Interspeech 2022
18-22 September 2022, Incheon, Korea

CS-CTCSCONV1D: Small footprint Speaker Verification with Channel Split Time-Channel-Time Separable 1-dimensional Convolution

Linjun Cai[1], *Yuhong Yang*[1,2,*], *Xufeng Chen*[1], *Weiping Tu*[1,2], *Hongyang Chen*[1]

[1]National Engineering Research Center for Multimedia Software, School of Computer Science, Wuhan University, Wuhan, China
[2]Hubei Key Laboratory of Multimedia and Network Communication Engineering, Wuhan, China

cailinjun, yangyuhong@whu.edu.cn

Abstract

We present an efficient small-footprint network for speaker verification. We start by introducing the bottleneck to the QuartzNet model. Then we proposed a Channel Split Time-Channel-Time Separable 1-dimensional Convolution (CS-CTCSConv1d) module, yielding stronger performance over the State-Of-The-Art small footprint speaker verification system. We apply knowledge distillation to further improve performance to learn better speaker embedding from the large model. We evaluate the proposed approach on Voxceleb dataset, obtaining better performances concerning the baseline method. The proposed model takes only 238.9K parameters to outperform the baseline system by 10% relatively in equal error rate (EER).

Index Terms: speaker verification, speaker embedding, small footprint, neural networks, knowledge distillation

1. Introduction

Automatic speaker verification (ASV) aims to verify a user's claimed identity given his or her speech segment. In the last few years, speaker verification systems utilizing deep neural network (DNN) have been tremendously successful. Many researches focus on overcoming the challenges in ASV task by designing different network backbones [1], different pooling functions [2, 3, 4], and loss functions[5, 6]. Over the years, DNN based speaker embedding learning has become the dominant method in this field, which is usually a time-delayed neural network (TDNN)[3, 7, 8, 9] or a convolutional neural network (CNN) [5, 10, 11]. These DNN-based methods improve the performances of ASV remarkably. However, these models comprise a large number of parameters and demand tremendous memory and computation resources. Speaker recognition systems usually work in small embedded devices where memory and computation resources are limited for the actual applications,. Therefore, a recent trend of deep neural network design of ASV systems is to explore efficient architectures to build small footprint low-latency models.

Some studies try to reduce the complexity of the ASV model by designing various models. For example, to reduce the computational complexity of the speaker verification model, Joon Son Chung et al. proposed Thin ResNet-34 and Fast ResNet-34, both with 1.4M parameters[5]. Thin ResNet-34 only uses one-quarter of the channels in each residual block compare with the original ResNet-34. Fast ResNet-34 reduces the input dimensions and prepose strides of 2 to cut down a half computation. Yoohwan Kwon et al. proposed performance optimized ResNet with 8M parameters by halving the number of channels[4]. This compression method is a natural approach. Koluguri, Nithin Rao, et al. proposed SpeakerNet with 5M parameters[12]. It comprises residual blocks with 1D depthwise separable convolutions, batch-normalization, and ReLU layers.

However, these model remains too large for devices like IoT terminals. Julien at al. has adopted the QuartzNet model to extract speaker embedding for embedded systems with only 237.5K parameters[13]. The basic repeated module consists of Time Channel Separable 1-dimensional Convolution (TCSConv1d) and max features map. Qingjian Lin et al. designed an asymmetric structure, where a large-scale ECAPA-TDNN model is applied for enrollment, and a small-scale model ECAPA-TDNNLite with 318K parameters extracts embedding during verification[14]. Without the large-scale ECAPA-TDNN model used to enroll, ECAPA-TDNNLite achieved similar performance but more training parameters to Julien's model. For these methods above, there are still some issues to be addressed. First, it is an intuitive method to introduce bottleneck to reduce the number of feature maps in the model. In addition, it is observed that a bottleneck of 1 achieved the best performance compared larger bottleneck ratio in [15]. Based on these observations, we introduce a novel module into QuartzNet, which can make a better trade-off between performance and the number of parameters.

On the other hand, some techniques can be adopted to improve the performance of the small-scale model. Wang, S et al. utilized knowledge distillation to narrow down the performance gap between large and small models with label-level and embedding-level loss functions[10]. It's observed that embedding-level knowledge distillation methods outperform the label-level one, which makes sense since the goal of ASV is not directly relevant to the predicted posteriors but corresponds to the speaker embedding instead.

Based on the above background and purpose, this paper presents a small footprint ASV model for embedded applications. The main contributions are as follows:

i) We present an improved efficient network. We propose the CS-CTCSConv1d module, where an efficient channel split operation followed by CTCSConv1d.

ii) We further design knowledge distillation methods to improve the performance of our proposed model. In this case, we extend our small-footprint model into a larger variant to serve as the teacher model.

iii) We demonstrate the efficacy of the proposal on the popular Voxceleb datasets. Experimental results with the proposed method get improvements compared with the state-of-the-art small-footprint model.

[*] corresponding author

10.21437/Interspeech.2022-913

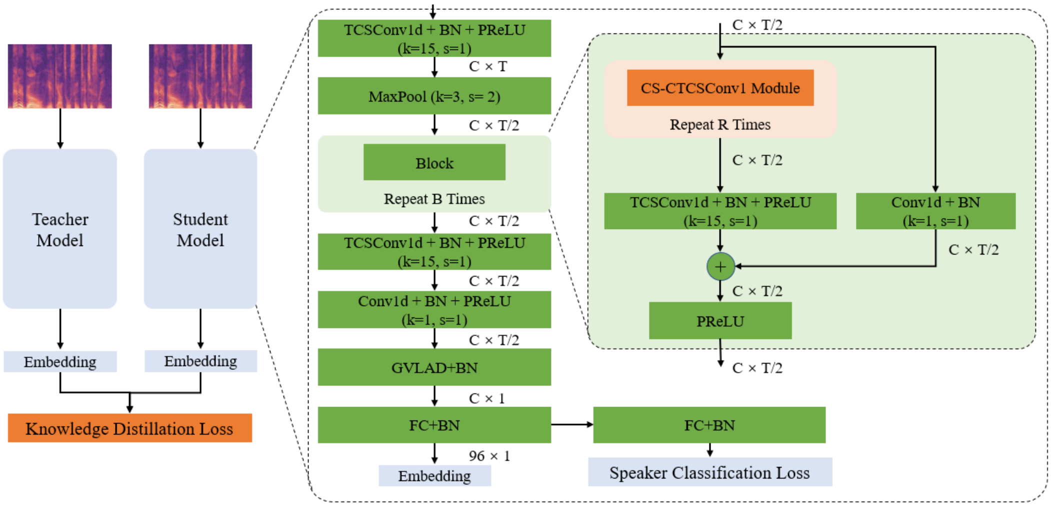


Figure 1: *The diagram for the proposed architecture, composed of proposed model and knowledge distillation.*

2. METHOD

2.1. Overall architecture

The diagram for our architecture is shown in Figure 1. The left part of the diagram is the knowledge distillation process, where the teacher model is only used in the training stage. The right part is the architecture of our network architecture composed of a QuartzNet-based embedding extractor and a Ghost Vector of Locally Aggregated Descriptors (GVLAD) pooling method.

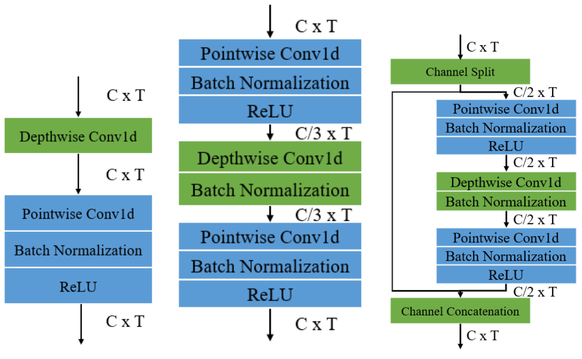
2.2. Design and motivation of CS-CTCSConv1D module

QuartzNet-based model consists of a series of block. The major computation of the architecture is from the repeated base module inside the block. So we focus on how to design the module. In the original QuartzNet, the module is TCSConv1d, as illustrated in Figure 2(a) containing four layers: 1) a depthwise convolution layer 2) a pointwise convolution layer 3) a normalization layer 4) a ReLU layer.

We first utilize the bottleneck structure introduced in the ResNet[16] by adding an additional pointwise convolution to the TCSConv1d. Although pointwise convolution is of quadratic time complexity ($\theta(C^2)$) with respect to the number (C) of channels, the total complexity will not increase by decreasing the number of the filters in the first pointwise convolution. The pointwise convolutional layer operates on each time frame independently but across all channels, while the depthwise convolutional layer with a kernel size of m operates on each channel individually but across m time frames. So we call the bottleneck as time-channel-time separable 1-dimensional convolution (CTCSConv1d), as shown in Figure 2(b). The depthwise convolution contributes a small portion of the computation in the module, so it allows kernels with much wider size of m. To keep the total computation cost comparable to the baseline system proposed by Julien et al., we use the bottleneck ratio of 3.

We further utilize a channel split operation to reduce input channels for the CTCSConv1d module. This enables the bottleneck ratio set to 1 with comparable computational complexity to CTCSConv1d module with a bottleneck ratio of 3. The proposed CS-CTCSConv1d module is illustrated in Figure 2(c).

Difference from Existing Methods. The proposed architecture has major differences from existing methods.



(a) TCSConv1d (b) CTCSConv1d (c) CS-CTCSConv1d

Figure 2: *An illustration of different modules.*

i) The module of block in Julien's model[13] first operates a TCSConv1d module without ReLU followed by a Max Features Map operation. TCSConv1d performs a 1-dimensional depthwise convolution that acts separately on channels, followed by a pointwise convolution that mixes channels. ii) Compared with the module in ShuffleNet v2[17], which utilizes two 1x1 convolutions, one depthwise convolution, channel split and channel shuffle operation, CS-CTCSConv1d operates on 1-D convolution and without channel shuffle operation. The whole ShuffleNet v2 network is built by stacking ShuffleNet v2 module repeatedly. Without channel shuffle operation, there is no information communication between the two branches. Nevertheless, in our proposed model shown in the right part of Figure 1, a TCSConv1d operates after the repeated CS-CTCSConv1d module, which provides the information communication between the two branches. To figure out the influence of the channel shuffle in the module, we experiment with a module by adding a channel shuffle operation.

Analysis on Complexities. Table 1 illustrates the complexities of the module mentioned before. Each module comprises depthwise convolution, pointwise convolution, batch normalization and ReLU. The pointwise convolution has much more complexity than other operations. By comparing TCSConv1d

Table 1: *Computational complexity comparison. The shape of the input is $C \times T$ where C is channels, and T is time frames. m is the kernel size of depthwise conv1d. Example MACs are calculated by C = 96, T = 200, m = 15.*

Module	Operator				Example MACs
	Depthwise Conv1d	Pointwise Conv1d	Batch Normalization	ReLU	
(a) TCSConv1d	$C \times T \times m$	$C \times C \times T \times 1$	$C \times T \times 2$	$C \times T$	2,169,792
(b) CTCSConv1d	$C/3 \times T \times m$	$2 \times C \times C/3 \times T \times 1$	$2 \times C/3 \times T \times 2 + C \times T \times 2$	$C/3 \times T + C \times T$	1,414,400
(c) CS-CTCSConv1d	$C/2 \times T \times m$	$2 \times C/2 \times C/2 \times T \times 1$	$3 \times C/2 \times T \times 2$	$2 \times C/2 \times T$	1,142,400
(d) Julien et al.	$C/2 \times T \times m$	$C/2 \times C \times T \times 1$	$C \times T \times 2$	–	1,104,000

and Julien at al., the max features map reduces computational costs by 49.1%. The additional computational costs of (c) CS-CTCSConv1d over Julien et al. are from the batch normalization and ReLU. Besides, the number of the input channels is C instead of C/2 for the first of the R repetitions in (c), producing more computational costs.

2.3. Knowledge distillation

We further utilize teacher-student learning to help the small-footprint model learn better speaker embedding from the well-trained teacher model. Input features are fed into the large-scale teacher model and the small-footprint student model respectively. We assume that given the same utterance, the embeddings extracted from each model are supposed to be the same. Thus, we define knowledge distillation to narrow down the cosine similarity between the speaker embeddings of the two models extracted from the same utterance:

$$\mathcal{L}_{kd} = -\sum_{i=1}^{B} \frac{\mathbf{e}_t^i \cdot \mathbf{e}_s^i}{\|\mathbf{e}_t^i\| \|\mathbf{e}_s^i\|}$$

where $\mathbf{e}_t^i$ represents the embedding computed by the teacher model for the i-th utterance in a mini-batch with batch size of B. $\mathbf{e}_s^i$ denotes the embedding computed by the student model.

We first pretrained the large model with speaker classification loss to get the well-trained teacher model. Then we apply speaker classification loss and knowledge distillation loss together to optimize the student model. The total loss is as follow:

$$\mathcal{L} = \mathcal{L}_{cl} + \lambda \mathcal{L}_{kd}$$

where $\mathcal{L}_{cl}$ is the speaker classification loss and λ is a hyperparameter for weighted summation.

3. Experimental setup

3.1. Dataset

Experiments are carried out on Voxceleb dataset[18, 19]. The network is trained on the development set of VoxCeleb2. We evaluated the model on the Voxceleb1 dataset.

We found that the data augmentation improves the large model's performance, but have an adverse effect for the smaller speed-optimized model in our experiments. The same conclusion is drawn in [4], so we only perform data augmentation on large models in the paper. We perform data augmentation with additive noise from MUSAN corpus[20] and the room impulse response (RIR) corpus[21].

3.2. Evaluation protocol

We report the performances of all systems in terms of equal error rates (EERs) and the minimum normalized detection cost (minDCf) in Voxceleb dataset. Scores are produced using the cosine similarity between embedding pairs in the trials. We set P_{target} = 0.01 and C_{FA} = C_{Miss} = 1 for minDCf. which are the same as the FFSVC [22] and NIST 2016 [23].

3.3. Training Details

Audio segments are converted into 64-dimensional MFCCs as the input feature computed from spectrograms using a 512 FFT size and a hamming window. We performed the short time Fourier transform on 25 ms with a 10 ms frame shift. We follow the setting in [13] with the number of the clusters K = 32, G = 3 for GVLAD and B = 5, R = 3 for QuartzNet. We set the number of channels C = 96 for our small-footprint model and C = 96 × 3 for large variant of our proposed model. The size of the output speaker embedding is 96. All models are trained using AAM-Softmax[24] and focal loss[25] as the speaker classification loss. The scaling and margin parameters of the AAM-Softmax loss function are respectively set to s = 30, m = 0.3. We set γ to 2 for focal loss and λ to 10 for knowledge distillation.

In the training process, each batch contains 128 speakers, each with one utterance of duration 2 to 5 in seconds. 2 to 5 means a random uniform sampling of each utterance from 2 to 5 seconds. Model parameters update through the Adam optimizer. We apply a weight decay of 0.0005 to prevent overfitting. A linear learning rate warm-up is employed in the first 25% epoch. The learning rate is initialized to zero, and linearly increases to 0.001. Then the learning rate is decreased by 50% every ten epochs. All models are trained for 100 epochs.

4. Results

4.1. Performance of CS-CTCSConv1d

Performance of different modules are shown in Table 2. We consider Julien's small footprint model as the baseline for the paper. To perform a fair comparison, we reproduce the model with the same experimental setup in Section 3. Our systems use a larger scale and a smaller margin in AAM-Softmax. Moreover, we apply a different learning rate schedule. As a result of these differences, the reproduced model trained with our implementation outperforms the results reported in [13]. The CTCSConv1d module with a bottleneck ratio of 3 affects the performance as expected due to the small input channels. Our proposed model using CS-CTCSConv1d system gives an average relative improvement of 6.0% in EER and 2.7% in minDCf over the baseline for each test set. Adding channel shuffle operation in our CS-CTCSConv1d module have very similar performance result, indicating that the channel shuffle has no contribution to the model. So we use model using CS-CTCSConv1d for the following experiments.

Table 2: *Performance of different modules using the number of the training parameters(# Params), Multiply-accumulate operations(MACs). MACs are measured for a 2-second input to the model. *This line is considered as the baseline system for our experiment.*

Model	# Params	MACs	VoxCeleb1-O		VoxCeleb1-E		VoxCeleb1-H	
			EER(%)	MinDCF	EER(%)	MinDCF	EER(%)	MinDCF
Julien at al.[13]	237.5K	-	3.31	-	-	-	-	-
Julien at al.(our impl)*	237.55K	22.9M	2.91	0.284	3.04	0.292	4.79	0.396
CTCSConv1d	258.915K	25.3M	3.31	0.344	3.41	0.337	3.41	0.33755
CS-CTCSConv1d	238.99K	23.2M	2.77	0.280	2.83	0.282	4.49	0.383
CS-CTCSConv1d + Channel Shuffle	238.99K	23.2M	2.78	0.249	2.76	0.279	4.53	0.381

Table 3: *Performance of knowledge distillation (KD). A. Small footprint model trained without KD. B. Large-scale teahcer model C. Small footprint model trained with KD.*

	Model	EER (%)		
		Vox-O	Vox-E	Vox-H
A	Ours	2.77	2.83	4.49
B.1	ECAPA-TDNN	1.14	1.32	2.46
B.2	Our large variant	2.07	2.12	3.50
C.1	Ours+KD (ECAPA-TDNN)	2.67	**2.76**	4.63
C.2	Ours+KD (Our large variant)	**2.62**	2.77	**4.44**

Table 4: *Comparison on inference speeds between the teacher and student models. Inference speeds are measured on two different CPUs with extracting embedding from 2-second utterances.*

Model	Inference Speed	
	Intel Xeon E5-2637	Raspberry Pi 4B
ECAPA-TDNN	88 ms ± 2.9ms	13.2 s ± 37.8 ms
Our large variant	122 ms ± 13.9 ms	1.9s ± 54.9 ms
CTCSConv1d	67.3 ms ± 2.5 ms	783.5 ms ± 7.3 ms
Julien (our impl)*	52.8 ms ± 5.0ms	551.1 ms ± 10.1 ms
CS-CTCSConv1d	46.4 ms ± 5.0 ms	573.1 ms ± 2.2 ms

4.2. Performance of knowledge distillation

A performance overview of the knowledge distillation described in Section 3 is given in Table 3.

We first train two large-scale teacher models that provide better performance but more computational costs. The first teacher model is ECAPA-TDNN, denoted as B.1. The large variant of our small footprint model is produced by multiplying the number of filters. When increasing the number of filters, the performance has no more marginal improvement until reaching three times the number of filters than the small footprint model. So we choose the model with channels $C = 288$ as the teacher model denoted as B.2. Although the ECAPA-TDNN outperforms our large variant, the performance of the small-footprint model using ECAPA-TDNN as teacher model has no margin improvement than using the large variant. A possible reason is that our large variant teacher model and student model has a very similar structure making it better learn the characteristics of the teacher network. With the knowledge distillation, our model further gets relative improvements of 5.4%, 2.1% in EER in Voxceleb1-O and VoxCeleb1-E. But the improvement in VoxCeleb1-H is limited for the more difficult test condition.

Table 5: *Comparison on the VoxCeleb-O trial with several previous approaches.*

Model	# Params	MACs	EER(%)	MinDCF
ECAPA-TDNN(our impl)	14.65M	2,685M	1.14	0.075
Our large variant	1.79M	177M	2.07	0.225
SpeakerNet-M[12]	5M	-	2.29	-
Thin ResNet-34[5]	1.4M	0.99G	2.36	-
Fast ResNet-34[5]	1.4M	0.45G	2.37	-
ECAPA-TDNNLite[14]	318K	-	3.00	0.292
Julien at al.[13]	237.5K	23M	3.31	-
Julien at al.(our impl)*	237.55K	22.9M	2.91	0.284
Ours	238.99K	23.2M	2.62	0.252

The inference speed are tested and compared in Table 4. It shows an interesting phenomenon that different models may have different performance results on different CPU models. ECAPA-TDNN outperforms our large variant model in Intel Xeon E5-2637, but on the other CPU, the opposite conclusion can be drawn. On embedded devices, the inference time is too long in the large models. Our model show competitive performance to the Julien's model.

A comparison of the previously proposed model and our proposed model is given in Table 5. Our model shows competitive performance among the small footprint models. Other models have lower EER, but require more parameters and expensive computational costs.

5. Conclusions

We proposed a novel small-footprint ASV model in this paper. It utilizes a novel module called CS-CTCSConv1d. We further explore the knowledge distillation to improve the performance of our proposed model. We extend our small footprint model into a larger model for knowledge distillation. The incorporation of CS-CTCSConv1d and knowledge distillation led to relative improvements of 10% in EER on average over the state-of-the-art small-footprint baseline systems on the VoxCeleb test sets.

6. Acknowledgements

This research is funded in part by the National Natural Science Foundation of China (62171326), Key Research and Development Program of Hubei Province (220171406), the Fundamental Research Funds for the Central Universities (2042021kf0039). The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

7. References

[1] 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.

[2] 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.

[3] B. Desplanques, J. Thienpondt, and K. Demuynck, “Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification,” in *INTERSPEECH*, 2020.

[4] Y. Kwon, H.-S. Heo, B.-J. Lee, and J. S. Chung, “The ins and outs of speaker recognition: lessons from voxsrc 2020,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5809–5813.

[5] J. S. Chung, J. Huh, S. Mun, M. Lee, H. S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, and I. Han, “In defence of metric learning for speaker recognition,” in *Interspeech*, 2020.

[6] X. Xiang, S. Wang, H. Huang, Y. Qian, and K. Yu, “Margin matters: Towards more discriminative deep neural network embeddings for speaker recognition,” in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 1652–1656.

[7] Y.-Q. Yu and W.-J. Li, “Densely connected time delay neural network for speaker verification.” in *INTERSPEECH*, 2020, pp. 921–925.

[8] 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.

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

[10] 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.

[11] Z. Chen, S. Wang, and Y. Qian, “Self-supervised learning based domain adaptation for robust speaker verification,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5834–5838.

[12] N. R. Koluguri, J. Li, V. Lavrukhin, and B. Ginsburg, “Speakernet: 1d depth-wise separable convolutional network for text-independent speaker recognition and verification,” *arXiv preprint arXiv:2010.12653*, 2020.

[13] J. Balian, R. Tavarone, M. Poumeyrol, and A. Coucke, “Small footprint text-independent speaker verification for embedded systems,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6179–6183.

[14] Q. Lin, L. Yang, X. Wang, X. Qin, J. Wang, and M. Li, “Towards lightweight applications: Asymmetric enroll-verify structure for speaker verification,” *arXiv preprint arXiv:2110.04438*, 2021.

[15] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, “Designing network design spaces,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 428–10 436.

[16] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[17] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, “Shufflenet v2: Practical guidelines for efficient cnn architecture design,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 116–131.

[18] J. S. Chung, A. Nagrani, and A. Zisserman, “Voxceleb2: Deep speaker recognition,” in *INTERSPEECH*, 2018.

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

[20] D. Snyder, G. Chen, and D. Povey, “Musan: A music, speech, and noise corpus,” *arXiv preprint arXiv:1510.08484*, 2015.

[21] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, “A study on data augmentation of reverberant speech for robust speech recognition,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5220–5224.

[22] X. Qin, M. Li, H. Bu, R. K. Das, W. Rao, S. Narayanan, and H. Li, “The ffsvc 2020 evaluation plan,” *arXiv preprint arXiv:2002.00387*, 2020.

[23] S. O. Sadjadi, T. Kheyrkhah, A. Tong, C. Greenberg, D. Reynolds, E. Singer, L. Mason, and J. Hernandez-Cordero, “The 2016 nist speaker recognition evaluation,” in *Proc. Interspeech 2017*, 2017, pp. 1353–1357. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2017-458

[24] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[25] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.