Serialized Multi-Layer Multi-Head Attention for Neural Speaker Embedding

Hongning Zhu[1,2]*, Kong Aik Lee*[3]*, Haizhou Li*[2]

[1]School of Computing, National University of Singapore, Singapore
[2]Department of Electrical and Computer Engineering, National University of Singapore, Singapore
[3]Institute for Infocomm Research, A*STAR, Singapore

hongning_zhu@u.nus.edu, lee_kong_aik@i2r.a-star.edu.sg, haizhou.li@nus.edu.sg

Abstract

This paper proposes a serialized multi-layer multi-head attention for neural speaker embedding in text-independent speaker verification. In prior works, frame-level features from one layer are aggregated to form an utterance-level representation. Inspired by the Transformer network, our proposed method utilizes the hierarchical architecture of stacked self-attention mechanisms to derive refined features that are more correlated with speakers. Serialized attention mechanism contains a stack of self-attention modules to create fixed-dimensional representations of speakers. Instead of utilizing multi-head attention in parallel, the proposed serialized multi-layer multi-head attention is designed to aggregate and propagate attentive statistics from one layer to the next in a serialized manner. In addition, we employ an input-aware query for each utterance with the statistics pooling. With more layers stacked, the neural network can learn more discriminative speaker embeddings. Experiment results on VoxCeleb1 dataset and SITW dataset show that our proposed method outperforms other baseline methods, including x-vectors and other x-vectors + conventional attentive pooling approaches by 9.7% in EER and 8.1% in DCF10^{-2}.

Index Terms: speaker embedding, speaker verification, attention mechanism, transformer

1. Introduction

Speaker verification aims to determine whether the speaker's identity of a test utterance is the same as the reference speech. In recent years, extracting speaker embeddings for speaker verification task has become the mainstream method. We have also witnessed a surge of interest in deep neural network (DNN) speaker embeddings [1]. These DNN-based embeddings achieve superior performance than i-vector [2] and has become state-of-the-art methods. The key of speaker embeddings is to find fixed-dimensional representations that can represent the voice characteristics of speakers.

Most deep neural networks for speaker embeddings consist of three components: (1) a DNN front-end for extracting frame-level features, (2) temporal aggregation of frame-level features to form utterance-level statistics, and (3) a speaker classifier followed by multi-class cross-entropy loss. The DNN front-end can be a time-delay neural network (TDNN) [3], convolutional neural network (CNN) [4, 5], recurrent neural network (RNN) [6, 7], or other neural network architectures [8, 9]. Pooling layers like global average pooling and statistics pooling are the most frequently used method for the temporal aggregation of frame-level features. It maps a variable-length sequence into a fixed-length representation. In x-vector embedding [3], its statistics pooling comprises the mean and standard deviation of frame-level features. A plain-vanilla statistics pooling assigns equal weights to each frame-level feature, which ignores the importance of some critical frames. Numerous recent works [10, 11, 12] have proposed attention-based techniques to address this problem. However, there is a common limitation in these prior methods that they only use one simple pooling layer to aggregate frame-level features to form a fixed-length vector.

In [13], it was shown that a stacked self-attention mechanism could achieve state-of-the-art results in natural language processing tasks [14, 15, 16]. In addition to the stacked self-attention at the encoder, the Transformer network [13] uses a multi-head topology, where attention functions are performed in parallel at each attention layer. However, the stacked self-attention modules compose a serialized architecture of the Transformer network, where it can learn from previous layers with attentive information. With a deeper utterance-level aggregation network, the speaker embeddings will have greater capacity and become more discriminative. We conjecture that, if multi-head topology is applied in a serialized manner, more layers can be stacked and capture more robust speaker characteristics.

In this paper, we present a method to extract speaker embedding, using a novel serialized multi-layer multi-head attention. We show that this method can capture representations that contain important information. With an input-aware query, the weighted statistics from different layers are calculated with self-attention, then fused into features of the next layer with temporal context. They are also aggregated in a serialized manner as the utterance-level speaker embedding for speaker verification. To the best of our knowledge, this study is the first to aggregate frame-level features into utterance-level embeddings with multi-layer attention network. We conduct experiments on VoxCeleb1 [17] dataset and Speakers in the Wild (SITW) [18] dataset. The experiments show the effectiveness over other conventional attentive pooling approaches.

The remainder of the paper is organized as follows. Section 2 reviews two conventional attentive pooling approaches. Section 3 describes the proposed serialized attention framework. The dataset and experimental setup are presented in Section 4. We discuss the results of these experiments in Section 5. The conclusion is finally drawn in Section 6.

2. Attention in Neural Speaker Embedding

Neural speaker embeddings are fixed-dimensional representations of speech utterances extracted with DNNs, among which x-vector [3] is the most widely used. In x-vector, temporal aggregation is used to convert frame-level features into a single fixed-dimensional vector. Then a fully-connected layer maps the utterance-level features to a speaker embedding. It is believed that certain frames are more unique and important for discriminating speakers than others. Instead of assigning equal weights to each frame, an attention mechanism is often applied

to obtain a set of weights as the importance of each frame over the input sequence.

2.1. Statistics pooling

Let $\mathbf{h}_t$ be the latent vector at the output of the frame processor network. By statistics pooling, we compute the mean and standard deviation of $\mathbf{h}_t$ along the temporal axis, $t = 1, 2, ..., T$. In particular, the first and second-order statistics are computed as follows:

$$\boldsymbol{\mu} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{h_t} \tag{1}$$

$$\boldsymbol{\sigma} = \frac{1}{T}\sqrt{\sum_{t=1}^{T}\mathbf{h_t}\odot\mathbf{h_t} - \boldsymbol{\mu}\odot\boldsymbol{\mu}} \tag{2}$$

where equal weights of $\alpha_t = \frac{1}{T}$ are assigned to all frames. The operator $\odot$ represents element-wise multiplication. The mean and standard deviation are concatenated as a fixed-dimensional representation and mapped to the speaker embedding vector via an affine transformation, typically, implemented with a fully-connected (FC) layer.

2.2. Attentive statistics pooling

Attentive statistics pooling [10] method aims to capture the temporal information focusing on the importance of frames. An attention model works in conjunction with the original embedding neural network and calculates a scalar score e_t for each frame, as follows:

$$e_t = \mathbf{v}^T f(\mathbf{W}\mathbf{h}_t + \mathbf{b}) + k \tag{3}$$

where $f(\cdot)$ is a non-linear activation function, such as tanh or ReLU. The scores are normalized over all frames with a softmax function as follows:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{\tau}^{T}\exp(e_\tau)} \tag{4}$$

The normalized scores are then used as the weights in the pooling layer to calculate a weighted mean $\tilde{\boldsymbol{\mu}}$ and a weighted standard deviation $\tilde{\boldsymbol{\sigma}}$:

$$\tilde{\boldsymbol{\mu}} = \sum_{t=1}^{T}\alpha_t\mathbf{h}_t \tag{5}$$

$$\tilde{\boldsymbol{\sigma}} = \sqrt{\sum_{t=1}^{T}\alpha_t\mathbf{h}_t\odot\mathbf{h}_t - \tilde{\boldsymbol{\mu}}\odot\tilde{\boldsymbol{\mu}}} \tag{6}$$

Compared to (1) and (2), the weights α_t in (5) and (6) are no longer uniformly distributed across the temporal axis. In this way, the utterance-level representation can focus on important frames that promotes a better speaker discrimination.

2.3. Self-attentive pooling

In [19], the authors used a more rigorous formulation based on a {value, key, query} tuple to construct the so-called self-attentive pooling mechanism.

Let $(\mathbf{v}_t, \mathbf{k}_t, \mathbf{q})$ be the {value, key, query} tuple. Here, $\mathbf{v}_t$ is the value vector with d_v dimensions, $\mathbf{q}$ is a time-invariant query with d_q dimensions, and $\mathbf{k}_t$ is the key vector with d_k dimensions. In [19], the $(\mathbf{v}_t, \mathbf{k}_t)$ is derived from different layers of the frame processor network, while $\mathbf{q}$ is a trainable parameter. The query vector maps the key vector sequence $[\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_T]$ to the weights $[\alpha_1, \alpha_2, ..., \alpha_T]$ via scaled dot-production attention and softmax function:

$$\alpha_t = \text{softmax}\left(\frac{\mathbf{q}\cdot\mathbf{k}_t}{\sqrt{d_k}}\right) \tag{7}$$

Note that the softmax is performed along the temporal axis. Finally, the weighted mean $\tilde{\boldsymbol{\mu}}$ and weighted standard deviation $\tilde{\boldsymbol{\sigma}}$ are computed in the same way as (5) and (6) with the weights α_t applied on the value vectors $\mathbf{v}_t$.

3. Serialized Multi-head Attention

In this section, we introduce the proposed serialized multi-layer multi-head attention mechanism. As depicted in Figure 1, the embedding neural network consists of three main stages, namely, a frame-level feature processor, a serialized attention mechanism, and a speaker classifier. The frame-level feature processor is the same as that in the x-vector [3], which uses a TDNN to extract high-level representations of the input acoustic features. The middle part of Figure 1, a serialized attention mechanism is used to aggregate the variable-length feature sequence into a fixed-dimensional representation. The top part of Figure 1 are feed-forward classification layers. Similar to the x-vector, the entire network is trained to classify input sequences into speaker classes.

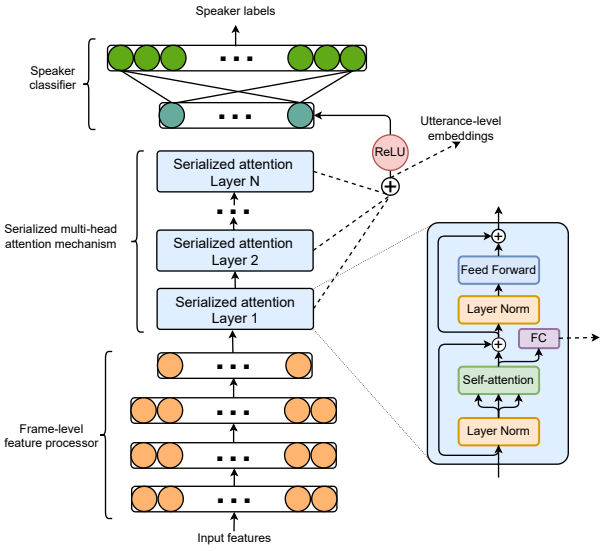

Figure 1: *Deep speaker embedding neural network with a serialized multi-head attention mechanism.*

3.1. Serialized attention

The serialized attention mechanism consists of a stack of N identical layers, and each layer is composed of two modules stacked together, i.e, a self-attention module and a feed forward module. We employ a residual connection around each of these modules. As in [20], layer normalization is applied on the input before the self-attention module and feed-forward module, separately. That is, the output of each sub-layer is $x + LayerNorm(Sublayer(x))$.

Instead of having multi-head attention in parallel, we propose to aggregate and propagate the information from one layer to the next in a serialized manner with stacked self-attention

modules. In the original multi-head attention, the input sequence is split into several homogeneous sub-vectors called heads. However, a deeper architecture of the aggregation network will increase the representational capacity, with more discriminative features can be learned and aggregated at different levels. In the proposed serialized attention mechanism, self-attention module is performed in a serialized manner, allowing the model to aggregate information with temporal context from deeper layers. Skip connection is used throughout self-attention module for training a much deeper network [21] and aggregating serialized heads. Specifically, from the n^{th} self-attention module ($n \in [1, ..., N]$), the weighted mean $\tilde{\boldsymbol{\mu}}$ and weighted standard deviation $\tilde{\boldsymbol{\sigma}}$ are obtained. After transformed by an affine transformation, it is converted to an utterance-level vector, also seen as a serialized head from layer n. The final utterance-level embedding is then obtained with the summation of the utterance-level vectors from all heads. After passing through a ReLU activation and Batch Normalization, it is then fed into classifier layers.

3.2. Input-aware self-attention

The attention function is mapping a query and a set of key-value pairs to an output [13]. Instead of using a fixed query for all utterances as in [19], we employ an input-aware unique query for each utterance. Considering that mean and standard deviation are capable of capturing the overall information and speech dynamics over an utterance, we use statistics pooling here to generate the query. As shown in Figure 2, consider an input sequence $[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_T]$ with $\mathbf{h}_t \in \mathbb{R}^d$, where T is the length of the input sequence. The model transforms the input sequence into the query $\mathbf{q}$ as follows:

$$\mathbf{q} = \mathbf{W}_q g(\mathbf{h}_t) \tag{8}$$

where $g(\cdot)$ is statistics pooling illustrated in Section 2.1, which is applied to calculate $[\boldsymbol{\mu}, \boldsymbol{\sigma}]$ with (1) and (2), and $\mathbf{W}_q \in \mathbb{R}^{d_k \times 2d}$ is a trainable parameter.

As for key-value pairs, in order to reduce the number of model parameters, the input sequence $[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_T]$ is directly assigned to the value sequence $[\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_T]$ of d dimensions without any extra computation. The key vector $\mathbf{k}_t$ is obtained by a linear projection with a trainable parameter $\mathbf{W}_k \in \mathbb{R}^{d_k \times d}$:

$$\mathbf{k}_t = \mathbf{W}_k \mathbf{h}_t \tag{9}$$

With $(\mathbf{v}_t, \mathbf{k}_t, \mathbf{q})$ as {value, key, query} tuple, the weights are computed via scaled dot-product attention as in (7). The first and second-order statistics are calculated the same as in (5) and (6). The weighted mean vector $\tilde{\boldsymbol{\mu}}$ is then added to all frames after an affine transformation in the residual connection.

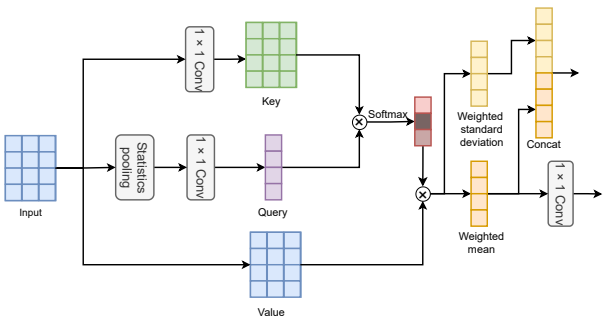

Figure 2: *Self-attention mechanism with input-aware query.*

3.3. Serialized multi-head embedding

After the self-attention module, the output from each self-attention layer is fed into a feed-forward module, which is to process the output to better fit the input for the next self-attention layer. The structure of the feed-forward module is the same as [13]. It consists of two linear transformations with a ReLU activation in between.

$$FFW(\mathbf{h}) = \mathbf{W}_2 f(\mathbf{W}_1 \mathbf{h} + \mathbf{b}_1) + \mathbf{b}_2 \tag{10}$$

where $\mathbf{h}$ is the input, $f(\cdot)$ is a ReLU function, and the linear transformations are different from layer to layer. $\mathbf{W}_1 \in \mathbb{R}^{d_{ff} \times d}$, $\mathbf{W}_2 \in \mathbb{R}^{d \times d_{ff}}$ with inner dimension d_{ff}, which can also be described as two convolutions with kernel size 1.

The utterance-level embedding of serialized attention mechanism is fed into one fully-connected layer and a standard softmax layer. The softmax layer with each of the nodes corresponds to the speaker labels in the training set. Hence, the system is trained as a speaker classifier. For training, we employ back-propagation with cross entropy loss. Once the system is trained, the utterance-level embedding is used as the serialized multi-head embedding for the speaker verification task.

4. Experiment

4.1. Dataset

The training set comprises the VoxCeleb2 [5] development dataset and it is an augmented version [3] with 5,994 speakers and 1,092,009 utterances.

Our experiment consists of four test sets from two different datasets: (1) SITW development core-core condition [18]; (2) SITW evaluation core-core condition; (3) the extended VoxCeleb1-E test set, which uses the entire Voxceleb1 [17] (both train and test splits), containing 1,251 speakers and 579,818 random pairs; (4) the hard VoxCeleb1-H test set, which consists of 550,894 pairs with the same nationality and gender, sampled from the entire VoxCeleb1 dataset. The utterances in these test sets vary in length from 4 seconds to 180 seconds. Hence, we could evaluate all models with both short and long utterances.

4.2. Experiments setup

The acoustic features used in all experiments are a combination of 23-dimensional Kaldi MFCC feature with a frame-length of 25ms and 3-dimensional pitch. The features are applied cepstral mean-normalization (CMN) with a sliding window of 3 seconds. A voice activity detection (VAD) is then used to remove silence frames.

To compare the proposed approach with other speaker embedding systems, we pick three baselines based on x-vector proposed in [3]: (i) statistics pooling as in the traditional x-vector. (ii) attentive statistics pooling proposed in [10]. (iii) self-attentive pooling [19], where the output of 4-th layer with 1-layer transformation network of 500 nodes is used as the keys. For frame-level layers in these x-vector systems, there are 512 channels in each of the first four TDNN layers, while 1500 in the fifth. ReLU is used as the non-linear activation for all the systems.

For the configuration of the proposed serialized attention mechanism, we use $d = 256$ and $d_k = 128$ in self-attention module, and the feed-forward dimension of $d_{ff} = 512$. To regularize the model, a dropout [22] of 0.1 is applied on the output of each module, before it is added to the input. For frame-level

Table 1: *Performance on SITW-dev and SITW-eval.* ***Boldface*** *denotes the best performance for each column.*

Embedding	Dim.	# params	SITW-dev			SITW-eval		
			EER(%)	DCF10^{-2}	DCF10^{-3}	EER(%)	DCF10^{-2}	DCF10^{-3}
statistic pooling [3]	512	4.47M	2.81	0.294	0.464	3.25	0.324	0.522
attentive statistic [10]	512	4.48M	2.54	0.276	0.462	3.17	0.304	0.491
self-attentive [19]	512	4.73M	2.77	0.288	0.447	3.09	0.305	**0.486**
ours ($N = 4$)	256	3.88M	2.54	0.283	0.478	3.09	0.314	0.515
ours ($N = 5$)	256	4.44M	2.27	0.286	0.479	2.98	0.308	0.519
ours ($N = 6$)	256	4.99M	**2.16**	**0.265**	**0.441**	**2.82**	**0.291**	0.499

Table 2: *Performance on VoxCeleb1-H and VoxCeleb1-E.* ***Boldface*** *denotes the best performance for each column.*

Embedding	Dim.	# params	VoxCeleb1-H			VocCeleb1-E		
			EER(%)	DCF10^{-2}	DCF10^{-3}	EER(%)	DCF10^{-2}	DCF10^{-3}
statistic pooling [3]	512	4.47M	4.50	0.425	0.660	2.57	0.283	0.497
attentive statistic [10]	512	4.48M	4.41	0.421	0.667	2.51	0.279	0.493
self-attentive [19]	512	4.73M	4.40	0.420	0.666	2.52	0.279	0.465
ours ($N = 4$)	256	3.88M	4.28	0.407	0.640	2.48	0.265	0.467
ours ($N = 5$)	256	4.44M	4.07	0.380	0.619	2.38	0.253	0.452
ours ($N = 6$)	256	4.99M	**3.99**	**0.375**	**0.616**	**2.36**	**0.242**	**0.431**

feature presscessor, the first three TDNN layers are the same as the x-vector system. The forth layer projects the dimension of frame-level features from 512 to 256 without non-linearity and Batch Normalization for a smaller number of parameters. The dimension of the first speaker classification layer is 256, which is equal to the speaker embedding dimension.

We use the ASV-Subtools [23] and the Kaldi toolkits [24] to implement the speaker embedding networks. During training, each utterance is chunked to a vector sequence of 200 frames, and the batch size is set to 512. We use the AdamW optimizer [25] with an initial learning rate of 10^{-3}, and decrease the learning rate every epoch for 9 epochs. After training the neural speaker embedding network, the Kaldi toolkit is used to train the PLDA model and the dataset used is the same as the training set. Before using PLDA, we use Linear Discriminant Analysis (LDA) to decrease the dimensionality to 128. Then the extracted embeddings are length-normalized and computed verification scores by the PLDA.

5. Results

In this section, we report the performance of serialized attention. We evaluate the experimental results in terms of Equal Error Rate (EER) and the minimum Decision Cost Function where prior target probability is set as 0.01 (DCF10^{-2}) and 0.001 (DCF10^{-3}).

5.1. Results on SITW

Table 1 presents the performance on SITW. Although there is a trade-off between performance and network size, we can obtain the best performance with $N = 6$ with 1.11M more parameters compared to $N = 4$. In comparison with baseline methods, the serialized attention mechanism achieves better performance on both development set and evaluation set, which improves by nearly 14.96% and 8.73% in EER on the dev and eval sets, respectively. Although self-attentive pooling offers the best performance of DCF10^{-3} on SITW-eval, our serialized attention achieves better performance in terms of EER and DCF10^{-2}.

5.2. Results on VoxCeleb1

Table 2 shows the performance on two VoxCeleb1 test sets. Here, the proposed method outperforms all the baseline methods by a significant margin with fewer or comparable parameters. The serialized attention with 6 layers has shown the best results of all the evaluated approaches. Comparing to statistics pooling, the 6-layer serialized attention achieves 11.33%, 11.76%, and 6.67%, relative improvements in terms of EER, DCF10^{-2} and DCF10^{-3}, respectively, in VoxCeleb1-H set. In VoxCeleb1-E set, relative improvements of 8.17%, 14.49%, 13.28% are also obtained in terms of EER, DCF10^{-2} and DCF10^{-3}, respectively. Besides, the 4-layer serialized multi-head attention is able to perform better than other single-layer attention-based pooling methods with 13% less parameters. It can be observed that stacking more layers will achieve further improvement. The results show that deeper serialized attention network makes the speaker embedding more discriminative.

6. Conclusion

In this paper, we proposes a new method to extract speaker embedding with a serialized multi-layer multi-head attention mechanism. Serialized attention mechanism contains a stack of self-attention modules to create fixed-dimensional representations for speaker verification. Unlike previous attention-based methods, weighted statistics from different layers are aggregated in a serialized manner. With an input-aware query, the proposed mechanism can capture more discriminative features. By increasing the stacked layers, consistent improvement is further obtained. Evaluating the proposed method on VoxCeleb1 and SITW dataset, our method outperforms other baseline methods by 9.7% in EER and 8.1% in DCF10^{-2}.

7. Acknowledgement

This research work is supported by Programmatic Grant No. A1687b0033 from the Singapore Government's Research, Innovation and Enterprise 2020 plan (Advanced Manufacturing and Engineering domain).

8. References

[1] K. A. Lee, O. Sadjadi, H. Li, and D. Reynolds, "Two decades into speaker recognition evaluation-are we there yet?" 2020.

[2] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[3] D. Snyder, 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.

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

[5] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.

[6] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[7] S. Novoselov, A. Shulipa, I. Kremnev, A. Kozlov, and V. Shchemelinin, "On deep speaker embeddings for text-independent speaker recognition," *arXiv preprint arXiv:1804.10080*, 2018.

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

[9] P. Safari, M. India, and J. Hernando, "Self-attention encoding and pooling for speaker recognition," *arXiv preprint arXiv:2008.01077*, 2020.

[10] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," *arXiv preprint arXiv:1803.10963*, 2018.

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

[12] M. India, P. Safari, and J. Hernando, "Self multi-head attention for speaker recognition," *arXiv preprint arXiv:1906.09890*, 2019.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[15] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860*, 2019.

[16] E. Strubell, P. Verga, D. Andor, D. Weiss, and A. McCallum, "Linguistically-informed self-attention for semantic role labeling," *arXiv preprint arXiv:1804.08199*, 2018.

[17] 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.

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

[19] Y. Liu, L. He, W. Liu, and J. Liu, "Exploring a unified attention-based pooling framework for speaker verification," in *2018 11th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 2018, pp. 200–204.

[20] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu, "On layer normalization in the transformer architecture," in *International Conference on Machine Learning*. PMLR, 2020, pp. 10 524–10 533.

[21] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[23] F. Tong, M. Zhao, J. Zhou, H. Lu, Z. Li, L. Li, and Q. Hong, "Asv-subtools: Open source toolkit for automatic speaker verification," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 6184–6188.

[24] 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.

[25] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.