Revisiting the Statistics Pooling Layer in Deep Speaker Embedding Learning

Shuai Wang, Yexin Yang, Yanmin Qian, Kai Yu

MoE Key Lab of Artificial Intelligence
SpeechLab, Department of Computer Science and Engineering
AI Institute, Shanghai Jiao Tong University, China
{feixiang121976, yangyexin, yanminqian, kai.yu}@sjtu.edu.cn

2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP) | 978-1-7281-6994-1/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISCSLP49672.2021.9362097

Abstract

The pooling function plays a vital role in the segment-level deep speaker embedding learning framework. One common method is to calculate the statistics of the temporal features, while the mean based temporal average pooling (TAP) and temporal statistics pooling (TSTP) which combine mean and standard deviation are two typical approaches. Empirically, researchers observe a big performance degradation in x-vector when removing the standard deviation. Based on this observation, in this paper, we designed a set of experiments to analyze the effectiveness of different statistics quantitatively, including the investigation and comparison on pooling functions based on standard deviation, covariance and ℓ_p-norm. Experiments are carried out on VoxCeleb and SRE16, and the results show that the second-order statistics based pooling functions yield better performance than TAP, and only the simple standard deviation can achieve the best performance on all the evaluation conditions.

Index Terms: speaker embedding, statistics pooling, speaker recognition

1. Introduction

Deep speaker embeddings are currently the dominating approach for speaker identity modeling. Unlike shallow model such as Gaussian mixture model (GMM) [1] or factor analysis [2, 3], deep neural network (DNN) has shown incredible nonlinear modelling power for complex data distribution [4, 5, 6]. Accordingly, one of the hottest topics is to use DNN for representation learning [7, 8], which aims to learn a highly compact and informative embedding to represent the original input.

In the speaker recognition field, the d-vector paradigm was the first well-known DNN based embedding learning framework, which uses a speaker-discriminative DNN to extract frame-level deep features and then average them to a single speaker embedding. However, despite the powerful DNN frontend, d-vector didn't show better results than the conventional i-vector system. As the following work, derived from the d-vector [4] framework, the x-vector structure [9] incorporates a statistics pooling layer to aggregate multiple frame-level deep features to a segment-level representation along the temporal axis, turning deep speaker embedding learning to a segment-level optimization problem. The x-vector style systems exhibits better performance on several popular datasets, including VoxCeleb [10, 11] and NIST SRE.

The success of x-vector comes in two folds, a more powerful learning machine (TDNN) and the segment-level training. Sharing the same idea, on the one hand, different architectures such as ResNet [12], inception network [13, 14] are investigated for the speaker embedding learning task. On the other hand, such a segment-level optimization strategy has been proven to be rather helpful for learning high-quality speaker embeddings [9]. To enable segment-level optimization, one pooling layer is needed to aggregate frame-level features to a single representation. Different pooling functions have been investigated in the literature, including the simple temporal average pooling (TAP) [15], temporal statistics pooling (TSTP) [9] and complicated pooling functions such as self-attentive pooling (SAP) [16], vector of locally aggregated descriptors (VLAD) [17] and learnable dictionary encoding (LDE) [18].

In this work, we will focus on the statistics based pooling functions and give a quantitative analysis of the impact of different statistics. Although TAP is used a lot in the speaker embedding learning task, especially for ResNet models [19, 20], it's always found helpful to use TSTP, which also considers the standard-deviation [9, 21, 16]. However, there is no systematic comparison and analysis of the impact of different statistics on the deep speaker embedding learning. We first derive the temporal standard deviation pooling (TSDP), which only considers the standard deviation of the input feature sequence, giving incredibly good results, which outperforms TAP and even TSTP based models. Motivated by this observation, we experimented with other high-order statistics such as covariance and ℓ_p-norm. Experiments are carried on two datasets, VoxCeleb [11] and NIST SRE 2016 [22] using two different backbones, i.e., TDNN and ResNet, and the results show the superiority of second-order statistics to the first-order mean and the incredible effectiveness of the simple standard deviation for the speaker embedding learning task.

2. Deep speaker embedding learning

A typical segment-level speaker embedding learning framework is shown in Figure 1. After several frame-level feature learning layers, a sequence of deep features would be aggregated into a segment-level representation, which would be projected to the speaker embedding by one or two segment-level affine layers. The whole network is optimized against the softmax-CE loss, aiming to discriminate the speakers in the training data.

In this work, we will focus on the simple statistics based pooling functions and investigate the impact of different statistics on the speaker embedding learning.

2.1. Temporal average pooling

For a given speech segment O, T frames of d-dimensional deep features $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_t, \ldots, \mathbf{x}_T\}$ can be obtained from the last frame-level layer, a pooling layer $\mathcal{P}$ is adopted to aggregate $\mathbf{X}$ to a single representation $\mathbf{z}$[1]. The temporal average pooling (TAP) simply computes the mean vector of $\mathbf{X}$ along the time axis as:

[1]Commonly, $\mathbf{z}$ is not the speaker embedding yet, and an affine layer will be used to transform $\mathbf{z}$ to embedding $\mathbf{e}$.

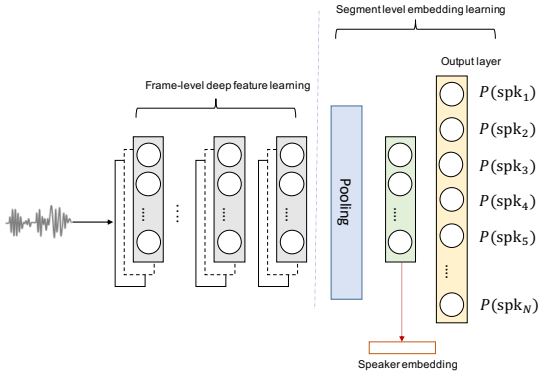

Figure 1: *Architecture of the typical deep speaker embedding learning system*

$$\mathcal{P}_{\text{tap}}(\mathbf{X}) = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}_t \tag{1}$$

2.2. Statistics pooling

In the x-vector framework, instead of the TAP which only uses the first-order statistics $\boldsymbol{\mu}$, a second-order statistics $\boldsymbol{\sigma}$ is appended to the mean vector and constructs the pooling function $\mathcal{P}_{\text{tstp}}$ as the concatenation of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, [2],

$$\mathcal{P}_{\text{tstp}}(\mathbf{X}) = [\boldsymbol{\mu}^\top, \boldsymbol{\sigma}^\top]^\top \tag{2}$$

where $\boldsymbol{\mu} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}_t$ and $\boldsymbol{\sigma} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\mathbf{x}_t - \boldsymbol{\mu})^2}$

3. Second-order statistics based pooling

3.1. Temporal standard deviation pooling

Although TSTP show better performance than TAP in the x-vector framework, there is no systematic analysis how much the standard deviation helps. To better analyze its effect, we derive the temporal standard deviation pooling (TSDP) as

$$\mathcal{P}_{\text{tsdp}}(\mathbf{X}) = \boldsymbol{\sigma} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\mathbf{x}_t - \boldsymbol{\mu})^2} \tag{3}$$

3.2. Global covariance pooling

Covariance based pooling has been investigated a lot in the image processing community [23, 24, 25]. Unlike the variance based pooling, covariance considers the correlation of different feature dimensions, yielding a more comprehensive description of data.

$$\boldsymbol{\Sigma} = \mathbf{X}\bar{\mathbf{I}}\mathbf{X}^T \tag{4}$$

where $\bar{\mathbf{I}} = \frac{1}{T}\left(\mathbf{I} - \frac{1}{T}\mathbf{1}\right)$, $\mathbf{I}$ and $\mathbf{1}$ are the $T \times T$ identity matrix and matrix of all ones. Instead of the original formula, we adopted the normalized version, iSQRT-COV [26, 23, 24], which applies iterative matrix square root normalization to boost the performance.

$$\boldsymbol{\Sigma}^{\frac{1}{2}} = \mathbf{U}\boldsymbol{\Lambda}^{\frac{1}{2}}\mathbf{U}^T \tag{5}$$

[2]It should be noticed that the formulas are not strictly defined for the vector based input, we use the notations just for simplicity

where $\mathbf{U}$ and $\boldsymbol{\Lambda}$ are the matrix of eigen-vectors and the diagonal matrix of eigenvalues of $\boldsymbol{\Sigma}$, respectively.

Then we define covariance based pooling (GCP) as

$$\mathcal{P}_{gcp} = \text{vech}(\boldsymbol{\Sigma}^{\frac{1}{2}}) \tag{6}$$

where vech is the half vectorization which takes the upper triangular part of a matrix and flatten it as a vector. Thus, the final output vector of $\mathcal{P}_{\text{gcp}}$ has the size of $\frac{d(d+1)}{2}$. More details could be referred to the paper and the open-source implementation[27].

3.3. ℓ_p-norm pooling

In our experiments, we observe an impressive performance improvement of TSDP, which is based on the second-order central moment standard deviation, we wonder whether high-order non-central moment such as power-averaging could also help. Accordingly, the temporal ℓ_p -norm pooling [3] is also investigated, which is stated as follows,

$$\mathcal{P}_{\text{tlpp}}(\mathbf{X}) = \frac{1}{T}\sqrt[p]{\sum_{t=1}^{T}\mathbf{x}_t^p} \tag{7}$$

When p is set to 1, ℓ_p -norm equals to the mean operation and $\mathcal{P}_{\text{tlpp}}$ is the same as $\mathcal{P}_{\text{tap}}$. In this work, we mainly investigate the usage of ℓ_2 -norm by setting $p = 2$.

4. Experiments

All the experiments are carried on two datasets: VoxCeleb and NIST SRE. Except for the difference in the data used, the other setups, such as feature preparing, neural network architectures and optimization strategy, are shared across the two sets of experiments. We will first describe the common experimental setups in Sec. 4.1. The detailed description of training and evaluation data along with the result presentation will be given in Sec. 4.2 and Sec. 4.3, respectively.

4.1. Experimental setups

4.1.1. Data preparation

The training data is cut to segments with random duration ranging from $2-4$s, following the Kaldi recipe[28, 29], however, we didn't use any data augmentation in most setups and used 40-dimensional Fbank as the input features. An energy-based VAD is applied to remove silent frames and all features are processed with Cepstral Mean Normalization (CMN).

4.1.2. Model architecture

For experiments on both datasets, two different popular speaker embedding front-ends are investigated: x-vector extracted from the 1-D convolutional TDNN and r-vector from the 2-D convolutional ResNet. The details are described as follows.

The TDNN structure is depicted in Table 1, which follows the standard x-vector system described in [9] and used in the Kaldi Recipe, except the pooling related layers changed according to different aggregation functions. The x-vector is extracted from the layer "segment1" in Table 1.

The ResNet architecture in [20] is used, which is shown in Table 2. The 34-layer version is used in this work. The r-vector is extracted from the layer "Dense" in Table 2.

[3]Also referred as power-average pooling in Pytorch, we add an additional $\frac{1}{T}$ to normalize the effect of utterance length

Table 1: *The TDNN based speaker embedding extractor. T denotes the sequence length, and N is the number of speakers. [30]*

Layer	Layer context	Input × output
frame1	$[t-2, t+2]$	200×512
frame2	$\{t-2, t, t+2\}$	1536×512
frame3	$\{t-3, t, t+3\}$	1536×512
frame4	$\{t\}$	512×512
frame5	$\{t\}$	512×1500
stats pooling	$[0, T]$	$1500 \times N_{\text{pool}}$
segment1	$\{0\}$	$N_{\text{pool}} \times 512$
segment2	$\{0\}$	512×512
projection	$\{0\}$	$512 \times$ N

Table 2: *The ResNet34 based speaker embedding extractor. T denotes the sequence length, and N is the number of speakers.*

Layer name	Structure	Output
Input	–	$40 \times T \times 1$
Conv2D-1	3×3, Stride 1	$40 \times T \times 32$
ResNetBlock-1	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 3$, Stride 1	$40 \times T \times 32$
ResNetBlock-2	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 4$, Stride 2	$20 \times \frac{T}{2} \times 64$
ResNetBlock-3	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 6$, Stride 2	$10 \times \frac{T}{4} \times 128$
ResNetBlock-4	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 3$, Stride 2	$5 \times \frac{T}{8} \times 256$
StatsPooling & Flatten	–	N_{pool}
Dense	–	256
Projection	–	N

For both the TDNN and ResNet systems, the pooling related layers are customized to fit different statistics. Accordingly, the width of the related layers can differ. N_{pool} used in TDNN and ResNet for different statistics are listed in Table 3. For all the TLPP based systems, p is set to 2 and the ℓ_p-norm is essentially ℓ_2-norm.

Table 3: *N_{pool} used in TDNN (Table 1) and ResNet (Table 2) for different pooling functions*

Pooling (Statistics)	TDNN	ResNet
TAP (mean)	1500	1280
TSDP (stddev)	1500	1280
TSTP (mean+stddev)	3000	2560
TLPP (ℓ_p-norm)	1500	1280
GCP (covariance)[3]	1275	1275

4.1.3. Neural network optimization

The optimization for all the models used in this work shares the same setups, and SGD with $1\text{e}{-4}$ momentum is used as the network optimizer. The learning rate is initially set to 0.1 and gradually reduced to $1\text{e}{-6}$ along with the training process. The training is paralleled on 4 GPUs, with a batch-size 64 on each GPU, resulting in a total batch-size of 256.

4.1.4. Evaluation metrics

We report the performance in terms of equal error rate (EER) and minimum detection cost function (minDCF), where p_{target} is set to 0.01.

4.2. Experiments on VoxCeleb

4.2.1. Dataset

VoxCeleb was released by Oxford and has been one of the most popular speaker recognition dataset. Two parts, VoxCeleb1 and VoxCeleb2 were included. In this work, the DEV set of VoxCeleb2 is used as the training data, which contains 5994 speakers and 1092009 utterances. The first part VoxCeleb1 is used as the evaluation set, and all three official trial lists (cleaned version) VoxCeleb_O, VoxCeleb_E and VoxCeleb_H are used for the evaluation.

4.2.2. Results

The results of VoxCeleb can be found in Table 4. As shown in the first two rows of both TDNN and ResNet systems, appending the standard deviation consistently outperform the original mean based systems, which is consistent with the observations in [16, 31]. All the second-order statistics based systems outperform the mean based systems, exhibiting the effectiveness of high-order statistics. The best results are obtained by using only the standard deviation, even exceeding the systems using TSTP, a $1.56\%, 1.78\%$ and 3.07% EER on the three evaluation sets. This observation shows that the simple concatenation might not be a proper way to integrate the information encoded in mean and standard deviation.

4.3. Experiments on SRE16

4.3.1. Dataset

The training set consists of two parts of data: SRE portion and SWBD portion. The former contains data selected from NIST SRE2004-2010 and the latter contains data selected from Switchboard dataset. To enable faster experiments, compared with other works[32], we adopted a more aggressive data filtering strategy, the final training list includes 62949 recordings from 3419 speakers. Systems are evaluated on the evaluation set of SRE16, including the Cantonese and Tagalog subsets. The unlabelled part of SRE16 is used for PLDA adaptation in the scoring phase.

4.3.2. Results

Results on SRE16 are shown in Table 5, which also contains both the TDNN and ResNet systems. Consistent with the VoxCeleb results in Table 4, the standard deviation pooling achieves the best performance on both Cantonese and Tagalog, with both TDNN and ResNet.

As shown in Table 4 and 5, TSDP achieves surprisingly the best performance and consistently outperforms the normal TSTP. Despite the good results on VoxCeleb, the performance on SRE16 is still not comparable to the ones in the literature [32, 33], where data augmentation is used to boost the system performance. Thus the standard deviation pooling based

[3]For GCP, a 1x1 convolution layer with BatchNorm and ReLU is applied before pooling to avoid enormous N_{pool}. To make the output dimension comparable with other pooling functions, the number of output channels is 50 for both TDNN and ResNet.

Table 4: *Results on Voxceleb Dataset using different statistics for pooling*

Architecture	Pooling Stats	voxceleb1_O		voxceleb1_E		voxceleb1_H	
		EER	minDCF	EER	minDCF	EER	minDCF
TDNN	mean	3.36	0.370	3.40	0.360	5.65	0.508
	mean+stddev	2.53	0.263	2.71	0.299	4.56	0.420
	ℓ_p-norm	2.50	0.313	2.69	0.296	4.54	0.435
	cov	2.66	0.300	2.94	0.309	4.90	0.458
	stddev	**2.41**	**0.260**	**2.59**	**0.270**	**4.33**	**0.399**
ResNet	mean	2.35	0.274	2.22	0.257	3.75	0.365
	mean+stddev	1.94	0.219	1.89	0.235	3.24	0.323
	ℓ_p-norm	1.81	0.232	1.85	0.219	3.15	0.314
	cov	1.95	0.242	2.11	0.257	3.57	0.347
	stddev	**1.56**	**0.183**	**1.78**	**0.209**	**3.07**	**0.305**

Table 5: *Results on SRE 2016 evaluation set, with unsupervised PLDA adaptation*

Architecture	Pooling Stats	Cantonese		Tagalog	
		EER	minDCF	EER	minDCF
TDNN	mean	7.21	0.542	17.21	0.888
	mean+stddev	5.81	0.469	15.34	**0.830**
	ℓ_p-norm	5.86	0.479	15.51	0.904
	cov	5.25	**0.448**	**13.99**	0.897
	stddev	**5.21**	0.454	14.16	0.856
ResNet	mean	5.36	0.436	15.58	0.883
	mean+stddev	4.26	0.371	12.91	0.845
	ℓ_p-norm	4.72	0.398	14.17	0.885
	cov	4.72	0.422	12.94	0.884
	stddev	**4.17**	**0.370**	**12.67**	**0.814**
TDNN (Aug)	mean+stddev	3.98	0.341	12.25	0.816
	stddev	3.72	0.350	12.28	0.824
ResNet (Aug)	mean+stddev	3.28	0.308	10.72	0.759
	stddev	3.28	0.287	10.79	0.745

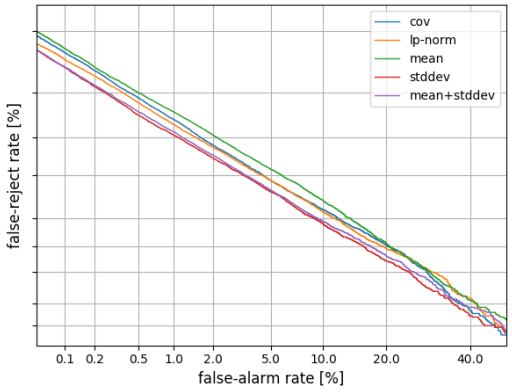

Figure 2: *DET Plot for ResNet systems on SRE16 Cantonese*

systems are also evaluated with Kaldi-style data augmentation, which are more competitive and shown as the bottom rows in Table 5. To give a more intuitive comparison between different pooling functions, we draw the DET plot with the ResNet results on SRE16 Cantonese, which is shown in Figure 2.

4.4. Investigation of Higher-order statistics

Besides the second-order statistics described above, we also investigated several types of higher-order statistics, although no better results were obtained, we would like to briefly summarize here.

1. For the ℓ_p-norm with $p = 3$ or $p = 4$, slightly worse results were obtained compared to p=2
2. Higher-order deviation $\sqrt[p]{\frac{1}{T}\sum_{t=1}^{T}(\mathbf{x}_t - \boldsymbol{\mu})^p}$ with $p = 3$ or $p = 4$ also underperform the standard deviation where $p = 2$.
3. The third-order skewness $\frac{1}{T}\sum_{t=1}^{T}\left(\frac{\mathbf{x}_t-\boldsymbol{\mu}}{\boldsymbol{\sigma}}\right)^3$ and fourth-order kurtosis $\frac{1}{T}\sum_{t=1}^{T}\left(\frac{\mathbf{x}_t-\boldsymbol{\mu}}{\boldsymbol{\sigma}}\right)^4$ used in [32] achieved even worse results than the simple first-order TAP.

5. Conclusion and future work

In this paper, we exhibit the impact of different statistics for segment-level speaker embedding learning. Based on different statistics such as mean, standard deviation, covariance and ℓ_p-norm, different backbones such as TDNN and ResNet are evaluated and compared, and experiments are carried out on VoxCeleb and SRE16 datasets. Consistent performance improvement is obtained using second-order statistics compared to the temporal average pooling, while the best performance can be achieved by the standard deviation based pooling.

It's interesting and surprising to see the effectiveness of the high order statistics, especially the standard deviation. *Standard deviation describes the fluctuation of the data, whereas conventionally, we believe the mean represents the information which exists throughout the whole sequence, including the speaker identity*. According to the experiments in [25] on text-dependent tasks, the variance-based aggregation also outperforms the mean based one in both d-vector and j-vector framework. The observation implies that dynamic information encoded by standard deviation not only contains the phonetic information but also provides speaker-dependent information. In future work, we will keep investigating on 1) a better strategy to integrate different statistics 2) a more advanced way to utilize the dynamic speaker identity information encoded in the speech sequence and new feature investigation.

6. Acknowledgements

This work was supported by the China NSFC projects (No. 62071288 and No. U1736202). Experiments have been carried out on the PI supercomputers at Shanghai Jiao Tong University.

7. References

[1] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[2] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," *CRIM, Montreal,(Report) CRIM-06/08-13*, 2005.

[3] 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, 2011.

[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 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014*. IEEE, 2014, pp. 4052–4056.

[5] Y. Liu, Y. Qian, N. Chen, T. Fu, Y. Zhang, and K. Yu, "Deep feature for text-dependent speaker verification," *Speech Communication*, vol. 73, pp. 1–13, 2015.

[6] 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 *Spoken Language Technology Workshop (SLT), 2016*. IEEE, 2016, pp. 165–170.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

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

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

[11] 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.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[13] C. Zhang and K. Koishida, "End-to-end text-independent speaker verification with triplet loss on short utterances," *Proc. Interspeech 2017*, pp. 1487–1491, 2017.

[14] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, "Deep speaker feature learning for text-independent speaker verification," *arXiv preprint arXiv:1705.03670*, 2017.

[15] S. Shon, H. Tang, and J. Glass, "Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-to-end model," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 1007–1013.

[16] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey, "Self-attentive speaker embeddings for text-independent speaker verification."

[17] W. Xie, A. Nagrani, J. S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5791–5795.

[18] W. Cai, Z. Cai, X. Zhang, X. Wang, and M. Li, "A novel learnable dictionary encoding layer for end-to-end language identification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5189–5193.

[19] S. Wang, Y. Yang, T. Wang, Y. Qian, and K. Yu, "Knowledge distillation for small foot-print deep speaker embedding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6021–6025.

[20] H. Zeinali, S. Wang, A. Silnova, P. Matějka, and O. Plchot, "But system description to voxceleb speaker recognition challenge 2019," *arXiv preprint arXiv:1910.12592*, 2019.

[21] S. Wang, Z. Huang, Y. Qian, and K. Yu, "Deep discriminant analysis for i-vector based robust speaker recognition," in *ISCSLP*, 2018.

[22] S. O. Sadjadi, T. Kheyrkhah, A. Tong, C. Greenberg, E. S. Reynolds, L. Mason, and J. Hernandez-Cordero, "The 2016 nist speaker recognition evaluation," 2017.

[23] P. Li, J. Xie, Q. Wang, and W. Zuo, "Is second-order information helpful for large-scale visual recognition?" in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2070–2078.

[24] P. Li, J. Xie, Q. Wang, and Z. Gao, "Towards faster training of global covariance pooling networks by iterative matrix square root normalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 947–955.

[25] S. Wang, H. Dinkel, Y. Qian, and K. Yu, "Covariance based deep feature for text-dependent speaker verification," in *International Conference on Intelligent Science and Big Data Engineering*. Springer, 2018, pp. 231–242.

[26] Q. Wang, J. Xie, W. Zuo, L. Zhang, and P. Li, "Deep cnns meet global covariance pooling: Better representation and generalization," *arXiv preprint arXiv:1904.06836*, 2019.

[27] X. Jiangtao and L. Peihua. Pytorch implementation of mpn-cov. [Online]. Available: https://github.com/jiangtaoxie/fast-MPN-COV/blob/master/src/representation/MPNCOV.py

[28] Kaldi. Sre16 speaker verification recipe in kaldi. [Online]. Available: https://github.com/kaldi-asr/kaldi/tree/master/egs/sre16

[29] 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, 2011.

[30] 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.

[31] S. Wang, Z. Huang, Y. Qian, and K. Yu, "Discriminative neural embedding learning for short-duration text-independent speaker verification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 11, pp. 1686–1696, 2019.

[32] L. You, W. Guo, L. Dai, and J. Du, "Deep neural network embedding learning with high-order statistics for text-independent speaker verification," *arXiv preprint arXiv:1903.12058*, 2019.

[33] J. Rohdin, T. Stafylakis, A. Silnova, H. Zeinali, L. Burget, and O. Plchot, "Speaker verification using end-to-end adversarial language adaptation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6006–6010.