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

Improving X-vector and PLDA for Text-dependent Speaker Verification

Zhuxin Chen, Yue Lin

NetEase Games AI Lab
{chenzhuxin,gzlinyue}@corp.netease.com

Abstract

Recently, the pipeline consisting of an x-vector speaker embedding front-end and a Probabilistic Linear Discriminant Analysis (PLDA) back-end has achieved state-of-the-art results in text-independent speaker verification. In this paper, we further improve the performance of x-vector and PLDA based system for text-dependent speaker verification by exploring the choice of layer to produce embedding and modifying the back-end training strategies. In particular, we probe that x-vector based embeddings, specifically the standard deviation statistics in the pooling layer, contain the information related to both speaker characteristics and spoken content. Accordingly, we modify the back-end training labels by utilizing both of the speaker-id and phrase-id. A correlation-alignment-based PLDA adaptation is also adopted to make use of the text-independent labeled data during back-end training. Experimental results on the SDSVC 2020 dataset show that our proposed methods achieve significant performance improvement compared with the x-vector and HMM based i-vector baselines.

Index Terms: speech verification, x-vector, PLDA, SDSVC 2020, short utterance

1. Introduction

Speaker verification (SV) is to determine whether two speech recordings are spoken by the same speaker or not. There are two categories of speaker verification systems: text-dependent speaker verification (TD-SV) and text-independent speaker verification (TI-SV). Compared with TD-SV, TI-SV is a process of verifying the identity without constraint on the speech content.

Fostered primarily by the NIST Speaker Recognition Evaluation challenge, most of this research have focused on the TI-SV task. There are two main approaches, namely i-vector [1] and x-vector [2, 3], used to produce speaker embedding. Both of them represent speech utterance by a low-dimensional fixed-length vector. Prior studies have found that x-vector leverage large-scale training datasets better than i-vector[2, 3]. In addition, the back-end similarity measurement plays an important role. For the x-vector based system with softmax loss, the PLDA back-end tends to outperform the cosine similarity back-end [4] since the softmax loss is not discriminative enough to optimize the embedding similarity [5, 6]. Recently, researchers also revealed that x-vectors contain the lexical content information in softmax-trained model[7]. Thus, the backend is crucial to deal with the phoneme-invariant problem in TI-SV.

Nowadays, due to the popularity of intelligent personal assistants, such as Siri, Alexa and Cortana, researchers have begun to pay more attention on TD-SV in which both the speaker and phrase are verified. Although it can be divided into two sub-tasks, a TI-SV system to identity the speaker and an automatic speech recognition (ASR) system to verify the phrase, it is laborious and complicated to build an ASR system, especially when the source language is low-resource.

In this paper, we mainly investigate one-pass approaches for x-vector and PLDA based TD-SV where speakers pronounce fixed pass-phrases to authenticate. The system produces one log-likelihood ratio (LLR) score that used to verify both of the speaker and phrase, without building an ASR system. We evaluated our systems on the text-dependent task of Short-duration Speaker Verification Challenge (SDSVC) 2020 [8]. The main contributions of this paper are summarized as follows:

- We analyzed the structure of neural network and investigated the optimal layer used to produce speaker embeddings. We found that the standard deviation part in the statistics pooling layer contains more content information than the first dense layer in segment-level.
- Inspired by the fact that x-vector based system is sensitive to the back-end model and the x-vector based embeddings also contain information about content, we simply modified the back-end training labels from speaker labels into mixture labels of speaker and phrase, and achieved promising result.
- Correlation alignment (CORAL) [9] is a method that aligns the distributions of out-of-domain and in-domain features in an unsupervised way. A CORAL based PLDA adaptation is employed to make use of the text-independent labeled data during back-end training, further improving the performance.

The rest of this paper is organized as follows. In Section 2, we review the related works on SV. The methodology are introduced in Section 3. Experimental setup and results will be given in Section 4. Finally, we draw a conclusion in Section 5.

2. Related Works

2.1. Research about text-dependent speaker verification

As far as we know, HMM based i-vector method achieves state-of-the-art performance in TD-SV task [10, 11]. In this method, Viterbi algorithm is adopted to obtain the phoneme information. In contrast to the conventional GMM based i-vector, which uses GMM for aligning the frames to Gaussian components, monophone HMMs are used for alignment and then computes the statistics.

In addition, deep speaker embeddings based methods are also proposed in TD-SV task. However, most of these systems are customized for specials keyword , such as ”OK Google” [12] and ”ni hao, mi ya” [13]. It does not take the Target-Wrong (TW) trial into account where the target speaker utters a wrong pass-phrase on the evaluation dataset.

2.2. The basic architecture of x-vector

Figure 1 depicts the basic architecture of DNN in the x-vector based system. In this system, the network consists of layers that extract speech characteristics at the frame-level, a statistics pooling layer that aggregates variable-length features to a

http://dx.doi.org/10.21437/Interspeech.2020-1188

fixed-dimensional vector, additional layers that operate at the segment-level, and finally a softmax output layer.

It has been shown that shallow TDNN is under-fitting when large-scale training data is available [14]. Various deeper neural network structures, such as extended TDNN (ETDNN)[15] and factorized TDNN (FTDNN) [16], are proposed to address this problem. There are some common designs shared across these models. First of all, most of the architectures use the statistics pooling layer. This layer, namely layer L_p in Figure 1, receives the output of the final frame-level layer as input, computes its mean and standard deviation, and then concatenates them together. Secondly, the dimension of the final frame-level layer is commonly higher compared with other TDNN and dense layers, except the presoftmax layer. Last but not least, a low-dimension layer L_x, as shown in Figure 1, is used to extract the speaker embeddings in most of TI-SV systems[2, 3].

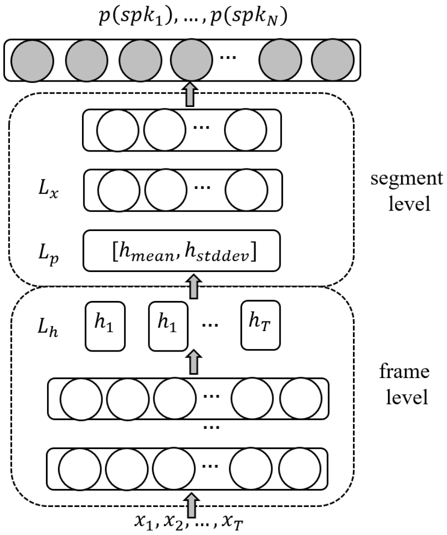

Figure 1: *The basic architecture of DNN in the x-vector system.*

3. Methodology

3.1. Overview

The proposed x-vector and PLDA based system for TD-SV have a similar calculation pipeline used in [2] for TI-SV. It can be divided into two parts, a DNN to produce embeddings that represent the information of speaker and phrase, followed by a back-end model to compare pairs of embeddings. We make several changes to the entire framework, constituting the main contributions of our work. In particular, we first investigate the optimal layer used to produce embeddings. We then modify the back-end training strategy accordingly. In addition, a correlation-alignment-based PLDA adaptation is employed to make use of the text-independent labeled data during back-end training, further improving the performance.

3.2. Acoustic features

The acoustic features are 40 dimensional filterbank (Fbank) coefficients with a frame-length of 25ms that are mean-normalized over a sliding window of up to 3 seconds. A frame-level energy-based VAD is employed to filter out non-speech frames.

3.3. Neural network architecture

To evaluate the effectiveness of our proposed strategies in this paper, two x-vector based neural network architectures, namely ETDNN and SpecAug-FTDNN-LSTM, were used in our experiments.

ETDNN: Compared to standard TDNN, the ETDNN architecture has a wider temporal context with interleaving dense and convolution layers in the frame-level layers. We used it as the x-vector baseline. The structure of ETDNN system is depicted in Table 1.

SpecAug-FTDNN-LSTM: This system consists of the architecture of SpecAugment, FTDNN and LSTM, as depicted in Table 2. SpecAugment is a newly proposed on-the-fly data augmentation and it showed very promising results in speech recognition [17]. FTDNN factorizes the weight matrix of TDNN layer into two low-rank matrices and the first matrice is constrained to be semi-orthogonal. Skip connections between the low-rank interior layers are created to reduce the risk of gradient vanishing. LSTM layers, with a delay of 3 frames, are inserted after the FTDNN layers to capture the long-range dependencies in speech. Different from ETDNN system, the dimension of the final frame-level layer is up to 2048.

We used Kaldi [18] to train these systems, with a mini-batch size of 128, an initial learning rate of 0.001 and a final learning rate of 0.0001 for 6 epochs.

Table 1: *The structure of ETDNN system.*

Layer	Layer Type	Context	Size
1	TDNN-ReLU-BN	t-2:t+2	512
2	Dense-ReLU-BN	t	512
3	TDNN-ReLU-BN	t-2,t,t+2	512
4	Dense-ReLU-BN	t	512
5	TDNN-ReLU-BN	t-3,t,t+3	512
6	Dense-ReLU-BN	t	512
7	TDNN-ReLU-BN	t-4,t,t+4	512
8	Dense-ReLU-BN	t	512
9	Dense-ReLU-BN	t	1536
10	Pooling(mean+stddev)	Full-seq	2*1536
11	Dense-ReLU-BN		512
12	Dense-ReLU-BN		512
13	Dense-Softmax		Num.spks.

Table 2: *The structure of SpecAug-FTDNN-LSTM system.*

Layer	Layer Type	Context Factor 1	Context Factor 2	Skip Conn. from Layer	Size	Inner Size
1	BN-SpecAug	t			40	
2	TDNN-ReLU-BN	t-2:t+2			512	
3	FTDNN-ReLU-BN	t-2,t	t,t+2		1024	256
4	FTDNN-ReLU-BN	t	t		1024	256
5	FTDNN-ReLU-BN	t-3,t	t,t+3		1024	256
6	FTDNN-ReLU-BN	t	t	3	1024	256
7	FTDNN-ReLU-BN	t-3,t	t,t+3		1024	256
8	FTDNN-ReLU-BN	t	t	2,4	1024	256
9	FTDNN-ReLU-BN	t-3,t	t,t+3		1024	256
10	FTDNN-ReLU-BN	t	t	4,6,8	1024	256
11	LSTM	t			1024	
12	LSTM	t			1024	
13	Dense-ReLU-BN	t			2048	
14	Pooling(mean+stddev)	Full-seq			2*2048	
15	Dense-ReLU-BN				512	
16	Dense-ReLU-BN				512	
17	Dense-Softmax				Num.spks.	

3.4. Embeddings representation

Although phoneme information in the embedding is useless and even degrades the performance of TI-SV system, it plays an important role in TD-SV system. As Figure 1 shows, based on a large amount of data, this network is trained to distinguish between speakers in the training set. The TDNN layers in frame-level focus on local feature extraction. The statistics pooling layer aggregates the high-level feature to a fixed-dimensional vector. The segment-level layers is used to capture speaker characteristics and eliminating irrelevant information. Therefore, the closer to the output layer, the less phoneme information.

Since it is also important to capture speaker characteristics over the entire utterance, we do not consider the frame-level layer to extract embeddings. We focus on the statistics pooling layer and the first dense layer in segment-level. There are four kinds of embeddings used in our experiment.

emb-xvector: Embeddings are extracted at layer L_x, before the nonlinearity.

emb-pool: Embeddings are extracted at statistics pooling layer, consist of the mean and standard deviation statistics of the output of layer L_h .

emb-mean: Embeddings are extracted using h_{mean} , the mean statistics of the output of layer L_h .

emb-stddev: Embeddings are extracted using $h_{stdddev}$, the standard deviation statistics of the output of layer L_h .

We evaluate emb-mean and emb-stddev respectively for two reasons. On the one hand, the mean statistics are difficult to make use of high-order information compared with standard deviation. On the other hand, the back-end models, including LDA and PLDA, generally assume that the distributions of embeddings are homogeneous Gaussian in the latent space. When we concatenate the statistics, it may fall into the problem of non-homogeneity and non-Gaussianality [19].

3.5. Back-end training strategy

We first train the back-end models using text-dependent dataset. After extracting the embedding as mentioned above, a simple yet effective operation is to modify the training labels of LDA and PLDA models. While the TI-SV system only uses the speaker tag as labels, our TD-SV system generates the news labels by combining both of the speaker tag and phrase tag, namely different phrases from the same speaker are considered as different labels.

In our experiment, we use a subset of Deepmine training set, totally about 863 speaker with a fixed set of ten different phrases, to train the back-end models. There are 8630 classes of the back-end training data after modification. Similar to [2], the embeddings are centered, dimensionality reduced using LDA and length normalized. We adjust the output dimension of LDA based on the results of the development set.

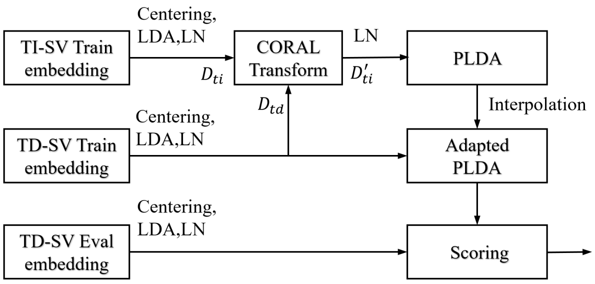

Figure 2: *Flow Diagram of PLDA Adaptation*

3.6. PLDA Adaptation

It is well known that the performance of SV system benefits from large-scale in-domain data. However, it would be prohibitively expensive to collect large amount of in-domain data for every application, especially for TD-SV task. Nowadays, large amount of text-independent data, such as VoxCeleb, are public and available. In this section, we propose a PLDA adaptation method that use both of TI-SV and TD-SV labeled data.

As shown in Figure 2, the vital components of our back-end model include CORAL transform and PLDA interpolation. During training, the embeddings extracted from the TI-SV and TD-SV training set are centered respectively and two LDA models are trained. While the model with TI-SV data only makes use of the speaker label, the other one with TD-SV data utilizes both of the speaker and phrase label using the strategy described in section 3.5. The CORAL is used to minimize the covariance distance between the TI-SV and TD-SV data by whitening and re-coloring. We use the CORAL transformed vectors to train the PLDA. Finally, the adaptive PLDA is achieved by linearly interpolating the TI-SV covariance with the TD-SV covariance. For the CORAL transformation, it can be described as following equations:

$$\mathrm{C}_{ti} = \mathrm{cov}(D_{ti}) + eye(size(D_{ti}, 2)) \tag{1}$$

$$\mathrm{C}_{td} = \mathrm{cov}(D_{td}) + eye(size(D_{td}, 2)) \tag{2}$$

$$D'_{ti} = D_{ti} * C_{ti}^{-1/2} * C_{td}^{1/2} \tag{3}$$

Where D_{ti} and D_{td} are the length normalized (LN) vectors after LDA projection. D'_{ti} is the CORAL transformed vector of TI-SV dataset.

During test, the embeddings are centered and dimensionality reduced using the model trained from TD-SV data.

4. Experiment

4.1. Training data

Our experiments are based on the text-dependent task of SDSVC 2020. The training data consists of Voxceleb1 [20], Voxceleb2 [21] and the training partition of DeepMine dataset [22].

To assess the performance of our proposed back-end training strategy, we only used Voxceleb1 and Voxceleb2 (totally 1,108,467 utterances from 7,363 speakers) to train the front-end extractors, since the DeepMine training set are considered as in-domain data. The data augmentation techniques described in [3] are applied.

To train the back-end models, we used Voxceleb1, Voxceleb2 and DeepMine training set. The Voxceleb datasets are considered as TI-SV labeled dataset and used for the adaptation of PLDA. DeepMine training set is a TD-SV labeled data that contains 963 speakers with a fixed set of ten Persian phrases, which is similar to Part1 of the RedDots [23]. We splited the DeepMine training set into two parts, with 863 speakers (89,801 utterances) as training set and 100 speakers (11,262 utterances) as development set for tuning parameters.

4.2. Evaluation

System performance is assessed on the evaluation of SDSVC 2020. The evaluation set is also drawn from DeepMine dataset with a fixed set of ten phrases, but it consists of five Persian and five English phrases. For each trial, model enrollment is done in a phrase and language-dependent way using three utterances.

The performance is reported in terms of equal error rate (EER) and the normalized detection cost function (MinDCF) as defined in SRE08 with $C_{miss} = 10$, $C_{FalseAlarm} = 1$ and $P_{Target} = 0.01$.

In addition, the score distribution is shown with Boxplot. Four kinds of trials, namely Imposter-Wrong(IW), Imposter-Correct(IC), Target-Wrong(TW), and Target-Correct(TC) are reported respectively [11].

Table 3: *The results (EER (%) / MinDCF) of ETDNN and SpecAug-FTDNN-LSTM with cosine similarity measurement on the evaluation set.*

Embedding	ETDNN	SpecAug-FTDNN-LSTM
emb-xvector	15.86 / 0.8490	14.84 / 0.8245
emb-mean	13.15 / 0.5772	14.01 / 0.6010
emb-pool	10.71 / 0.5124	11.06 / 0.5101
emb-stddev	10.05 / 0.5033	**9.84 / 0.4738**

Table 4: *The results (EER (%) / MinDCF) of ETDNN with PLDA back-end on the evaluation set.*

Embedding	original	modified	adaption
emb-xvector	11.02 / 0.6507	2.73 / 0.1405	2.60 / 0.1318
emb-mean	–	2.58 / 0.1124	2.49 / 0.1085
emb-pool	–	2.48 / 0.1032	2.41 / 0.0982
emb-stddev	–	2.15 / 0.0884	**2.11 / 0.0856**

Table 5: *The results (EER (%) / MinDCF) of SpecAug-FTDNN-LSTM with PLDA back-end on the evaluation set.*

Embedding	original	modified	adaption
emb-xvector	11.27 / 0.6682	2.53 / 0.1333	2.43 / 0.1258
emb-mean	–	2.37 / 0.0956	2.28 / 0.0908
emb-pool	–	2.39 / 0.0953	2.28 / 0.0884
emb-stddev	–	1.98 / 0.0743	**1.93 / 0.0714**

Table 6: *The SDSVC baselines and the results of SpecAug-FTDNN-LSTM system on the evaluation set.*

System	EER(%)	MinDCF
SDSVC x-vector baseline	9.05	0.5290
SDSVC i-vector/HMM baseline	3.49	0.1464
SpecAug-FTDNN-LSTM(our)	**1.72**	**0.0625**

4.3. Experimental setup and results

We first compared the embedded layers used to produce embeddings using cosine similarity measurement. The embeddings were centered and normalized, and then the inner dot products were computed. As shown in Table 3, it achieved the best results for both MinDCF and EER by using the standard deviation statistics as embeddings for both ETDNN and SpecAug-FTDNN-LSTM systems. The mean statistics with less discrimination even degraded the performance of statistics pooling layer based systems. Figure 3 depicts the score distributions with cosine similarity measurement in ETDNN system. It is shown that Imposter-Correct trials with consistent pass-phrases tended to achieve higher scores compared with Imposter-Wrong trials, especially using the standard deviation statistics as embeddings. Therefore, the standard deviation statistics contain more content information than the first dense layer in segment-level.

We then evaluated the proposed approach that modifying the back-end training labels and the strategy of PLDA adaption. The dimension of LDA projection was set to 512 for all the systems in Table 4 and Table 5. As shown in the first line of Table 4, three systems used the same DNN extractor namely ETDNN and the embeddings were extracted using emb-xvector layer. With different back-end training labels, the modified pipeline significantly outperforms the original one. Figure 4 depicts the score distributions with PLDA back-end in ETDNN system. Our proposed approach obviously increased the gap between the Target-Correct and Target-Wrong trials, making them to be better distinguished. Table 4 and Table 5 also summarize the performance of PLDA adaption as described in section 3.6. As can be seen, the PLDA adaptation achieved better performance in four kinds of embeddings in both ETDNN and SpecAug-FTDNN-LSTM systems.

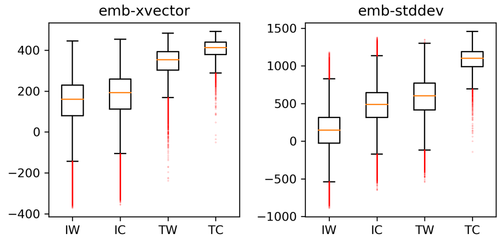

Figure 3: *Boxplot of score distributions with cosine similarity measurement in the development set using ETDNN system.*

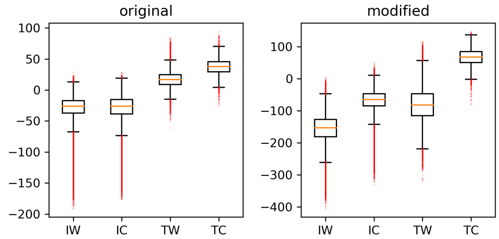

Figure 4: *Boxplot of score distributions with PLDA back-end in the development set using ETDNN system.*

Finally, we compared our SpecAug-FTDNN-LSTM system with the SDSVC baselines [8] on the evaluation set. As shown in Table 6, the performance of SpecAug-FTDNN-LSTM system is further improved by increasing the dimension of LDA projection to 1500. Overall, our best single system significantly outperforms both x-vector and HMM based i-vector baselines, reducing the EER about 81% and 51% respectively, and the MinDCF about 88% and 57% respectively.

5. Conclusion

In this paper, we investigated the pipeline based on x-vector and PLDA for short-duration text-dependent speaker verification. Our findings are:

- The statistics pooling layer, especially the standard deviation statistics, contains more content information than the first dense layer in segment-level.
- Back-end models are important for text-dependent speaker verification system. It is useful to extend the labels from speaker-id to a mixture of speaker-id and phrase-id, namely different phrases from the same speaker are considered as different labels.
- Correlation alignment based adaptation enable the text-dependent speaker verification system to make better use of text-independent labeled data.
- The result of SpecAug-FTDNN-LSTM system outperforms the ETDNN and HMM based i-vector systems significantly.

For reasons of space, we only reported the single system that used the out-of-domain dataset to train the DNN extractors. Our submission to the challenge showed that the performance could be further improved by using deeper neural network and combining the DeepMine dataset during training[24].

6. 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] 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.

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *ICASSP*, 2018, pp. 5329–5333.

[4] Y. Zhu and B. Mak, "Orthogonal training for text-independent speaker verification," in *ICASSP*, 2020, pp. 6584–6588.

[5] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *CVPR*, 2017, pp. 212–220.

[6] J. S. Chung, J. Huh, S. Mun, M. Lee, H. S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, and I. Han, "In defence of metric learning for speaker recognition," in *Interspeech*, 2020.

[7] D. Raj, D. Snyder, D. Povey, and S. Khudanpur, "Probing the information encoded in x-vectors," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 726–733.

[8] 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.

[9] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[10] H. Zeinali, L. Burget, H. Sameti, O. Glembek, and O. Plchot, "Deep neural networks and hidden markov models in i-vector-based text-dependent speaker verification." in *Odyssey*, 2016, pp. 24–30.

[11] 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. 25, no. 7, pp. 1421–1435, 2017.

[12] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *ICASSP*, 2014, pp. 4052–4056.

[13] 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," in *Interspeech*, 2019, pp. 4045–4049.

[14] J. Villalba, N. Chen, D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, J. Borgstrom, F. Richardson, S. Shon, F. Grondin *et al.*, "State-of-the-art speaker recognition for telephone and video speech: the jhu-mit submission for nist sre18," in *Interspeech*, 2019, pp. 1488–1492.

[15] 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, pp. 5796–5800.

[16] D. Snyder, J. Villalba, N. Chen, D. Povey, G. Sell, N. Dehak, and S. Khudanpur, "The jhu speaker recognition system for the voices 2019 challenge," in *Interspeech*, 2019, pp. 2468–2472.

[17] 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.

[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] Y. Cai, L. Li, D. Wang, and A. Abel, "Deep normalization for speaker vectors," *arXiv preprint arXiv:2004.04095*, 2020.

[20] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Interspeech*, 2017, pp. 2616–2620.

[21] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Interspeech*, 2018, pp. 1086–1090.

[22] H. Zeinali, L. Burget, and J. Cernocky, "A multi purpose and large scale speech corpus in Persian and English for speaker and speech recognition: the DeepMine database," in *Proc. ASRU 2019 The 2019 IEEE Automatic Speech Recognition and Understanding Workshop*, 2019.

[23] K. A. Lee, A. Larcher, G. Wang, P. Kenny, N. Brümmer, D. v. Leeuwen, H. Aronowitz, M. Kockmann, C. Vaquero, B. Ma *et al.*, "The reddots data collection for speaker recognition," in *Interspeech*, 2015, pp. 2996–3000.

[24] Z. Chen, D. Chen, H. Ding, and Y. Lin, "The netease games system description for text-dependent sub-challenge of sdsvc 2020." [Online]. Available: https://sdsvc.github.io/descriptions/Team14_Task1.pdf