarded as a trainable variable of a model. This is similar to [24]. The other infers an utterance-level time-variant (UTV) RIR, which is invariant inside one utterance but varies among different utterances. The UTV-RIRs are predicted by an additional trainable neural network that uses the same conditional features as neural vocoders. We add the proposed reverberation module to the PSP of the HiNet vocoder, and experiments are conducted on a multi-speaker reverberant speech database with various types of reverberation conditions, including unseen ones. Furthermore, a multi-task training strategy that uses both reverberant and corresponding dry waveforms is also investigated.

This paper is organized as follows: In Section 2, we briefly review the NSF and HiNet vocoders. In Section 3, we give details on our proposed reverberation module and neural RIR estimators. Section 4 reports our experimental results. Conclusions are given in Section 5.

2. Brief view of NSF and HiNet vocoders

The NSF models [19] generate speech waveforms from input acoustic features through time-domain non-linear transformations. They include three modules: a conditional module that upsamples input acoustic features such as F0 and mel-spectrogram, a source module that outputs a sine-based source signal given the F0, and a dilated-convolution-based filter module that transforms the source signal into an output waveform.

The NSF models are suitable for applications where the users want to precisely control the F0 of the output waveform.

HiNet [20] is a neural vocoder that produces speech waveforms from input acoustic features by predicting amplitude and phase spectra hierarchically. The HiNet vocoder consists of two predictors, an ASP and a PSP. The ASP uses acoustic features as input and predicts frame-level log amplitude spectra (LAS). Then, the F0 and LAS predicted by the ASP are sent into the PSP for phase spectra prediction. Finally, the predicted amplitude and phase spectra are combined to reconstruct speech waveforms by STFS.

In our latest work [21], the ASP consisted of multiple convolutional layers for converting acoustic features into the LAS. Generative adversarial networks (GANs) were also newly introduced into the ASP; the ASP was used as a generator, and two discriminators were adopted. Both discriminators consisted of multiple convolutional layers, which operate convolution along with either the frequency or time axis of the input LAS, respectively. The training of the ASP is based on a Wasserstein GAN [25] loss together with the mean square error (MSE) between the predicted LAS and natural ones.

The PSP conducts two steps: neural waveform generation and phase spectrum extraction. The neural waveform generator was based on a customized NSF vocoder [19] with three modifications for better phase recovery: 1) the use of LAS as input, 2) pre-calculation of the initial phase of the sine-based excitation signal for each voiced segment at the training stage, and 3) the use of a combined loss function including MSE on amplitude spectra, waveform loss, and correlation loss.

3. Proposed methods

When a speech waveform signal $\boldsymbol{d} = [d_1, \ldots, d_T]^\top$ of length T is produced in a closed room, it propagates to an observation point through a direct path, reflects off walls and surrounding objects and becomes a reverberant signal. By assuming that the RIR of a room can be approximated by the finite impulse response sequence $\boldsymbol{h} = [h_1, \ldots, h_L]^\top$ [26, 27], where $h_1 = 1$ denotes the direct path, the received reverberant signal $\boldsymbol{r} = [r_1, \ldots, r_T]^\top$ can be written as

$$\boldsymbol{r} = \boldsymbol{d} * \boldsymbol{h}. \tag{1}$$

On the basis of this principle, we propose a reverberation module for the HiNet vocoder. This module accepts the output waveform of the PSP in the HiNet vocoder as input, which is illustrated in Fig. 1.[1] Although we can directly compute Eq. (1) through convolution in the time domain, in order to reduce the computational cost, we implement the convolution in the frequency domain as

$$\boldsymbol{r} = \mathcal{F}^{-1}[\mathcal{F}(\boldsymbol{d}) \odot \mathcal{F}(\boldsymbol{h})], \tag{2}$$

where $\mathcal{F}$, $\mathcal{F}^{-1}$, and $\odot$ represent the FFT, inverse FFT, and element-wise product, respectively.

There are two ways to parameterize and estimate the value of the RIR $\boldsymbol{h}$[2]:

- **Global time-invariant (GTI) RIR**: inspired by DDSP [24], the RIR $\boldsymbol{h}$ is assumed to be time-invariant and shared for all speech data, and the values of its coefficients $\{h_1, \ldots, h_L\}$ are learned from the training data. Note that, because $h_1 = 1$, only $L - 1$ elements in $\boldsymbol{h}$ need to be learned.

[1] We also tried to add a reverberation module based on a causal convolution network for the ASP, but it was not effective.

[2] We also tried to parameterize the RIR as an exponentially decaying function with a trainable decay rate, but the learned RIR was intractable.

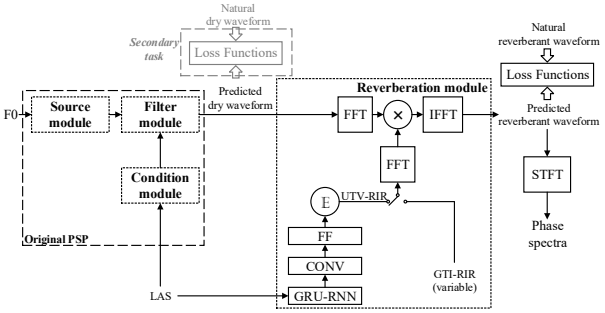


Figure 1: *Flowchart of PSP with reverberation module. Here FF, CONV, and GRU-RNN represent feed-forward, convolutional, and unidirectional GRU-based recurrent layers, respectively, and $\times$ and $\mathbb{E}$ denote element-wise product and temporal averaging operation, respectively.*

- **Utterance-level time-variant (UTV) RIR**: the RIR $\boldsymbol{h}$ is assumed to be invariant for one utterance, but different utterances acquire different $\boldsymbol{h}$. The value of $\boldsymbol{h}$ is predicted from the input LAS by a small conditional network that consists of trainable recurrent layers, convolutional layers, feed-forward layers, and a temporal average pooling layer as shown in Fig. 1. The temporal average pooling layer averages the hidden features of all of the frames and gives a single vector as the predicted $\boldsymbol{h}$.

The GTI-RIR is expected to be suitable for scenarios where we want to learn the RIR of one time-invariant acoustic environment, while the UTV-RIR is suitable for more general cases where the speech data is recorded in several different acoustic environments. During training, the reverberation module and the PSP are jointly optimized by a loss function consisting of multi-resolution spectral distortions [19] between the output of the reverberation module and the natural reverberant waveform.

For cases where the dry waveforms of the reverberant data are also available (e.g., when reverberation data are generated from clean data through simulation or replaying), we further investigate a multi-task training strategy that uses not only reverberant data but also dry waveforms. As the gray region in Fig. 1 shows, the loss function of the secondary task is a combination of MSE on LAS, waveform loss, and correlation loss [20] between a generated dry waveform and the natural dry waveform. The whole loss function is the sum of the loss functions of the main and secondary tasks.

4. Experiments

4.1. Data and feature configuration

A multi-speaker reverberant speech database[3] [28] was used in our experiments. From the database, we used a reverberant subset of 28 speakers that contained 11,572 utterances and 18 reverberation types (9 rooms $\times$ 2 microphones positions). We randomly divided this subset into a training set (11,012 utterances) and validation set (560 utterances). Regarding the test set, there were three scenarios below in our experiments:

T1 Two unseen speakers' reverberant data with 6 unseen reverberation types (3 rooms $\times$ 2 microphone positions), 824 utterances in total;

T2 Two unseen speakers' reverberant data with the same 18 reverberation types as in the training set, 832 utterances in total;

T3 Dry speech version of **T**1.

[3] https://doi.org/10.7488/ds/1425

The original 48-kHz waveforms were down sampled to 24-kHz for the experiments. The acoustic features included the 80-dimensional mel-spectrogram, F0 extracted using YAPPT [29], and a voiced/unvoiced flag. The LAS used by HiNet was computed using 2048 FFT points. All features were extracted with a frame shift and length of 12 and 50 ms, respectively.

4.2. Experimental models

We compared the following variants in the experiments[4]:

N-BL: The harmonic-plus-noise NSF model [30] was included as a reference model without the reverberation module. The number of model parameters was around 1.2×10^6.

H-BL: Baseline HiNet vocoder without the proposed reverberation module. We used the baseline ASP configuration in our previous work [21] and added two convolution layers having 512 and 1024 channels, respectively, to the discriminator that operated along with the frequency axis. This was motivated by the increased number of FFT points of the LAS. We adopted the same PSP used as the baseline in our previous work [21], but GANs were not used here. The loss function was a combination of MSE on LAS, waveform loss, and correlation loss. Note that the NSF module in the PSP is slightly different from **N-BL** (see details in [20]). The number of model parameters was around 6.2×10^7 for the ASP and 7.2×10^6 for the PSP.

H-GTI: HiNet with the GTI-RIR-based reverberation module integrated into PSP. The RIR length was 6,000. The model configuration and size were the same as **H-BL** except for the increased 5,999 trainable parameters for GIT-RIR.

H-UTV: HiNet with the UTV-RIR-based reverberation module integrated into the PSP. The RIR length was 6,000. The trainable neural network that converts LAS to RIR consisted of a unidirectional GRU layer with 1,024 nodes, a convolution layer with 1,024 nodes and a kernel-size of 11, and a feed-forward linear layer with 5,999 output nodes. Other settings were the same as those of **H-BL**. The number of model parameters for the PSP was increased by 2.4×10^7 compared with **H-BL**. Since the UTV-RIR is non-autoregressive, the increased model size did not cause an obvious degradation of generation efficiency.

H-UTV-MT: same as **H-UTV** but with the secondary task using dry waveforms during training.

4.3. Main experiments

Our main experiments focused on the reverberation effect and speech quality. We compared **N-BL**, **H-BL**, **H-GTI**, and **H-UTV** under testing scenarios **T**1 and **T**2 using both objective and subjective evaluations.

4.3.1. Objective evaluation – T60 comparisons –

T60 estimation errors [31] were used as the objective metric to evaluate the reverberation effects. T60 is also called the reverberation time, and it is defined as the time it takes for sound to decay by 60 dB after the source has been switched off. We used an open source toolkit [32] to blindly estimate T60 from the reverberant speech. The T60 estimation errors were calculated as the difference between the estimated T60 and the ground-truth T60 (T60n) reported in the database paper [28].

We calculated the T60 estimation errors for the output waveforms from all of the experimental models. For reference, we also calculated the errors for the natural reverberant waveform and the output waveform from the PSP in the HiNet models (denoted by **P-***). Figure 2 shows box plots of T60 estimation errors for utterances with T60n = 0.362s under test scenario **T**1. Note that the T60 estimated errors for natural reverberant speech were non-zero because blind estimation of T60 is not error-free.

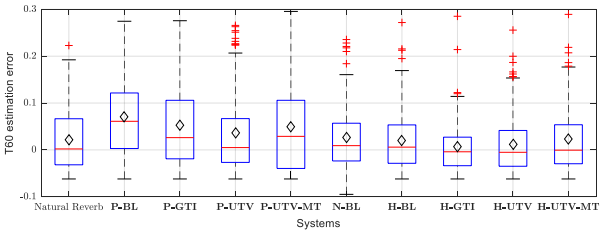


Figure 2: *Box plots of T60 estimation errors for utterances with T60n = 0.362s under test scenario **T**1.*

Figure 2 demonstrates that both **P-GTI** and **P-UTV** had smaller errors than **P-BL**, which indicates the usefulness of the proposed reverberation module in the PSP component of HiNet. Furthermore, **P-UTV** had a smaller error than **P-GTI**, suggesting that UTV-RIR is more effective than GTI-RIR in modeling unseen reverberation types. By comparing **P-*** with **H-***, we see that **H-*** had smaller errors than **P-***. This suggests that the ASP is able to produce the reverberation effect by a moderate amount even though the ASP has no explicit reverberation module. The performance differences among **H-*** vocoders are small. Additionally we can observe that **N-BL** had smaller errors than **P-BL** while **H-BL** had marginally smaller errors than **N-BL**.

4.3.2. Subjective evaluation

We conducted two types of listening tests on the crowdsourcing platform Amazon Mechanical Turk[5] with anti-cheating considerations [33] to evaluate the reverberation effect and speech quality, respectively. In each test, 20 test-set utterances were generated for each test scenario by each experimental model, and these utterances were evaluated by about 40 English native listeners.

Reverberation effect: The first test was a similarity test on the reverberation effect. Listeners were asked to first listen to the natural dry and reverberant audio tracks. They were then asked to listen to a few test audio tracks and assign a score from 1 to 9 to each, where a higher score denoted a reverberation effect more similar to that in the natural reverberant audio tracks. The audio tracks generated from the PSP in the HiNet models were directly used for the listening test, and they are denoted as **P-***. Furthermore, to investigate the impact of the proposed reverberation module, the listening test used the audio tracks generated from the PSP after the trained reverberation module was removed, and they are denoted as **P-*(dry)**.

The results for test scenarios **T**1 (unseen reverberant type) and **T**2 (seen reverberant type) are plotted in Figure 3. As expected, the similarity scores of **P-GTI** and **P-UTV** had higher means and medians than those of **P-BL** in both **T**1 and **T**2. This means that the proposed module generated reverberation that was perceptually more similar to the natural reverberant speech. Furthermore, **P-UTV** outperformed **P-GTI** in **T**1. These results were consistent with the results for the T60 estimation errors in Section 4.3.1. For **T**2, however, **P-UTV** did not

[4]Examples of generated speech can be found at `http://home.ustc.edu.cn/~ay8067/reverb/demo.html`. Scripts and toolkits for the NSF model can be found at `https://github.com/nii-yamagishilab/project-CURRENNT-scripts`

[5]`https://www.mturk.com`.

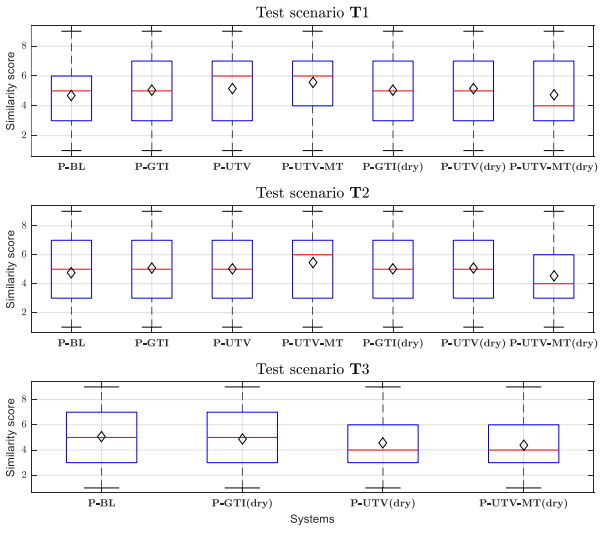


Figure 3: *Box plot of reverberant effect similarity scores for all test scenarios. Here, black diamonds and red lines represent mean and median.*

outperform **P-GTI**, which indicates that **P-UTV** may be more suitable for unknown reverberation conditions. Unfortunately, the similarity scores of **P-*(dry)** remained similar to **P-BL**. It seems that the evaluated models did not have de-reverberation ability, i.e., they could not generate perfect dry waveforms giving reverberant acoustic features. One possible reason may be that the reverberation module was jointly trained with the rest of the network. This point is further investigated together with multi-task learning in the next section.

Quality: The second test was a MUSHRA (MUltiple Stimuli with Hidden Reference and Anchor) test [34] done to compare the quality of the generated waveforms. The average MUSHRA scores and their 95% confidence intervals are shown in Figure 4. The reference audio tracks in the MUSHRA test for **T**1 and **T**2 were the natural reverberant waveforms.

As Figure 4 shows, **H-GTI** and **H-UTV** had higher MUSHRA scores than **H-BL** for both **T**1 and **T**2, suggesting that the reverberation module in the PSP was helpful for improving the quality of synthetic speech for HiNet. The MUSHRA scores for **T**2 were higher than those for **T**1, which is reasonable since unseen reverberation conditions are more challenging to model. We can also see that the difference between **H-GTI** and **H-UTV** was not significant. Utterance-dependent RIR estimation seems to be important for modeling multiple reverberation types as the T60 comparison and the similarity test suggest, but it does not improve the perceived quality of generated waveforms. Finally, **H-BL** outperformed **N-BL**, and this indicates that the reverberant speech generated from the HiNet vocoder sounded better than that from the NSF vocoder with the current configurations.

4.4. Additional analysis

Finally, we analyzed two additional configurations.

Multi-task training using dry waveforms: Models using the multi-task training are denoted as ***-UTV-MT**. By comparing **P-UTV-MT** with **P-UTV** in Figure 2, we see that using the multi-task training did not reduce the T60 estimation error. However, as the similarity test results in Figure 3 show, **P-UTV-MT** had a higher mean and median than **P-UTV**. Furthermore, the median of **P-UTV-MT(dry)** was 4.0, while that of **P-UTV-MT** was 6.0 for **T**1 and **T**2, and the differences were larger than those between **P-UTV** and **P-UTV(dry)**. These results suggest that multi-task training using dry waveforms as additional supervision makes the functional role of the proposed reverberation module more explicit. Regarding the quality, there was no obvious difference between between **H-UTV-MT** and **H-UTV** as shown in Figure 4.

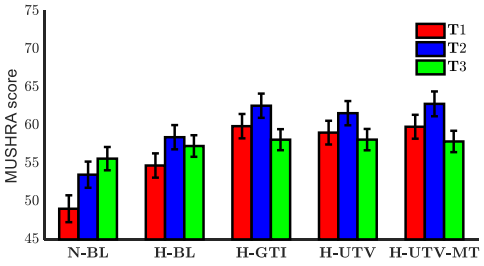


Figure 4: *Average MUSHRA scores with 95% confidence interval for all test scenarios.*

Use of dry acoustic features (T3): If the proposed framework is well generalized, it should be able to generate dry speech with high quality when we input dry acoustic features. This was the purpose of **T**3. The results in Figure 3 show that the medians or the mean similarity scores of **P-GTI(dry)**, **P-UTV(dry)**, and **P-UTV-MT(dry)** were lower than those of the corresponding models in **T**1 and **T**2. In other words, these models generated waveforms that were perceptually closer to the natural dry waveforms when using dry input acoustic features. These results are encouraging. However, the generated waveforms were not sufficiently close to the natural dry waveforms, so there is still room for improvement. The results of the quality comparisons are shown in Figure 4. The reference tracks used for the MUSHRA test were natural waveforms without reverberation. From the MUSHRA listening test, we can see that the quality scores for **T**3 were similar to those of **T**1 for unseen conditions.

5. Conclusion

In this paper, we proposed a neural reverberation module and integrated it into non-autoregressive source-filter-based neural vocoders. The reverberation module uses RIRs for convolving waveforms generated by the vocoders as the standard signal processing method does, but the RIRs are estimated jointly with other parameters of the neural vocoder or predicted by another trainable network. The former approach, called GIT-RIR, uses a globally invariant vector and is directly trained on a reverberant dataset. The latter approach, called UTV-RIR, uses another network to estimate a different RIR for each utterance. We conducted experiments by adding the proposed reverberation module to the PSP of the HiNet vocoder. Objective and subjective evaluation results indicated that the proposed reverberation module is helpful for modeling the reverberation effect and improving the quality of reverberant speech generated by the HiNet vocoder. We also confirmed that the UTV-RIR was better than the GTI-RIR when modeling multiple unseen reverberation types. For future work, we plan to apply the reverberation module to other neural vocoders.

Acknowledgments: This work was partially supported by a JST CREST Grant (JPMJCR18A6, VoicePersonae project), Japan, MEXT KAKENHI Grants (19K24371, 16H06302, 17H04687, 18H04120, 18H04112, 18KT0051), Japan, and the National Natural Science Foundation of China under Grant 61871358 and the China Scholarship Council (CSC). The experiments were partially conducted on TSUBAME 3.0 of Tokyo Institute of Technology.

6. References

[1] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[2] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *arXiv preprint arXiv:1612.07837*, 2017.

[3] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *Proc. ICML*, 2018, pp. 2410–2419.

[4] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg *et al.*, "Parallel WaveNet: Fast high-fidelity speech synthesis," in *Proc. ICML*, 2018, pp. 3918–3926.

[5] W. Ping, K. Peng, and J. Chen, "ClariNet: Parallel wave generation in end-to-end text-to-speech," in *arXiv preprint arXiv:1807.07281*, 2019.

[6] R. Prenger, R. Valle, and B. Catanzaro, "WaveGlow: A Flow-based Generative Network for Speech Synthesis," in *Proc. ICASSP*, 2019, pp. 3617–3621.

[7] A. Tamamori, T. Hayashi, K. Kobayashi, K. Takeda, and T. Toda, "Speaker-dependent WaveNet vocoder," in *Proc. Interspeech*, 2017, pp. 1118–1122.

[8] T. Hayashi, A. Tamamori, K. Kobayashi, K. Takeda, and T. Toda, "An investigation of multi-speaker training for WaveNet vocoder," in *Proc. ASRU*, 2017, pp. 712–718.

[9] N. Adiga, V. Tsiaras, and Y. Stylianou, "On the use of WaveNet as a statistical vocoder," in *Proc. ICASSP*, 2018, pp. 5674–5678.

[10] Y. Ai, H.-C. Wu, and Z.-H. Ling, "SampleRNN-based neural vocoder for statistical parametric speech synthesis," in *Proc. ICASSP*. IEEE, 2018, pp. 5659–5663.

[11] Y. Ai, J.-X. Zhang, L. Chen, and Z.-H. Ling, "DNN-based spectral enhancement for neural waveform generators with low-bit quantization," in *Proc. ICASSP*, 2019, pp. 7025–7029.

[12] J. Lorenzo-Trueba, T. Drugman, J. Latorre, T. Merritt, B. Putrycz, and R. Barra-Chicote, "Robust universal neural vocoding," *arXiv preprint arXiv:1811.06292*, 2018.

[13] L.-J. Liu, Z.-H. Ling, Y. Jiang, M. Zhou, and L.-R. Dai, "WaveNet vocoder with limited training data for voice conversion," in *Proc. Interspeech*, 2018, pp. 1983–1987.

[14] K. Kobayashi, T. Hayashi, A. Tamamori, and T. Toda, "Statistical voice conversion with WaveNet-based waveform generation," in *Proc. Interspeech*, 2017, pp. 1138–1142.

[15] Z.-H. Ling, Y. Ai, Y. Gu, and L.-R. Dai, "Waveform Modeling and Generation Using Hierarchical Recurrent Neural Networks for Speech Bandwidth Extension," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 5, pp. 883–894, 2018.

[16] Y. Cui, X. Wang, L. He, and F. K. Soong, "A new glottal neural vocoder for speech synthesis," in *Proc. Interspeech*, 2018, pp. 2017–2021.

[17] L. Juvela, V. Tsiaras, B. Bollepalli, M. Airaksinen, J. Yamagishi, and P. Alku, "Speaker-independent raw waveform model for glottal excitation," in *Proc. Interspeech 2018*, 2018, pp. 2012–2016.

[18] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," in *Proc. ICASSP*, 2019, pp. 5891–5895.

[19] X. Wang, S. Takaki, and J. Yamagishi, "Neural Source-Filter Waveform Models for Statistical Parametric Speech Synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 402–415, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8915761/

[20] Y. Ai and Z.-H. Ling, "A neural vocoder with hierarchical generation of amplitude and phase spectra for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 839–851, 2020.

[21] Y. Ai and Z.-H. Ling, "Knowledge-and-data-driven amplitude spectrum prediction for hierarchical neural vocoders," *arXiv preprint arXiv:2004.07832*, 2020.

[22] F. Gunnar, *The acoustic theory of speech production*. The Hague, The Netherlands: Mouton, 1960.

[23] Y. Zhao, X. Wang, L. Juvela, and J. Yamagishi, "Transferring neural speech waveform synthesizers to musical instrument sounds generation," in *Proc. ICASSP*, 2020, pp. 6269–6273.

[24] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," *Proc. ICLR*, 2020.

[25] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.

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

[27] J. Benesty, M. M. Sondhi, and Y. Huang, *Springer handbook of speech processing*. Springer, 2007.

[28] C. Valentini-Botinhao and J. Yamagishi, "Speech enhancement of noisy and reverberant speech for text-to-speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 8, pp. 1420–1433, 2018.

[29] K. Kasi and S. A. Zahorian, "Yet another algorithm for pitch tracking," in *Proc. ICASSP*, vol. 1, 2002, pp. 361–364.

[30] X. Wang and J. Yamagishi, "Using cyclic noise as the source signal for neural source-filter-based speech waveform model," *arXiv preprint arXiv:2004.02191*, 2020.

[31] N. D. Gaubitch, H. W. Loellmann, M. Jeub, T. H. Falk, P. A. Naylor, P. Vary, and M. Brookes, "Performance comparison of algorithms for blind reverberation time estimation from speech," in *Proc. IWAENC*, 2012, pp. 1–4.

[32] M. Jeub, "Blind reverberation time estimation," 2015. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/35740-blind-reverberation-time-estimation

[33] S. Buchholz and J. Latorre, "Crowdsourcing preference tests, and how to detect cheating," in *Proc. Interspeech*, 2011, pp. 3053–3056.

[34] I. Recommendation, "Method for the subjective assessment of intermediate sound quality (MUSHRA)," *ITU, BS*, pp. 1543–1, 2001.

An Effective Perturbation based Semi-Supervised Learning Method for Sound Event Detection

Xu Zheng[1], *Yan Song*[1], *Jie Yan*[1], *Li-Rong Dai*[1], *Ian McLoughlin*[1,2], *Lin Liu*[3]

[1]National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei, China.
[2]ICT Cluster, Singapore Institute of Technology, Singapore
[3]iFLYTEK Research, iFLYTEK CO., LTD, Hefei, China

{zx980216, yanjie17}@mail.ustc.edu.cn, {songy, ivm, lrdai}@ustc.edu.cn, linliu@iflytek.com

Abstract

Mean teacher based methods are increasingly achieving state-of-the-art performance for large-scale weakly labeled and unlabeled sound event detection (SED) tasks in recent DCASE challenges. By penalizing inconsistent predictions under different perturbations, mean teacher methods can exploit large-scale unlabeled data in a self-ensembling manner. In this paper, an effective perturbation based semi-supervised learning (SSL) method is proposed based on the mean teacher method. Specifically, a new independent component (IC) module is proposed to introduce perturbations for different convolutional layers, designed as a combination of batch normalization and dropblock operations. The proposed IC module can reduce correlation between neurons to improve performance. A global statistics pooling based attention module is further proposed to explicitly model inter-dependencies between the time-frequency domain and channels, using statistics information (*e.g.* mean, standard deviation, max) along different dimensions. This can provide an effective attention mechanism to adaptively re-calibrate the output feature map. Experimental results on Task 4 of the DCASE2018 challenge demonstrate the superiority of the proposed method, achieving about 39.8% F1-score, outperforming the previous winning system's 32.4% by a significant margin.
Index Terms: sound event detection, semi-supervised learning, independent component analysis, statistics pooling

1. Introduction

Sound event detection (SED) is the task of determining when and where target event categories occur in continuous audio. SED has attracted significant research attention due to its wide application in real-world systems, such as robotics [1], smart home devices [2], health care, and audio based indexing and retrieval [3, 4]. With the development of deep learning techniques, several mainstream deep neural networks (DNN), such as CNN, RNN and CRNN, have recently achieved state-of-the-art SED performance [4, 5, 6].

However, real-life SED is challenging, in part due to the lack of large-scale well annotated audio datasets, which are generally expensive and time-consuming to collect. Semi-supervised learning (SSL) SED methods that can exploit real data (which is either weakly labeled – without timestamp – or is unlabeled) to improve system performance, have thus drawn increasing research interest. Recent Detection and Classification of Acoustic Scenes and Events (DCASE) challenges have included a task for the evaluation of SSL based SED in domestic environments with weakly labeled audio data. There are several semi-supervised learning based methods in the literature,

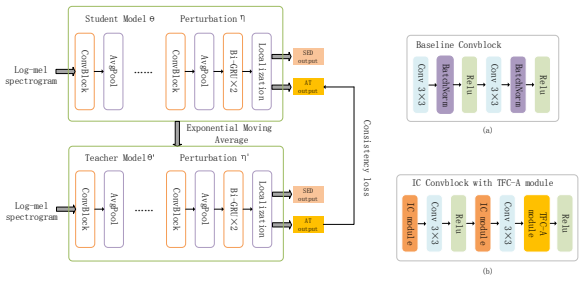


Figure 1: *(Left) framework of mean teacher based semi-supervised SED learning for large-scale weakly labeled data, and Fig. The common Conv-Bn-Relu baseline convblock. (b) Our proposed IC convblock with TFC-A module.*

including self-training [7], temporal ensembling (TE) [8], virtual adversary training (VAT) [9] and mean teacher (MT) [10]. In [7], the self-training based method was proposed to exploit the unlabeled data by generating pseudo-labels using models trained with small-size labeled data. In [8, 9, 10], perturbation based methods were proposed under the *smoothness* assumption, which indicates that two data points close to each other in feature space are likely to have the same label [11]. Among these methods, MT has shown promising SED performance in DCASE challenges, where the teacher acts as an ensemble of the students to generate the targets for SSL, and the consistency cost is employed as a regularization term. The key to effective MT is choosing suitable data and/or model perturbation to form a better teacher model from the student model and thus improve target quality. However, simply applying randomized data augmentation or dropout may not be optimal for introducing effective perturbation. In [12], a spec-augment technique was applied to improve data augmentation while in [13], different teacher and student models were exploited to perform SED and AT respectively.

In this paper, we propose an effective perturbation based semi-supervised learning (SSL) method based on mean teacher, as shown in Fig. 1. This includes a new independent component (IC) module, designed as a combination of batch normalization(BN) [14] and dropblock [15] operations. Its goal is two-fold: (1) Apply perturbations to the input of the internal convolutional layer to learn a better teacher and (2) Construct whitened input for convolutional filters in each intermediate convolutional layer. Compared to dropout, the dropblock drops units in a contiguous region of a feature map,

which can reduce the spatial correlation of the input. The IC module (BN+Dropblock) can approximate independent component analysis (ICA), which is traditionally implemented by two steps: the zero-phase component analysis (ZCA) to whiten the network activations, and rotation operations to get the final independent components [16]. That is, BN replaces the ZCA, while the dropblock reduces the dependencies within activations.

Furthermore, motivated by "squeeze-and-excitation" [17], which models inter-dependencies between the channels, a global statistics pooling based attention module is further proposed to explicitly model inter-dependencies between the time-frequency domain and channels using statistics information (*e.g.* mean, standard deviation, max) computed along different dimensions. This provides an effective attention mechanism which can adaptively re-calibrate the output feature map.

To evaluate the effectiveness of the proposed methods, extensive experiments have been conducted on DCASE2018 challenge task4 benchmarks. The experimental results show its superiority with 39.79% F1-score compared to 32.4% in the winning system.

2. Baseline method

In this section, we will firstly introduce the mean teacher based framework for SED [10], as shown in left of Fig. 1. We adopt CRNN as the backbone architecture [18]. The CNN part is composed of 5 convolutional blocks, followed by two Bi-GRU layers to model long-term relationships and a localization module.

Since the task 4 of DCASE challenges focuses on weakly labeled data, which consists of audio tagging(AT) and SED tasks. In mean teacher, two CRNNs (namely, teacher and student) with the same architecture are used. And the teacher model is updated by exponential moving average of the student model parameters. The consistency loss $L_{consist}$ is defined as the expected distance between the prediction of teacher (with weights θ' and perturbation $\eta\prime$) and student model (with weights θ and perturbation η), which is

$$L_{consist} = MSE(\mathbf{S}_{\theta_{AT}}(\mathbf{x};\eta), \mathbf{T}_{\theta'_{AT}}(\mathbf{x};\eta\prime)) \qquad (1)$$

In the baseline SED system, spec-augment [19] is applied to the input of CRNNs to perform input perturbation. This may be further improved by adding perturbation to the intermediate convolutional layer, such as dropout [20]. We will introduce our proposed method, in which the IC convolutional block(convblock) is used, instead of baseline convblock, to generate perturbation to intermediate convolutional layer. Furthermore, an attention mechanism based on global statistics information is proposed to improve the effectiveness of convblock output.

3. The perturbation based semi-supervised learning

As aforementioned, the proposed perturbation based SSL framework is obtained by replacing the traditional convblock (in Fig. 1(a)) with IC convblock with TFC-A module (in Fig. 1(b)) The IC module is placed before the convolutional layer. It is worth noting that this is different from the operations in common practice, the BN layer is placed after the convolutional layer, followed by a non-linear activation. The TFC-A module, meanwhile, is proposed to explicitly model inter-dependencies. between the time-frequency domain and channels. Details of the IC and attention modules will be described in the following subsections.

3.1. Independent Component(IC) module

Data or model perturbation plays an important role in the mean teacher SSL method. Generally, the perturbation is applied to the input spectrograms, via data augmentation techniques including Gaussian noise and spec-augment [19] . From this perspective, dropout [20] can be considered as a type perturbation applied to the input of intermediate layer. However, it is shown that the dropout is less effective for convolutional layer than fully connected layer [15]. This may perhaps be caused by the fact that activation units in convolutional layer are spatially correlated, and the information of the dropped units can be partially recovered by the surrounding units. Furthermore, proposed by Li et al. [21], if dropout is placed before BN, it may lead to biased estimation of the mean and standard variation hyper-parameters in BN.

In this paper, an effective IC module, which consists of the BN and dropblock operator, is proposed. The BN is applied to normalize the input to distribution with zero mean and unit variance. And then in dropblock, which is a structured form of dropout, contiguous regions of a neurons are set to zero. By randomly dropping neurons in this way, the IC module can effectively introduce the perturbation to the input of convolutional layer. IC module in fact produces various student models, leading to a better teacher obtained by ensembling student models.

Besides, the IC module can also approximate the feature whitening operation, such as ZCA and ICA [16], which may facilitate the training procedure. Specially, ICA composes of two steps: 1) ZCA to de-correlate the input features, 2) Rotation to reduce the dependence. However, ICA is generally computational complex, especially for whitening the activations of a wide neural network.

For dropblock, there are two main hyper-parameters, $drop\_prob$ and $block\_size$. The $drop\_prob$ is defined same as dropout. The $block\_size$ defines the dropping area in activation map, and when $block\_size = 1$, dropblock resembles standard dropout.

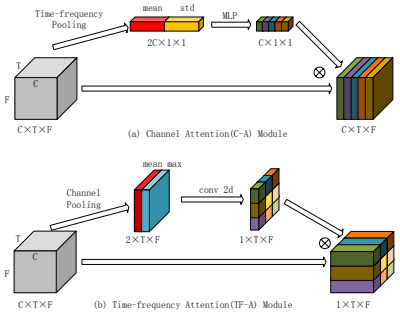


Figure 2: *Illustration of our proposed (a) channel attention (C-A) module, and (b) time-frequency attention (TF-A) module.*

3.2. Statistics pooling based attention modules

Given an intermediate feature map $\mathbf{U} \in \mathbb{R}^{C\times T\times F}$, the global statistics pooling based attention module consists of two parts: 1) Channel-attention module(C-A) with an 1D channel attention map $\mathbf{M}_C \in \mathbb{R}^C$, and 2) Time-frequency module(TF-A) with a

2D time-frequency attention map $\mathbf{M}_{TF} \in \mathbb{R}^{T\times F}$ where T, F and C are time, frequency and channel dimension respectively.

Different statistics information from $\mathbf{U}$ (*e.g.* mean, standard deviation, max) is exploited for C-A and TF-A.

3.2.1. Channel Attention(C-A)

As shown in Fig. 2(a), the statistics pooling is used to calculate the mean $\boldsymbol{\mu} \in \mathbb{R}^C$ and standard deviation $\boldsymbol{\sigma} \in \mathbb{R}^C$ over the time-frequency domain

$$\boldsymbol{\mu} = \frac{1}{T\times F}\sum_{i=1}^{T}\sum_{j=1}^{F}\mathbf{u}(i,j) \tag{2}$$

$$\boldsymbol{\sigma} = \sqrt{\frac{1}{T\times F}\sum_{i=1}^{T}\sum_{j=1}^{F}\mathbf{u}^2(i,j) - \boldsymbol{\mu}^2} \tag{3}$$

The output of statistics pooling $\mathbf{z}$ is obtained by concatenating the mean and standard $\mathbf{z} = [\boldsymbol{\mu}; \boldsymbol{\sigma}]$. The channel attention map $\mathbf{M}_C$ is calculated via series of non-linear operations,

$$\mathbf{M}_C = \sigma(\mathbf{W}_2(\delta(\mathbf{W}_1\mathbf{z}))) \tag{4}$$

where $\mathbf{W}_1 \in \mathbb{R}^{2C\times\frac{C}{r}}$, $\mathbf{W}_2 \in \mathbb{R}^{\frac{C}{r}\times C}$ denote FC layers, $\sigma(\cdot)$ and $\delta(\cdot)$ denote sigmoid and ReLU respectively, r is the reduction rate.

3.2.2. Time-frequency Attention(TF-A)

As shown in Fig. 2(b), in TF-A module, a 2D feature map $\mathbf{M}_{TF} \in \mathbb{R}^{T\times F}$ is used to provide the attention on time-frequency domain. It is obtained by first calculating the mean and max information over channels, followed by a convolutional layer and a sigmoid activation

$$\mathbf{M}_{TF} = \sigma(\mathbf{f}^{k\times k}[AvgPool(\mathbf{U}); MaxPool(\mathbf{U})]) \tag{5}$$

where $\mathbf{f}$ denotes a convolutional layer with kernel size $k \times k$.

3.2.3. Arrangement of Attention Modules

As aforementioned, the C-A and TF-A may provide complementary attentive information from their feature maps. Given C-A and TF-A modules, there are different arrangements of them, namely a sequential or parallel arrangement. For sequential arrangement of C-A and TF-A modules, we can have "C-A + TF-A" arrangement

$$\begin{aligned}\mathbf{U}' &= \mathbf{M}_C(\mathbf{U}) \otimes \mathbf{U} \\ \mathbf{U}'' &= \mathbf{M}_{TF}(\mathbf{U}') \otimes \mathbf{U}'\end{aligned} \tag{6}$$

and reverse ordered sequential arrangement "TF-A + C-A"

$$\begin{aligned}\mathbf{U}' &= \mathbf{M}_{TF}(\mathbf{U}) \otimes \mathbf{U} \\ \mathbf{U}'' &= \mathbf{M}_C(\mathbf{U}') \otimes \mathbf{U}'\end{aligned} \tag{7}$$

For parallel module is implemented as,

$$\mathbf{U}' = \mathbf{M}_{TF}(\mathbf{U}) \otimes \mathbf{M}_C(\mathbf{U}) \otimes \mathbf{U} \tag{8}$$

In experiments, we will evaluate different arrangement of C-A and TF-A modules.

4. Experiments Setup

4.1. Dataset

The experiments are conducted on the benchmark dataset from Task4 of the DCASE 2018 Challenge [22]. The dataset contains 1578 weakly-labeled training clips, 14412 unlabeled in-domain training clips, 39999 unlabeled out-of-domain training clips, 288 development clips and 880 evaluation clips. The average length of occurrence of each event class is presented in Table 1, indicating the very significant variance in duration between events. In our experiments, we utilize weakly-labeled clips with unlabeled in-domain clips as training set, and evaluate the performance on publicly available evaluation set.

Table 1: *Average length and median filter size of each class in the development dataset.*

Sound event label	**Average length(s)**	**Median filter size(s)**
Alarm_bell_ringing	1. 53	0.50
Blender	5. 35	1.75
Cat	0. 81	0.26
Dishes	0. 56	0.16
Dog	1. 03	0.34
Electric_shaver_toothbrush	7. 42	2.46
Frying	9. 34	3.10
Running_water	5. 61	1.85
Speech	1. 51	0.50
Vacuum_Cleaner	8. 66	2.86

4.2. Feature Extraction

The input features used in the proposed system are log-mel spectrograms, which are extracted from the audio signal resampled at 32 kHz. The spectrogram uses 64 Mel-scale filters and a window size of 32ms with 50% overlap between windows. As a result, each 10-second sound clip is transformed into a 2D time-frequency representation with a size of (640 × 64).

4.3. Experimental Settings

The neural networks are trained using the Adam optimizer [23], where the maximum learning rate is set to 0.001, and the total training epochs are set to 100. Specifically, there is a rampup for the learning rate over the first 20 epochs, and an adaptive median filter is used for backend processing. The filter size for each event category is selected according to Table 1.

For IC modules, we performed a set of experiments to determine that a sensible $block\_size$ is 5 for this configuration. In the TFC-A modules, similar empirical testing found a good reduction rate r is 8, and a reasonable kernel size for the convolutional layer in the TF-A module is 5×5.

Event based macro-F1 is used as the main metric For SED tasks. The experiment results are all evaluated by the sed_eval toolbox [24]. Onsets are evaluated with a collar tolerance of 200ms. Tolerance for offsets is computed per event as the maximum of 200ms or 20% of event length.

5. Results and Discussion

In experiments, we evaluate the performance of with perturbation based SED systems including: 1) IC(dropblock): with IC

Table 2: *Results of our proposed methods.*

System	Macro F1, %
Winner's system [25]	32.4
IC(dropblock)	39.30
TF-A + C-A	39.50
IC(dropblock)+TF-A + C-A	**39.79**

module only to introduce perturbation on input of intermediate convolutional layer, 2) TF-A + C-A: with the sequential arrangement of TF-A and C-A to introduce the attention mechanism for output, and 3) IC(dropblock + TF-A + C-A): the combination of both 1) and 2). As shown in Table 2, the proposed IC module, as well as TFC-A module can achieve F1-score over 39.00%. In addition, by combining the IC(dropblock) and "TF-A + C-A" modules, the F1-score can achieve 39.79%, which significantly outperforms the previously winning score of 32.4% [25]. The experiments results demonstrate the effectiveness of our proposed methods. A detailed ablation analysis will be conducted in the next subsections.

5.1. Evaluations of dropblock or dropout in IC module

We further evaluate the performance of dropout and dropblock in IC module with different $drop\_prob$. As shown in Table 3, the CRNN baseline in Fig. 1(a) achieves an F1-score of 36.17%, with 3.77% gain over winner's system in DCASE2018 challenges. This may come from the superiority of the CRNN system as well as the adaptive median filter (in Table 1) for backend processing. The performance of system using IC module is further improved compared with the baseline system.

Furthermore, we can see that with the same $drop\_prob$, the performance of IC(dropblock) is generally better than IC(dropout), indicating that IC(dropblock) provides more effective perturbations for convolutional layers. Specifically, among different $drop\_prob$, IC(dropblock) with $drop\_prob$=0.05 provides best F1-score of 39.30%, relative 3.13% improvement of our baseline system. To further analyze the effectiveness of TFC-A modules, we conduct several ablation experiments.

Table 3: *SED results from evaluating the IC modules.*

ConvBlock	drop_prob	Macro F1, %
Baseline convblock	-	36.17
IC(no perturbation)	0	37.74
IC(dropblock)	0.05	**39.30**
IC(dropblock)	0.10	38.10
IC(dropblock)	0.20	36.49
IC(dropout)	0.05	38.34
IC(dropout)	0.10	37.28
IC(dropout)	0.20	35.86

5.2. Evaluations of different TFC-A

In the proposed attention method, different types of statistics information (*e.g.* mean, standard deviation, max) are used for TF-A and C-A. Results in Table 4, reveal quite wide differences in performance for the different types of statistics in the C-A module. Specially C-A(mean), same as Squeeze-and-Excitation [17], can improve the performance from 37.74% (baseline without C-A) to 38.68%. On the contrary, C-A(max) only achieves the performance of 25.25%, indicating that C-A(max) may loss the information of the overlapped events. In addition, C-A(mean std) incorporates the standard deviation as a second-order statistic information, and achieves F1-score of 39.00%. Finally, TF-A(mean-max) provides a slight improvement over the baseline system. This indicates that the class-wise information tends to be relatively strongly encoded in the channel dimension.

Table 4: *SED performance of different TFC-A modules.*

TFC-A module	Macro F, %
-	37.74
C-A (mean)	38.68
C-A (max)	25.25
C-A (mean std)	**39.00**
TF-A (mean max)	38.04
Parallel (TFC-A)	37.96
C-A + TF-A	37.54
TF-A + C-A	**39.50**

We also evaluate the performance of combining TF-A or C-A modules in parallel or sequential fashion, as shown in Table 4. We can see that sequential arrangement (TF-A + C-A) can achieve the F1-score of 39.50%, performing best with the same settings. Interestingly, the reverse ordered sequential arrangement (C-A + TF-A), performs worst. From the previous experiments we already know the importance of the C-A module. It seems that performing TF-A first strengthens the information on the time-frequency domain but it is necessary to apply C-A to emphasize the important channels. In the converse case, the C-A first sequential attention is overruled by the subsequent TF-A operation, thus reducing performance. The parallel TFC-A also provides slight improvement to the baseline system, but worse than the sequential attention module(TF-A + C-A).

6. Conclusion

In this paper, a novel perturbation based semi-supervised learning method, combining batch normalization with dropblock, is investigated. This not only provides perturbation to the convolutional layer but also whitens their inputs, to improve semi-supervised SED performance. Experimental results reveal the proposed dropblock based IC modules outperform conventional dropout, providing more effective perturbations to the convolutional layers. Furthermore, statistical pooling based attention module is used to explicitly model inter-dependency between time-frequency and channel domains. Among mean, standard deviation and max computed on different dimensions, we sequentially apply time-frequency attention, followed by channel attention, performing best. By combining the IC module and TFC-A module, the final F1-score of 39.8%, significantly outperforms the 32.4% achieved by the previously published winning system. In future, we hope to exploit other perturbation and attention types for semi-supervised SED.

7. Acknowledgement

This work was supported by National Natural Science Foundation of China (Grant No. U1613211).

8. References

[1] R. Takeda and K. Komatani, "Sound source localization based on deep neural networks with directional activate function exploiting phase information," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 405–409, 2016.

[2] A. Southern, F. Stevens, and D. Murphy, "Sounding out smart cities: Auralization and soundscape monitoring for environmental sound design," *The Journal of the Acoustical Society of America*, vol. 141, no. 5, pp. 3880–3880, 2017.

[3] S. Goetze, J. Schroder, S. Gerlach, D. Hollosi, J.-E. Appell, and F. Wallhoff, "Acoustic monitoring and localization for social care," *Journal of Computing Science and Engineering*, vol. 6, no. 1, pp. 40–50, 2012.

[4] Y. Wang, S. Rawat, and F. Metze, "Exploring audio semantic concepts for event-based video retrieval," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1360–1364.

[5] I. McLoughlin, H.-M. Zhang, Z.-P. Xie, Y. Song, and W. Xiao, "Robust sound event classification using deep neural networks," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 23, pp. 540–552, Mar. 2015.

[6] J. Yan, Y. Song, W. Guo, L.-R. Dai, I. McLoughlin, and L. Chen, "A region based attention method for weakly supervised sound event detection and classification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 755–759.

[7] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models." *WACV/MOTION*, vol. 2, 2005.

[8] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," *arXiv preprint arXiv:1610.02242*, 2016.

[9] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[10] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in neural information processing systems*, 2017, pp. 1195–1204.

[11] Y. Luo, J. Zhu, M. Li, Y. Ren, and B. Zhang, "Smooth neighbors on teacher graphs for semi-supervised learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8896–8905.

[12] J. Yan, Y. Song, L.-R. Dai, and I. McloughLin, "Task-aware mean teacher method for large scale weakly labeled semi-supervised sound event detection," in *ICASSP 2020*, 2020, pp. 1195–1204.

[13] L. Lin, X. Wang, H. Liu, and Y. Qian, "What you need is a more professional teacher," *arXiv preprint arXiv:1906.02517*, 2019.

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

[15] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Dropblock: A regularization method for convolutional networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 727–10 737.

[16] H. Oja and K. Nordhausen, "Independent component analysis," *Encyclopedia of Environmetrics*, vol. 3, 2006.

[17] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[18] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Large-scale weakly supervised audio classification using gated convolutional neural network," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 121–125.

[19] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[21] X. Li, S. Chen, X. Hu, and J. Yang, "Understanding the disharmony between dropout and batch normalization by variance shift," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2682–2690.

[22] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," *arXiv preprint arXiv:1807.10501*, 2018.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

[25] L. JiaKai, "Mean teacher convolution system for DCASE 2018 task 4," *Tech. Rep., DCASE Challenge*, 2018.

An Effective Speaker Recognition Method Based on Joint Identification and Verification Supervisions

Ying Liu[1], *Yan Song*[1], *Yiheng Jiang*[1], *Ian McLoughlin*[1,2], *Lin Liu*[3], *Lirong Dai*[1]

[1]National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei, China.
[2]ICT Cluster, Singapore Institute of Technology
[3]iFLYTEK Research, iFLYTEK CO., LTD., Hefei, Anhui 230088, China.

{ly1004, jiangyh}@mail.ustc.edu.cn, {songy, ivm, lrdai}@ustc.edu.cn, linliu@iflytek.com

Abstract

Deep embedding learning based speaker verification methods have attracted significant recent research interest due to their superior performance. Existing methods mainly focus on designing frame-level feature extraction structures, utterance-level aggregation methods and loss functions to learn discriminative speaker embeddings. The scores of verification trials are then computed using cosine distance or Probabilistic Linear Discriminative Analysis (PLDA) classifiers. This paper proposes an effective speaker recognition method which is based on joint identification and verification supervisions, inspired by multi-task learning frameworks. Specifically, a deep architecture with convolutional feature extractor, attentive pooling and two classifier branches is presented. The first, an identification branch, is trained with additive margin softmax loss (AM-Softmax) to classify the speaker identities. The second, a verification branch, trains a discriminator with binary cross entropy loss (BCE) to optimize a new triplet-based mutual information. To balance the two losses during different training stages, a ramp-up/ramp-down weighting scheme is employed. Furthermore, an attentive bilinear pooling method is proposed to improve the effectiveness of embeddings. Extensive experiments have been conducted on VoxCeleb1 to evaluate the proposed method, demonstrating results that relatively reduce the equal error rate (EER) by 22% compared to the baseline system using identification supervision only.

Index Terms: speaker verification, mutual information learning, attentive bilinear pooling, multi-task framework

1. Introduction

Speaker recognition (SR) is the task of automatically determining whether a given utterance belongs to a certain speaker identity. According to different recognition settings, SR can be categorized into either speaker identification (SID) which classifies a given utterance as being from a specific speaker, or speaker verification (SV), which is a binary classification problem that determines whether two given speech utterances belong to same speaker or not. Compared to SID, SV is an open-set recognition task with no overlap between training and test set, which is closely related to representation learning.

Over the past few decades, the most popular SV methods have been based on i-vector followed by Probabilistic Linear Discriminative Analysis (PLDA) [1, 2], in which the i-vector representation is generally learned in an unsupervised manner. Recently, deep embedding learning based SV methods have attracted significant interest due to their superior performance. Compared to traditional i-vector systems, deep learning based SV methods may benefit from the discriminative characteristics and large receptive-field of deep neural networks (DNNs).

Existing deep embedding learning architectures include time-delay DNN (TDNN) [3], convolutional neural network (CNN) [4, 5, 6], and Long Short-Term Memory (LSTM) networks [7]. Generally, these architectures consist of a frame-level feature extractor, an utterance-level aggregator and a classifier, which can be optimized in an end-to-end way.

Many recent works have focused on utterance-level aggregation methods, $e.g.$, statistical pooling [3], attentive pooling [8], bilinear pooling [4], and dictionary based pooling methods [9, 10]. Meanwhile, other works have proposed different loss functions, including triplet loss [11, 12], center loss [9], triplet-center loss [13], angular softmax loss (A-Softmax) [9] and additive margin softmax loss (AM-Softmax) [14, 15]. However, in most deep embedding learning based methods, the network architectures are trained under identification supervision, optimized for the SID task. Meanwhile, for SV tasks, the verification score between utterance pairs is computed via cosine distance or through an additional trainable backend (*e.g.* PLDA). However, it is still difficult to directly incorporate an effective backend into a deep embedding learning architecture [4].

In this paper, an effective speaker recognition method is proposed based on joint identification and verification supervisions. This is inspired by the multi-task learning framework, as shown in Fig. 1 and detailed in Section 3. Specifically, this includes a deep architecture with frame level feature extractor, attentive pooling and two branches of classifiers. The first branch is similar to deep embedding learning, in which a speaker classifier is optimized via AM-Softmax loss to discriminate the learned speaker embeddings. The second branch optimizes a new triplet-based mutual information (T-MI) between positive and negative samples extracted from the embedding space, inspired by triplet loss [11, 12] and mutual information based representation learning [16]. As with generative adversarial networks (GANs), we train a discriminator to separate them, using binary cross entropy loss (BCE). To prevent the issue of an imbalance between AM-softmax and BCE loss at different training stages, we introduce a ramp-up/ramp-down weighting scheme. In addition, a new attentive bilinear pooling method (ABP) is proposed, aiming to improve performance by aggregating features along the temporal axis.

To evaluate the effectiveness of the proposed method, extensive experiments have been conducted on the Voxceleb1 benchmark [17]. By jointly optimizing the identification and verification, our method can relatively reduce the equal error rate (EER) by 22% compared to the baseline system using identification supervision only.

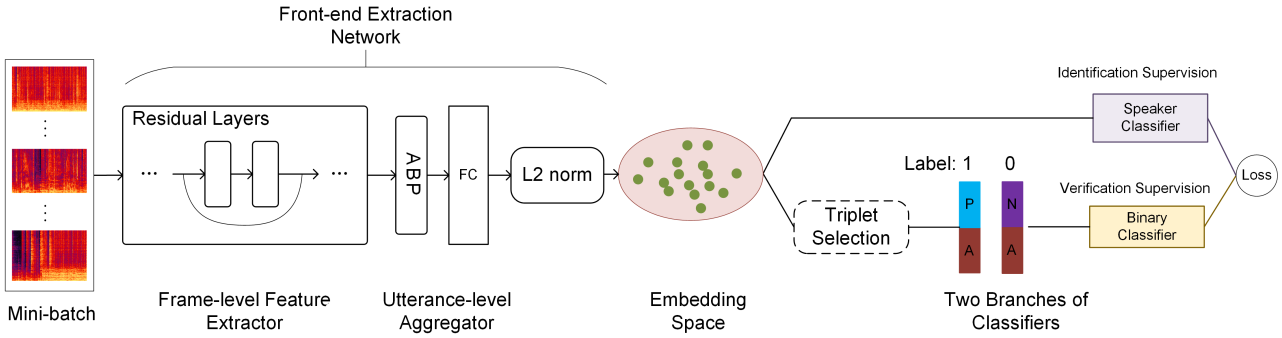


Figure 1: *Framework of the proposed SV method based on joint identification and verification supervisions.*

2. Overview of the proposed multi-task learning framework

The proposed multi-task learning based speaker recognition framework is shown in Fig. 1. It consists of frame-level feature extractor, utterance-level aggregator, and two branches of classifiers.

The frame-level feature extractor is adapted from the existing ResNet-18 architecture [18], which comprises an input convolutional layer and 4 residual stages. The main difference lies in that we keep the temporal and frequency dimensions of feature maps in each residual stage unchanged, and insert a transition layer to reduce the frequency dimension.

The aggregator is then followed to map the extracted frame-level features into utterance-level representations. In this paper, a novel attentive bilinear pooling (ABP) method is introduced to improve the effectiveness of embeddings, detailed in Section 3.3. Then an embedding layer, implemented by a fully connected (FC) layer, is added to make a nonlinear transformation and dimension reduction to obtain speaker embeddings.

The speaker embeddings are firstly length normalized and then fed into two branches of classifiers for multi-task learning. The first, an identification branch, is implemented by a FC layer and trained with AM-Softmax loss for SID task. The second, a verification branch, accomplishes the SV task by first constructing the positive and negative pairs from the selected triplet, and then training a binary classifier with BCE loss. Finally, a ramp-up/ramp-down weighting scheme is employed to balance the AM-softmax and BCE loss for multi-task learning.

During testing, we can either extract speaker embeddings from the embedding layer for the enrolment and test set, and then use a traditional PLDA backend to calculate verification scores, or directly use the output of the verification branch, giving scores in an end-to-end fashion.

3. Methods

3.1. Triplet-based mutual information (T-MI) learning

Mutual Information (MI) of statistical dependence is a promising tool for learning representations in an unsupervised way [16]. Given two random variables x and y, MI can be defined as

$$\begin{aligned} MI(x;y) &= \int_x \int_y p(x,y) \log \left\{ \frac{p(x,y)}{p(x)p(y)} \right\} dxdy \\ &= D_{KL}\{p(x,y) \| p(x)p(y)\} \end{aligned} \tag{1}$$

where D_{KL} is the Kullback-Leibler (KL) divergence between the joint distribution $p(x,y)$ and product of marginals $p(x)p(y)$. The MI is minimized when the random variable x and y are statistically independent, and is maximized when they contain identical information. For SV task, inspired by triplet loss [12], we can construct triplet (x_a, x_p, x_n), where x_a and x_p are utterances from the same speaker, x_a and x_n are from the different speakers. And then discriminative speaker representations can be effectively learned by maximizing MI between x_a and x_p, and minimizing it between x_a and x_n. This is logical, however, MI is hard to measure directly.

Fortunately, previous research [19] has found it possible to optimize the MI within an encoder-discriminator framework. Motivated by [16], a verification branch is designed as the discriminator using T-MI learning. Specifically, the front-end extraction network, including a frame-level feature extractor and an utterance-level aggregator, is used as the encoder, denoted by $f_\theta(\cdot)$. Embeddings of the triplet can be obtained by feeding it through the network. Then positive embedding pair $(f_\theta(x_a), f_\theta(x_p))$ and negative embedding pair $(f_\theta(x_a), f_\theta(x_n))$ are formed and passed through the verification branch, implemented by a binary classifier denoted by $g_\phi(\cdot)$, for discriminating verification. The positive pair and the negative pair are labeled '1' and '0' respectively, and the standard binary cross entropy loss (BCE) is used as the objective function to train the classifier:

$$\begin{aligned} L_{ver} = \frac{1}{N} \sum_{i=1}^{N} &- \log \left\{ g_\phi(f_\theta(x_a^i) \oplus f_\theta(x_p^i)) \right\} \\ &- \log \left\{ 1 - g_\phi(f_\theta(x_a^i) \oplus f_\theta(x_n^i)) \right\} \end{aligned} \tag{2}$$

where $\oplus$ denotes the concatenation operator. The BCE loss in Eq. (2) actually estimates the *Jansen-Shannon* divergence between positive and negative distributions, which is similar to the KL-based definition of MI [16].

The main difference to [16] is that we construct triplet with respect to label information in a mini-batch, which in fact introduces the verification supervision. Therefore, the output sigmoid probability of the verification branch can be used as a similarity measure of two embeddings, which conveniently allows the system to be trained end-to-end without PLDA backend or cosine distance calculation. Given input pair (x_1, x_2), the verification score can be obtained as:

$$score(x_1, x_2) = g_\phi(f_\theta(x_1) \oplus f_\theta(x_2)) \tag{3}$$

3.2. Joint optimization of identification and verification

As discussed above, the multi-task system is optimized jointly with identification and verification supervisions. For SID task, AM-Softmax loss is used as the objective function:

$$L_{iden} = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s\cdot(cos\theta_{y_i}-m)}}{e^{s\cdot(cos\theta_{y_i}-m)}+\sum_{j=1,j\neq y_i}^{c} e^{s\cdot cos\theta_j}} \tag{4}$$

where s is the scale parameter and m is the margin. In our experiments, we set s = 18 and m = 0.1. The total loss for joint optimization is the weighted sum of the identification loss and verification loss:

$$L = \lambda L_{iden} + \mu L_{ver} \tag{5}$$

where the λ and μ are weight parameters. To balance these two loss components at different training stages, a ramp-up/ramp-down weighting scheme is introduced. The weight μ starts from zero and ramps up along the curve $\mu(t) = \mu_0 e^{-5\{1-t/T_1\}^2}$, and similarly, λ ramps down according to the curve $\lambda(t) = \lambda_0 e^{-5\{(t-T_2)/(T_3-T_2)\}^2}$, where t is training epoch, μ_0 is the final value of μ, λ_0 is the initial value of λ, [0, T_1] and [T_2, T_3] are the durations of ramp-up and ramp-down periods respectively. In our experiments, μ_0 and λ_0 are set to 1.

3.3. Attentive bilinear pooling (ABP)

Inspired by [4, 20], an attentive bilinear pooling (ABP) method is further utilized to force the model to pay more attention to useful information for aggregation. It calibrates the output frame-level features with learnable convolutional layer to provide frame-wise attention mechanism.

Specifically, let $\mathbf{H} \in \mathbf{R}^{L\times D}$ be the frame-level feature map captured by the hidden layer below the self-attention layer, where L and D are the number of frames and feature dimension respectively. Then the attention map $\mathbf{A} \in \mathbf{R}^{L\times K}$ can be obtained by feeding $\mathbf{H}$ into a 1×1 convolutional layer followed by softmax non-linear activation, where K is the number of attention heads. The 1^{st}-order and 2^{nd}-order attentive statistics of $\mathbf{H}$, denoted by $\boldsymbol{\mu}$ and $\boldsymbol{\sigma^2}$, can be computed similar as cross-layer bilinear pooling [4], which is

$$\begin{aligned} \boldsymbol{\mu} &= T_2(T_1(\mathbf{H}^T\mathbf{A})) \\ \boldsymbol{\sigma^2} &= T_2(T_1(\mathbf{H}^T(\mathbf{A}\odot\mathbf{A})) - (T_1(\mathbf{H}^T\mathbf{A}))\odot T_1(\mathbf{H}^T\mathbf{A})) \end{aligned} \tag{6}$$

where $T_1(x)$ is the operation of reshaping x into a vector, and $T_2(x)$ includes a signed square-root step and a L2-normalization step. $\odot$ represents the Hadamard product. The output of ABP is the concatenation of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma^2}$.

It is worth noting that the proposed ABP method derives the attention map using the softmax along temporal axis to obtain the attention for each frame. And the attentive 2^{nd}-order statistics information is further exploited for aggregation, similar as statistics pooling in [21]. This is different from the existing pooling methods, such as NetVLAD [10] and learnable dictionary encoding (LDE) [9], which mainly focus on deriving Baum-Welch statistics over the channel dimension.

Compared to multi-head attentive pooling [20], ABP further normalizes the length of statistics before concatenation, which is able to extract more robust embeddings and achieve better performance.

Table 1: *Detailed configuration of the front-end extraction network.*

Layer	Structure	Stride	O/p size
Conv1	$7\times7,\ 16$	1×1	$L\times35\times16$
Res1	$\begin{bmatrix}3\times3,\ 16\\3\times3,\ 16\end{bmatrix}\times2$	1×1	$L\times35\times16$
Trans1	$3\times3,\ 32$	1×2	$L\times17\times32$
Res2	$\begin{bmatrix}3\times3,\ 32\\3\times3,\ 32\end{bmatrix}\times2$	1×1	$L\times17\times32$
Trans2	$3\times3,\ 64$	1×2	$L\times8\times64$
Res3	$\begin{bmatrix}3\times3,\ 64\\3\times3,\ 64\end{bmatrix}\times2$	1×1	$L\times8\times64$
Trans3	$3\times3,\ 128$	1×2	$L\times3\times128$
Res4	$\begin{bmatrix}3\times3,\ 128\\3\times3,\ 128\end{bmatrix}\times2$	1×1	$L\times3\times128$
Trans4	$3\times3,\ 128$	1×2	$L\times1\times128$
ABP	-	-	$1\times(128\times2K)$
FC	$(128\times2K)\times128$	-	1×128
L2norm	-	-	1×128

4. Experimental setup and results

4.1. Dataset and input features

To investigate performance of the proposed system, we conducted extensive experiments using VoxCeleb1 benchmark [17] which contains over 140,000 utterances from 1251 speakers. The training set is the development portion without data augmentation, containing 1,211 speakers and the evaluation set consists of 37,720 trial pairs from 40 speakers.

The feature extraction process uses Kaldi [22]. In our implementation, 41-dimensional filter bank outputs (FBank) are used as acoustic features, obtained from 25ms windows with 10ms shift between frames. We apply mean-normalization over a sliding window of 3s, and use voice activity detection (VAD) to remove silent segments. The features from the training dataset are randomly truncated into short slices ranging from 2 to 4s. For evaluation, utterances are divided equally into 10 slices with 4s duration.

4.2. Model configuration

The detailed configuration of the front-end extraction network is summarized in Table 1, where L denotes variable-length data frames. The input layer consists of a single convolutional layer with kernel size of 7x7, stride of 1x1 and channel dimension of 16. Four residual stages include [2,2,2,2] basic blocks with 16, 32, 64, 128 channels respectively, and each basic block having 2 convolutional layers with filter sizes of 3x3 and a stride of 1x1. The transition layer comprises a convolutional layer with kernel size of 1x1 and stride of 1x2. After the four stages, the frequency dimension of the feature map is reduced to 1. For ABP, the output dimension $128\times2K$ is varied with different attention heads K.

The identification branch is implemented by a FC layer with units equal to the number of speaker categories. We should note that when computing the AM-Softmax loss, the weight of this layer need to be normalized. The verification branch comprises two FC layers followed by sigmoid activation.

The mini-batch size for training is set to 128, containing 64 speakers with 2 utterances from each. All neural networks are implemented using the PyTorch framework [25]. The network is optimized using stochastic gradient descent (SGD) [26] with

Table 3: *Results for verification on VoxCeleb1 dataset. (AP refers to average pooling and SP refers to statistics pooling)*

System	Aggregation	Loss	Training set	Similarity	EER, %
i-vector+PLDA [17]	-	-	Voxceleb1	PLDA	8.80
x-vector [23]	SP	Softmax	Voxceleb1	cosine	11.3
				PLDA	7.1
ResNet-34 [5]	SP	Softmax	Voxceleb1	cosine	5.01
				PLDA	4.74
ResNet-34 [9]	LDE	A-Softmax	Voxceleb1	cosine	4.56
ResNet-20 [14]	AP	AM-Softmax	Voxceleb1	cosine	4.30
ResNet-50 [24]	AP	Softmax+Contrastive	Voxceleb2	cosine	4.19
Thin ResNet-34 [10]	NetVLAD	AM-Softmax	Voxceleb2	cosine	3.32
ResNet-18 (Ours)	ABP	Softmax	Voxceleb1	cosine	3.76
ResNet-18 (Ours)	ABP	AM-Softmax	Voxceleb1	cosine	3.51
Multi-task ResNet-18 (Ours)	ABP	-	Voxceleb1	Verification output	**2.94**

momentum of 0.95 and weight decay of 5e-4. Each network is trained for 60 epochs with initial learning rate of 0.1, gradually decreasing to 0.0001. The durations of ramp-up and ramp-down periods are set to [0, 25] and [25, 40] epochs respectively. The performance is evaluated in terms of equal error rate (EER).

4.3. Results

4.3.1. Evaluation on different number of attention heads K

In Table 2, we study the effect of different number of attention heads K in proposed ABP method. Same as most existing deep embedding learning based SV methods, these results are obtained by using the modified ResNet-18 with Softmax loss to learn speaker embeddings first, and evaluating the verification scores with cosine distance measure. From Table 2, we can see that the EER reduces from 4.07% to 3.76% when K increases from 2 to 16. This indicates that increasing the number of attention heads can improve the effectiveness of the proposed ABP method. However, large value of K may lead to large model size and high computational complexity. In the following experiments, we only report the results with $K = 16$, considering the trade-off between effectiveness and efficiency.

4.3.2. Main results

The main results are reported in Table 3. We compared the performance of three systems including: 1) ResNet-18 with Softmax loss, 2) ResNet-18 with AM-Softmax loss, and 3) Multi-task ResNet-18. The first two systems are implemented following the existing deep embedding learning based methods, which compute the verification scores via cosine distance measure. The multi-task ResNet-18 is implemented using the proposed speaker recognition method based on joint identification and verification supervisions, and the performance is evaluated according to the output of the verification branch directly.

From Table 3, we see that the proposed system outperforms all other SV methods by a large margin. Specifically, ResNet-18 with Softmax loss achieves an EER of 3.76%, which is better than the systems in [5, 9, 14, 23], demonstrating the effectiveness for embedding learning of our modified ResNet-18 architecture and ABP method. Thanks to the role of the margin parameter, ResNet-18 with AM-Softmax loss achieves an EER of 3.51%, which is a slight improvement compared with the Softmax model. This result is also better than 4.30% in ResNet-34 using LDE aggregation in [9] with the same experimental settings, and comparable to the 3.32% in Thin ResNet-34 using NetVLAD aggregation in [10] with much larger Voxceleb2 training set. This indicates the superiority of our proposed ABP method.

The EER is further reduced to 2.94% with Multi-task ResNet-18, outperforming almost all other deep embedding learning based SV systems in the same situation. It is worth noting that we could further extract speaker embeddings to train an additional PLDA. When we add a PLDA step, the joint effect of the additional verification supervision allows performance to be further improved to 2.81%.

Table 2: *Results on different numbers of attention heads K.*

K	2	4	8	16
EER, %	4.07	3.91	3.82	**3.76**

5. Conclusion

In this paper, inspired by a multi-task framework, an effective speaker recognition method based on joint identification and verification supervision is proposed. Specifically, a deep architecture with convolutional feature extractor, attentive pooling and two branches of classifiers is presented. The first, an identification branch, is trained with AM-Softmax loss for speaker identity classification. The second, a verification branch, trains a discriminator with BCE loss to optimize the MI between positive and negative samples extracted from the embedding space. To balance these two losses at different training stages, a novel ramp-up/ramp-down weighting scheme is employed and, furthermore, a novel attentive bilinear pooling method is proposed. This further improves the effectiveness of embeddings. Experiments conducted on the Voxceleb1 benchmark yield exceptional results, demonstrating the effectiveness of the proposed model for the SV task.

6. Acknowledgements

This work was supported by National Natural Science Foundation of China (Grant No. U1613211), the Leading Plan of CAS (XDC08030200), and by Key Science and Technology Project of Anhui Province (Grant No. 17030901005).

7. References

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

[2] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Proc. Odyssey*, 2010.

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey *et al.*, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. ICASSP*, 2018, pp. 5329–5333.

[4] Z. Gao, Y. Song, I. McLoughlin, W. Guo, and L. Dai, "An improved deep embedding learning method for short duration speaker verification," in *Proc. Interspeech*, 2018, pp. 3578–3582.

[5] W. Cai, J. Chen, and M. Li, "Analysis of length normalization in end-to-end speaker verification system," in *Proc. Interspeech*, 2018.

[6] Y. Jiang, Y. Song, I. McLoughlin, Z. Gao, and L. Dai, "An effective deep embedding learning architecture for speaker verification," in *Proc. Interspeech*, 2019, pp. 4040–4044.

[7] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *Proc. ICASSP*, 2018, pp. 4879–4883.

[8] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *Proc. Interspeech*, 2018, pp. 2252–2256.

[9] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," in *arXiv preprint arXiv:1804.05160*, 2018.

[10] W. Xie, A. Nagrani, J. S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," *Proc. ICASSP*, 2019.

[11] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, and X. Liu, "Deep speaker: an end-to-end neural speaker embedding system," in *arXiv preprint arXiv:1705.02304*, 2017.

[12] S. Novoselov, V. Shchemelinin, A. Shulipa, A. Kozlov, and I. Kremnev, "Triplet loss based cosine similarity metric learning for text-independent speaker recognition," in *Proc. Interspeech*, 2018, pp. 2242–2246.

[13] Y. Jiang, Y. Song, J. Yan, L. Dai, and I. McLoughlin, "Triplet-center loss based deep embedding learning method for speaker verification," in *Proc. APSIPA*, 2019.

[14] M. Hajibabaei and D. Dai, "Unified hypersphere embedding for speaker recognition," in *arXiv preprint arXiv:1807.08312*, 2018.

[15] Y. Liu, L. He, and J. Liu, "Large margin softmax loss for speaker verification," in *Proc. Interspeech*, 2019, pp. 2873–2877.

[16] M. Ravanelli and Y. Bengio, "Learning speaker representations with mutual information," in *Proc. Interspeech*, 2019, pp. 1153–1157.

[17] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A large-scale speaker identification dataset," in *Proc. Interspeech*, 2017.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. ICASSP*, 2018, pp. 5334–5338.

[19] P. Brakel and Y. Bengio, "Learning independent features with adversarial nets for non-linear ica," in *arXiv preprint arXiv:1710.05050*, 2017.

[20] Y. Zhu, T. Ko, D. Snyder, B. Mak *et al.*, "Self-attentive speaker embeddings for text-independent speaker verification," in *Proc. Interspeech*, 2018, pp. 3573–3577.

[21] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," in *Proc. Interspeech*, 2017, pp. 999–1003.

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget *et al.*, "The Kaldi speech recognition toolkit," in *Proc. ASRU*. IEEE Signal Processing Society, 2011.

[23] S. Shon, H. Tang, and J. Glass, "Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-toend model," in *arXiv preprint arXiv:1809.04437*, 2018.

[24] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Proc. Interspeech*, 2018, pp. 1086–1090.

[25] A. Paszke, S. Gross, S. Chintala, G. Chanan *et al.*, "Automatic differentiation in pytorch," 2017.

[26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.

Speaker Adaptive Training for Speech Recognition Based on Attention-over-Attention Mechanism

Genshun Wan[1]*,Jia Pan*[1]*,Qingran Wang*[2]*,Jianqing Gao*[2]*,Zhongfu Ye*[1]

[1]University of Science and Technology of China
[2]iFlytek Research, iFlytek Co., Ltd.

{gswan,panjia}@mail.ustc.edu.cn, {qrwang2,jqgao}@iflytek.com, yezf@ustc.edu.cn

Abstract

In our previous work, we introduced a speaker adaptive training method based on frame-level attention mechanism for speech recognition, which has been proved an effective way to do speaker adaptive training. In this paper, we present an improved method by introducing the attention-over-attention mechanism. This attention module is used to further measure the contribution of each frame to the speaker embeddings in an utterance, and then generate an utterance-level speaker embedding to perform speaker adaptive training. Compared with the frame-level ones, the generated utterance-level speaker embeddings are more representative and stable. Experiments on both the Switchboard and AISHELL-2 tasks show that our method can achieve a relative word error rate reduction of approximately 8.0% compared with the speaker independent model, and over 6.0% compared with the traditional utterance-level d-vector-based speaker adaptive training method.

Index Terms: speech recognition, speaker adaptive training, attention-over-attention

1. Introduction

Recently, deep neural networks (DNNs) such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have become the mainstream structure of automatic speech recognition (ASR) [1–3]. However, when it comes to the speakers with special accents or pronunciation habits, the accuracy of ASR may suffer a significant reduction. In response, speaker adaptive training (SAT) is one of the effective approaches to improve the performance of ASR on these conditions.

The most widely used methods of SAT can be classified into two categories: auxiliary features and adversarial learning. Auxiliary features that contain information about speakers are used to perform speaker adaptive training. Speaker i-vectors or bottleneck vectors, obtained by a pretrained speaker recognition model, can be used with the acoustic features together to make the acoustic model generalize better to different speakers [4–7]. In addition, speaker codes [8–10] can also be used to represent speaker characteristics. To highlight the importance of the speaker embeddings in adaptation, the authors in [11, 12] try to generate the speaker-dependent (SD) parameters via a controller network that takes speaker embeddings as input, and the controller network is shared among all speakers. Another type of methods to perform SAT is using the adversarial learning scheme. Similar to the methods used in domain adaptation [13–15], the acoustic model and the speaker classification model are jointly optimized via adversarial learning [16]. In [17], a reconstruction network is trained to predict the input speaker i-vector. The mean-squared error loss of the i-vector reconstruction and the cross-entropy loss of the acoustic model are jointly optimized through adversarial multi-task learning.

The speaker adaptive training methods mentioned above should be helpful when a number of adaptation data is provided. However, in real-world ASR systems, the collection of sufficient speaker data is very difficult, particularly the labeled data. Insufficient data will introduce an inaccurate speaker embedding and make a sharp decline in performance for speaker adaptive training. As one of the mainstream approaches, the i-vector-based speaker adaptive training method takes the i-vectors obtained in advance as the speaker embeddings. However, their performance is unsatisfactory because the i-vector is obtained without regard to the speech recognition task.

In order to provide a dynamic speaker embedding associated with the speech recognition performance, a speaker adaptive training method based on attention mechanism is proposed in our previous work [18]. The i-vectors of all speakers in the training data are obtained as a static memory in advance. For each frame, the closest speaker i-vector is selected with attention mechanism which is learned jointly with the acoustic model from the training data. However, subject to the limited and partially invalid frames, the speaker representation would be unstable and uniform to a certain extent.

In this paper, we propose a speaker adaptive training method for speech recognition based on attention-over-attention mechanism. For each utterance, the nearest d-vectors are selected and then recombined to an utterance-level aggregated vector by attention-over-attention mechanism. The aggregated vector is connected with the acoustic model to provide the information about the current speaker. Compared with the traditional utterance-level d-vectors or the frame-level aggregated d-vectors metioned in [18], aggregated utterance-level vectors can provide a more accurate and robust speaker representation to improve the recognition accuracy. Experiments on the Switchboard and AISHELL-2 tasks show that the proposed method achieves a significant improvement over the SD method based on utterance-level d-vectors.

2. Related Work

Since the speaker representation is essential to the speaker adaptive training, there have been intensive researches to optimize the speaker embeddings and create the direct relationship between the speaker embeddings and the speech recognition performance with limited resources. As mentioned above, an attention based speaker adaptive training method is proposed in [18]. As illustrated in Fig. 1, the framework mainly consists of two parts: the main network and the attention module.

The main network of the proposed method is the same as other structures of acoustic model, including feedforward neural networks, CNNs and RNNs. The main network plays two roles: acoustic modeling and providing information for the attention module. As a kind of weak information, speak char-

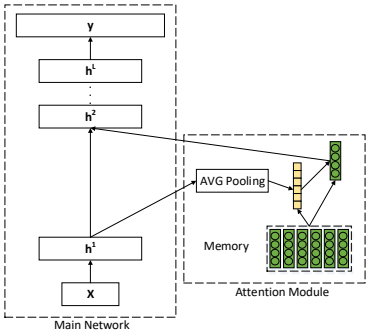


Figure 1: *The framework of the speaker adaptive training method based on attention mechanism.*

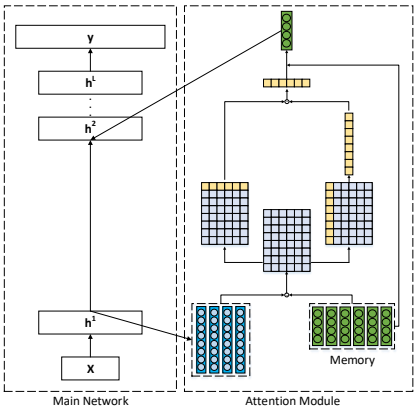


Figure 2: *The framework of the speaker adaptive training method based on attention-over-attention mechanism.*

acteristics need to be extracted and used before it is removed finally by deep neural networks. Therefore, the outputs of the hidden layers near the input layer are provided for the attention module.

The attention module is equipped mainly to select the vectors that are most similar to the current frame from the memory and combine them into a vector named the frame-level aggregated speaker vector. The memory consists of a group of vectors, such as i-vectors [19] and d-vectors [20], which are easily distinguished from each other by its corresponding speaker. As an effective representation of the speaker, the aggregated speaker vector based on attention mechanism is used together with the acoustic features to do speaker adaptive training.

Experiments on the Switchboard task show that the speaker adaptive training method based on attention mechanism could achieve a decent performance improvement compared to that of the i-vector-based speaker adaptive training method.

3. The Proposed Method

3.1. Motivation

In our previous work [18], the frame-level aggregated speaker vectors based on attention mechanism is used to represent the frame-level speaker embeddings. However, because only the history part of current utterance can be used to gather the speaker information during the process of attention module at each frame, especially for the first few frames, the speaker representation would be unstable and uniform. In addition, during the process of gathering the speaker information, an average pooling is used to obtain the information. Average pooling means all the history frames have the same importance. When there are some abnormal frames with little speaker information, such as silence or environmental noise frame, average pooling strategy is obviously unreasonable to generate the representative speaker embeddings. That is to say, effective speaker information may be further weaken by the partially invalid frames. In order to make better use of the long-term speaker information and then form a representative utterance-level speaker embedding, the attention mechanism should focus not only on the importance distribution of each vector in the memory at each frame, but also the importance distribution of each frame. In order to maintain the coherence and consistency of the speaker embeddings within an utterance, we tend to make use of the utterance-level embeddings with attention mechanism.

Attention mechanism is widely used in many fields, such as machine translation and speech recognition. By putting different weights on different types of information, the process of model training becomes more flexible. Common attention mechanisms include: soft attention [21], hard attention [22] and self-attention [23]. In paper [24], another attention mechanism is placed over the existing attention to further strengthen the importance of each individual attention part. This kind of attention mechanism covering two dimensional space, called attention-over-attention mechanisms, offers a potential path to generate robust utterance-level embeddings. Further attention in time dimension can weaken the influence of some abnormal frame-level speaker embedding. Therefore, the embedding generated from all the frames of this utterance can be uniform and stable.

Starting with the generation of representative and robust utterance-level speaker embeddings, we propose a speaker adaptive training method for speech recognition based on attention-over-attention mechanism

3.2. SAT Based on Attention-over-Attention Mechanism

As illustrated in Fig. 2, the main structure of the speaker adaptive training method based on attention-over-attention mechanism is similar with our previous work. We replace the common attention mechanism by attention-over-attention mechanism to generate a more representative and stable speaker embedding.

In our study, the d-vectors of the speakers in training set are extracted as the memory. To obtain the d-vectors, a neural network is pre-trained by speaker discriminative criteria such as cross-entropy or triplet loss. Then, the output of the last hidden layer is obtained to produce a frame-level speaker representation, and all the frame-level representations are then averaged to form an utterance-level speaker embedding called the d-vector. Finally, the simplified method of clustering such as K-means is adopted to reduce the number of base vectors in a memory. Assuming that the memory has N vectors, the memory is denoted by $\mathbf{m} = \{\mathbf{m}_1,\mathbf{m}_2,...,\mathbf{m}_N\}$, in which $\mathbf{m}_i$ represents the i-th vector in this memory.

Given an utterance with T speech frames, the acoustic features of the main network are represented by $\mathbf{X} = \{\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_T\}$, where $\mathbf{x}_t$ represents the feature vector at the frame t. The corresponding outputs of the l-th hidden layer of the main network are denoted as $\mathbf{H}^l = \{\mathbf{h}_1^l,\mathbf{h}_2^l,...,\mathbf{h}_T^l\}$.

After obtaining the output of the hidden layer near the input layer $\mathbf{h}^{low}$ and the memory $\mathbf{m}$, we calculate a similarity degree matrix, which indicates the similarity scores between speaker information for each frame and each vector in the memory. We compute the matrix $\mathbf{M} \in \mathbb{R}^{|T|\times|N|}$ by the dot product between

the transformation output vector of $\mathbf{h}^{low}$ at the frame t and the i-th memory vector.

$$M(t,i) = (\mathbf{W}^m \mathbf{h}_t^{low}) \odot \mathbf{m}_i^\top \tag{1}$$

Based on the similarity degree matrix $\mathbf{M}$, we apply a row-wise softmax function to get the similarity scores for each row. We denote $\alpha(t) \in \mathbb{R}^{|N|}$ as the memory-level attention at the frame t. $\alpha(t)$ indicates the similarity degree to each vector at this frame, as described in the following formula.

$$\begin{aligned} \alpha(t) &= softmax(M(t,1), ..., M(t,|N|)) \\ \alpha &= [\alpha(1), \alpha(2), ..., \alpha(|T|)] \end{aligned} \tag{2}$$

The contributions to the speaker embeddings of each frame are obvious different, particularly those featured as environmental noise, and so on. On the contrary, the utterance-level speaker embeddings are more robust than the frame-level ones. Instead of averaging α at all the frames to form a final attention score, another attention mechanism is introduced to determine the importance of each individual attention.

We first calculate a column-wise softmax attention to get the similarity scores for each column, and denote $\beta(i) \in \mathbb{R}^{|T|}$ as the frame-level attention at the memory i. $\beta(i)$ indicates the importance degree of each frame corresponding to the i-th vector in the memory, as described in the following formula.

$$\beta(i) = softmax(M(1,i), ..., M(|T|, i)) \tag{3}$$

Then, we average all the $\beta(i)$ to get an averaged attention β. β is also an attention score vector with T dimensions, and it can be taken as the final importance degree of each frame.

$$\beta = \frac{1}{N} \sum_{i=1}^{N} \beta(i) \tag{4}$$

Since α is a combination of memory-level attention from 1 to T and β is a frame-level attention, we can calculate the matrix multiplication between α and β to get the final attention score $\mathbf{a}$ in an utterance. Attention score $\mathbf{a}$ is a N-dimensional vector.

$$\mathbf{a} = \alpha \odot \beta^\top \tag{5}$$

The normalized attention values $\mathbf{a}$ are used to compute a weighted sum of the vectors in the memory and formula the final utterance-level aggregated speaker vector $\mathbf{c}$.

$$\mathbf{c} = \sum_{i=1}^{N} \alpha_i \mathbf{m}_i \tag{6}$$

Finally, we connect the aggregated speaker vector with the main network, and the main network generates speaker-normalized representations using the speaker information in the aggregated speaker vector. For each frame, the aggregated speaker vector $\mathbf{c}_t$ is the same as $\mathbf{c}$. A simple connection method is concatenating the aggregated speaker vector with the outputs of the hidden layers of the main network as $\hat{\mathbf{h}}_t^{low} = [\mathbf{h}_t^{low\top}, \mathbf{c}_t{}^\top]^\top$.

The final loss function at the cross-entropy training stage of the proposed method is described by the following formula:

$$\mathcal{L} = \sum_{s=1}^{S} \sum_{t=1}^{T_s} \log p(y_t^s | \mathbf{x}_t^s, \mathbf{m}) \tag{7}$$

In Eq.(7), $\mathbf{x}_t^s$ and y_t^s indicate the acoustic feature vector and the triphone state label for frame t in utterance s. T_s is the number of frames of utterance s, and S is the number of total utterances in the training set.

4. Experiments and result analysis

4.1. Experimental Setup

We evaluated the performance of the proposed approach on both English and Mandarin speech recognition tasks.

The English training data of the Switchboard (SWB) task [25] consists of 20-hour English CALLHOME and 309-hour Switchboard-I dataset, including a total of 5110 speakers. The SWB part of NIST 2000 Hub5 evaluation set is taken as test set, and it contains 1831 utterances from 40 speakers in total. The Mandarin training data of AISHELL-2 task [26] consists of 1000 hours of clean audio segments recorded via the iPhone channel from 1991 speakers, including 1293 speakers with slight northern accents, 678 speakers with southern accents and 20 speakers with other accents. The test set contains 5000 utterances from 10 speakers, and each speaker has approximately half an hour of audio segments.

4.2. Baseline systems

The SI baseline was trained with a VGG-like [27] model architecture based on frame-level cross-entropy criterion. The inputs of the model were the 40-dimensional log Mel-scale filter-bank features. The architecture of the model mainly consisted of convolutional and pooling layers, and each convolutional layer was equipped with a standard ReLU activation function. We shuffled the utterances in training data and grouped them into mini-batches with a limit of 2048 frames per minibatch to speed up training. Stochastic gradient descent was used as the optimizer, and the initial learning rate was set to 0.02. All subsequent experiments were performed by the CAFFE toolkit [28] and run on a server equipped with 4 Tesla P40 GPUs.

In our paper, speaker d-vectors are taken as additional inputs to perform speaker adaptive training. The speaker verification network included five convolutional layers. The utterances belonging to the same speaker were concatenated and split into audio segments, each of which had 500 frames. 64-dimensional log Mel-scale filter-bank features were taken as the input.

The d-vector-based SD models were evaluated at both the speaker and utterance levels. During the testing steps, the utterance-level d-vectors were extracted from each utterance separately and the speaker-level d-vectors were extracted using all the utterances from the same speaker. Table 1 reports the word error rate (WER) of the baseline models on SWB task. The performance at the utterance level is much worse than that at the speaker level.

Table 1: *Performance of the baseline models on the SWB task.*

Method	**WER**	**WERR**
SI baseline	13.8	–
SD baseline(speaker-level)	13.0	5.8%
SD baseline(utterance-level)	13.5	2.2%

4.3. Results of the proposed method

For speaker adaptive training method based on the d-vector memory, all speaker-level vectors in the training set were clustered into 128 classes via the K-means algorithm.

Table 2 reports the performance of the proposed method on the SWB task. For our previous work, speaker adaptive training method based on the traditional frame-level attention mecha-

nism achieves a relative 4.3% WER reduction (WERR) compared with the SI model and a relative 2.3% WER reduction over the utterance-level d-vector-based SD model. When we substitute the traditional frame-level attention mechanism with the utterance-level attention-over-attention (AOA) mechanism, the proposed method achieves a relative 8.0% WER reduction over the SI baseline model and a relative 5.9% WER reduction over the utterance-level d-vector-based SD model. In addition, the result of SAT based on AOA mechanism also has lower WER than the speaker-level d-vector-based SD model. If we just average the memory-level attention at all the frames to form the final utterance-level attention directly, no significant improvement can be achieved. For a deep convolution neural network, computational complexity hardly changes at all. Compared with the traditional frame-level attention or utterance-level average attention mechanism, attention-over-attention further strengthen the discrimination among each frame-level speaker embeddings based on the utterance-level long information. In light of this, the generation of the aggregated speaker vector is more robust.

Table 2: *Performance of the proposed method on the SWB task.*

Method	**WER**	**WERR**
SI baseline	13.8	–
SAT with traditional frame-level Att. [18]	13.2	4.3%
SAT with utterance-level average Att.	13.1	5.1%
SAT with utterance-level AOA	12.7	8.0%

We also presented the results on AISHELL-2 task. The results shown in Table 3 are consistent with the results on SWB task. The proposed method achieves a relative 8.3% WER reduction over the SI model and a relative 7.0% WER reduction over the utterance-level d-vector-based SD model.

Table 3: *Performance of the proposed method on the AISHELL-2 task.*

Method	**WER**	**WERR**
SI baseline	7.2	–
SD baseline(speaker-level)	6.9	4.2%
SD baseline(utterance-level)	7.1	1.4%
SAT with traditional frame-level Att. [18]	6.9	4.2%
SAT with utterance-level AOA	6.6	8.3%

To verify the improvement of the speaker embeddings, we compared the aggregated speaker d-vectors with the utterance-level d-vectors with t-distributed stochastic neighbor embedding (t-SNE) [29] on test data. 10 utterances from the same speaker were first randomly picked, and then the utterance-level d-vectors based on attention-over-attention were obtained directly during the attention module. And the utterance-level d-vectors based on the traditional frame-level attention could be generated by averaging all the frame-level aggregated speaker vectors in an utterance. For comparison, traditional utterance-level d-vectors were also extracted for each utterance of each speaker.

As shown in Fig. 3, the aggregated speaker vectors based on different attention mechanism are all closer to the speaker-level d-vector than the traditional utterance-level ones. And compared with the traditional frame-level attention mechanism, the

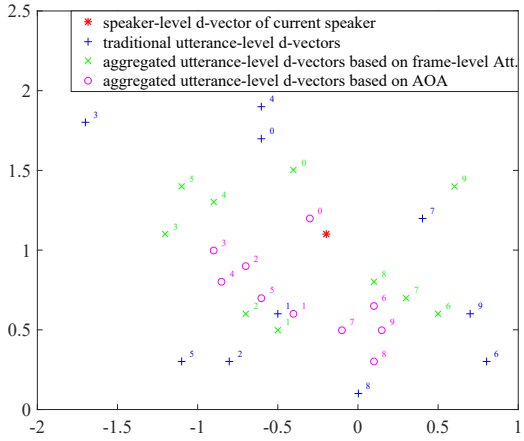


Figure 3: *t-SNE of the different speaker vectors in test set.*

aggregated speaker vectors based on attention-over-attention are more concentrated, which indicates the offsets influenced by the frame with little speaker information are effectively reduced. In order to solidify the conclusion, we calculated the euclidean distance between the utterance-level speaker vector and speaker-level d-vector for all the utterances of all the speakers in the test set, and got the mean and variance of the distance. As shown in Table 4, compared with the aggregated utterance-level d-vectors based on the traditional frame-level attention mechanism, the mean of euclidean distance of the aggregated utterance-level d-vectors based on attention-over-attention mechanism is relatively closer while the variance of the euclidean distance has great advantages. The results prove the superiority of the attention-over-attention mechanism again.

Table 4: *The mean and variance of euclidean distance between different utterance-level d-vectors and speaker-level d-vectors.*

Utterance-level d-vectors	**Mean**	**Variance**
traditional d-vectors	1.43	0.13
aggregated d-vectors based on Att.	1.29	0.10
aggregated d-vectors based on AOA	1.24	0.06

5. Conclusions

In this study, we have proposed a speaker adaptive training method for speech recognition with attention-over-attention mechanism, which can be used to measure the speaker information contribution of each memory vector and each frame. Thus, the utterance-level aggregated vectors are more representative and stable. The results on the Switchboard and AISHELL-2 task show that our proposed approach can achieve relative word error rate reductions of 8.0% and 8.3% compared with the speaker independent model respectively, and 6.0%-7.0% compared to that of the traditional utterance-level d-vector-based SAT method. The utterance-level aggregated speaker vectors based on attention-over-attention mechanism yielded relative word error rate reductions of approximately 4.0% compared with the frame-level attention mechanism.

6. References

[1] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proceedings of Interspeech*, 2014, pp. 338–342.

[2] O. Abdel-Hamid, A. rahman Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Proceedings of ICASSP*, 2012, pp. 4277–4280.

[3] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. Lecun, "Very deep multilingual convolutional neural networks for lvcsr," in *Proceedings of ICASSP*, 2016, pp. 4955–4959.

[4] P. Karanasou, Y. Wang, M. J. Gales, and P. C. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *Proceedings of Interspeech*, 2014, pp. 2180–2184.

[5] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proceedings of ASRU*, 2013, pp. 55–59.

[6] Y. Miao, H. Zhang, and F. Metze, "Speaker adaptive training of deep neural network acoustic models using i-vectors," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 23, no. 11, pp. 1938–1949, 2015.

[7] P. Cardinal, N. Dehak, Y. Zhang, and J. Glass, "Speaker adaptation using the i-vector technique for bottleneck features," in *Proceedings of Interspeech*, 2015, pp. 2867–2871.

[8] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code," in *Proceedings of ICASSP*, 2013, pp. 7942–7946.

[9] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, "Direct adaptation of hybrid dnn/hmm model for fast speaker adaptation in lvcsr based on speaker code," in *Proceedings of ICASSP*, 2014, pp. 6389–6393.

[10] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 22, no. 12, pp. 1713–1725, 2014.

[11] X. Cui, V. Goel, and G. Saon, "Embedding-based speaker adaptive training of deep neural networks," in *Proceedings of Interspeech*, 2017, pp. 122–126.

[12] Y. Zhao, J. Li, S. Zhang, L. Chen, and Y. Gong, "Domain and speaker adaptation for cortana speech recognition," in *Proceedings of ICASSP*, 2018, pp. 5984–5988.

[13] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," *JMLR Workshop and Conference Proceedings*, vol. 37, pp. 1180–1189, 2015.

[14] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Proceedings of NIPS*, 2016, pp. 343–351.

[15] Z. Meng, J. Li, Y. Gong, and B. Juang, "Adversarial teacher-student learning for unsupervised domain adaptation," in *Proceedings of ICASSP*, 2018, pp. 5949–5953.

[16] Z. Meng, J. Li, Z. Chen, Y. Zhao, V. Mazalov, Y. Gong, and B. Juang, "Speaker-invariant training via adversarial learning," in *Proceedings of ICASSP*, 2018, pp. 5969–5973.

[17] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, and et al., "English conversational telephone speech recognition by humans and machines," in *Proceedings of Interspeech*, 2017.

[18] J. Pan, D. Liu, G. Wan, J. Du, Q. Liu, and Z. Ye, "Online speaker adaptation for lvcsr based on attention mechanism," in *Proceedings of APSIPA*, 2018, pp. 183–186.

[19] N. Dehak, P. 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.

[20] E. Variani, X. Lei, E. McDermott, I. Lopez, Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proceedings of ICASSP*, 2014, pp. 4052–4056.

[21] M. T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *Computer Science*, 2015.

[22] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," *Computer Science*, pp. 2048–2057, 2015.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N, Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *In Advances in Neural Information Processing Systems*, pp. 6000–6010, 2017.

[24] Y. Cui, Z. Chen, S. Wei, S. Wang, T. Liu, and G. Hu, "Attention-over-attention neural networks for reading comprehension," *ACL 2017*, pp. 593–602, 2017.

[25] J. Godfrey, E. Holliman, and J. McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Proceedings of ICASSP*, 1992, pp. 517–520.

[26] J. Du, X. Na, X. Liu, and H. Bu, "Aishell-2: Transforming mandarin asr research into industrial scale," in *CoRR*, 2018.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *CoRR*, 2014.

[28] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, 2014, pp. 675–678.

[29] L. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

Speaker Code Based Speaker Adaptive Training Using Model Agnostic Meta-learning

Huaxin Wu[1]*, Genshun Wan*[2]*, Jia Pan*[2]

[1]iFlytek Research, iFlytek Co., Ltd
[2]University of Science and Technology of China

hxwu2@iflytek.com, gswan@mail.ustc.edu.cn, panjia@mail.ustc.edu.cn

Abstract

The performance of automatic speech recognition systems can be improved by speaker adaptive training (SAT), which adapts an acoustic model to compensate for the mismatch between training and testing conditions. Speaker code learning is one of the useful ways for speaker adaptive training. It learns a set of speaker dependent codes together with speaker independent acoustic model in order to remove speaker variation. Conventionally, speaker dependent codes and speaker independent acoustic model are jointly optimized. However, this could make it difficult to decouple the speaker code from the acoustic model. In this paper, we take the speaker code based SAT as a meta-learning task. The acoustic model is considered as meta-knowledge, while speaker code is considered as task specific knowledge. Experiments on the Switchboard task show that our method can not only learn a good speaker code, but also improve the performance of the acoustic model even without speaker code.

Index Terms: automatic speech recognition, speaker adaptive training, model-agnostic meta-learning

1. Introduction

Recently, the accuracy of automatic speech recognition (ASR) has been greatly improved by the use of deep neural network (DNN) acoustic models such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) [1–3]. However, the performance is still unsatisfactory if the acoustic condition of the test data is mismatched to that of the training data, such as for speakers who have not been seen by the acoustic model. In response, speaker adaptive training (SAT) is one of the effective approaches to improve the performance of ASR on these conditions. SAT reduces the mismatch by removing speaker variance during training of the acoustic model, and it allows the acoustic model to focus solely on modelling phonetic variations.

In recent years, a lot of SAT approaches have been proposed. They can be divided into two groups: auxiliary feature methods and adversarial learning methods.

Auxiliary feature methods use auxiliary features to inform the acoustic model about speaker identity. In [4–7], speaker i-vectors or bottleneck vectors are obtained using a pretrained speaker recognition model. Then, acoustic features concatenated with the corresponding speaker vectors are fed to a DNN-based acoustic model. In [8–10], the authors use speaker codes which are learned together with the acoustic model to represent speaker identity information. In order to highlight the importance of the speaker embeddings in adaptation and provide speaker identity information more effectively, [11, 12] try to generate the speaker dependent parameters via a controller network that takes speaker embeddings as input.

Adversarial learning is also used to perform speaker adaptive training. Inspired by the methods used in domain adaptation [13–15], [16] optimizes the acoustic model and the speaker classification model jointly via adversarial learning. In [17], a reconstruction network is trained to predict the input speaker i-vector. The mean-squared error loss of the i-vector reconstruction and the cross-entropy loss of the acoustic model are jointly optimized through adversarial multitask learning.

Despite the progress, it is still a challenge of speaker adaptation to improve performance on test data as much as possible without overfitting, which is especially important in a rapid adaptation setting when we use only a small amount of adaptation data. Auxiliary feature methods need speaker identity information, but how to get the information is a question that needs to be considered. Features similar to i-vector [4–7] are extracted from other pre-trained models. They may not fit perfectly with the current acoustic models. The speaker code method [8–10] is a useful way to provide information about speaker identity. But speaker codes and the acoustic model are optimized together. This training strategy makes it difficult to decouple speaker identity information from the acoustic model. What we really want is a speaker code that is only related to the speaker identity, and an acoustic model that has nothing to do with the speaker identity.

The adversarial learning methods aim to map the input speech frames from different speakers into the invariant hidden features of the speaker, so as to use the representation of the normalized speaker factors for further classification tasks. They do not perform adaptive training on the test speakers. For example, when we already have a small number of labeled speaker utterances, how to use those labeled utterances to improve the model's performance on unlabeled utterances of the same speaker is something that the adversarial learning method cannot do.

For this purpose, we introduce a meta-learning method called MAML [18] to the speaker code based SAT framework. We consider automatic speech recognition on a specific speaker as a specific task. The acoustic model learns meta-knowledge across all speakers, and the speaker code learns task-specific knowledge which indicates the speaker identity. We evaluated the effectiveness of the proposed method on the Switchboard dataset. The experiments show that our method can not only learn a useful speaker code to improve the performance on target speaker, but also improve the performance of the acoustic model even without speaker code.

The remainder of this paper is organized as follows. In Section 2, we briefly review the speaker code method and the standard MAML algorithm as related works. In Section 3, we present two different methods to apply the MAML algorithm to speaker code based speaker adaptive training. In Section 4, we report and discuss our experimental results on the Switchboard

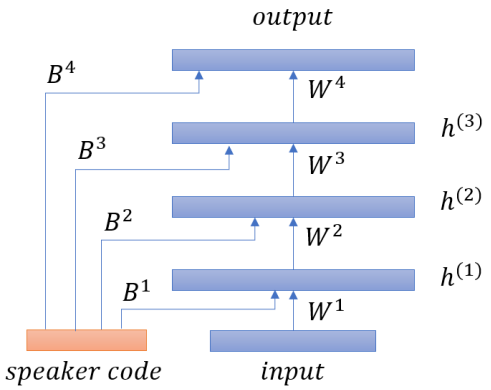


Figure 1: *Illustration of the model structure for speaker code based SAT.*

task. Finally, the paper is concluded in Section 5.

2. Related Work

2.1. Speaker code based SAT

Assuming that we have an (L+1)-layer DNN acoustic model consisting of weight matrices, denoted as $\mathbf{W}^l(1 \leq l \leq L+1)$, and the data come from C different speakers in total, we should have C different speaker codes, denoted as $\mathbf{s}^{(c)}(1 \leq c \leq C)$. Each speaker code is simply a vector, whose dimension can be freely adjusted. As shown in Fig. 1, these speaker codes are fed into some particular layers of DNN through another set of connection weights, denoted as $\mathbf{B}^l(l \in \mathcal{L})$, $\mathcal{L}$ stands for the number of layers connected with the speaker codes. For any layer $l(l \in \mathcal{L})$, it receives input features from both the lower layer $l-1$ and the speaker code, the output features in these layers are computed as follows:

$$\mathbf{h}^l = \mathbf{W}^l \mathbf{h}^{l-1} + \mathbf{B}^l \mathbf{s}^{(c)} \quad (\forall l \in \mathcal{L}) \tag{1}$$

In the learning process, both speaker codes and their connection weight matrices are all randomly initialized. The weights of the DNN can be initialized from a pretrained ASR model. After that, all of these parameters are jointly learned using the standard BP algorithm. In the testing stage, a new speaker code is learned based on a small amount of adaptation data for each speaker while the other parameters of the acoustic model are frozen. The learned speaker code is used for all the utterances of the corresponding speaker. Experiments on the TIMIT and Switchboard task have shown that the speaker code method is quite effective to adapt large DNN models using only a small amount of adaptation data.

2.2. Model-Agnostic Meta-Learning

Model-Agnostic Meta-Learning (MAML) [18] is a popular meta-learning framework. MAML learns initialization parameters θ_0 by meta training $\mathcal{M}_{train}$ such that the model can perform well on query set after a few steps of gradient descent. Support set $\mathcal{S}$ are used to calculate loss used for gradient computation. Suppose model f is initialized as f_{θ_0}, let $\theta_N = Adapt(\theta_0; L, \mathcal{S}, N)$ be the model parameters updated through N steps of gradient descent where the loss function is L computed on support set $\mathcal{S}$. The optimization problem is defined as Eq.(2), which minimizes the loss of f_{θ_N} on query set $\mathcal{Q}$:

$$\min_{\theta_0} L(\theta_N; \mathcal{Q}) = \min_{\theta_0} L(Adapt(\theta_0; L, \mathcal{S}, N); \mathcal{Q}) \tag{2}$$

In speech applications, MAML has been applied to ASR. For example, in [19], MAML is proved helpful for cross-language speech recognition. The results showed that MAML based approach significantly outperforms the state-of-the-art multitask pretraining approach on all target languages.

3. Proposed Method

For some parametric model f_θ, MAML aims to find a set of initial parameters θ_0 which can be used to fast adapt to any new task sampled from the same distribution. In other words, the parameters θ_0 could be regarded as meta-knowledge. For speaker code based SAT method, $\mathbf{W}^l$ as well as $\mathbf{B}^l$ are speaker independent, so they can alse be regarded as meta-knowledge across SAT procedure. This is our motivation of using MAML in speaker code based SAT. We propose two different methods to apply MAML.

3.1. SAT with zero-initialized speaker code (SAT-ZISC)

In the speaker code based SAT method, the speaker independent parameters are denoted as $\theta = (\mathbf{W}, \mathbf{B})$. The adaptation procedure, or "inner loop", is formulated as following:

$$\mathbf{s}_N^{(c)} = Adapt(\mathbf{s}_0^{(c)}, \theta; L, \mathcal{S}, N) \tag{3}$$

where $\mathbf{s}_0^{(c)}$ is the initial speaker code for speaker c, L is the loss function, $\mathcal{S}$ is the support set which contains utterances of speaker c. During the adaptation procedure, or "inner loop", we freeze speaker independent parameters θ, and update initial speaker code by gradient descent. For example, when using one step of gradient update, the process can be described as following:

$$\mathbf{s}^{(c)} \leftarrow \mathbf{s}^{(c)} - \alpha \nabla_{\mathbf{s}^{(c)}} L(\mathbf{s}^{(c)}, \theta) \tag{4}$$

Generally, N adaptation steps could be applied to get adapted speaker code $\mathbf{s}_N^{(c)}$.The step size α and the number of steps N are fixed as hyperparameter.

When we get adapted speaker code $\mathbf{s}_N^{(c)}$, we could compute the loss on query set $\mathcal{Q}$, which contains some different uttreances of the same speaker. And then we update the speaker independent parameters θ by gradient descent. The final optimization problem, what we refered as the "outer loop" of meta-learning, is defined as Eq.(5):

$$\min_{\theta} L(\mathbf{s}_N^{(c)}; \mathcal{Q}) = \min_{\theta} L(Adapt(\mathbf{s}_0^{(c)}, \theta; L, \mathcal{S}, N); \mathcal{Q}) \tag{5}$$

Fig. 2 shows the architecture of the proposed SAT-ZISC method. Since the acoustic model weights $\mathbf{W}$ are initialized from a pretrained ASR model and the connection weights $\mathbf{B}$ are initialized from scratch, we initialize all speaker codes $\mathbf{s}_0^{(c)}(1 \leq c \leq C)$ to zero vector, so they have no effect on the original acoustic model at the beginning, and make the training process more stable. At the end of the "inner loop " learning, the speaker code $\mathbf{s}_0^{(c)}$ will be saved and used as a initial speaker code for next inner-loop learning.

It is worth mentioning that only the speaker independent parameters θ are updated in the "outer loop" while the adapted speaker code keeps unchanged, which means we do not need to compute second derivatives as the original MAML does.

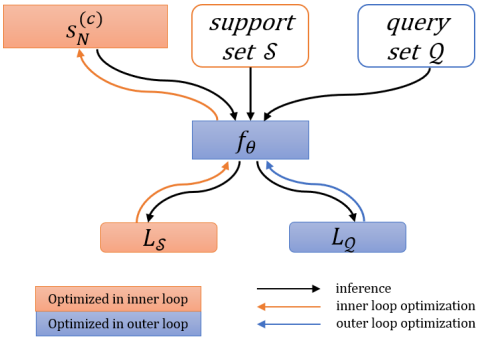


Figure 2: *Overview of the architecture of SAT-ZISC.*

In the testing stage, a new speaker code is first initialized to be zero, and then it is updated based on the derivatives of the adaptation data as Eq.(3). The learned speaker code will be fed to the model as in Eq.(1) for testing purpose.

3.2. SAT with meta-initialized speaker code (SAT-MISC)

Learning a good speaker code from a zero-initialized vector might be a hard work. In this section, we employ a new mothed called meta-initialized speaker code. Here not only the model parameters $\theta = (\mathbf{W}, \mathbf{B})$, but also the initial speaker code $\mathbf{s}_0$ are regarded as speaker independent parameters. As Fig. 3 shows, all speakers share a same initial speaker code $\mathbf{s}_0$. In the process of speaker adaptation, the initial speaker code is updated based on adaptation data of the speaker, to become a speaker dependent code $\mathbf{s}_N^{(c)}$:

$$\mathbf{s}_N^{(c)} = Adapt(\mathbf{s}_0, \theta; L, \mathcal{S}, N) \tag{6}$$

The final optimization problem, or "outer loop" is a little defferent from Eq.(5). It is formulated as following:

$$\min_{\theta, \mathbf{s}_0} L(\mathbf{s}_N^{(c)}; \mathcal{Q}) = \min_{\theta, \mathbf{s}_0} L(Adapt(\mathbf{s}_0, \theta; L, \mathcal{S}, N); \mathcal{Q}) \tag{7}$$

After training, we aim to get a speaker independent model and a initial speaker code $\mathbf{s}_0$ that is more suitable to speaker adaptation.

It is worth mentioning that during the training stage, we use second order derivatives to train the initial speaker code $\mathbf{s}_0$ according to Eq.(7). This could bring a significant computational expense. For computation efficiency, some previous works [18, 20] ignored the second-order term, which were alse known as First-order MAML(FOMAML). But we found that the training process was very difficult to converge when using FOMAML in our experiments. Finally we used second-order MAML in the SAT-MISC method, and its training speed was about 30% slower than the SAT-ZISC method.

4. Experiments

4.1. Dataset

All experiments were performed on the Switchboard(SWB) dataset. The training data of the SWB task [21] consists of 20-hour English CALLHOME and 309-hour Switchboard-I dataset, including a total of 5110 speakers. The SWB part of NIST 2000 Hub5 evaluation set is taken as test set, which contains 1831 utterances from 40 speakers in total. We use 20 utterances for each speaker to do speaker adaptive training. So the final test set contains 1031 utterances.

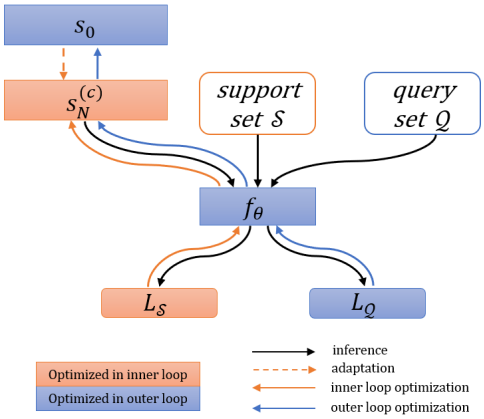


Figure 3: *Overview of the architecture of SAT-MISC.*

4.2. Baseline setup

The speaker independent baseline is trained with a VGG-like [22] model architecture based on frame-level cross-entropy criterion. The inputs of the model were the 40-dimensional log Mel-scale filter-bank features. The architecture of the model mainly consisted of convolutional and pooling layers, and each convolutional layer was equipped with a standard ReLU activation function. We shuffled the utterances in training data and grouped them into minibatches with a limit of 2048 frames per minibatch to speed up training. Stochastic gradient descent was used as the optimizer, and the initial learning rate was set to 0.02.

The speaker code baseline is trained based on the speaker independent baseline model. We connected the speaker code to the first layers of each convolutional block (layer conv0, conv1, conv5, conv9, conv13) in the VGG-like model. The connection method is as described in the Eq.(1). The DNN weights $\mathbf{W}$ were initialized from the speaker independent baseline and the connection weights $\mathbf{B}$ were randomly initialized. At the beginning, we initialized the speaker code $\mathbf{s}^{(c)}$ randomly, but found that the training process was very difficult to converge. So we initialized all speaker codes to zero vector. After that, all of the parameters were jointly optimized. The learning rate of $\mathbf{W}$ and $\mathbf{s}^{(c)}$ was set to 0.02, and the learning rate of $\mathbf{B}$ was set to 0.4. Tabel 1 reports the word error rate(WER) of the baseline models.

Table 1: *Performance of the baseline models on SWB.*

Method	**WER**	**WERR**
baseline	13.8	–
SC-baseline with adapted speaker code	13.5	2.2%

4.3. Results of the proposed method

In the training stage of our proposed methods, the hyperparameter step size α in MAML is set to 0.02. The rest of the hyperparameters are set the same as the SC-baseline model. Tabel 2 reports the performance of the our proposed methods. For SC-baseline model, firstly we fed the model with a zero initialized speaker code that has no effect on acoustic parameters. It achieves only a relative 0.7% WER reduction (WERR) compared with the baseline model. After training on 20 utterances of the target speaker to get an adapted spaeker code,

it achieves a relative 2.2% WER reduction compared with the baseline model.

As for the SAT-ZISC method, we alse fed the model with a zero initialized speaker code to test the performance of the speaker independent acoustic model. It achieves a relative 3.6% WER reduction compared with the baseline model, which is much better than SC-baseline. When using an adapted speaker code, it achieves a relative 4.3% WER reduction compared with the baseline model.

When receiving a zero initialized speaker code, the SAT-MISC method achieves a relative 3.6% WER reduction compared with the baseline model. In this method, the model has an initial speaker code, and it is a part of speaker independent parameters. When we fed the model with the initial speaker code, it achieves a relative 4.3% WER reduction compared with the baseline model. If we get an adapted spaeker code based on this initial speaker code, we will get a result of a relative 5.8% WER reduction compared with the baseline model.

Table 2: *Performance of the propoesd method on SWB.*

Method	**WER**	**WERR**
baseline	13.8	–
SC-baseline with adapted speaker code	13.5	2.2%
SC-baseline with zero speaker code	13.7	0.7%
SAT-ZISC with zero speaker code	13.3	3.6%
SAT-ZISC with adapted speaker code	13.2	4.3%
SAT-MISC with zero speaker code	13.3	3.6%
SAT-MISC with initial speaker code	13.2	4.3%
SAT-MISC with adapted speaker code	13.0	5.8%

The results show that both SAT-ZISC and SAT-MISC are able to get a better speaker independent acoustic model. This can be considered as the benefit of meta-learning. In the process of ASR, speaker independent acoustic model is a meta-knowledge across all speakers. Applying MAML to speaker adaptive training makes the model easier to extract these meta-knowledge. When using adapted speaker code, SAT-MISC is superior than SAT-ZISC. This result shows that learning from a good initialized speaker code is better than learning from zero.

We also investigated the impact of the number of adaptation steps N in proposed motheds. We find that SAT-ZISC need more steps to learn a useful speaker code than SAT-MISC. As Tabel 3 shows, SAT-ZISC need 12 adaptation steps to achieve best result while SAT-MISC need only 2 adaptation steps. This is a proof that learning from a good initialized speaker code is much easier than learning from zero. Tabel 3 also shows that more adaptation steps will not bring further improvement, and too many adaptation steps may degrade the performance of the model due to overfitting.

5. Conclusions

In this study, we have proposed two speaker code based speaker adaptive training methods with meta-learning approach. Both of the methods use the MAML algorithm. The speaker code is updated in the MAML's inner loop, and the speaker independent parameters are optimized in the MAML's outer loop. The results on the Switchboard task show that our methods not only learn a suitable speaker code, but alse significantly improve the performance of the acoustic model. Both of the acoustic models of our method achieve a relative 3.6% WER reduction (WERR) compared with the baseline model. After using adapted speaker code, the SAT-ZISC method achieves a relative 4.3% WER reduction compared with the baseline, and the SAT-MISC method achieves a relative 5.8% WER reduction compared with the baseline. In future work, we plan to use more corpora to evaluate the effectiveness of SAT-ZISC and SAT-MISC extensively. Besides, based on MAML's modelagnostic property, our approaches can be applied to a wide range of network architectures.

Table 3: *Impact of different adaptation steps on SWB.*

Method	**adaptation steps**	**WER**	**WERR**
baseline	–	13.8	–
SAT-ZISC	0	13.3	3.6%
	2	13.3	3.6%
	6	13.3	3.6%
	12	13.2	4.3%
	20	13.3	3.6%
	35	13.4	2.9%
SAT-MISC	0	13.2	4.3%
	2	13.0	5.8%
	6	13.2	4.3%

6. References

[1] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proceedings of Interspeech*, 2014, pp. 338–342.

[2] O. Abdel-Hamid, A. rahman Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Proceedings of ICASSP*, 2012, pp. 4277–4280.

[3] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. Lecun, "Very deep multilingual convolutional neural networks for lvcsr," in *Proceedings of ICASSP*, 2016, pp. 4955–4959.

[4] P. Karanasou, Y. Wang, M. J. Gales, and P. C. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *Proceedings of Interspeech*, 2014, pp. 2180–2184.

[5] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proceedings of ASRU*, 2013, pp. 55–59.

[6] Y. Miao, H. Zhang, and F. Metze, "Speaker adaptive training of deep neural network acoustic models using i-vectors," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 23, no. 11, pp. 1938–1949, 2015.

[7] P. Cardinal, N. Dehak, Y. Zhang, and J. Glass, "Speaker adaptation using the i-vector technique for bottleneck features," in *Proceedings of Interspeech*, 2015, pp. 2867–2871.

[8] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code," in *Proceedings of ICASSP*, 2013, pp. 7942–7946.

[9] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, "Direct adaptation of hybrid dnn/hmm model for fast speaker adaptation in lvcsr based on speaker code," in *Proceedings of ICASSP*, 2014, pp. 6389–6393.

[10] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 22, no. 12, pp. 1713–1725, 2014.

[11] X. Cui, V. Goel, and G. Saon, "Embedding-based speaker adaptive training of deep neural networks," in *Proceedings of Interspeech*, 2017, pp. 122–126.

[12] Y. Zhao, J. Li, S. Zhang, L. Chen, and Y. Gong, "Domain and speaker adaptation for cortana speech recognition," in *Proceedings of ICASSP*, 2018, pp. 5984–5988.

[13] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," *JMLR Workshop and Conference Proceedings*, vol. 37, pp. 1180–1189, 2015.

[14] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Proceedings of NIPS*, 2016, pp. 343–351.

[15] Z. Meng, J. Li, Y. Gong, and B. Juang, "Adversarial teacher-student learning for unsupervised domain adaptation," in *Proceedings of ICASSP*, 2018, pp. 5949–5953.

[16] Z. Meng, J. Li, Z. Chen, Y. Zhao, V. Mazalov, Y. Gong, and B. Juang, "Speaker-invariant training via adversarial learning," in *Proceedings of ICASSP*, 2018, pp. 5969–5973.

[17] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, and et al., "English conversational telephone speech recognition by humans and machines," in *Proceedings of Interspeech*, 2017.

[18] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1126–1135.

[19] J.-Y. Hsu, Y.-J. Chen, and H.-y. Lee, "Meta learning for end-to-end low-resource speech recognition," *arXiv preprint arXiv:1910.12094*, 2019.

[20] A. Nichol and J. Schulman, "Reptile: a scalable metalearning algorithm," *arXiv preprint arXiv:1803.02999*, vol. 2, p. 2, 2018.

[21] J. Godfrey, E. Holliman, and J. McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Proceedings of ICASSP*, 1992, pp. 517–520.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *CoRR*, 2014.