INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

The XMUSPEECH System for Short-Duration Speaker Verification Challenge 2020

Tao Jiang[1], *Miao Zhao*[1], *Lin Li*[2], *Qingyang Hong*[1]

[1]School of Informatics, Xiamen University, China
[2]School of Electronic Science and Engineering, Xiamen University, China
lilin@xmu.edu.cn,qyhong@xmu.edu.cn

Abstract

In this paper, we present our XMUSPEECH system for Task 1 in the Short-duration Speaker Verification (SdSV) Challenge. In this challenge, Task 1 is a Text-Dependent (TD) mode where speaker verification systems are required to automatically determine whether a test segment with specific phrase belongs to the target speaker. We leveraged the system pipeline from three aspects, including the data processing, front-end training and back-end processing. In addition, we have explored some training strategies such as spectrogram augmentation and transfer learning. The experimental results show that the attempts we had done are effective and our best single system, a transfered model with spectrogram augmentation and attentive statistic pooling, significantly outperforms the official baseline on both progress subset and evaluation subset. Finally, a fusion of seven subsystems are chosen as our primary system which yielded 0.0856 and 0.0862 in term of minDCF, for the progress subset and evaluation subset respectively.

Index Terms: speaker recognition, text-dependent, data augmentation, transfer learning

1. Introduction

This paper describes the XMUSPEECH team's submissions for Task 1 of the Short-duration Speaker Verification (SdSV) Challenge 2020 [1]. The proposed challenge evaluates SdSV in the degree of phonetic overlap between enrollment and test utterances (cross-lingual). This is the first challenge focusing on the Short-duration speaker recognition system benchmarking and varying degrees of phonetic variability analysis. This challenge was splited into two separate tracks: Task 1 and Task 2. Task 1 of the SdSV Challenge 2020 is the text-dependent (TD) speaker verification mode and Task 2 is the text-independent (TI) mode. Our team only participant in Task 1. In this task, the systems are required to automatically determine whether a test segment with a specific phrase was spoken by the target speaker when given a test segment of speech and the target speaker's enrollment data. That means if the phrase of test segment is not equal to the target speaker's phrase, it should also be regarded as an imposter. As a consequence, Task 1 is a 2-fold verification task in which both the speaker and phrase should be verified correctly.

Currently, there are two main approaches for TD speaker verification: i-vector [2] and deep embeddings. The deep embedding systems have attracted much attention as its simple training procedure and excellent performance [3, 4, 5]. Compared to i-vector framework, deep embedding methods show superior performances especially when given sufficient training data. Specially, the x-vectors are extracted from a time delay neural network (TDNN) [6] with a temporal statistics pooling layer. Then the x-vectors from the test and enrolled utterance were scored with cosine distance [7] or a probabilistic linear discriminant analysis (PLDA) backend [8]. Since the introduction of x-vectors into the field of speaker verification, many attempts have been made to improve the performances of the TD speaker verification systems. At data level, D.Snyder [5] has proved that x-vectors are data-driven and the model training with data augmentation can improve its performance under the severe environments. Besides, due to the lack of sufficient datasets for TD speaker verification, the strategy of transfer learning has also been investigated and confirmed its effectiveness in [9]. The main idea of transfer learning is to train a TI deep speaker embedding model and obtain its parameters as the TD model's initial parameters. In terms of network architecture, many new structures have been proposed to improve the performance of speaker verification, such as extended TDNN (ETDNN) [10] and factorized TDNN (FTDNN) [11]. Compared to standard TDNN architecture, ETDNN has a slightly wider temporal context and FTDNN factorizes the parameter matrices into smaller matrices, which make them achieve better performances. What's more, based on the assumption that frame-level features associated with different phonemes have different contribution to the speaker representation, the attention mechanism was introduced to TD speaker verification community [12, 13].

This paper is organized as follows: Section 2 describes the setup of the datasets for the challenge. Section 3 introduces the features and three types of DNN embedding-based speaker verification systems. The performance of our systems on both progress and evaluation subset is reported and analyzed in Section 4. Finally, Section 5 concludes the paper with final remarks and future work.

2. Datasets

2.1. Training Data

2.1.1. Task 1 In-Domain Data

DeepMine [14, 15] is a speech database in Persian and English designed to build and evaluate text-dependent, text-prompted, and text-independent speaker verification, as well as Persian speech recognition systems. For Task1, the in-domain training data is the Part 1 of DeepMine dataset, which contains more than 101,000 utterances from 963 speakers. In this dataset, each speaker utters 5 Persian phrases, and some would utter 5 more English phrases.

There are four types of trials in Task 1.

- Target-Correct (TC): The target speaker utters the correct pass-phrase
- Target-Wrong (TW): The target speaker utters the wrong pass-phrase

http://dx.doi.org/10.21437/Interspeech.2020-1704

- Imposter-Correct (IC): The imposter speaker utters the correct pass-phrase
- Imposter-Wrong (IW): The imposter speaker utters the wrong pass-phrase

Since Task 1 only consider the TC as target and the rest will be considered as imposter, we no longer use the form of a label given by a speaker. In other words, the same speaker is distinguished by phrase labels, so the number of in-domain data's *spkid* increased from 963 to 8886.

2.1.2. Opening Dataset

This challenge adopted a fixed training condition where the system should only be trained using a designated dataset. Under the fixed condition of the challenge, we also used the following datasets in our submissions except the Task 1 in-domain data.

- VoxCeleb-1 [16] : a dataset which contains more than 100,000 utterances for 1,251 celebrities, which are extracted from videos uploaded to YouTube.
- VoxCeleb-2 [17] : a dataset which contains more than 1,000,000 utterances for 6112 celebrities, which are extracted from videos uploaded to YouTube.

2.2. Development Dataset

In our systems, we extract 63 speakers from the Task 1 in-domain data to form the development set *(Dev)* since there is no separated development data provided for the challenge. Besides, the tags of the development set are treated like the Task 1 in-domain data mentioned above. As a result, the in-domain training set *(Train)* has only 900 speakers which we used to training models.

2.3. Augmentation Dataset

Data augmentation not only can increase the diversity and number of training data, but also can reduce the domain mismatch between the enrolled and test set data [5]. To obtain robust speaker embedding on SV systems, we applied two augmentation strategies.

2.3.1. Traditional Augmentation

We utilized Kaldi toolkit [18] to add noise to the data and we use the following augmentation datasets.

- Applied room impulse response (RIR) from the AIR dataset[1]
- Augmented with Musan [19][2] babble
- Augmented with Musan noise
- Augmented with Musan music

2.3.2. Spectrogram Augmentation

Inspired by the data augmentation from computer vision, D.S.Park et al proposed Spectrogram Augmentation (SpecAug) [20], an augmentation method that operates on the log spectrogram of the input audio, rather than the raw audio itself as shown in Figure 1. As this augmentation is applied directly to the feature inputs of a neural network without requiring any additional data, it is simple and computationally cheap to apply.

[1]http://www.openslr.org/resource/28
[2]http://www.openslr.org/resource/17

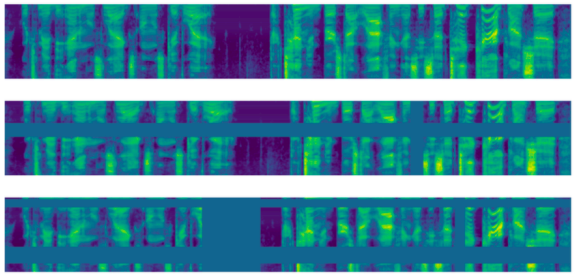

Figure 1: *SpecAugment: Multi-masking on acoustic feature*

3. Speaker Recognition System

Our goal is to design robust systems for the TD speaker verification. We will therefore present several systems including TDNN x-vector systems, ETDNN x-vector systems and Transfered x-vector systems. All of these systems are trained by using Pytorch toolkits [21] and the training chunk sizes are all set to 200.

3.1. Feature Extraction

Three features including Mel-frequency cepstral coefficient (MFCC), Filterbank features (FBank) and Perceptual linear predictive features (PLP) are adopted in our systems. The settings of feature extraction are:

- 23-dimensional Kaldi MFCC with Pitch - 16kHz, frequency limits 40-7800Hz, 25ms frame length, 3-dimensional pitch.
- 40-dimensional Kaldi FBank - 16kHz, frequency limits 40-7800Hz, 25ms frame length.
- 20-dimensional Kaldi PLP with Pitch - 16kHz, frequency limits 40-7800Hz, 25ms frame length, 3-dimensional pitch.

All the features were applied cepstral mean-normalization (CMN) with a sliding window of 3 seconds.

3.2. Speaker Model

3.2.1. TDNN x-vector

For the TDNN x-vector system, the 5-layer TDNN is trained to distinguish speakers of training dataset. The slicing parameters for the five TDNN layers are :$\{t-2, t-1, t, t+1, t+2\}$, $\{t-2, t, t+2\}$, $\{t-3, t, t+3\}$, $\{t\}$, $\{t\}$. The statistics pooling layer is used to calculate the mean and standard deviation on all frames of the input segment. The segment level representation is then inputted into two fully connected layers to classify the speakers in the training set. After training, the speaker vector is extracted from the 512-dimensional affine component of the first fully connected layer.

Here are some details about TDNN x-vetor training:

- Three features were used including MFCC, PLP and FBNAK.
- Except for the baseline, all other models use SpecAug during model training. Besides, the frequency mask parameter and time mask parameter are set to 0.2.
- A warm restart technique for stochastic gradient descent [22] was used to improve the training performance.

Table 1: *ETDNN X-vector architecture*

Layer	Layer Type	Context	Size
1	TDNN-ReLU	t-2:t+2	512
2	Dense-ReLU	t	512
3	TDNN-ReLU	t-2,t,t+2	512
4	Dense-ReLU	t	512
5	TDNN-ReLU	t-3,t,t+3	512
6	Dense-ReLU	t	512
7	TDNN-ReLU	t-4,t,t+4	512
8	Dense-ReLU	t	512
9	Dense-ReLU	t	512
10	Dense-ReLU	t	1500
11	Pooling(mean+stddev)	Full-seq	2x1500
12	Dense(Embedding)-ReLU		512
13	Dense-ReLU		512
14	Dense-Softmax		Num.spks

Warm restart technique reinitialize the learning rate in every cycle which can effectively help the model jump out of the local or saddle points and promote the model to converge to a flatter area. The parameters T_{Max} and T_{multi} are set to 3 and 2 respectively, which means there are three decay cycles and the length of each cycle are $3, 6$ and 12. In each cycle, the learning rate decays from 0.001 to $4e-8$.

- All experiments can be reproduced with ASV-Subtools [3] which is open sourced by XMUSPEECH.

3.2.2. ETDNN x-vector

Compared with the standard TDNN architecture, the ETDNN [10] has a slightly wider temporal context due to extra convolutional layers and use the interleaving of dense layers between the convolutional layers. The basic architecture of ETDNN is shown in Table 1, the first 10 layers are operated on speech frames and used as inputs to pooling layer. The x-vector is extracted from the output of an affine layer which follows the pooling layer. Similar to the previous TDNN model, the ETDNN model also uses three features MFCC, PLP and FBank. Besides, the setup for SpecAug and warm restart technique are the same with TDNN model.

3.2.3. Transfered x-vector

Lack of a dataset which contains sufficient speakers and phrase information has always been a problem in the field of TD speaker verification. As a consequence, the TD deep embedding model trained by a small TD dataset cannot learn the distinguishable speaker information very well, and its performance is poor and unstable, which may cause overfitting. However, the dataset without text annotation is easier to collect than the TD dataset, and now there are many TI datasets such as VoxCeleb 1&2 which meet the requirement of the fixed condition of the challenge. Therefore, we applied the transfer learning strategy on our experiments. Specifically, the fine-tuning steps are as follows:

1. The VoxCeleb $1+2$ was used to train the TI deep speaker model which based on TDNN architecture.
2. The parameters, obtained by the TI model, which not included the output layer's parameters, are used as TD model's initial parameters.
3. The in-domain dataset $Train$ was used to train a new model.

[3]https://github.com/Snowdar/asv-subtools

Table 2: *The results of the base x-vector in Dev set with different PLDA parameters*

PLDA Back-End	*Dev*	
x-vector_mfcc_baseline	EER%	minDCF
submean+norm	0.53	0.0614
submean+lda256+norm	0.58	0.0793
submean+whiten+norm	0.51	0.0621
submean+lda256+whiten+norm	0.58	0.0789

To get a robust transferred model, the TI model we used should achieve a good performance on its source domain, which yields 2.16% EER in test of VoxCeleb 1. We only used MFCC feature to train the TI model due to the time limit, so we didn't compare with other features in the part of transfered model. In addition, since transfered model using the parameters obtained from TI model as initial parameters, the new model converges quickly. Therefore, the parameters of warm restart, T_{max} and T_{multi} are set to 3 and 1, respectively, after our experimental comparisons.

To further makes the transfer learning match the task of target domain, we experiment with an attentive pooling mechanism in transfer architecture. The strategy we adopt is similar to that proposed in [13]. Specifically, the weight e_t for each frame are calculated as:

$$e_t = \nu^T f\left(Wh_t + b\right) + k \tag{1}$$

where $f(.)$ is ReLU function in our experiment. Then the weight score are normalized over all frames:

$$\alpha_t = \frac{exp\left(e_t\right)}{\sum_{\tau}^{T} exp\left(e_\tau\right)} \tag{2}$$

The normalized score α_t is then used as the weight in the pooling layer to calculate the weighted average vector:

$$\tilde{\mu} = \sum_{\tau}^{T} \alpha_t h_t \tag{3}$$

4. Experiments

4.1. Back-End Processing

We used the Kaldi toolkit to train the PLDA model and the dataset is corresponding to the *Train* set. The PLDA is used to calculate the scores between enrolled and test x-vectors. The PLDA scorings between the target and test embeddings were computed using the batch likelihood ratio.

To get a better PLDA model, we used the *Train* to train the PLDA with different strategies such as submean, linear discriminant analysis (LDA) which reduced the dimension to 256, whiten and so on. As we can see from the Table 2, the x-vectors obtained from our baseline model are used to compare these strategies. We found that the PLDA model training with whitening, submean and normalization achieved a better result, while the usage of LDA degrade the performance. As a result, all of our submitted systems used submean, whitening and normalization for PLDA training.

Table 3: *The EER% and minDCF results of our single systems on the Task 1 of SdSV Challenge 2020 on the Progress subset and Evalution subset*

#	Feature	Systems	Progress subset		Evaluation subset	
			minDCF	EER%	minDCF	EER%
0	MFCC	official i-vector/HMM baseline [23]	0.1472	3.47	0.1464	3.49
1	MFCC	TDNN-Xvector	0.1210	3.05	0.1219	3.04
2	MFCC	TDNN-Xvector-SpecAug	0.1149	2.90	0.1156	2.92
3	MFCC	ETDNN-Xvector-SpecAug	0.1091	2.61	0.1089	2.64
4	PLP	TDNN-Xvector-SpecAug	0.1171	2.98	0.1176	3.02
5	PLP	ETDNN-Xvector-SpecAug	0.1078	2.66	0.1082	2.68
6	FBank	TDNN-Xvector-SpecAug	0.1066	2.71	0.1077	2.72
7	FBank	ETDNN-Xvector-SpecAug	0.1056	2.66	0.1059	2.67
8	MFCC	Transfer-Xvector-SpecAug	0.1067	2.63	0.1069	2.63
9	MFCC	Transfer-Xvector-SpecAug-Attentive	**0.1016**	**2.48**	**0.1024**	**2.51**

Table 4: *Performance of the average fusion system on the Progress subset and Evaluation subset of Task 1 in SdSV Challenge 2020*

System	Progress subset		Evaluation subset	
	minDCF	EER%	minDCF	EER%
fusion 2+4+6	0.0979	2.53	0.0979	2.55
fusion 3+5+7	0.0923	2.37	0.0935	2.39
fusion 2+3+4+9	0.0904	2.34	0.0907	2.36
fusion 3+5+7+9	0.0863	2.22	0.0873	2.25
fusion 2+3+4+5+6+7+9	**0.0856**	**2.22**	**0.0862**	**2.24**

4.2. Result and Analysis

We train our single systems with different features and different architectures. For this challenge, the whole trials are divided into two subsets: a progress subset (30% trials) and an evaluation subset (70% trails). Thus we present our experiment results on the progress subset and evaluation subset in both EER% and minimum Detection Cost Function (minDCF) in Table 3.

4.2.1. Single Systems

We start with TDNN-Xvector on *Train* which outperforms official baseline about 17% in minDCF for both progress subset and evaluation subset. Based on this, we verified the effectiveness of SpecAug. As we can see from Table 3, the results of system 1 and system 2 confirm that applying SpecAug to network training can further improve the performance, which yields about 5% relative improvement on both subsets. As a consequence, our subsequent systems all use this strategy during training.

Comparing the results of system 2, 4, 6 and system 3, 5, 7, respectively, it can be clearly seen that ETDNN-Xvector performs better than TDNN-Xvector under the same training strategy. Besides, with the same architecture, we can also see the FBank feature performs better than MFCC and PLP.

Our best single system is the Transfer-Xvector-SpecAug-Attentive, which achieves 0.1016 minDCF and 2.48% EER on progress subset, and 0.1024 minDCF and 2.51% EER on evaluation subset. In Table 3, we also can see that using attentive pooling rather than average pooling is effective in transfered model which yields 4.75% and 4.21% relative improvement on progress subset and evaluation subset, respectively.

4.2.2. Fusion Systems

The fusion strategy is the average weight and the performances of the fused systems on the progress subset and evaluation set are shown in Table 4. Compared to the best single system Transfer-Xvector-SpecAug-Attentive, all score fusion systems perform better, which show that the fusion system is more robust and stable. Finally, our best fusion system consists of seven subsystems, which achieved 0.0856 and 0.0862 in minDCF for two subsets, respectively, and yields 15.7% and 15.8% relative improvements in term of minDCF for progress subset and evaluation subset over the best single system.

5. Conclusion

This paper presents the system submitted by XMUSPPECH in the TD task of SdSV challenge 2020. To obtain robust systems for the task, we have made various attempts in data processing, x-vector extractors selecting and model training. The experimental results showed that our systems tasking such process could significantly improve the performance compared to the official baseline. In addition, we found that SpecAug could be successfully implemented during network training, and we confirmed that the strategy of transfer learning which adopted a TI deep speaker embedding model to a TD model was effective. Finally, the fusion system demonstrated the best performance on both the progress subset and evaluation subset.

6. Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No.61876160).

7. References

[1] H. Zeinali, K. A. Lee, J. Alam, and L. Burget, "Short-duration speaker verification (SdSV) challenge 2020: the challenge evaluation plan," *arXiv preprint arXiv:1912.06311*, 2019.

[2] 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.

[3] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 165–170.

[4] 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.

[5] 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.

[6] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[7] N. Dehak, R. Dehak, J. R. Glass, D. A. Reynolds, P. Kenny *et al.*, "Cosine similarity scoring without score normalization techniques." in *Odyssey*, 2010, p. 15.

[8] P. Kenny, T. Stafylakis, P. Ouellet, M. J. Alam, and P. Dumouchel, "PLDA for speaker verification with utterances of arbitrary duration," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7649–7653.

[9] X. Qin, D. Cai, and M. Li, "Far-field end-to-end text-dependent speaker verification based on mixed training data with transfer learning and enrollment data augmentation," *Proc. Interspeech 2019*, pp. 4045–4049, 2019.

[10] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5796–5800.

[11] D. Povey, G. Cheng, Y. Wang, K. Li, and S. Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Interspeech 2018*, 2018.

[12] F. R. rahman Chowdhury, Q. Wang, I. L. Moreno, and L. Wan, "Attention-based models for text-dependent speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5359–5363.

[13] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *INTERSPEECH*, 2018.

[14] H. Zeinali, H. Sameti, and T. Stafylakis, "DeepMine speech processing database: Text-dependent and independent speaker verification and speech recognition in Persian and English." in *Odyssey*, 2018, pp. 386–392.

[15] H. Zeinali, L. Burget, J. Černockỳ *et al.*, "A multi purpose and large scale speech corpus in Persian and English for speaker and speech recognition: the DeepMine database," *arXiv preprint arXiv:1912.03627*, 2019.

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

[17] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.

[18] 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.

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

[20] 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.

[21] 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.

[22] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[23] H. Zeinali, H. Sameti, and L. Burget, "HMM-based phrase-independent i-vector extractor for text-dependent speaker verification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. PP, pp. 1–1, 04 2017.