Y-Vector: Multiscale Waveform Encoder for Speaker Embedding

Ge Zhu[1], *Fei Jiang*[1,2], *and Zhiyao Duan*[1]

[1]University of Rochester, Rochester, NY, USA
[2]Beijing Institute of Technology, Beijing, China
{ge.zhu, fei.jiang, zhiyao.duan}@rochester.edu

Abstract

State-of-the-art text-independent speaker verification systems typically use cepstral features or filter bank energies as speech features. Recent studies attempted to extract speaker embeddings directly from raw waveforms and have shown competitive results. In this paper, we propose a novel multi-scale waveform encoder that uses three convolution branches with different time scales to compute speech features from the waveform. These features are then processed by squeeze-and-excitation blocks, a multi-level feature aggregator, and a time delayed neural network (TDNN) to compute speaker embedding. We show that the proposed embeddings outperform existing raw-waveform-based speaker embeddings on speaker verification by a large margin. A further analysis of the learned filters shows that the multi-scale encoder attends to different frequency bands at its different scales while resulting in a more flat overall frequency response than any of the single-scale counterparts.

Index Terms: speaker verification, speaker embedding, raw waveform, multi-scale learning

1. Introduction

In recent years, the development of deep representations of speech utterances has made a breakthrough in speaker verification in terms of accuracy. Variani *et al.* [1] first trained a deep neural network (DNN) to extract utterance-level features (d-vector), achieving comparable performance to the previous state of the art, i-vector [2]. Since then, various deep embedding models have been proposed. Among them, x-vector [3] and its variants [4, 5, 6] are the most prominent, achieving state-of-the-art performance in many datasets and tasks [7], [8].

However, the above-mentioned DNN models are still built upon handcrafted feature inputs such as Mel-Frequency Cepstral Coefficients (MFCCs), which have long been used since Gaussian Mixture Model-Universal Background Models (GMM-UBM). Although MFCCs are designed based on human perceptual evidence, they are not necessarily optimal for speaker recognition tasks and could lose important information during the transform. Thanks to deep learning, there has been a trend on learning feature representations from raw data (e.g., time domain waveforms) to breakthrough the limit of feature engineering [9, 10].

Speaker verification research also witnessed an increased effort on developing time-domain deep neural network approaches. Taking raw waveforms as the input, a 1-d convolutional layer is usually applied as the first layer, where the set of filters behave like the Short-Time Fourier Transform (STFT), resulting in time-varying filter responses for future layers to process. In [11], Muckenhirn *et al.* first applied a Convolution Neural Network (CNN) based architecture for speaker verification and achieved competitive results to i-vector on Voxforge dataset. By analyzing the frequency response of the learned filters, they found that the first layer of the CNN was able to implicitly model the fundamental frequency (F_0). To efficiently learn meaningful filters, Ravanelli and Bengio proposed SincNet [12, 13] to constrain the free filters in the first convolutional layer with parameterized sinc functions. Jung *et al.* [14] later utilized this sinc-convolution layer with RawNet [15] and feature map scaling, and marginally outperformed the existing best spectrogram based system. In [16], Lin and Mak adapted the architecture in wav2vec [17] and achieved an equal error rate (EER) of 1.95% on the VoxCeleb1-O test set.

However, the set of filters in a convolutional layer typically has the same kernel size. This makes it difficult to learn high-frequency and low-frequency components simultaneously for wide-band signals. One idea is to split one convolution branch into several parallel branches with different scales, similar to InceptionNet [18] in computer vision. In this way, different groups of parameters for the convolution layer, including the number of filters, kernel size and stride size, can be independently determined, and filters at each scale can respond to different frequency components efficiently. Multi-scale convolutions have also been successfully used in acoustic modeling for speech recognition tasks from the raw waveform [19, 20]. This also motivates us to learn time-domain multi-scale representations for speaker verification.

In this paper, we present a new time-domain speaker embedding (Y-vector) based on a novel multi-scale waveform encoder. Compared to existing time-domain approaches [16], the proposed system uses a multi-scale waveform encoder to capture broadband responses. It also uses a time-frequency squeeze-excitation (tf-SE) attention module to re-calibrate the importance across time and frequency domains, and a TDNN for frame aggregation. Extensive experiments are conducted on the VoxCeleb1-O, VoxCeleb1-H and VoxCeleb1-E test sets. Results show that Y-vector outperforms existing time-domain speaker verification systems by a large margin. Further analysis shows that the multi-scale encoder responds to different frequency bands at its different scales, while resulting in a more flat overall frequency response than its single-scale counterparts.

2. Proposed System

The proposed Y-vector system is shown in Fig. 1. First, the multi-scale waveform encoder uses two filtering layers to take the same raw waveform input into multiple streams operating at different temporal resolutions. The filtered embeddings are then concatenated and then go through three (tf-SE) convolutional downsampling blocks. Finally, this representation is fed into a frame aggregator, implemented as a TDNN with additive margin softmax (AM-Softmax) to extract speaker embeddings using a speaker classification task. We now describe the details of each stage.

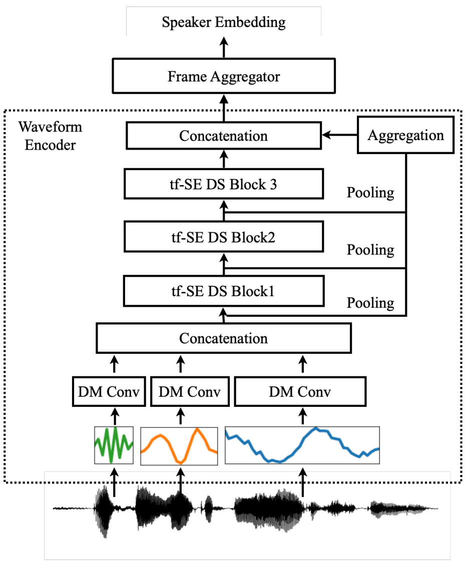

Figure 1: *Block diagram of our proposed Y-vector system. In the waveform encoder part, the three wave shaped curves demonstrate learned filters at different temporal resolutions. (DM: dimension match, DS: downsampling)*

2.1. Multi-scale Filtering Layer

Different from STFT that uses analytical Fourier basis as the filters, raw waveform encoders learn convolution filters to process the waveform. They need to use a small stride size, because a large size would lead to low temporal resolution and the loss of information. Therefore, in [19], Zhu *et al.* used a small stride convolutional layer followed by max pooling. Similarly, in wav2vec [17], Schneider *et al.* applied a five-convolutional-layer encoder with a series of relatively small strides of {5, 4, 2, 2, 2}, to decrease the time dimension gradually. With this strategy, the final sequence length of audio samples becomes 160 times smaller than the original input, while important information can passes through the layers more easily for learning good representations for speakers [16].
However, the above methods both use a single-scale convolution filterbank to process the waveform, which limits the frequency responses to the input speech signal. To be specific, short filters do not have the sufficient length to respond to low frequencies, while long filters can be inaccurate in modeling high frequencies as the signal might be non-stationary within the filter window.
In this work, we propose to extend the encoder into a multi-scale setting. To do so, we replace the first two convolution layers of wav2vec with three parallel branches at different scales, which are reflected by the filter size. Each branch consists of two layers of 1-d convolution, shown in the lower part of Fig.1. The first layer is used for primary filtering and the second dimension-match layer aims to compensate for the resulted output dimension differences. Therefore, the multiplication of the stride sizes is a constant across the three branches. It is also possible to use more than two layers, but in this paper, we only investigate the two-layer case.

2.2. tf-SE Downsampling Block

After concatenating the feature maps from different branches, we use three convolution blocks to further downsample them into a feasible size for the subsequent frame aggregator. The architecture of the downsampling block can be written as:

$$Y = tf\text{-SE}(\text{ReLU}(\text{Norm}(\text{Dropout}(\text{Conv}(X))), \qquad (1)$$

where $X, Y \in \mathcal{R}^{F \times T}$ are the input and output feature maps, respectively, F is the number of filters and T is the sequence length in time, tf-SE denotes for the temporal and frequency squeeze and excitation module. This idea is borrowed from the attention module for 2D signal processing [21, 22], where the importance of different channels of the embeddings is re-calibrated through squeeze and excitation networks. In our work, the re-calibration is performed in time and frequency dimensions instead, in a sequential manner. Specifically, we first aggregate the global information of the whole utterance from the input using average pooling along time, then re-scale the frequency dimension through:

$$X' = \sigma(\mathbf{W}_1 AvgPool_{1:T}(X) + \mathbf{b}_1) \odot X, \qquad (2)$$

where $\mathbf{W}_1$ is a matrix of size $F \times F$, $\mathbf{b}_1$ is a bias vector of size $F \times 1$, $\odot$ denotes for element-wise multiplication and σ denotes for sigmoid function. During time re-calibration, we borrow the idea of temporal gating in [16], re-scaling feature maps at every time frame t with a scalar factor. Then the resulting output feature at frame t can be written as:

$$Y_t = \sigma(\mathbf{W}_2(X'_t) + \mathbf{b}_2)) \odot X'_t, \qquad (3)$$

where $\mathbf{W}_2$ is a matrix of size $F \times 1$ and $\mathbf{b}_2$ is a bias scalar. By collecting the global "time" information in the first step, all of the "frequency" components will be re-calibrated with the excitation module. Similarly, the second step performs as a gate mechanism, all of the "time" frames will be re-weighted.

2.3. Multi-level Feature Map Aggregation

The original wav2vec only uses the feature map of the last layer for further processing. Although deeper layer features are usually more complex and contribute more to the final representations, Lee *et al.* [23] found that features extracted by early layers are also helpful through skip connections. Therefore, to further improve the accuracy, we concatenate feature maps at different layers at each time frame. To do so, max pooling is used to downsample earlier-layer feature maps to the same frame rate as that of the last layer.

3. Experiments

3.1. Dataset

In our experiment, we use the VoxCeleb corpora [24, 25] to train and evaluate our system and comparison systems for speaker verification. VoxCeleb is a free large-scale text-independent dataset collected from public multimedia data, where acoustic conditions are not controlled. Specifically, we employ the VoxCeleb2 development dataset for training, which contains 2,442 hours of recordings from 5,994 speakers. VoxCeleb1 is used for testing. It contains three test sets: (i) VoxCeleb1-O: the original verification test set consists of 37,611 random pairs from 40 speakers; (ii) VoxCeleb1-E: a list of 579,818 random pairs; (iii) VoxCeleb1-H: a list of 550,894 pairs with the same nationality and gender. Among these test sets, VoxCeleb1-E and VoxCeleb1-H cover all the 1251 speakers in VoxCeleb1.

For the evaluation metric, we compute Equal Error Rate (EER) and the minimum of the normalized detection cost function (minDCF) at $C_{miss} = C_{fa} = 1$ and $P_{Target} = 10^{-2}$ on these test sets to measure speaker verification accuracy. At test phase, we apply cosine score backend to all systems to measure the similarity between testing pair embeddings and calculate EER by adjusting the decision threshold.

3.2. Multiscale Architecture

The detailed multi-scale waveform encoder of Y-vector is shown in Table 1. Here we fix the ratio between stride size and kernel size to 0.5, which is equivalent to a 50% overlap ratio in STFT. In our study, we use a TDNN [3] as the frame aggregator[1].

Table 1: *System Y-vector-5. Numbers in brackets are convolution parameters: number of channels, kernel size and stride size, respectively.*

Group	Conv. Parameters		
	Branch 1	Branch 2	Branch 3
Multi-scale Filtering	$[90, 12, 6]$ $[160, 5, 3]$	$[90, 18, 9]$ $[160, 5, 2]$	$[90, 36, 18]$ $[192, 5, 1]$
Concatenation	-		
Downsampling	$[512, 5, 2]$ $[512, 3, 2]$ $[512, 3, 2]$		

For the ablation study in Section 3.4.2, we design several variants of the proposed system. Specifically, we compare systems with different number of channels in the first layer, as it can be viewed as the "frequency resolution" counterpart in STFT. We also compare systems with different total decimation rates of the first two layers, i.e., the multiplication of their stride sizes, as this indicates how fast the time dimension is reduced. We also investigate the effectiveness of multi-level aggregation and tf-SE components. Details are listed in Table 2.

Table 2: *Different multi-scale waveform encoder variants explored in ablation study.*

System	# of Channels	Decimation Rate	ML Aggreg.	tf-SE.
Y-vector-1	150	24		
Y-vector-2	150	24	✓	
Y-vector-3	270	24	✓	
Y-vector-4	270	18	✓	
Y-vector-5	270	18	✓	✓

3.3. Training Details

At the preprocessing stage, we simply normalize the raw waveform of each utterance by its maximum value. No voice activity detection (VAD) module is used. All of the recordings from VoxCeleb2 are used without filtering out speakers with short utterances, and we did not perform any data augmentation tricks either. For each utterance, we randomly crop 3.9s for batchifying to feed to the neural network.

For the TDNN frame aggregator, we empirically find that layer normalization works better than batch normalization in our system. We also apply L2 regularization on the last two fully connected layers combined with LeakyReLU activation functions with a negative slope of 0.2, following the method mentioned in [30]. As for the AM-Softmax loss function, the scale factor and margin are set to 30 and 0.35 respectively. For training, we use Stochastic Gradient Descent (SGD) with an initial learning rate of 0.01 and a momentum of 0.9. The learning rate decays by a factor of 0.5 for every 60 epochs. We train the system for 300 epochs, and in each epoch, we randomly sample 240,000 utterances from the whole training set. The batch size is set to 96.

[1]Code is available at https://github.com/gzhu06/Y-vector

3.4. Results

3.4.1. Comparison with Other Systems

In this section, we compare the proposed Y-vector with other recent speaker embedding networks using various features. The results are listed in Table 3. It can be seen that Y-vector significantly outperforms all other raw waveform-based systems and spectrogram-based systems on both VoxCeleb1-E and VoxCeleb1-H. It is noted that one comparison method, modified-wav2spk, is our implementation of wav2spk with the proposed multi-scale encoder using the same number of input channels; we also remove the original temproal gating because it already appears in tf-SE modules. Comparing with an MFCC-based system [27] with a similar backbone neural architecture, we can see that Y-vector also achieves better performance. One might argue that this difference might be due to the complexity of the 5-layer convolution waveform encoder in Y-vector instead of the benefit of raw waveform input. To verify this, we build another system ('3 CNN + x-vector') that takes MFCC as input, and feeds it to the last three convolution layers of the waveform encoder followed by the TDNN aggregator. We used a 3-layer CNN because it has been shown in [31] that standard mel-filterbanks can be approximated by two convolution operations. However, as can be seen in Table 3, the MFCC system still underperforms Y-vector systems significantly. It is worth to mention that this result does not suggest that the proposed waveform-based system outperforms the dominant spectrum-based systems in practice. More thorough investigations are needed on experimental settings, model architectures and optimization tricks are needed.

3.4.2. Ablation Study of the Y-vector Architecture

In this section, we compare five variants of the proposed Y-vector system listed in Table 2. From Y-vector-1 to Y-vector-2, we see improvement on both VoxCeleb1-E and VoxCeleb1-H, showing the effectiveness of multi-level feature aggregation. When increasing frequency resolution of filters (from Y-vector-2 to Y-vector-3) and decreasing total decimation rate (from Y-vector-3 to Y-vector-4), the performance both improves. Finally, when comparing Y-vector-4 with Y-vector-5, we see that although the tf-SE block does not improve the performance in VoxCeleb-E, it does improve on VoxCeleb-H, which is comprised of *harder* trials with more similar utterances in each trial.

3.4.3. Multi-scale Versus Single-scale

We also compare the multi-scale waveform encoder with three different single-scale encoders. We use a filter size of 10, 20, 40 respectively in the three single-scale encoders with each contains 96 channels. For the multi-scale encoder, we use the above three filter sizes in parallel branches to model high, middle, and low frequency information, respectively. For a fair comparison, we use only 32 filters in each branch so that the total number of filters is equal to that of the single-scale encoders, which means that the number of parameters is nearly the same.

The evaluation results of the four encoders are shown in Fig.2. We can see that the "low" encoder performs the

Table 3: *EER (%) comparison on different test sets. All models are trained on the VoxCeleb2 training set and scored with cosine similarity. A statistical significance test is performed using a bootstrap procedure [26]: Because Vox1-E and Vox1-H are much larger than Vox1-O, an absolute value of 0.05 of EER difference for Vox1-E and Vox1-H is already outside the 95% confidence interval for all methods, while for Vox1-O the EER difference has to be larger than 0.15. (∗: results copied from the references. †: our implementation. SP: statistical pooling.)*

Method	**Feature**	**Aggregation**	**Loss**	**VoxCeleb1-O**		**VoxCeleb1-E**		**VoxCeleb1-H**	
				EER	minDCF	EER	minDCF	EER	minDCF
Monteiro et al. [27]	MFCC	SP	E2E	2.51^*	-	2.53^*	-	4.69^*	-
3 CNN+x-vector†	MFCC	SP	AM-softmax	2.82	0.284	2.96	0.302	4.91	0.422
Xie et al. [28]	Spectrogram	GhostVLAD	Softmax	3.24^*	-	3.13^*	-	5.06^*	-
Nagrani et al. [29]		GhostVLAD	Softmax	2.87^*	0.310^*	2.95^*	-	4.93^*	-
RawNet2 [14]	Raw Waveform	GRU	Softmax	$\mathbf{2.48}^*$	-	2.87^*	-	4.89^*	-
wav2spk† [16]		Gating + SP	AM-softmax	3.00	0.281	2.78	0.280	4.56	0.390
modified wav2spk		SP	AM-softmax	2.69	0.278	2.62	0.261	4.28	0.371
Y-vector-1 (ours)		SP	AM-softmax	2.78	0.269	2.64	0.270	4.33	0.377
Y-vector-2 (ours)		SP	AM-softmax	2.77	0.270	2.50	0.263	4.17	0.376
Y-vector-3 (ours)		SP	AM-softmax	2.79	0.258	2.47	0.256	4.07	0.366
Y-vector-4 (ours)		SP	AM-softmax	2.60	0.239	2.39	0.248	4.00	0.354
Y-vector-5 (ours)		SP	AM-softmax	2.72	0.261	**2.38**	**0.241**	**3.87**	**0.339**

worst, "mid" and "high" encoders perform similarly with each other, while the "multi" encoder improves the EER over other systems on Vox1-E and Vox1-H slightly.

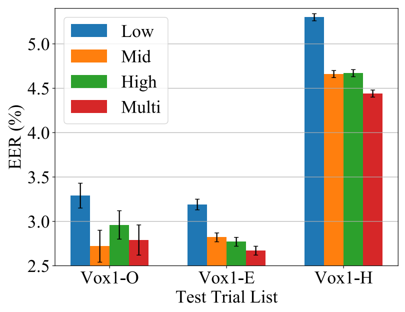

Figure 2: *Test set EER (%) comparing single- and multi-scale versions of the proposed system. Error bars show a 95% confidence interval.*

To further investigate the attributes of the multi-scale filters in the encoder, we compute the Cumulative Frequency Response (CFR) of the learned filters in the previous section as [9]:

$$CFR = \sum_{k=1}^{M} \frac{\mathcal{F}_k}{\|\mathcal{F}_k\|_2}, \tag{4}$$

where $\mathcal{F}_k$ is the magnitude frequency response of the filter f_k computed with a 256-point discrete Fourier transform, and M is the number of filters. The CFRs of the learned filters in the three single-scale and the multi-scale waveform encoders are shown in the top and bottom of Fig. 3, respectively. As shown in the top figure, the three single-scale encoders focus on different frequency bands but none of them is capable of modeling a wide frequency range. In contrast, the bottom figures shows that the multi-scale encoder has a much more flat overall CFR covering the entire frequency range with less than 5 dB fluctuations. This owes to the filters in the three parallel branches responding to different frequency bands. Note that the overall CFR has a peak between 400 Hz and 1000 Hz, suggesting that this frequency band plays a more important role in speaker recognition.

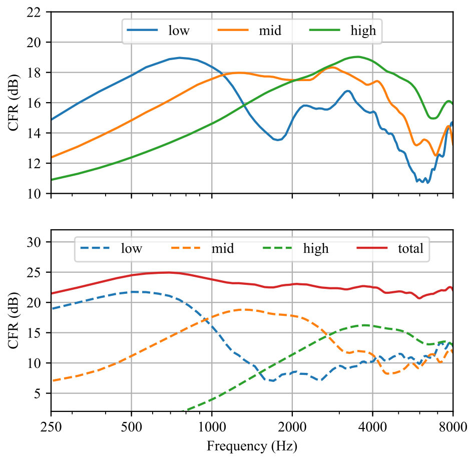

Figure 3: *Top: CFRs of the learned filters at different scales of the single-scale waveform encoder. Bottom: CFRs of the learned filters of the multi-scale waveform encoder and its single-scale branch.*

4. Conclusion

In this paper, we proposed a multi-scale raw waveform speaker embedding system and demonstrated its effectiveness in speaker verification. To be specific, the multi-scale waveform encoder uses three convolution branches with different time scales to compute speech features from the waveform, which are then processed by squeeze-and-excitation blocks and a multi-level feature aggregator. On speaker verification task, the proposed embedding significantly outperforms both time-domain and several MFCC-based speaker embedding systems, on both VoxCeleb1-H and VoxCeleb1-E. Future work includes the investigation of learnable filter bank architectures and cross-domain tasks especially under channel mismatch conditions.

5. Acknowledgements

This work was supported by National Science Foundation grant No. 1741472 and funding from Voice Biometrics Group.

6. References

[1] 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 Proceedings*. IEEE, 2014, pp. 4052–4056.

[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, 2011.

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors: robust DNN embeddings for speaker recognition," in *IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, 2018, pp. 5329–5333.

[4] D. Garcia-Romero, D. Snyder, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "x-Vector DNN Refinement with Full-Length Recordings for Speaker Recognition," in *Proc. Interspeech*, 2019, pp. 1493–1496.

[5] D. Garcia-Romero, G. Sell, and A. Mccree, "MagNetO: X-vector Magnitude Estimation Network plus Offset for Improved Speaker Recognition," in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, 2020, pp. 1–8.

[6] A. McCree, G. Sell, and D. Garcia-Romero, "Speaker Diarization Using Leave-One-Out Gaussian PLDA Clustering of DNN Embeddings," in *Proc. Interspeech*, 2019, pp. 381–385.

[7] A. Nagrani, J. S. Chung, J. Huh, A. Brown, E. Coto, W. Xie, M. McLaren, D. A. Reynolds, and A. Zisserman, "Voxsrc 2020: The second voxceleb speaker recognition challenge," 2020.

[8] S. O. Sadjadi, C. Greenberg, E. Singer, D. Reynolds, L. Mason, and J. Hernandez-Cordero, "The 2019 NIST Speaker Recognition Evaluation CTS Challenge," in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, 2020, pp. 266–272.

[9] D. Palaz, M. Magimai-Doss, and R. Collobert, "End-to-end acoustic modeling using convolutional neural networks for hmm-based automatic speech recognition," *Speech Communication*, vol. 108, pp. 15–32, 2019.

[10] H. Muckenhirn, V. Abrol, M. Magimai-Doss, and S. Marcel, "Understanding and Visualizing Raw Waveform-Based CNNs," in *Proc. Interspeech*, 2019, pp. 2345–2349.

[11] H. Muckenhirn, M. Magimai.-Doss, and S. Marcell, "Towards directly modeling raw speech signal for speaker verification using cnns," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 4884–4888.

[12] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," in *IEEE Spoken Language Technology Workshop*, 2018, pp. 1021–1028.

[13] M. Ravanelli and Y. Bengio, "Interpretable convolutional filters with sincnet," *arXiv preprint arXiv:1811.09725*, 2018.

[14] J. weon Jung, S. bin Kim, H. jin Shim, J. ho Kim, and H.-J. Yu, "Improved RawNet with Feature Map Scaling for Text-Independent Speaker Verification Using Raw Waveforms," in *Proc. Interspeech 2020*, 2020, pp. 1496–1500.

[15] J.-w. Jung, H.-s. Heo, j.-h. Kim, H.-j. Shim, and H.-j. Yu, "Rawnet: Advanced end-to-end deep neural network using raw waveforms for text-independent speaker verification," *Proc. Interspeech*, pp. 1268–1272, 2019.

[16] W. Lin and M.-W. Mak, "Wav2Spk: A Simple DNN Architecture for Learning Speaker Embeddings from Waveforms," in *Proc. Interspeech 2020*, 2020, pp. 3211–3215.

[17] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised Pre-Training for Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 3465–3469.

[18] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[19] Z. Zhu, J. Engel, and A. Y. Hannun, "Learning multiscale features directly from waveforms," in *Proc. Interspeech*, 2016.

[20] P. von Platen, C. Zhang, and P. Woodland, "Multi-span acoustic modelling using raw waveform signals," in *Proc. Interspeech*, 2019.

[21] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[22] S. Yadav and A. Rai, "Frequency and temporal convolutional attention for text-independent speaker recognition," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6794–6798.

[23] J. Lee and J. Nam, "Multi-level and multi-scale feature aggregation using pretrained convolutional neural networks for music auto-tagging," *IEEE Signal Processing Letters*, vol. 24, no. 8, pp. 1208–1212, 2017.

[24] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Proc. Interspeech*, 2018.

[25] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: a large-scale speaker identification dataset," in *Proc. Interspeech*, 2017.

[26] E. Haasnoot, A. Khodabakhsh, C. Zeinstra, L. Spreeuwers, and R. Veldhuis, "Feerci: A package for fast non-parametric confidence intervals for equal error rates in amortized o(m log n)," in *2018 International Conference of the Biometrics Special Interest Group (BIOSIG)*, 2018, pp. 1–5.

[27] J. Monteiro, I. Albuquerque, J. Alam, R. D. Hjelm, and T. Falk, "An end-to-end approach for the verification problem: learning the right distance," in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 13–18 Jul 2020, pp. 7022–7033.

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

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

[30] H. Zeinali, L. Burget, J. Rohdin, T. Stafylakis, and J. Cernocky, "How to improve your speaker embeddings extractor in generic toolkits," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 6141–6145.

[31] J. Andén and S. Mallat, "Deep scattering spectrum," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4114–4128, 2014.