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

Vector-Based Attentive Pooling for Text-Independent Speaker Verification

Yanfeng Wu[1], *Chenkai Guo*[2], *Hongcan Gao*[2], *Xiaolei Hou*[2], *Jing Xu*[1]

[1]College of Artificial Intelligence, Nankai University, Tianjin, China
[2]College of Computer Science, Nankai University, Tianjin, China
{yanfeng_wu,gaohongcan,houxiaolei}@mail.nankai.edu.cn, {guochenkai,xujing}@nankai.edu.cn

Abstract

The pooling mechanism plays an important role in deep neural network based systems for text-independent speaker verification, which aggregates the variable-length frame-level vector sequence across all frames into a fixed-dimensional utterance-level representation. Previous attentive pooling methods employ scalar attention weights for each frame-level vector, resulting in insufficient collection of discriminative information. To address this issue, this paper proposes a vector-based attentive pooling method, which adopts vectorial attention instead of scalar attention. The vectorial attention can extract fine-grained features for discriminating different speakers. Besides, the vector-based attentive pooling is extended in a multi-head way for better speaker embeddings from multiple aspects. The proposed pooling method is evaluated with the x-vector baseline system. Experiments are conducted on two public datasets, VoxCeleb and Speaker in the Wild (SITW). The results show that the vector-based attentive pooling method achieves superior performance compared with statistics pooling and three state-of-the-art attentive pooling methods, with the best equal error rate (EER) of 2.734 and 3.062 in SITW as well as the best EER of 2.466 in VoxCeleb.

Index Terms: text-independent speaker verification, deep neural network, pooling mechanism, vector-based attention

1. Introduction

Speaker verification (SV) serves as a common research topic to check “who spoke” based on the voices, which can be applied in various fields, such as criminal investigation, judicial forensics and telephone identification. Given a slice of speech, an SV system aims to identify if it belongs to a specific person. SV systems can be generally classified into two typical categories according to the content of utterances: text-dependent and text-independent. Text-dependent SV systems require the texts of utterances to be fixed; while text-independent SV systems have no restrictions on the texts of utterances, which is the focus of this work.

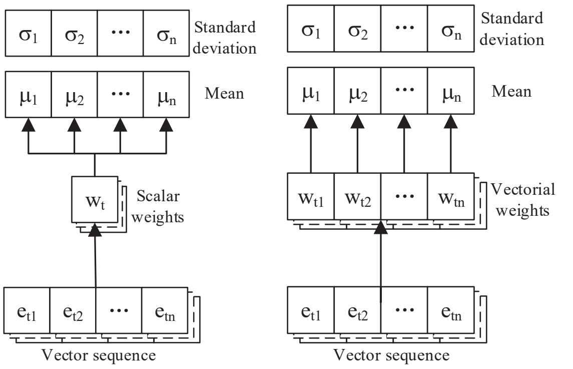
Over the past decade, text-independent SV tasks commonly have relied on the combination of i-vector-based representation [1] and a probabilistic linear discriminant analysis (PLDA) [2] backend classifier. Recently, with the great success of deep learning over a wide range of machine learning tasks, an increasing number of studies introduce deep neural network (DNN) relevant techniques into the SV area, and achieve competitive verification performance compared with traditional i-vector-based works. In detail, DNN-based SV systems can be divided into two classes: deep speaker embedding systems and end-to-end systems. Deep speaker embedding systems [3–6] train a DNN instead of i-vector to produce speaker embeddings and use a separately trained PLDA classifier to measure embedding pairs; while the end-to-end DNN SV systems try to put the two stages of deep speaker embedding systems together. The end-to-end systems usually train the DNN architecture with various kinds of loss functions (e.g., triplet loss [7, 8], adversarial loss [9, 10] and others [11, 12]) to discriminate between the same-speaker and different-speaker pairs and adopt the cosine distance scores to evaluate the test utterances. Among DNN-based SV systems, x-vector [3], as well as its modified version [13–17], achieves superior performance than i-vector and becomes the state-of-the-art method for current SV works.

For text-independent SV, since the input utterances have variable lengths, most DNN-based SV systems employ a pooling layer to aggregate the variable-length frame-level vector sequence so as to obtain a fixed-dimensional utterance-level vector. In the implementation of x-vector, a statistics pooling layer is employed by computing the mean and standard deviation of frame-level features. However, the statistics pooling normally assigns equal weight to each frame-level vector, which ignores the importance of some critical frames during the training, resulting in a challenge for the performance improvement. To address the challenge, recent studies were proposed to integrate the attention mechanism into the pooling layer. For instance, Okabe *et al.* [18] proposed an attentive statistics pooling method which acquires weighted mean and standard deviations of frame-level features, and the weights are calculated by an attention mechanism. Zhu *et al.* [19] introduced a pooling method where the weights are determined by a self-attention mechanism with multiple attention heads. India *et al.* [20] presented a self multi-head attention method where the weights are calculated considering different parts of the sequence. However, there is a common limitation in prior methods that the attention weight for each frame-level vector is naturally computed as a scalar. As a result, each element of a frame-level vector has an equal attention weight when computing the weighted mean and standard deviation, leading to insufficient extraction of important features.

Under the insight, inspired by recent generalized pooling in sentence embedding [21], we present a self vector-based attentive pooling method for text-independent SV, where the vectorial attention instead of scalar attention is adopted. The vector-based attention can extract fine-grained features and more discriminative information from the encoded representations. Besides, we extend the vector-based attentive pooling in a multi-head way so as to discriminate speakers from multiple aspects. We implement the pooling methods in the x-vector baseline system and compare the proposed pooling method with four state-of-the-art pooling methods. Experiments conducted on two public datasets, VoxCeleb [22–24] and Speaker in the Wild (SITW) [25] demonstrate the effectiveness of the proposed method. The remainder of this paper is organized as follows. Section 2 introduces the proposed pooling method. The experimental setup and results are presented in Section 3 and Section 4 respectively. Finally, the conclusion is given in Section 5.

http://dx.doi.org/10.21437/Interspeech.2020-1422

2. Attentive pooling method



(a) Attentive statistics pooling. (b) Vector-based attentive pooling.

Figure 1: *Structure of two attentive pooling methods.*

In this section, we review the scalar attentive pooling method called attentive statistics pooling [18] for text-independent SV, and then describe the proposed vector-based attentive pooling method.

2.1. Attentive statistics pooling

Attentive statistics pooling method aims to capture more information for important frame-level features with respect to a long-term feature variation. Given a sequence of encoded hidden representation, this method assigns a weight over each representation of the sequence through a trainable layer and then obtains respective weighted mean and weighted standard deviation of these representations as the utterance-level representation.

Considering that the frame-level encoded representation is a sequence of hidden vectors, denoted as $H = [h_1, ..., h_t, ..., h_T] \in \mathbb{R}^{T\times N}$ with $h_t = [e_{t1}, ..., e_{tn}, ..., e_{tN}]$ where T is the number of frames and N is the dimension of each hidden vector h_t. The relevant scalar weight for each element of the vector h_t can be defined as follows:

$$w_t = \tau(v^T f(Wh_t + b)), \tag{1}$$

where τ is the softmax activation function calculated as follows:

$$\tau(e_t) = \frac{\exp(e_t)}{\sum_{t=1}^{T}\exp(e_t)}. \tag{2}$$

In Eq. (1), $W \in \mathbb{R}^{N\times N}$ is the weight matrix; $b \in \mathbb{R}^N$ is the bias item; $v \in \mathbb{R}^N$ is the weight vector; $f(\cdot)$ is a non-linear activation function, e.g. ReLU.

Given the set of weights over all elements of the sequence, the weighted mean vector μ and weighted standard deviation vector σ can be obtained by the following equations:

$$\mu = \sum_{t=1}^{T} w_t h_t, \tag{3}$$

$$\sigma = \sqrt{\sum_{t=1}^{T} w_t h_t \odot h_t - \mu \odot \mu}, \tag{4}$$

where $\odot$ denotes the element-wise product.

Finally, the utterance-level representation E is the concatenation of μ and σ:

$$E = [\mu; \sigma]. \tag{5}$$

Note that the weighted mean μ and standard deviation σ effectively reflect the speaker's nature in terms of the temporal variations over long-term contexts, whose effectiveness has been proved in recent studies [18, 19]. Therefore, we also generate the concatenated weighted mean and standard deviation vector as the output of pooling layer in the implementation of the vector-based attentive pooling method.

2.2. Vector-based attentive pooling method

The attentive statistics pooling method provides the attention weight w_t for each vector h_t at the frame t as a scalar, where each element e_{tn} of the h_t has the same weight. Nonetheless, each element can have different weight for a better discriminative utterance-level representation. Based on above observations, we present a self vector-based attentive pooling method for text-independent SV, as shown in Figure 1.

The vector-based attention was first proposed in [21] for sentence embedding. Such attention provides a vectorial attention weight for each encoded hidden representation, which collects more discriminative information than traditional scalar attention. The vector-based attention weight over all frames can be represented as a matrix $A = [a_1, ..., a_t, ..., a_T] \in \mathbb{R}^{T\times N}$ with $a_t = [w_{t1}, ..., w_{tn}, ..., w_{tN}]$. The a_t is the vectorial weight for each element of the vector h_t at the frame t, and the matrix can be computed as:

$$A = \tau(W_2 f(W_1 H^T + b_1) + b_2)^T, \tag{6}$$

where $W_1 \in \mathbb{R}^{d_a\times N}$ and $W_2 \in \mathbb{R}^{N\times d_a}$ are weight matrices; $b_1 \in \mathbb{R}^{d_a}$ and $b_2 \in \mathbb{R}^N$ are bias items; d_a is a hyper-parameter. The softmax function ensures that the sum of all elements is 1 in every column of the weight matrix A. Each element w_{tn} of the vector a_t is the attention weight for the element e_{tn} of the h_t. The scalar attention is a special case of the vector-based attention when the element in a_t is equal to each other, which can be represented as

$$w_{t1} = ... = w_{tn} = ... = w_{tN}, \forall t \in \{1, ..., T\}. \tag{7}$$

In order to make the vectorial attention discriminate features from multiple aspects, we extend the pooling method in a multi-head way as following equation:

$$A^i = \tau(W_2^i f(W_1^i H^T + b_1^i) + b_2^i)^T, \forall i \in \{1, ..., I\}, \tag{8}$$

where A^i is the vectorial weight matrix generated by i-th attention head and I is the number of attention heads.

The i-th weighted mean vector μ^i can be calculated by summing up the element-wise products of each frame-level vector h_t and the vectorial weight a_t^i:

$$\mu^i = \sum_{t=1}^{T} \alpha_t^i \odot h_t, \forall i \in \{1, ..., I\}. \tag{9}$$

And the i-th weighted standard deviation vector σ^i can be acquired as follows:

$$\sigma^i = \sqrt{\sum_{t=1}^{T} \alpha_t^i \odot h_t \odot h_t - \mu^i \odot \mu^i}, \forall i \in \{1, ..., I\} \tag{10}$$

Finally, the output E' of the vector-based attentive pooling layer is the vector concatenating μ^i and σ^i in all I attention heads:

$$E' = [\mu^1; ...; \mu^I; \sigma^1; ...; \sigma^I] \tag{11}$$

Table 1: *Experimental results on SITW. **Boldface** values are the best results.*

Embedding(Attention heads)	Parameters	Development			Evaluation		
		EER(%)	DCF10^{-2}	DCF10^{-3}	EER(%)	DCF10^{-2}	DCF10^{-3}
i-vector [1]	-	5.622	0.4799	0.6814	6.534	0.5211	0.6939
statistics pooling [3]	4.4M	3.389	0.3367	0.5315	3.581	0.3639	0.5618
self multi-head attention [20]	3.6M	3.157	0.3152	0.5270	3.417	0.3464	0.5367
attentive statistics pooling [18]	6.7M	2.772	0.3042	0.5083	3.280	0.3302	0.5371
self-attentive pooling(1) [19]	5.2M	3.003	0.3094	0.4784	3.472	0.3409	0.5489
self-attentive pooling(2) [19]	6.7M	2.965	0.3190	0.5044	3.390	0.3421	0.5272
self-attentive pooling(5) [19]	11M	3.157	0.3062	0.4804	3.144	0.3355	0.5136
vector-based attentive pooling(1)	5.9M	2.849	**0.2808**	0.4628	**3.062**	0.3218	0.5044
vector-based attentive pooling(2)	8.9M	**2.734**	0.2845	**0.4601**	**3.062**	**0.3195**	**0.4955**
vector-based attentive pooling(3)	12M	3.080	0.2919	0.4801	3.253	0.3402	0.5314

If the vector-based attention is multi-head, a penalty term P is added to the cross-entropy loss function:

$$P = \rho \sum_{i=1}^{I} \sum_{j=i+1}^{I} \max(\lambda - ||A^i - A^j||_F^2, 0) \qquad (12)$$

where ρ and λ are corresponding hyper-parameters; $|| \cdot ||_F$ represents the Frobenius norm of matrix. The penalty term encourages the diversity of the attention matrices across different heads of attention so that each head can collect dissimilar information.

3. Experimental setup

3.1. Datasets and baselines

The experiments are conducted on two public datasets, VoxCeleb [22, 23] and SITW [25], which have been commonly used in relevant comparative experiments [26–28]. The training set is the development portion of VoxCeleb2 without any data augmentation technique, which contains 5994 speakers and over one million utterances. There are three test sets in the two datasets including SITW development core-core condition, SITW evaluation core-core condition and VoxCeleb1 test portion. The VoxCeleb dataset contains short utterances with the average length of 8 seconds, while the duration of utterances in SITW dataset ranges from 6 seconds to 180 seconds. Hence, all models in our experiment can be evaluated with both short and long utterances.

There are two baseline systems in our experiment, x-vector [3] and i-vector [1]. In x-vector system, the proposed vector-based attention pooling is compared with three baseline pooling methods: (i) statistics pooling of traditional x-vector, (ii) attentive statistics pooling [18], (iii) self multi-head attention based pooling [20], and (iv) self-attentive pooling with single head, two heads and five heads [19]. For better comparison, the vector-based attentive pooling is set with a single head, two heads and three heads, respectively.

3.2. Model Configuration

For the i-vector baseline, the input acoustic features are 24-dimension MFCCs with deltas and delta-deltas, which have a total of 72 dimensions. For the x-vector baseline, MFCCs of 30 dimensions are used. All features are acquired from 25ms windows with 10ms shift between frames. In addition, we exploit mean normalization with a 3-second window, and adopt energy-based voice active detection (VAD) to remove the non-speech frames. The data preprocessing step is handled by Kaldi toolkit [29].

The parameters of baseline systems and methods are configured as corresponding literature. In i-vector system, the dimension of i-vector is 400. In x-vector system, the dimension of the first four frame-level layers and two utterance-level FC layers is set as 512; while the dimension of fifth frame-level layer is set as 1500. For the self multi-head attention based pooling, the number of heads is set as 15. For the self-attentive pooling, the parameter d_a is set to 500. For the proposed vector-based attentive pooling, the parameter d_a, ρ and λ are set to 500, 1 and 1 respectively.

The embedding extraction and network training of all DNN-based systems are implemented on an existing Tensorflow tookit [30], where the standard softmax with cross entropy loss function and stochastic gradient descent (SGD) optimizer with momentum of 0.9 are employed to compute the loss score. The batch size is set as 128. The learning rate is set to 0.001 initially, and then is continuously halved once the validation loss fails to decrease for a while till it goes down below 10^{-6}. Afterwards, the outputs of the second utterance-level layer are extracted as speaker embeddings. We use the PLDA model implemented in Kaldi as the backend classifier to compute verification scores for all comparative systems. The extracted embeddings are centered and projected to 200-dimensional vectors using linear discriminative analysis (LDA), which are then length-normalized and computed by the PLDA.

4. Result

We evaluate the experimental results in terms of equal error rate (EER) and the minimum of normalized detection cost function for which the prior target probability is set as 0.01 (DCF10^{-2}) and 0.001 (DCF10^{-3}).

4.1. Results on SITW

Table 1 shows the performance on SITW. All attentive pooling methods outperform statistics pooling and the two-head vector-based attentive pooling achieves the best performance in almost all metrics, which suggests the proposed method is effective for text-independent SV. In terms of single-head attention whose parameters are less than multi-head attention, the vector-based pooling also results in better performance than other attentive methods in almost all aspects except for the EER of Development set. This result indicates that the vector-based pooling

Table 2: *EER(%) for different durations on SITW.* ***Boldface*** *values are the best results.*

Embedding(Attention heads)	**Development**				**Evaluation**			
	<15s	**15-25s**	**25-40s**	**>40s**	**<15s**	**15-25s**	**25-40s**	**>40s**
i-vector [1]	6.452	5.609	5.331	5.287	7.901	6.789	5.507	6.426
statistics pooling [3]	3.970	3.341	3.026	3.172	4.289	3.585	2.753	3.916
self multi-head attention [20]	3.722	3.222	2.882	3.172	4.063	3.127	2.863	4.217
attentive statistics pooling [18]	3.226	**2.745**	2.738	2.870	3.612	3.204	2.753	3.715
self-attentive pooling(1) [19]	3.722	**2.745**	2.882	2.870	4.063	3.432	2.753	4.217
self-attentive pooling(2) [19]	3.226	3.103	2.594	3.172	3.837	3.280	**2.533**	4.016
self-attentive pooling(5) [19]	4.218	3.103	2.882	2.870	3.386	3.204	**2.533**	3.614
vector-based attentive pooling(1)	**2.978**	2.983	2.594	2.719	3.386	3.127	**2.533**	**3.313**
vector-based attentive pooling(2)	3.226	2.864	**2.305**	**2.568**	**3.160**	**2.975**	2.643	3.514
vector-based attentive pooling(3)	3.226	3.222	2.594	3.323	3.612	3.432	**2.533**	3.514

Table 3: *Experimental results on VoxCeleb.* ***Boldface*** *values are the best results.*

Embedding	**EER(%)**	**DCF**10^{-2}	**DCF**10^{-3}
i-vector [1]	5.657	0.5016	0.6593
statistics [3]	2.556	0.3079	0.5582
self multi-head [20]	2.709	0.2804	**0.4032**
attentive statistics [18]	2.593	0.2947	0.4322
self-attentive(1) [19]	2.667	0.3002	0.4307
self-attentive(2) [19]	2.773	0.2940	0.4877
self-attentive(5) [19]	2.635	0.2887	0.4041
vector-based attentive(1)	2.582	0.2894	0.5126
vector-based attentive(2)	**2.466**	**0.2726**	0.4286
vector-based attentive(3)	2.641	0.3070	0.4107

makes the network to collect more discriminative information for speaker embeddings with a single attention head. In addition, the single-head vector-based attentive pooling has only 5.9M parameters, less than that of attentive statistics pooling, which reduces the computational cost of the network.

In order to evaluate the performance of pooling methods on different durations, we divide the test utterances in SITW dataset into four groups based on duration. The duration of four groups are shorter than 15s, between 15s and 25s, between 25s and 40s, and longer than 40s, respectively. Table 2 reports the EER results of all utterance groups. The single-head and two-head vector-based attentive pooling achieves the best EERs in almost all duration conditions except for 15-25s of the development set, which indicates that this method is robust to various durations. In addition, we observe that the performance of some utterance groups is lower for most methods, e.g. groups shorter than 15s in both two testsets and groups longer than 40s in SITW Evaluation testset, which suggests that speaker verification of very short and long utterances is more challenging. Our proposed vector-based attentive pooling achieves improvements over other pooling methods in these two duration groups, demonstrating the effectiveness of our method in both short and long utterances.

4.2. Results on Voxceleb

Table 3 shows the performance on VoxCeleb. The vector-based attentive pooling with two attention heads lead to the best EER as well as DCF10^{-2}. Surprisingly, most attentive pooling methods are not more effective than statistics pooling. Only two-head vector-based attentive pooling outperforms statistics pooling in terms of EER. The main reason may be that the short duration of test utterances in VoxCeleb fails to make the network emphasize important features. The experimental results suggest that attentive pooling methods have little impact on performance improvement in VoxCeleb.

4.3. Impact of multi-head attention

Different from self-attentive pooling which performs better with more attention heads, the vector-based attentive pooling achieves the best performance with two attention heads rather than three attention heads, which demonstrates the two-head vector-based attentive pooling can extract sufficient information to discriminate speakers. Apart from that, it can be seen in Table 1 that adding more attention heads results in substantial increase of the network parameters. When the number of attention heads is set for the best performance, the vector-based attentive pooling not only performs better but also reduces the number of attention heads as well as the network parameters compared with the self-attentive pooling.

5. Conclusion

In this paper, we propose a vector-based attentive pooling method for text-independent speaker verification, which introduces vectorial attention into the pooling layer of the network architecture. The vectorial attention can extract more discriminative information from the frame-level output. Besides, the vector-based attentive pooling is extended in a multi-head way so as to collect information from multiple aspects.

We evaluate the proposed method with the x-vector baseline system and compare it with statistics pooling and three state-of-the-art attentive pooling methods. Experiments conducted on VoxCeleb and SITW demonstrate the effectiveness of the proposed method. The vector-based attentive pooling achieves the best EER, DCF10^{-2} and DCF10^{-3} in SITW, and the best EER and DCF10^{-2} in VoxCeleb. In the future, we plan to incorporate the proposed method into more DNN-based network architectures and evaluate the effect of different configurations.

6. Acknowledgements

This work is supported by Science and Technology Planning Project of Tianjin, China (Grant No.18ZXZNGX00310), Tianjin Natural Science Foundation (Grant No. 17JCZDJC30700 and 19JCQNJC00300), and Fundamental Research Funds for the Central Universities of Nankai University (Grant No.6319140).

7. References

[1] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[2] S. Ioffe, “Probabilistic linear discriminant analysis,” in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Springer Berlin Heidelberg, 2006, pp. 531–542.

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

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

[5] G. Bhattacharya, J. Alam, and P. Kenny, “Deep speaker embeddings for short-duration speaker verification,” in *Proc. Interspeech 2017*, 2017, pp. 1517–1521.

[6] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, “Deep speaker feature learning for text-independent speaker verification,” in *Proc. Interspeech 2017*, 2017, pp. 1542–1546.

[7] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, “Deep speaker: an end-to-end neural speaker embedding system,” 2017.

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

[9] D. Michelsanti and Z. Tan, “Conditional generative adversarial networks for speech enhancement and noise-robust speaker verification,” in *Proc. Interspeech 2017*, 2017, pp. 2008–2012.

[10] F. Kreuk, Y. Adi, M. Cisse, and J. Keshet, “Fooling end-to-end speaker verification with adversarial examples,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 1962–1966.

[11] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, “Generalized end-to-end loss for speaker verification,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4879–4883.

[12] W. Cai, J. Chen, and M. Li, “Exploring the encoding layer and loss function in end-to-end speaker and language recognition system,” in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 74–81.

[13] J. S. Chung, A. Nagrani, E. Coto, W. Xie, M. McLaren, D. A. Reynolds, and A. Zisserman, “Voxsrc 2019: The first voxceleb speaker recognition challenge,” *ISCA Challenges*, 2019.

[14] 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)*, 2019, pp. 5796–5800.

[15] M. Mclaren, D. Castán, M. K. Nandwana, L. Ferrer, and E. Yilmaz, “How to train your speaker embeddings extractor,” in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 327–334.

[16] J. Villalba, N. Chen, D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, J. Borgstrom, F. Richardson, S. Shon, F. Grondin, R. Dehak, L. P. García-Perera, D. Povey, P. A. Torres-Carrasquillo, S. Khudanpur, and N. Dehak, “State-of-the-Art Speaker Recognition for Telephone and Video Speech: The JHU-MIT Submission for NIST SRE18,” in *Proc. Interspeech 2019*, 2019, pp. 1488–1492.

[17] S. Novoselov, A. Shulipa, I. Kremnev, A. Kozlov, and V. Shchemelinin, “On deep speaker embeddings for text-independent speaker recognition,” in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 378–385. [Online]. Available: http://dx.doi.org/10.21437/Odyssey.2018-53

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

[19] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey, “Self-attentive speaker embeddings for text-independent speaker verification,” in *Proc. Interspeech 2018*, 2018, pp. 3573–3577.

[20] M. India, P. Safari, and J. Hernando, “Self multi-head attention for speaker recognition,” in *Proc. Interspeech 2019*, 2019, pp. 4305–4309.

[21] Q. Chen, Z. Ling, and X. Zhu, “Enhancing sentence embedding with generalized pooling,” in *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, August 2018, pp. 1815–1826.

[22] A. Nagrani, J. S. Chung, and A. Zisserman, “Voxceleb: A large-scale speaker identification dataset,” in *Proc. Interspeech 2017*, 2017, pp. 2616–2620.

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

[24] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, “Voxceleb: Large-scale speaker verification in the wild,” *Computer Science and Language*, 2019.

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

[26] 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)*, 2018, pp. 1007–1013.

[27] C. Zhang, K. Koishida, and J. H. L. Hansen, “Text-independent speaker verification based on triplet convolutional neural network embeddings,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 9, pp. 1633–1644, 2018.

[28] Y. Jiang, Y. Song, I. McLoughlin, Z. Gao, and L.-R. Dai, “An Effective Deep Embedding Learning Architecture for Speaker Verification,” in *Proc. Interspeech 2019*, 2019, pp. 4040–4044.

[29] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, “The kaldi speech recognition toolkit,” in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011.

[30] Y. Liu, L. He, and J. Liu, “Large margin softmax loss for speaker verification,” in *Proc. Interspeech 2019*, 2019, pp. 2873–2877.