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

NEC-TT Speaker Verification System for SRE'19 CTS Challenge

Kong Aik Lee[1], *Koji Okabe*[1], *Hitoshi Yamamoto*[1], *Qiongqiong Wang*[1], *Ling Guo*[1], *Takafumi Koshinaka*[1], *Jiacen Zhang*[2], *Keisuke Ishikawa*[2], *Koichi Shinoda*[2]

[1]NEC Corporation, Japan
[2]Tokyo Institute of Technology, Japan
{kongaik.lee, k-okabe}@nec.com

Abstract

The series of *speaker recognition evaluations* (SREs) organized by the National Institute of Standards and Technology (NIST) is widely accepted as the de facto benchmark for speaker recognition technology. This paper describes the NEC-TT speaker verification system developed for the recent SRE'19 CTS Challenge. Our system is based on an x-vector embedding front-end followed by a thin scoring back-end. We trained a very-deep neural network for x-vector extraction by incorporating residual connections, squeeze-and-excitation networks, and angular-margin softmax at the output layer. We enhanced the back-end with a tandem approach leveraging the benefit of supervised and unsupervised domain adaptation. We obtained over 30% relative reduction in error rate with each of these enhancements at the front-end and back-end, respectively.

Index Terms: speaker recognition, benchmark evaluation

1. Introduction

Benchmark evaluations and challenges have been the major driving force advancing speaker recognition technology [1, 2, 3, 4, 5]. Among these, the series of speaker recognition evaluations (SREs) conducted by NIST are the most prominent and influential. From the first SRE in 1996 [6] to the recent editions [7, 8, 9], how speaker comparison is carried out has changed substantially. Modern approaches first represent the enrollment and test utterances as fixed-length vectors – the so-called speaker embedding. These embeddings are then compared using a simple inner product, or more commonly scored using a probabilistic linear discriminant analysis (PLDA) back-end. This paper presents the key advancements in this approach and benchmarks its performance with the latest NIST SRE'19.

The SRE'19 marks the latest event in the series of SRE conducted by NIST. The benchmark evaluation comprises two parts – CTS Challenge and Multimedia Challenge. In the CTS Challenge, the *train set* consists of English utterances while the *test set* consists of Tunisian Arabic utterances, which poses a substantial mismatch with the train set. For the Multimedia Challenge, the major challenge is the *multi-speaker test* scenario, for which an additional diarization module has to be used to determine the target speaker (if any) from a given test segment. This is coupled with a face recognition from video task. Concerning the speaker recognition task, the core technology used in both challenges share similar components - feature extraction, VAD, x-vector extraction, score normalization and calibration. This paper presents the technical details of the datasets, sub-system development, and analysis of the NEC-TT submission to the SRE'19 CTS Challenge.

The aim of this paper is twofold. Firstly, we present techniques that we have found effective for SRE'19. In this regard, we found that residual connection is a key ingredient to train very-deep neural network for extracting speaker embedding. We also present a better solution to handle domain mismatch. Secondly, we share our insights and findings on what works for speaker recognition in general and possible future directions.

2. Train, Dev and Augmentation sets

The SRE'19 evaluation set consists of narrowband conversational telephone speech (CTS) drawn from the *call-my-net 2* (CMN2) corpus. The design of CMN2 corpus follows the protocol reported in [10]. In particular, the CMN2 corpus consists of recordings spoken in Tunisian Arabic, which were collected over the traditional *public switched telephone network* (PSTN) and the more recent *voice over IP* (VOIP). Different from that of the SRE'19-Eval set, the training set comprises mainly English utterances and were collected over landline and mobile PSTN networks. These differences in terms of languages (Tunisian Arabic *vs.* English) and channel (VoIP *vs.* PSTN) causes a considerable mismatch between the trained model and test data. As we shall illustrate further in this paper, our results show that such domain mismatch could be dealt with by adding small amount of in-domain data to the training set, and also by adapting the PLDA scoring back-end.

Table 1 shows the list of corpora that we used for SRE'19 CTS Challenge. Most part of the training set listed in Table 1 were provided by NIST and LDC. It encompasses the Fisher, Switchboard and SRE' 04, 05, 06, 08, 10, 12, 16 datasets, which have been used extensively in previous SREs. We also used audio-from-video (AfV) data after down-sampling them to 8 kHz. This includes VoxCeleb-1 and VoxCeleb-2 corpora [4] and an in-house dataset collected from YouTube by ourselves. The in-house dataset has a considerable size of $3,672$ speakers, $383,575$ utterances, and a total duration of 977 hours.

Also listed in Table 1 are audio and noise datasets used for data augmentation. These include MUSAN [11], PRISM [12], and a collection of room impulse responses (RIR) prepared for the REVERB Challenge [13].

3. NEC-TT Speaker Verification System

Fig. 1 shows the functional blocks in NEC-TT 2019 speaker verification system. In the following, we highlight those components that had contributed to good performance in SRE'19 CTS Challenge.

3.1. Pre-processing

We use a DNN-based voice activity detection (VAD) algorithm to discard non-speech frames. The DNN receives a sequence of 40-dimensional *mel frequency cepstral coefficients* (MFCCs) extracted from each frame of 25 ms (with 10-ms shift), and produces frame-wise posterior probabilities of *voice*, *noise*, and *silence* classes. The DNN was trained with the Fisher corpus.

http://dx.doi.org/10.21437/Interspeech.2020-1132

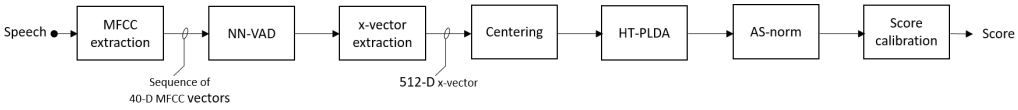

Figure 1: *NEC-TT 2019 speaker verification system comprises a pipeline of functional blocks that converts an input test utterance to x-vector embedding (i.e., the front-end) and scoring with a heavy-tailed PLDA (HT-PLDA) back-end.*

Table 1: *List of speech corpora and audio datasets designated as train, development, and data augmentation sets for SRE'19 CTS Challenge.*

Usage	**Dataset**
Train (out-of-domain)	SRE'04-05-06-08-10-12-16 Swb-2 Phase I, II, III Swb-Cell Part 1, 2 Fisher 1, 2 VoxCeleb 1, 2
Train (in-house)	NEC-TT in-house AfV corpus
Train (in-domain)	SRE'18-Eval SRE'18-CMN2-Unlabeled
Dev (in-domain)	SRE'18-Dev
Data Augmentation	MUSAN, PRISM Reverb Challenge RIR

See [14] for details.

3.2. X-vector extraction

Our variant of x-vector extractor [15] consists of forty-three TDNN layers [16] with residual connections [17] and squeeze-and-excitation (SE) blocks [18, 19]. The detail structures is shown in Table 2. The pooling layer uses a two-head *attentive statistics pooling* [20] in the same way as in [21, 22]. *Additive margin softmax* loss [23] was used for optimizing the network. The 512-dimension bottleneck features from `utt1` layer was used as the x-vector embedding.

Data augmentation was used to increase the amount of training data and improve robustness. This include adding noise (e.g., babble, music), imposing channel noise (codec) and convolutive variation (e.g., room reverberation) to the original audio recordings, as follows:

- Adding noise, music, and mixed speech drawn from the MUSAN [11] and PRISM [12] database at various SNR;
- Adding reverberation by using simulated room impulse responses (RIR) [24], and real RIR drawn from the REVERB challenge database [13]; and
- Encoding speech segments with an AMR codec at 6.7 and 4.75 kbps.

With the above strategy, we increase the amount of training data by three folds.

3.3. Scoring back-end

Heavy-tailed PLDA (HT-PLDA) [25, 26] was used as the scoring back-end. In HT-PLDA, the observations $\boldsymbol{r}$ follows a t-distribution:

$$P(\boldsymbol{r}|\boldsymbol{z}) = T(\boldsymbol{r}|\boldsymbol{F}\boldsymbol{z}, \boldsymbol{W}, \nu), \quad (1)$$

where $\boldsymbol{z}$ is the latent variable, $\boldsymbol{F}$ is the factor loading matrix, $\boldsymbol{W}$ is a positive definite precision matrix, and ν is known as the degrees of freedom. Our previous report [21] shows that heavy-tailed PLDA performs better than the Gaussian PLDA.

Table 2: *Structure of our x-vector extractor with forty-three layers. The notation $\{k, d, o\}$ indicates the configuration of a TDNN, i.e., 1D convolution layer with kernel size k, dilation rate d, output dimension o, and $[\cdot]$ denotes a residual block.*

Layer	Structure
`frame1`	$\{5, 1, 512\}$
`frame2`	$\{1, 1, 512\}$
`frame3:12`	$\begin{bmatrix}\{3,2,512\}\\ \{1,1,512\}\end{bmatrix} \times 5$
`frame13:22`	$\begin{bmatrix}\{3,3,512\}\\ \{1,1,512\}\end{bmatrix} \times 5$
`frame23:32`	$\begin{bmatrix}\{3,4,512\}\\ \{1,1,512\}\end{bmatrix} \times 5$
`frame33:42`	$\begin{bmatrix}\{3,5,512\}\\ \{1,1,512\}\end{bmatrix} \times 5$
`frame43`	$\{1, 1, 1500\}$
`pool`	Two-head attentive statistics
`utt1`	512
`utt2`	512
`output`	Additive margin softmax

3.4. Score normalization and calibration

Scores from all the sub-systems were subject to score normalization before calibration and fusion. To this end, we use symmetric normalization (s-norm) with an adaptive cohort selection scheme [27]. Cohorts were selected from the SRE'18 CMN2 Unlabelled set, which matches well the development, and therefore the evaluation set. To a certain extent, score normalization using the in-domain cohort set performs a score-level domain adaptation.

At the end of the pipeline is the score calibration, whereby scores are scaled and shifted with a linear function. The parameters are trained by optimizing the *log-likelihood ratio cost* (Cllr). We refer the readers to [28] for further discussion on Cllr and its implementation. Score calibration was performed using the `Dev (in-domain)` set.

4. Key Advances

We enhanced NEC-TT SRE'19 speaker verification system at two fronts, namely, (i) a deeper x-vector extractor with residual connections, squeeze-and-excitation (SE) blocks, and angular softmax, and (ii) tandem approach to domain adaptation.

4.1. Very-deep x-vector extractor

A typical x-vector extractor consists of three functional blocks, namely, (i) a frame processor (`frame`) consisting of multiple layers of TDNN, (ii) a statistical pooling layer (`pool`), and (iii) utterance classification (`utt`). The x-vector embedding is extracted from the first classification layer before the activation function) [15].

Table 2 shows the neural network architecture of our x-

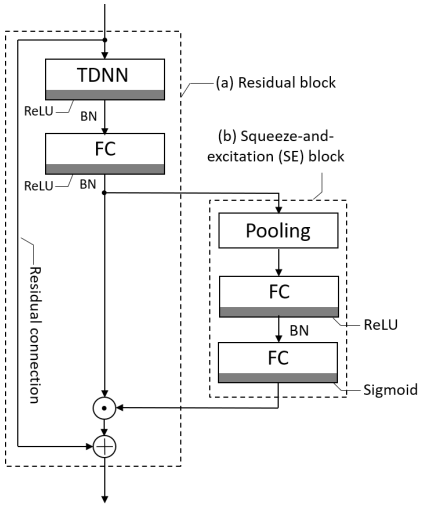

Figure 2: *(a) A residual block comprises a TDNN layer, a fully-connected (FC) layer, and a residual connection. (b) Squeeze-and-excitation (SE) block attached to output of a residual block. The shaded box indicates activation function (ReLU and Sigmoid in this case), and BN denotes batch normalization.*

Table 3: *Performance comparison of x-vector extractor with increasing depth.*

	SRE'18-Dev		SRE'19-Eval	
	EER (%)	Min Cost	EER (%)	Min Cost
Five layers	4.03	0.281	3.95	0.369
Forty-three layers	3.40	0.229	2.68	0.255

vector extractor. The frame processor was extended from five layers, which are typical in most implementations [15, 21], to forty three TDNN layers. The increased depth is made possible by the use of residual connection [17]. As shown in Fig. 2(a), each residual block consists of a TDNN followed by a fully connected (FC) layer. A residual connection passes the input to the TDNN directly to the output of the fully connected layer. This connection allows gradients to flow through the residual block directly, and annihilates the vanishing gradient problem in training very-deep neural networks. In Table 2, FC layers are implemented as TDNNs with kernel size $k = 1$, dilation factor $d = 1$, with $o = 512$ output channels. We used four residual blocks, each repeated five times, with different dilation factors.

Table 3 shows a comparison of x-vector extractors with increasing depth. We kept the pooling and classification layers the same for the comparison. By increasing the number of layers from 5 to 43, we obtain significant reduction in EER and Min Cost across the two test sets. On SRE'18-Dev set, the reduction amounts to 15.6% and 18.5% in EER and Min Cost, respectively. On SRE'19-Eval set, we obtained over 30% of improvement in EER and Min Cost.

4.2. Angular softmax

One key feature that leads to the effectiveness of x-vector is the discriminative cross-entropy loss that optimizes the inter-class separation via the output softmax layer. Angular softmax [23] was proposed with the intention to promote intra-class compactness while attaining inter-class separation. In this work, we used the variant referred to as the *additive margin softmax* [23], which is defined as follows:

$$S(y_i) = \frac{e^{s \cdot cos(\theta_{y_i}) - m}}{e^{s \cdot cos(\theta_{y_i}) - m} + \sum_{j=1; j \neq y_i}^{C} e^{s \cdot cos(\theta_j)}} \quad (2)$$

Here, θ_{y_i} is the angle between the input vector (to the softmax layer) and the weight vector of class y_i among the C target classes. The parameters s and m control the size of the angular margin introduced to the classification loss. We refer the readers to [23, 29] for further details.

The effects of angular softmax could be seen by comparing `front1`, with `front2` and `front3` in Table 4. Compared to the `front1` baseline using the plain vanilla softmax, `front3` gives a relative improvement of 19.7% and 7.5% in Min Cost on SRE'18-Dev and SRE'19-Eval sets, respecitvely, while the impacts on EER is marginal. Though effective, our results show that angular softmax is relatively sensitive to the values of parameters s and m. This can be seen by comparing the performance of `front2` and `front3` with different values used for these parameters.

4.3. Squeeze and excitation

The goal of the squeeze-and-excitation (SE) [19] is to find a set of weights (within zero and one) to be assigned to each output dimension (or channel) of the TDNN layer. SE is realized with a small neural network attached to the output of the TDNN layers. Figure 2(b) shows a SE block attached to a residual block. The output of the FC layer is pooled across time (i.e., *squeeze*) to obtain a mean vector and its standard deviation. These are used to estimate a set of weights to apply on (i.e., *excite*) individual output dimensions of the residual block. The objective is to add content-awareness by weighting these dimensions depending on the input sequence.

We experimented with two configurations. In the first, five SE blocks were attached to residual-blocks $\{4, 8, ..., 20\}$. In the second, ten SE blocks were used on residual-blocks $\{2, 4, ..., 20\}$. Comparing the performance of `front3`, with `front4` and `front5` in Table 4, there is no apparent benefit by adding SE networks. We even observe considerable degradation on Min Cost, though the impacts on EER is negligible.

4.4. Domain adaptation

Parameter tuning and optimization of the x-vector extractor were carried out using the `Train (out-of-domain)` set listed in Table 1. The training set is different from that of the SRE'19-Eval set in terms of languages (Tunisian Arabic *vs.* English) and channel (VoIP *vs.* PSTN). As such, mismatch between the model and the test data is expected.

Domain adaptation was performed on the PLDA using the SRE'18 data, which has the same domain as the SRE'19-Eval set. In this regard, we use a tandem approach [30] leveraging the benefit of supervised [31] and unsupervised [32] PLDA adaptation. Our strategy is as follows. First, the `Train (out-of-domain)` set was used to produce an out-of-domain PLDA. Unsupervised domain adaptation was then applied with CORAL+ [32] using the in-domain SRE'18 set to produce a pseudo in-domain PLDA (note that speaker labels were not used in the adaptation). On the other hand, we trained an in-domain PLDA using the SRE'18 dataset. The final PLDA is obtained as a linear interpolation between the pseudo in-domain PLDA and the in-domain PLDA (i.e., covariance ma-

Table 4: *Comparison of seven different configurations of x-vector extractor. The train set consists of out of domain (OOD), in-domain (InD), and in-house subsets as shown in Table 1.*

Front-end	Train set			A-Softmax		SE	SRE'18-Dev		SRE'19-Eval	
	OOD	InD	In-house	s	m	#blocks	EER (%)	Min Cost	EER (%)	Min Cost
`front1`	✓			-	-	-	3.40	0.229	2.68	0.255
`front2`	✓			32	0.20	-	3.65	0.214	2.66	0.259
`front3`	✓			40	0.15	-	3.52	0.184	2.48	0.236
`front4`	✓			40	0.15	5	3.41	0.212	2.42	0.245
`front5`	✓			40	0.15	10	3.21	0.209	2.50	0.248
`front6`	✓	✓		40	0.15	-	2.99	0.152	2.31	0.217
`front7`	✓		✓	40	0.15	-	3.50	0.209	2.50	0.237
Fusion							3.16	0.160	2.17	0.208

Table 5: *Comparison of supervised, unsupervised, and tandem approaches to domain adaptation of PLDA.*

	SRE'18-Dev		SRE'19-Eval	
	EER (%)	Min Cost	EER (%)	Min Cost
OOD PLDA	4.38	0.284	3.58	0.298
Unsupervised [32]	3.69	0.203	3.01	0.258
Supervised [31]	3.57	0.215	2.63	0.248
Tandem (this study)	3.52	0.184	2.48	0.236

trices interpolation). We used an equal weight of 0.5 in the interpolation.

Table 5 shows the comparison of supervised [31], unsupervised [32], and our tandem approaches to domain adaptation of PLDA. The x-vectors were extracted using `front3` (see Table 4). As expected, supervised adaptation gives a better results than the unsupervised counterpart since speaker labels were used in the former. Nevertheless, the performance gap is rather small. The tandem approach leverages the strength of both techniques. Compared to the OOD PLDA, the improvement amounts to 19.6% reduction in EER and 35.2% reduction in Min Cost on SRE'18-Dev set, while 30.7% reduction in EER and 20.8% reduction in Min Cost were obtained on SRE'19-Eval set.

5. Sub-systems and Fusion

NEC-TT primary submission to SRE'19 CTS Challenge was a fusion system consisting of seven sub-systems. All sub-systems used the x-vector PLDA pipeline as shown in Fig. 1. We trained seven x-vector front-ends with different configurations as shown in Table 4. As described earlier in Section 4.2, `front1` used softmax, while `front2` and `front3` used additive margin softmax. Based on `front3`, SE blocks, as described in Secion 4.3, were inserted to produce `front4` and `front5`. Based on `front3`, `front6` used additional in-domain SRE'18-Eval set, while `front7` used additional in-house AfV dataset. An adapted HT-PLDA back-end was used to produce seven sub-systems.

The x-vector extractors `front1` to `front5` were trained using the out-of-domain train set. We investigated the effectiveness of adding in-domain and our in-house AfV data to the train set. Besides the total speech duration, what we aim to achieve is the expansion of the x-vector output layer with an increased number of speakers in the train set.

We added SRE'18-Eval set (200 speakers) to the out-of-domain train set ($\approx 14,200$ speakers) and used them to train `front6` in Table 4. Similarly, `front7` was trained with our in-house AfV data ($\approx 3,600$ speaker) together with the out-of-domain train set. Comparing `front6` to `front3` in Table 4, we observe a consistent improvement on SRE'18-Dev and SRE'19-Eval sets. On SRE'19-Eval set, the improvement amounts to 6.9% and 8.1% in EER and Min Cost, respectively. Notice that the improvement is attained on-top of the domain adaptation applied on the PLDA. This result is surprising given the smaller size of the in-domain data compared to the out-of-domain train set. Adding our in-house AfV data increases significantly the size of the train set to $\approx 17,800$ speakers. However, comparing `front7` to `front3` in Table 4, we did not observe any benefit of using this set of data. One insight that we could derive here is that adding a small amount of in-domain training data helps the x-vector extractor to perform better on the test set.

The scores from the seven sub-systems were fused to form NEC-TT primary submission. The weights were manually selected based on their performance on the SRE'19 online leaderboard. The fusion results are shown in the last row of Table 4. Compared to the single best (`front6`), the fusion gives an improvement of 6.1% and 4.1% in EER and Min Cost, respectively, on SRE'19-Eval set. The fusion system performs better than all sub-systems except `front6` on SRE'18-Dev set. This is likely due to over-fitting of `front6` on the development set. Considering the computational complexity of each front-end, the performance gain is marginal.

6. Conclusions

We have presented the NEC-TT 2019 speaker verification system, and reported its performance on the recent NIST SRE'19 CTS Challenge. We showed that significant improvement could be attained by increasing the depth of the x-vector extractor from 5 to 43 layers with the use of residual connections. The performance improvement amounts to 30% in EER and Min Cost on the SRE'19-Eval set. We also found that x-vector extractor trained with angular softmax gives lower Min Cost but not on the EER, while the use of squeeze-and-excitation (SE) network does not affect much the performance. We confirmed that domain mismatch could be dealt with effectively by adapting the PLDA back-end, and by adding in-domain data in training the x-vector extractor. The former led to 30.7% and 20.8% relative improvement in EER and Min Cost, respectively, on SRE'19-Eval set. The latter led to 6.9% and 8.1% in EER and Min Cost, respectively. We expect that further improvement could be obtained with better techniques in adapting the x-vector extractor to handle domain mismatch. Finally, we found that the perks of fusing multiple x-vector sub-systems is slimmer compared to our previous experience with shallow-structure sub-systems. These are points for future work.

7. References

[1] K. A. Lee, O. Sadjadi, H. Li, and D. Reynolds, "Two decades into speaker recognition evaluation - are we there yet?" *Computer Speech & Language*, vol. 61, p. 101058, 2020.

[2] C. S. Greenberg, L. P. Mason, S. O. Sadjadi, and D. A. Reynolds, "Two decades of speaker recognition evaluation at the National Institute of Standards and Technology," *Computer Speech & Language*, vol. 60, p. 101032, 2020.

[3] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The Speakers in the Wild (SITW) speaker recognition database," in *Proc. Interspeech*, 2016, pp. 818–822.

[4] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, vol. 60, p. 101027, 2020.

[5] K. A. Lee, A. Larcher, G. Wang, P. Kenny, N. Brümmer, D. v. Leeuwen, H. Aronowitz, M. Kockmann, C. Vaquero, B. Ma, H. Li, T. Stafylakis, M. J. Alam, A. Swart, and J. Perez, "The RedDots data collection for speaker recognition," in *Proc. Interspeech*, 2015, pp. 2996–3000.

[6] A. F. Martin and M. A. Przybocki, "The NIST speaker recognition evaluations: 1996-2001," in *Odyssey: The Speaker and Language Recognition Workshop*, 2001.

[7] K. A. Lee, V. Hautamaki, T. Kinnunen, H. Yamamoto, K. Okabe, V. Vestman, J. Huang, G. Ding, H. Sun, A. Larcher, R. K. Das, H. Li, M. Rouvier, P.-M. Bousquet, W. Rao, Q. Wang, C. Zhang, F. Bahmaninezhad, H. Delgado, and M. Todisco, "I4U submission to NIST SRE 2018: Leveraging from a decade of shared experiences," in *Proc. Interspeech*, 2019, pp. 1497–1501.

[8] P. Matejka, O. Plchot, O. Glembek, L. Burget, J. Rohdin, H. Zeinali, L. Mosner, A. Silnova, O. Novotny, M. Diez, and J. Černocky, "13 years of speaker recognition research at BUT, with longitudinal analysis of nist sre," *Computer Speech & Language*, vol. 63, p. 101035, 2020.

[9] J. Villalba, N. Chen, D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, J. Borgstrom, L. P. García-Perera, F. Richardson, R. Dehak, P. A. Torres-Carrasquillo, and N. Dehak, "State-of-the-art speaker recognition with neural network embeddings in NIST SRE18 and Speakers in the Wild evaluations," *Computer Speech & Language*, vol. 60, p. 101026, 2020.

[10] K. Jones, S. Strassel, K. Walker, D. Graff, and J. Wright, "Call my net corpus: A multilingual corpus for evaluation of speaker recognition technology," in *Proc. Interspeech 2017*, 2017, pp. 2621–2624. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2017-1521

[11] D. Snyder, G. Chen, and D. Povey, "MUSAN: a music, speech, and noise corpus," in *arXiv:1510.08484*, 2015.

[12] L. Ferrer, H. Bratt, L. Burget, H. Cernocky, O. Glembek, M. Graciarena, A. Lawson, Y. Lei, P. Matejka, O. Plchot *et al.*, "Promoting robustness for speaker modeling in the community: the PRISM evaluation set," in *Proceedings of NIST 2011 workshop*, 2011.

[13] K. Kinoshita, M. Delcroix, S. Gannot, E. A. Habets, R. Haeb-Umbach, W. Kellermann, V. Leutnant, R. Maas, T. Nakatani, B. Raj *et al.*, "A summary of the REVERB challenge: state-of-the-art and remaining challenges in reverberant speech processing research," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, p. 7, 2016.

[14] H. Yamamoto, K. Okabe, and T. Koshinaka, "Robust i-vector extraction tightly coupled with voice activity detection using deep neural networks," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2017, pp. 600–604.

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

[16] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Interspeech 2015*, 2015, pp. 3214–3218.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[18] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[19] J. Zhou, T. Jiang, Z. Li, L. Li, and Q. Hong, "Deep speaker embedding extraction with channel-wise feature responses and additive supervision softmax loss function," in *Proc. Interspeech*, 2019, pp. 2883–2887.

[20] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *Proc. Interspeech*, 2018, pp. 2252–2256.

[21] K. A. Lee, H. Yamamoto, K. Okabe, Q. Wang, L. Guo, T. Koshinaka, J. Zhang, and K. Shinoda, "The nec-tt 2018 speaker verification system," *Proc. Interspeech*, pp. 4355–4359, 2019.

[22] ——, "Nec-tt system for mixed-bandwidth and multi-domain speaker recognition," *Computer Speech & Language*, vol. 61, p. 101033, 2020.

[23] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, 2018.

[24] T. Ko, V. Peddinti, M. S. Daniel Povey, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *Proc. IEEE ICASSP*, 2017, pp. 5220–5224.

[25] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Odyssey: Speaker and Language Recognition Workshop*, 2010.

[26] A. Silnova, N. Brümmer, Garcia-Romero, D. Snyder, and L. Burget, "Fast variational bayes for heavy-tailed plda applied to i-vectors and x-vectors," in *Proc. Interspeech*, 2018.

[27] D. Colibro, C. Vair, E. Dalmasso, K. Farrell, G. Karvitsky, S. Cumani, and P. Laface, "Nuance - politecnico di torino's 2016 nist speaker recognition evaluation system," in *Proc. Interspeech*, 2017, pp. 1338–1342.

[28] N. Brümmer and E. de Villiers, "The bosaris toolkit user guide: Theory, algorithms and code for binary classifier score processing," *Documentation of BOSARIS toolkit*, 2011.

[29] Y. Liu, L. He, and J. Liu, "Large margin softmax loss for speaker verification," in *Proc. INTERSPEECH*, 2019.

[30] Q. Wang, K. Okabe, K. A. Lee, and T. Koshinaka, "A generalized framework for domain adaptation of plda in speaker recognition," in *Proc. ICASSP*, 2020, pp. 6619–6623.

[31] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Proc. Interspeech*, 2011, pp. 249–252.

[32] K. A. Lee, Q. Wang, and T. Koshinaka, "The CORAL+ algorithm for unsupervised domain adaptation of plda," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5821–5825.