Depth-First Neural Architecture With Attentive Feature Fusion for Efficient Speaker Verification

Bei Liu, *Student Member, IEEE*, Zhengyang Chen, *Student Member, IEEE*, and Yanmin Qian, *Senior Member, IEEE*

***Abstract*—Deep speaker embedding learning based on neural networks has become the predominant approach in speaker verification (SV) currently. In prior studies, researchers have investigated various network architectures. However, rare works pay attention to the question of how to achieve a better trade-off on model performance and computational complexity. In this paper, we focus on efficient architecture design for speaker verification. Firstly, we systematically study the effect of the network depth and width on performance and empirically discover that *depth is more important than the width of networks for speaker verification task*. Based on this observation, we propose a novel depth-first (DF) architecture design rule. By applying it to ResNet and ECAPA-TDNN, two new families of much deeper models, namely DF-ResNets and DF-ECAPAs, are constructed. In addition, to further boost the performance of small models in the low computation regime, two novel attentive feature fusion (AFF) schemes, including sequential AFF (S-AFF) and parallel AFF (P-AFF), are proposed to dynamically fuse features in a learnable way. Experimental results on the VoxCeleb dataset show that the newly proposed DF-ResNets and DF-ECAPAs can achieve a much better trade-off on performance and complexity than the original ResNet and ECAPA-TDNN. Moreover, small models can further obtain up to 40% relative improvement in EER by adopting AFF scheme with negligible computational cost. Finally, a comprehensive comparison with various other published SV systems illustrates that our proposed models achieve the best trade-off on performance and complexity in both low and high computation scenarios.**

***Index Terms*—Attentive feature fusion, depth-first architecture, ECAPA-TDNN, ResNet, speaker verification.**

I. Introduction

SPEAKER verification (SV) is a task to verify the persons' claimed identities according to the biometric characteristics of their voices. Given enrollment and testing utterances, a SV system can automatically determine whether they belong to the same speaker or not. In general, two modules exist in a SV system. One is an embedding extractor which can extract speaker embeddings from utterances. The other measures similarity between the extracted embeddings. Traditionally, i-vector [1] combined with probabilistic linear discriminant analysis (PLDA) [2] is the widely-used approach. With the thriving of deep learning, neural networks have been highly applied in this task and achieved encouraging results [3]. Typically, DNN-based SV systems consist of a frame-level feature extractor, a segment-level embedding aggregator and a speaker classifier. Given an utterance, frame-level feature representation is firstly generated by neural network. Then a fixed-dimension speaker embedding is obtained through a pooling layer. Finally, a multi-class speaker classifier is adopted to train these systems. To further improve systems' performance and robustness, researchers have made great efforts in different aspects, including network backbones [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], pooling mechanisms [16], [17], [18], [19] and loss functions [20], [21], [22].

Concerning network backbones, diverse architectures have sprung up over the recent years, which can be roughly divided into four different types: time-delay neural network (TDNN) [5], [6], [7], [8], [10], [11], convolutional neural network (CNN) [9], Transformer [13], [14] and multi-layer perceptrons (MLP) [15]. With the ability to capture signals' temporal dynamics under wide context, TDNN is naturally suitable for speech processing [23]. [5] firstly introduces a TDNN system for text-independent speaker verification. x-vector [6] and its variants [7], [8] are proposed to further improve the performance. Subsequently, ECAPA-TDNN [11] provides impressive results by making several architectural enhancements to the original x-vector. Unlike TDNN-based systems, the winner [9] of VoxSRC-2019 proves that 2D convolutional neural network ResNet [24] also works surprisingly well for the SV task, which not only releases a strong baseline but also makes CNN popular in the SV field. In addition, a Transformer-based system is presented in [13] by adopting the self-attention encoder to extract speaker embeddings, which is motivated by Transformer's roaring success in other fields [25], [26], [27]. [14] further introduces the local information modeling into Transformer to improve the performance. Plus, [15] builds a pure MLP network without convolution or self-attention operation.

Although the existing models obtain promising results for SV task, neural networks are becoming larger and more complicated in order to pursue better performance. Big models are not only computationally unfriendly which require massive storage and computing resources, but also computationally inefficient due to the fact that the performance gains are very limited

Manuscript received 4 August 2022; revised 12 January 2023 and 20 April 2023; accepted 23 April 2023. Date of publication 5 May 2023; date of current version 15 May 2023. This work was supported in part by China NSFC projects under Grants 62122050 and 62071288, and in part by the Shanghai Municipal Science and Technology Commission Project under Grant 2021SHZDZX0102. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Zheng-Hua Tan. *(Corresponding author: Yanmin Qian.)*

The authors are with the X-LANCE Lab, Department of Computer Science and Engineering and MoE Key Laboratory of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: beiliu@sjtu.edu.cn; zhengyang.chen@sjtu.edu.cn; yanminqian@sjtu.edu.cn).

Digital Object Identifier 10.1109/TASLP.2023.3273417

when the number of parameters reaches a certain threshold. On the other hand, small models generally have an obvious performance gap with big models. How to achieve a good trade-off on model performance and computational complexity has been rarely discussed in the SV field. In fact, this is a crucial problem for SV applications. This paper explores efficient architecture design towards a better model performance and complexity trade-off for SV task in both low and high computation regimes.

Firstly, we systematically study the effect of: 1) depth and width of network; 2) scaling-up strategy on the SV system's performance. Prior works rarely discuss this question and the existing network scaling-up methods are mostly ad-hoc and heuristic. For example, ResNet-based SV systems consists of four computational stages each of which contains several residual blocks. Due to the memory limit, the number of channels in each block is directly reduced by half in [9]. On the contrary, [28] proposes a thin-ResNet trunk architecture by decreasing the block number of each stage. Additionally, [29] simply adopts the original ResNet in the experiments. Similar phenomenon exists in TDNN-based systems. [11] merely doubles the filter number in the convolutional layers to obtain a bigger model ECAPA(C=1024) based on ECAPA(C = 512). Subsequently, [30] further increases the channel number to 2048 and adds an extra layer in a brute-force way. In brief, how the depth and width of network affect SV system's performance is still not well understood. Besides, whether there exists a more principled scaling-up strategy for the SV task has not been fully explored. Based on our empirical analyses, it is observed that depth is more important than the width of networks for speaker verification task. Accordingly, we propose the depth-first (DF) architecture design rule through which new base models are constructed by significantly deepening ResNet and ECAPA-TDNN while decreasing or maintaining its complexity. Next, a special scaling-up strategy is introduced to yield two new families of much deeper models, namely DF-ResNets and DF-ECAPAs respectively.

Moreover, we introduce a novel attentive feature fusion (AFF) scheme to further boost small models' performance in low computation condition. It is widely admitted that small models are much easier for deployment while the performance gap with large models is significant. How to make small models have comparable performance to large ones is an urgent and challenging task for speaker verification. Previous studies [31], [32] attempts to bridge the gap via knowledge distillation. In this article, a light-weight attentive feature fusion module is proposed which can significantly improve small models' performance with negligible computational overhead. The existing feature fusion methods used in DNN-based SV systems are fix-weighted and non-learnable, which are not capable of modeling dynamic interactions among features. For example, element-wise addition between features is adopted in residual blocks of ResNet [9]. And ECAPA-TDNN utilizes concatenation operation in multi-feature aggregation layer. In contrast, our attentive feature fusion is designed to achieve dynamic fusion among features. It exploits attention modules to generate fusion weights based on the features' content in a learnable way. In particular, two different fusion strategies are proposed, including sequential AFF and parallel AFF.

More specifically, given the success of deep speaker embedding learning, this paper focuses on efficient architecture design for speaker verification with the purpose of achieving a better trade-off on model performance and complexity in both low and high computation scenarios, which is mostly ignored by previous studies. The main contributions of this work are summarized as follows:

1) The question of how the depth and width of networks affect SV systems' performance is systematically studied. Empirical results reveal that depth is more important than the width of networks for speaker verification task.
2) Based on the above observation, the depth-first (DF) architecture design rule is proposed. By applying it to ResNet and ECAPA-TDNN, two new base models are built. Next, a special scaling-up strategy is developed to yield two new families of much deeper models, named as DF-ResNets and DF-ECAPAs respectively.
3) To further boost small models' performance in low computation condition, A novel attentive feature fusion scheme is designed to replace conventional methods, which can make small models have comparable performance to large ones with negligible computational cost.
4) Finally, a comprehensive comparison with previous SV systems is presented, demonstrating that our proposed models achieve the best trade-off on performance and complexity in both low and high computation scenarios.

II. Review on Deep Speaker Embedding Learning

In recent years, deep speaker embedding learning has become the predominant approach for speaker verification. ResNet [24] and ECAPA-TDNN [11] are the two most widely used network backbones, providing the state-of-the-art performance. In this article, they serve as the baseline models.

A. ResNet

ResNet is firstly proposed by [24] for image recognition. In VoxSRC-2019, [9] introduces r-vector based on ResNet for speaker verification and wins the competition. Since then, it has become one of the most popular models in the SV field [29], [33], [34], [35], [36], [37], [38], [39].

Specifically, the input acoustic feature of ResNet-based SV system is a 3-dimensional tensor $1 \times F \times T$ where 1, F and T represent the channel, frequency and time dimension respectively. For SV task, Fbank or MFCC features are generally extracted from audio raw waveform. As Table II illustrates, the following is a 2-dimensional convolutional layer whose output is a $C \times F \times T$ feature map where C indicates the number of channels. Then, there exist four computational stages each of which contains several residual blocks. There are two different types of residual block: basic and bottleneck block, which consist of 1×1 or 3×3 convolutional operation, BatchNorm and non-linear function. After each stage, the number of channels will double while the frequency and time dimension will be reduced by half via setting stride as 2. Next, a statistical pooling

layer is adopted to map the variable-length frame-level features into a fixed-length embedding where mean and standard deviation are calculated along the time dimension and the results are concatenated together. Subsequently, a fully-connected layer is utilized to project the resulting vector into a low-dimensional speaker embedding. The network can be effectively trained by multi-class objective function.

B. ECAPA-TDNN

ECAPA-TDNN [11] is an enhanced TDNN variant of x-vector [6], which obtains promising results in various speaker verification competitions, including VoxSRC-2019 [11], VoxSRC-2020 [30], SdSVC-2021 [37], VoxSRC-2021 [40].

Different from ResNet, ECAPA-TDNN uses 1D convolutional layers. The input feature is a 2-dimensional representation $F \times T$, indicating the frequency and time dimension respectively. As Table IV shows, it is firstly processed by a 1D dilated convolutional layer. The output is a $C \times T$ feature map where C denotes the channel number. Then, the result is fed into three successive SE-Res2Blocks, each of which consists of two 1D dilated convolutional layers, one 1D dilated Res2Block [41] and one 1D squeeze-excitation (SE) block [42]. Besides, a multi-layer feature aggregation module is specially designed to integrate hierarchical speaker information residing in various network layers by concatenating the output feature maps from the above three SE-Res2Blocks. After that, the authors propose an attentive statistical pooling layer which can focus more on speaker-specific properties along the channel and time dimension through self-attention mechanism. Similar to ResNet-based SV systems, a dense layer is followed to reduce the dimension of the pooled vector. Finally, AAM-softmax [21] is used as the loss function for training.

III. Depth-First Architecture Design for Deep Speaker Embedding

In this section, we first investigate the effect of a network's depth and width on the SV system performance. Based on the empirical analyses, a novel depth-first (DF) architecture design rule and a special scaling-up strategy are proposed. By applying them to the current state-of-the-art system ResNet and ECAPA-TDNN, two new model families named as DF-ResNets and DF-ECAPAs, are designed respectively.

A. Investigating the Effect of Depth and Width

For a convolutional neural network, depth and width are the two essential factors to affect its performance [43], [44]. It is a common way to obtain a series of models with different performances under various resource constraints by scaling network's depth or width [45], [46], [47], [48], [49], [50]. However, as mentioned above, the current scaling methods in the SV field are mostly ad-hoc and heuristic. How the depth and width of a network affect SV system's performance is still not well understood. In particular, it is not yet clear which dimension plays a more important role in the SV task.

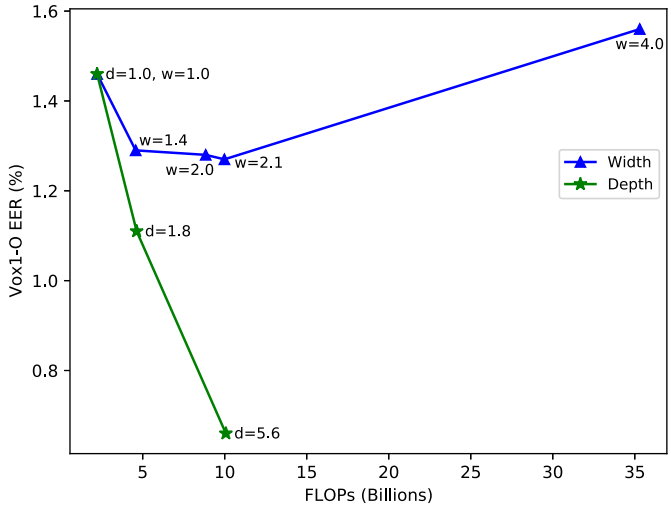


Fig. 1. Speaker verification performance comparison of scaling up ResNet18 with different network width (w) and depth (d) coefficients.

In this part, we systematically investigate the impact of depth and width of a network on SV system's performance based on ResNet. In the experiments, ResNet18, with the number of channels in each stage as [32, 64, 128, 256], is adopted as the base model. During scaling-up process, width coefficient w and depth coefficient d are the ratios to the channel and layer number of the base model respectively. By setting different (w, d) pairs, we obtain two series of models under different FLOPs. The specific scaling configurations are listed below:

- Base model: width=[32, 64, 128, 256], depth=18
- $w = 1.0$, $d = 1.8$: width = [32, 64, 128, 256], depth = 34
- $w = 1.0$, $d = 5.6$: width = [32, 64, 128, 256], depth = 101
- $w = 1.4$, $d = 1.0$: width = [46, 92, 184, 368], depth = 18
- $w = 2.0$, $d = 1.0$: width = [64, 128, 256, 512], depth = 18
- w=2.1, $d = 1.0$: width = [68, 136, 272, 544], depth = 18
- $w = 4.0$, $d = 1.0$: width = [128, 256, 512, 1024], depth = 18

From Fig. 1, it can be obviously seen that the Vox1-O EER saturates very quickly when widening the network with larger w. Specifically, the performance almost reaches the limit after $w = 1.4$. When further largening w, the FLOPs significantly increases while the result even becomes worse ($w = 4.0$ vs. $w = 1.0$). This illustrates that increasing the width of a network can not consistently boost the SV system's performance. On the contrary, continuous improvements can be achieved by deepening the base model from $d = 1.0$ to $d = 5.6$. Notably, the performance gains of deepening ResNet18 are much more significant than widening it under similar FLOPs. For example, compared to the base model, $w = 1.4$ and $d = 1.8$ result in the same FLOPs increase. However, the result of $d = 1.8$ is much better than $w = 1.4$. The performance gap between $w = 2.1$ and $d = 5.6$ is even getting larger, which reveals that increasing the depth is a more computationally efficient operation than increasing the width. All the above empirical analyses lead us to the following observation:

Observation - Depth is more important than the width of networks for speaker verification task.

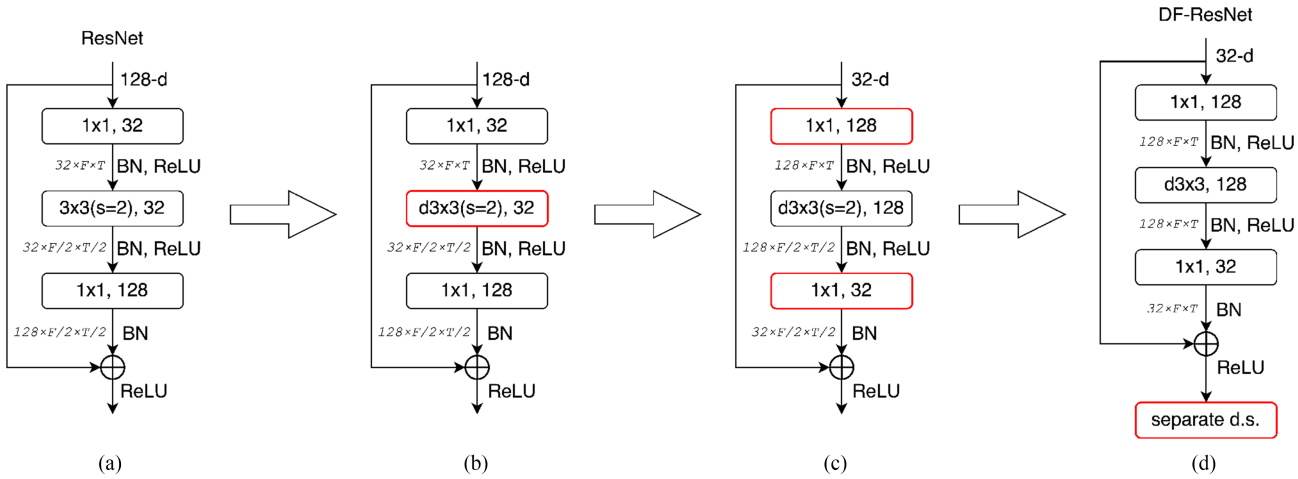


Fig. 2. (a)–(d) The roadmap from ResNet18 to DF-ResNet56. (a) The bottleneck block in the original ResNet. (b) Substitute the standard convolution with depthwise convolution. (c) Move down 1×1 convolution with 32 channels and move up 1x1 convolution with 128 channels. Also, change the channel number of depthwise convolution from 32 to 128. (d) A separate downsampling layer placed after the residual block. **separate d.s.** represents separate downsampling.

B. Depth-First Design Rule

Based on the above observation, we hypothesize that performance improvements can be achieved by largely increasing the depth of a network for speaker verification. Therefore, we propose a novel depth-first (DF) design rule and a special scaling-up strategy. Specifically, the depth-first rule implies that depth has a higher priority than width in our architecture design. It should be emphasized that this is not equal to increasing the number of layers in a network directly. The core idea behind the DF design lies in significantly deepening a network while maintaining its complexity. Speaking of a network's complexity, parameter number and FLOPs are measured. To achieve this goal, several design choices are carefully made. By applying the DF design rule to both ResNet and ECAPA-TDNN, two new base models are obtained. Subsequently, a special strategy is adopted to scale up the resulting base models. Consequently, two novel families of much deeper models, named DF-ResNets and DF-ECAPAs, are constructed respectively. The trajectories from ResNet and ECAPA-TDNN to DF-ResNets and DF-ECAPAs are presented in the following section.

C. Depth-First ResNets

In this section, we first describe the process of deepening ResNet18 into DF-ResNet56 (Depth-First ResNet56) while maintaining the model complexity according to depth-first design rule. Then, a new family of DF-ResNets is constructed by scaling up DF-ResNet56 in a specific way. Fig. 2 schematically presents the roadmap from ResNet18 to DF-ResNet56. Table I provides the corresponding changes of the number of parameters, FLOPs and performance during this process.

1) A Roadmap From ResNet18 to DF-ResNet56: The following are our specific design choices.

Basic block → bottleneck block: We start from a ResNet18 model. As Table II shows, it comprises of 4 stages, each of which has 2 basic blocks. The number of channels in each stage is [32, 64, 128, 256] respectively. Firstly, the basic block is substituted with the bottleneck block originating from [24] (Fig. 2(a)), which increases the number of layers from 18 to 26. Although the parameter number doubles and FLOPs are increased by 30%, the model's performance surprisingly becomes worse, as Table I indicates. This implies that the original bottleneck block in [24]

TABLE I
THE ROADMAP FROM RESNET18 TO DF-RESNET56 AND THE CORRESPONDING CHANGES OF PARAMETER NUMBER, FLOPS AND PERFORMANCE EER (%)

System	# Params	FLOPs	Vox1-O
ResNet18	4.11M	2.22G	1.48
basic block → bottleneck block	8.74M	2.93G	1.68
conv2d → depthwise conv2d	7.18M	1.75G	1.96
invert dimension	2.89M	1.94G	2.20
separate downsampling	3.14M	1.41G	1.65
increase layer number	4.49M	2.66G	0.96

TABLE II
DETAILED CONFIGURATION COMPARISON BETWEEN RESNET18 AND DF-RESNET56

Stage	ResNet18	DF-ResNet56
conv1	3×3, 32, stride 1	3×3, 32, stride 1
res2	$\begin{bmatrix} 3 \times 3, 32 \\ 3 \times 3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1, 128 \\ d3 \times 3, 128 \\ 1 \times 1, 32 \end{bmatrix} \times 3$
separate downsample	—	3×3, 64, stride 2
res3	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1, 256 \\ d3 \times 3, 256 \\ 1 \times 1, 64 \end{bmatrix} \times 3$
separate downsample	—	3×3, 128, stride 2
res4	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1, 512 \\ d3 \times 3, 512 \\ 1 \times 1, 128 \end{bmatrix} \times 9$
separate downsample	—	3×3, 256, stride 2
res5	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1, 1024 \\ d3 \times 3, 1024 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
pooling	Global Statistical Pooling	Global Statistical Pooling
FC	(5120, 256)	(5120, 256)
# params	4.11×10^6	4.49×10^6
FLOPs	2.22×10^9	2.66×10^9

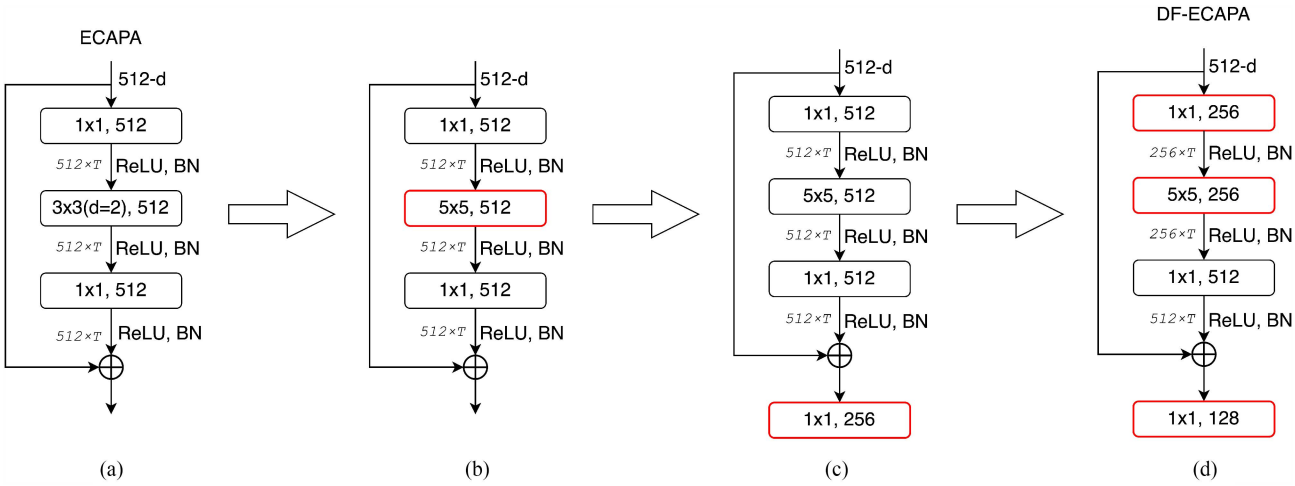


Fig. 3. (a)–(d) The roadmap from ECAPA(C = 512) to DF-ECAPA52. For simplicity, SE block is ignored in this figure. (a) The original SE-Res2Block in ECAPA(C = 512). (b) Replace 3×3 dilated convolution with 5×5 standard convolution. (c) Downsample the channel number by half. (d) Shrink the channel number of the first two layers in SE-Res2Block by half in DF-ECAPA244.

is not computationally efficient, which motivates us to re-design it.

conv2d → depthwise conv2d: The original bottleneck block adopts the traditional 2-dimensional convolution operation with 3×3 kernel size. Inspired by [47], we attempt to replace the traditional convolution with the depthwise convolution (Fig. 2(b)) in order to reduce FLOPs. It is a variant of grouped convolution which is widely used in lightweight models. We can see from Table I that this change can reduce FLOPs to 1.75 G along with a slight decrease in parameters. As a result, the performance EER further degrades to 1.96%.

Invert dimension: The previous step reduces FLOPs by 1.6x, however, the number of parameters is still large (7.18 M). We adopt the inverted block design in [48] to build an inverted bottleneck block by swapping the position of the first 1×1 convolution with 32 channels and the third 1×1 convolution with 128 channels in the original bottleneck block (Fig. 2(b) to (c)). Additionally, we further increase the channel number of depthwise convolution from 32 to 128. The above two choices can significantly decrease the parameter number from 7.18 M to 2.89 M at a slight cost of FLOPs. In the meanwhile, the EER temporarily reaches the highest point 2.20%.

Separate downsampling: The original ResNet performs the spatial downsampling at the beginning of each stage by directly setting the stride of convolutional layer as 2, as shown in Fig. 2(a). Instead, a separate downsampling layer [51] is employed to replace the traditional method. Specifically, this layer contains a standard 3×3 convolution equipped with stride 2 and a BatchNorm operator (Fig. 2(d)), which is placed at the end of each stage except for the last one to achieve the same resolution downsampling effect as Table II displays. This modification results in a slight increase in the number of parameters and a large decrease in FLOPs. At the same time, the EER is significantly reduced from 2.20% to 1.65%.

Increase layer number: After the above preparations, the parameter number and FLOPs are decreased by 25% and 37% so that we get some room to deepen the network. Following the stage compute ratio from [51], the number of blocks is increased from [2, 2, 2, 2] to [3, 3, 9, 3] for each stage, providing us the resulting model named as DF-ResNet56. This step leads to a significant reduction in EER from 1.65% to 0.96%. Compared to the original ResNet18, DF-ResNet56 achieves **35%** relative improvement on EER with a slight increase in parameter number and FLOPs. Table II presents the detailed comparison between ResNet18 and DF-ResNet56 in terms of architectural structure and computational complexity.

2) Construct a Family of DF-ResNets: In this part, we will construct a new model family according to a special scaling-up strategy. Specifically, in order to align with ResNet18/34/101, we increase the number of blocks B of DF-ResNet56 in a specific ratio for each stage. In the meanwhile, the channel number C of DF-ResNet56 stays the same. Consequently, a much deeper model family, DF-ResNet56/110/179/233, is built. The detailed configurations are listed below:

- DF-ResNet56: C = [32, 64, 128, 256], B = [3, 3, 9, 3]
- DF-ResNet110: C = [32, 64, 128, 256], B = [3, 3, 27, 3]
- DF-ResNet179: C = [32, 64, 128, 256], B = [3, 8, 45, 3]
- DF-ResNet233: C = [32, 64, 128, 256], B = [3, 8, 63, 3]

D. Depth-First ECAPAs

Similarly to DF-ResNets, we first present the details about how to increase the depth of ECAPA-TDNN without adding extra complexity in this part. Subsequently, a new family of DF-ECAPAs is constructed following a specific scaling-up method. Fig. 3 displays the process from ECAPA(C = 512) to DF-ECAPA52. Correspondingly, Table III reflects the fluctuations of parameter number, FLOPs and performance EER (%) in details.

1) A Roadmap From ECAPA(C = 512) to DF-ECAPA52: The specific design decisions is provided below.

Kernel size: Our starting point is an ECAPA(C = 512) model. We first investigate the behavior of different convolution kernel sizes and dilation spaces between the kernel points. The original SE-Res2Block in ECAPA(C = 512) adopts dilated convolution

TABLE III
THE ROADMAP FROM ECAPA(C = 512) TO DF-ECAPA52 AND THE CORRESPONDING CHANGES OF PARAMETER NUMBER, FLOPS AND PERFORMANCE EER (%)

System	# Params	FLOPs	Vox1-O
ECAPA(C=512)	6.39M	1.05G	0.97
kernel size 5×5	6.56M	1.08G	0.94
ResNet-ify	1.94M	0.34G	2.43
increase block number	5.64M	1.05G	0.83

TABLE IV
DETAILED CONFIGURATION COMPARISON BETWEEN ECAPA(C = 512) AND DF-ECAPA52. D INDICATES THE DILATION SPACE

Stage	ECAPA(C=512)	DF-ECAPA52
conv1	$5 \times 5, 512$	$5 \times 5, 512$
res2	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3 (d = 2), 512 \\ 1 \times 1, 512 \\ \text{Squeeze-Excitation} \end{bmatrix}$	$\begin{bmatrix} 1 \times 1, 512 \\ 5 \times 5, 512 \\ 1 \times 1, 512 \\ \text{Squeeze-Excitation} \end{bmatrix} \times 4$
res3	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3 (d = 3), 512 \\ 1 \times 1, 512 \\ \text{Squeeze-Excitation} \end{bmatrix}$	$\begin{bmatrix} 1 \times 1, 256 \\ 5 \times 5, 256 \\ 1 \times 1, 256 \\ \text{Squeeze-Excitation} \end{bmatrix} \times 8$
res4	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3 (d = 4), 512 \\ 1 \times 1, 512 \\ \text{Squeeze-Excitation} \end{bmatrix}$	$\begin{bmatrix} 1 \times 1, 128 \\ 5 \times 5, 128 \\ 1 \times 1, 128 \\ \text{Squeeze-Excitation} \end{bmatrix} \times 4$
aggregate	Multi-layer Feature Aggregation	Multi-layer Feature Aggregation
pooling	Attentive Statistical Pooling	Attentive Statistical Pooling
FC	(3072, 256)	(768, 256)
# params	6.39×10^6	5.64×10^6
FLOPs	1.05×10^9	1.05×10^9

to increase the receptive field (Fig. 3(a)). Although dilated convolution is a cheap operator with no increase in parameters, we claim that it sacrifices the power of modeling complex relationships. In fact, the larger receptive field can be achieved by simply increasing the kernel size. Therefore, we decide to remove the dilated convolution and adopt the standard convolution with large kernel size instead. In the experiments, several kernel sizes are examined including 3, 5 and 7. We notice that the performance becomes better with larger kernel but saturates at 7×7. Finally, we choose to stick with the standard 5×5 convolution (Fig. 3(b)), which has a slightly better performance under similar FLOPs than the original ECAPA(C = 512) as shown in Table III.

ResNet-ify: In the original ECAPA(C = 512), there exist three successive SE-Res2Blocks each of which outputs the feature map with the same shape $C \times T$ where C is set to 512, as Table IV illustrates. In order to reduce the parameter number and FLOPs, we follow the design paradigm of ResNet and downsample the channel number C by a factor of 2 after each block as shown in Fig. 3(c). Regarding the scale dimension s in SE-Res2Block, we follow the original configuration of ECAPA(C = 512) and set it to 8 for $C = 512$. When the channel number is downsampled by half, s is reduced by a factor of 2 accordingly. As a result, these design decisions significantly reduce the parameter number from 6.56 M to 1.94 M and FLOPs from 1.08 G to 0.34 G respectively. The performance EER degrades to 2.43%.

Increase block number: After the above steps, the parameter number and FLOPs are significantly reduced by 3.3x and 3.1x respectively. It is time to deepen the model by increasing the block number. Following ResNet, we introduce the stage idea to build three computational stages in total each of which contains several SE-Res2Blocks proposed in the previous step. In our design, the number of blocks in each stage is set to [4, 8, 4] respectively. This step significantly reduces the EER from 2.43% to 0.83%, yielding the final model DF-ECAPA52 which largely outperforms the original ECAPA(C = 512) by relative **15%** with similar parameters and FLOPs. The specific comparison between them is illustrated in Table IV.

2) Construct a Family of DF-ECAPAs: Based on the above DF-ECAPA52, we build a much deeper family of DF-ECAPAs to align with ECAPA(C = 512)/(C=1024) through a specific scaling-up strategy. For the width, we follow the channel expansion idea in bottleneck block of ResNet and shrink the channel number C in SE-Res2Block by a factor of 2 (Fig. 3(d)). Then the number of blocks B in each stage is increased to [16, 48, 16]. Finally, the corresponding DF-ECAPA52/244 are constructed. The configurations are summarized below:

- DF-ECAPA52: C = [512, 256, 128], B = [4, 8, 4]
- DF-ECAPA244: C = [256, 128, 64], B = [16, 48, 16]

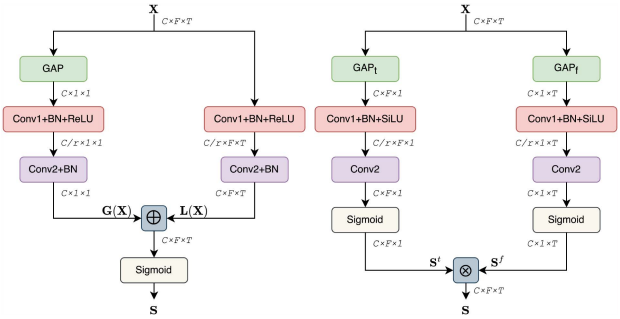
IV. ATTENTIVE FEATURE FUSION SCHEME

In the previous section, we build two new families of much deeper models, namely DF-ResNets and DF-ECAPAs, which can achieve a much better trade-off on performance and complexity than the original ResNet and ECAPA-TDNN in both low and high computation scenarios. Still, there exist significant performance gap between small and large models. In this section, we design a novel attentive feature fusion (AFF) scheme to further boost small models' performance in low computation condition with negligible computational cost. The details of the proposed method and its application in ResNet, ECAPA-TDNN, DF-ResNets and DF-ECAPAs are presented below.

A. Attention Modules

In the original ResNet and ECAPA-TDNN, feature fusion is simply implemented via element-wise addition or concatenation. Instead, we introduce attentive feature fusion scheme which can achieve dynamic fusion among different features by using attention modules to generate fusion weights based on the feature contents in a learnable way. It is worth emphasizing that the proposed AFF scheme is compatible with various attention modules such as SE [42], T-SE [52] and fwSE [37]. In the experiments, we explore two different attention mechanisms, including multi-scale channel attention (MS-CAM) [53] and coordinate attention (CA) [54]. Different from the existing methods, MS-CAM and CA aim to simultaneously capture different cross-dimension interactions in features to enhance representation ability. Taking 2D convolution as an example, Fig. 4(a) and (b) illustrate the detailed components of them.

MS-CAM: As Fig. 4(a) shows, MS-CAM consists of two branches to aggregate the local and global context information along the channel dimension respectively. Given a 3D feature $\mathbf{X} \in \mathbb{R}^{C \times F \times T}$ where C, F and T mean the channel, frequency and time dimension respectively, the local context



(a) 2D convolution MS-CAM (b) 2D convolution CA

Fig. 4. Attention modules.

$\mathbf{L}(\mathbf{X}) \in \mathbb{R}^{C\times F\times T}$ can be calculated through:

$$\mathbf{L}(\mathbf{X}) = \mathcal{B}(\text{Conv2}(\text{ReLU}(\mathcal{B}(\text{Conv1}(\mathbf{X}))))) \tag{1}$$

where Conv1 and Conv2 are point-wise convolution with the channel number of C/r and C respectively. r is the channel reduction ratio. $\mathcal{B}$ refers to BatchNorm. ReLU denotes rectified linear unit.

Similarly, we can obtain the global context $\mathbf{G}(\mathbf{X}) \in \mathbb{R}^{C\times 1\times 1}$ by:

$$\mathbf{G}(\mathbf{X}) = \mathcal{B}(\text{Conv2}(\text{ReLU}(\mathcal{B}(\text{Conv1}(\text{GAP}(\mathbf{X})))))) \tag{2}$$

where GAP stands for global average pooling.

Based on the above $\mathbf{L}(\mathbf{X})$ and $\mathbf{G}(\mathbf{X})$, we can get the attention map $\mathbf{S} \in \mathbb{R}^{C\times F\times T}$ through:

$$\mathbf{S} = \sigma(\mathbf{L}(\mathbf{X}) \oplus \mathbf{G}(\mathbf{X})) \tag{3}$$

where $\oplus$ represents the broadcasting addition. σ is the sigmoid function.

We utilize the resulting attention map $\mathbf{S}$ as the fusion weights for attentive feature fusion in Section IV-B.

CA: CA aims to encode direction-aware information into the generated attention maps, as shown in Fig. 4(b). Specifically, for a 3D feature $\mathbf{X} \in \mathbb{R}^{C\times F\times T}$, two separate attention maps are independently processed along the time and frequency dimension respectively. The time attention map $\mathbf{S}^t \in \mathbb{R}^{C\times F\times 1}$ can be generated as follows:

$$\mathbf{S}^t = \sigma(\text{Conv2}(\text{SiLU}(\mathcal{B}(\text{Conv1}(\text{GAP}_\text{t}(\mathbf{X})))))) \tag{4}$$

where GAP_t is average pooling along the time dimension. SiLU [55] denotes sigmoid-weighted linear unit.

Likewise, the frequency attention map $\mathbf{S}^f \in \mathbb{R}^{C\times 1\times T}$ can be obtained via:

$$\mathbf{S}^f = \sigma(\text{Conv2}(\text{SiLU}(\mathcal{B}(\text{Conv1}(\text{GAP}_\text{f}(\mathbf{X})))))) \tag{5}$$

where GAP_f is average pooling along the frequency dimension.

Finally, $\mathbf{S}^t$ and $\mathbf{S}^f$ are combined together to generate the final attention map $\mathbf{S} \in \mathbb{R}^{C\times F\times T}$ by:

$$\mathbf{S} = \mathbf{S}^t \otimes \mathbf{S}^f \tag{6}$$

where $\otimes$ means the broadcasting multiplication.

Also, the resulting attention map $\mathbf{S}$ can be used as the fusion weights in the following attentive feature fusion.

B. Attentive Feature Fusion

Inspired by [53], two different attentive feature fusion (AFF) schemes are proposed, including sequential AFF (S-AFF) and parallel AFF (P-AFF). Also, AFF can be divided into binary fusion and multiple fusion according to input feature number.

1) Binary Fusion: Binary fusion implies that there are two input features needed to be fused. The following part presents the sequential and parallel AFF strategies for binary fusion.

S-AFF: As illustrated in Fig. 5, the input features $\mathbf{X}, \mathbf{Y}$ are firstly element-wise added together. Then, the resulting feature is fed into MS-CAM or CA module to output the attention map $\mathbf{S}$ serving as fusion weights. After that, we re-scale the original $\mathbf{X}$ and $\mathbf{Y}$ by multiplying $\mathbf{S}$ and $1 \ominus \mathbf{S}$ respectively. Finally, the fused feature $\mathbf{Z}$ is obtained by adding the weighted features together. The whole calculation process of S-AFF can be summarized as:

$$\mathbf{S} = \text{MS-CAM/CA}(\mathbf{X} + \mathbf{Y}) \tag{7}$$

$$\mathbf{Z} = \mathbf{S} \otimes \mathbf{X} + (1 \ominus \mathbf{S}) \otimes \mathbf{Y} \tag{8}$$

where $\ominus$ represents the broadcasting subtraction. $\otimes$ is the element-wise multiplication.

P-AFF: Different from S-AFF, for two feature maps $\mathbf{X}, \mathbf{Y}$, P-AFF firstly feeds them into MS-CAM or CA module separately and generates the corresponding attention map $\mathbf{S}^\mathbf{X}$ and $\mathbf{S}^\mathbf{Y}$ in parallel. Next, the original $\mathbf{X}$ and $\mathbf{Y}$ are re-scaled by using the resulting attention map $\mathbf{S}^\mathbf{X}$ and $\mathbf{S}^\mathbf{Y}$ as fusion weights. The final fused feature $\mathbf{Z}$ is calculated as follows:

$$\mathbf{S}^\mathbf{X} = \text{MS-CAM}_1/\text{CA}_1(\mathbf{X}) \tag{9}$$

$$\mathbf{S}^\mathbf{Y} = \text{MS-CAM}_2/\text{CA}_2(\mathbf{Y}) \tag{10}$$

$$\mathbf{Z} = \mathbf{S}^\mathbf{X} \otimes \mathbf{X} \otimes (1 \ominus \mathbf{S}^\mathbf{Y}) + (1 \ominus \mathbf{S}^\mathbf{X}) \otimes \mathbf{Y} \otimes \mathbf{S}^\mathbf{Y} \tag{11}$$

2) Multiple Fusion: Multiple fusion means to fuse three or more features. The above mentioned sequential and parallel AFF strategies for binary fusion can be easily extended to multiple fusion by simply adding extra features. Taking three input features $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$ as an example, the specific multiple fusion processes of S-AFF and P-AFF are presented below.

S-AFF: Similar to (7), $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$ are firstly added together. Then, MS-CAM or CA module is utilized to yield the fusion weight $\mathbf{S}$. Subsequently, a learnable fusion parameter w_i, followed by the softmax function to generate I_i, is employed to learn the importance of each input feature. Finally, the re-scaled features by multiplying σ_i and $\mathbf{S}$ are added to obtain the fused feature $\mathbf{F}$.

$$\mathbf{S} = \text{MS-CAM/CA}(\mathbf{X} + \mathbf{Y} + \mathbf{Z}) \tag{12}$$

$$\mathbf{F} = I_1 \otimes \mathbf{S} \otimes \mathbf{X} + I_2 \otimes \mathbf{S} \otimes \mathbf{Y} + I_3 \otimes \mathbf{S} \otimes \mathbf{Z} \tag{13}$$

$$I_i = \frac{e^{w_i}}{\sum_j e^{w_j}}, i, j = 1, 2, 3 \tag{14}$$

where w_i is learnable fusion parameter which is normalized into σ_i through the softmax function ((14)).

P-AFF: Similar to binary case, $\mathbf{X}, \mathbf{Y}$ and $\mathbf{Z}$ are firstly fed into MS-CAM or CA module in parallel and the corresponding

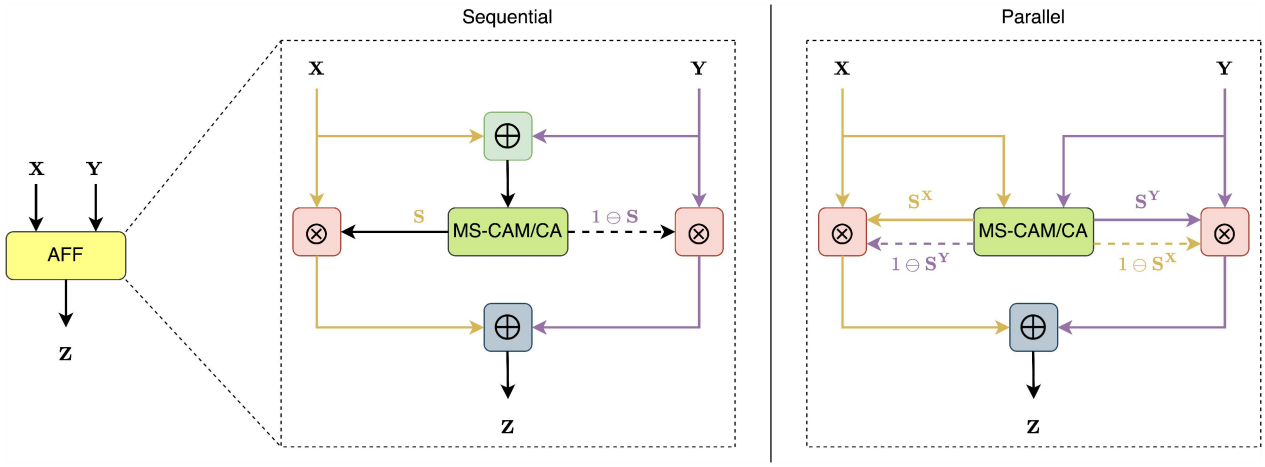


Fig. 5. Sequential and parallel AFF schemes for binary fusion. **Sequential:** add two features first. Then attention module takes the result to generate fusion weights. **Parallel:** process two features in parallel and generate fusion weights independently.

attention maps $\mathbf{S}^{\mathbf{X}}, \mathbf{S}^{\mathbf{Y}}$ and $\mathbf{S}^{\mathbf{Z}}$ are calculated independently. Next, we re-weight the original features separately to generate the fused feature $\mathbf{F}$ as (18) shows.

$$\mathbf{S}^{\mathbf{X}} = \text{MS-CAM}_1/\text{CA}_1(\mathbf{X}) \tag{15}$$

$$\mathbf{S}^{\mathbf{Y}} = \text{MS-CAM}_2/\text{CA}_2(\mathbf{Y}) \tag{16}$$

$$\mathbf{S}^{\mathbf{Z}} = \text{MS-CAM}_3/\text{CA}_3(\mathbf{Z}) \tag{17}$$

$$\begin{aligned}\mathbf{F} = {} & \mathbf{S}^{\mathbf{X}} \otimes \mathbf{X} \otimes (1 \ominus \mathbf{S}^{\mathbf{Y}}) \otimes (1 \ominus \mathbf{S}^{\mathbf{Z}}) \\ & + (1 \ominus \mathbf{S}^{\mathbf{X}}) \otimes \mathbf{S}^{\mathbf{Y}} \otimes \mathbf{Y} \otimes (1 \ominus \mathbf{S}^{\mathbf{Z}}) \\ & + (1 \ominus \mathbf{S}^{\mathbf{X}}) \otimes (1 \ominus \mathbf{S}^{\mathbf{Y}}) \otimes \mathbf{S}^{\mathbf{Z}} \otimes \mathbf{Z}\end{aligned} \tag{18}$$

C. Application

1) ResNet/DF-ResNets: In the residual block of ResNet and DF-ResNets (Fig. 2), there exists binary feature fusion where the element-wise addition between features is adopted. Our proposed AFF module can be easily integrated into ResNet and DF-ResNets by simply replacing the original element-wise addition in every residual block.

2) ECAPA-TDNN/DF-ECAPAs: For ECAPA-TDNN and DF-ECAPAs, element-wise addition in SE-Res2Block (Fig. 3) can be replaced by binary AFF module and concatenation in multi-layer feature aggregation (Table IV) can be substituted with multiple AFF module.

V. Experimental Setups

A. Dataset and Data Augmentation

Voxceleb1&2 [56], [57] are large-scale benchmark datasets for speaker identification and verification which contain over 6000 celebrities' interview audio data collected from YouTube videos. In the experiments, we evaluate the proposed methods by training on the Voxceleb2 dev set which consists of around 2,200 hours data including 1,092,009 utterances from 5,994 speakers. For testing, the three official released trial sets are adopted to measure performance. Specifically, 37,720 trials coming from 40 speakers are included in Vox1-O. 581,480 trials are sampled from 1251 speakers in Vox1-E. And Vox1-H has 552,536 trials with 1190 speakers. Meanwhile, to increase the diversity and richness of the training data, we utilize three data augmentation techniques which are listed as follows:

- Speed Perturb: As [58] states, speed perturbation can be adopted to diversify speakers. Sox is used to change the utterance speed by 0.9 or 1.1 time, which yields 3,276,027 training utterances from 17,982 speakers.
- Online Data Augmentation: According to [59], we add the noise from MUSAN and reverberation from RIR to utterances in an on-the-fly manner.
- SpecAugment: SpecAugment is first introduced in [60] and has been widely used in speech-related tasks. Following this method, we randomly mask the frequency and time dimension of the extracted acoustic features.

B. Implementation Details

All the proposed models are implemented using PyTorch framework. For the acoustic features, 80-dimensional Fbank is extracted from raw waveform with by setting window size as 25 ms and frame shift as 10 ms. During training, a 200-frame chunk is randomly cropped from one utterance. We adopt AAM-softmax [21] as the loss function with the configuration of 0.2 margin and 32 scale. The AdamW with 0.05 weight decay is utilized as the training optimizer. The total number of training epoch is set to 165 with the exponential scheduler as learning rate regulator.

C. Evaluation Metrics

For evaluation criterion, trial scores are calculated using cosine distance. For a pair of enrollment and testing speaker embedding $\boldsymbol{\eta}_e, \boldsymbol{\eta}_t$, cosine similarity is measured by (19) where $\langle \cdot, \cdot \rangle$ stands for the inner product between two embeddings, $\| \cdot \|$ is the magnitude of embeddings. Subsequently, adaptive score normalization (AS-Norm) [61] is utilized to normalize the resulting cosine scores. And we set the imposter cohort size as

TABLE V
RESULTS COMPARISON BETWEEN THE ORIGINAL RESNETS AND PROPOSED DF-RESNETS

System	# Params	FLOPs	EER (%)		
			Vox1-O	Vox1-E	Vox1-H
ResNet18	4.11M	2.22G	1.48	1.52	2.72
ResNet34	6.63M	4.63G	0.96	1.01	1.86
ResNet101	15.89M	10.07G	0.62	0.80	1.48
DF-ResNet56	4.49M	2.66G	0.96	1.09	1.99
DF-ResNet110	6.98M	5.15G	0.75	0.88	1.64
DF-ResNet179	9.84M	8.64G	0.62	0.80	1.51
DF-ResNet233	12.33M	11.17G	0.58	0.76	1.44

600. The equal error rate (EER) is used to report performance.

$$sim_{\cos}(\boldsymbol{\eta_e}, \boldsymbol{\eta_t}) = \frac{\langle \boldsymbol{\eta_e}, \boldsymbol{\eta_t} \rangle}{\|\boldsymbol{\eta_e}\| \|\boldsymbol{\eta_t}\|} \tag{19}$$

VI. RESULTS AND ANALYSIS

In this section, we first evaluate the proposed depth-first architecture design rule in Section VI-A. Then, the detailed results of attentive feature fusion scheme on small models in low computation condition are presented in Section VI-B. Section VI-C analyses the trade-off on model performance and complexity. In Section VI-D, we provide a comprehensive comparison between our proposed model families and various previous SV systems.

A. Evaluation on Depth-First Architecture Design

1) DF-ResNets: We first evaluate the performance of DF-ResNets proposed in Section III-C2. The original ResNet18/34/101 are implemented as the baselines.

From Table V, it can be observed that DF-ResNets achieve promising results in both low and high computation regimes. Take DF-ResNet56 as an example, 35%, 28% and 27% relative improvements are obtained on Vox1-O, Vox1-E and Vox1-H respectively with similar complexity compared to ResNet18. The same phenomenon can be seen for large models. This illustrates the effectiveness and superiority of our proposed depth-first architecture design rule. We can conclude that the deeper the model, the better the performance is under a fixed parameter and FLOPs constraint for speaker verification task. This is consistent with our empirical results in Section III-A. On the other hand, we can see from Table I that the newly-proposed computational block is more efficient than the bottleneck block in ResNet, which reflects that the series of design choices in Section III-C1 are effective and powerful. At the same time, DF-ResNets display the outstanding scalability in which performance can be consistently boosted with the increase in depth. It reveals that our special scaling-up strategy is simple and effective.

2) DF-ECAPAs: In this part, we make a comparative analysis between DF-ECAPAs introduced in Section III-D2 and the original ECAPA-TDNN.

In the original ECAPA-TDNN, the authors provide two different model configurations in terms of the channel number (C=512 or 1024). However, increasing the width of ECAPA-TDNN is not a computationally efficient choice. From Table VI, we can see that doubling the number of channels leads to the increase in parameters by 2.3x and FLOPs by 2.5x respectively, but the performance gains are not promising. In fact, there exists significant redundant computing in the original configuration. Instead, our design choices proposed in Section III-D1 focus on the depth of model. By shrinking the width, we successfully deepen ECAPA-TDNN while maintaining its complexity. Compared with ECAPA(C = 512), the resulting DF-ECAPA52 achieves the relative improvements in EER by 15%, 18%, 20% on Vox1-O, Vox1-E, Vox1-H respectively under similar complexity. Moreover, following the channel expansion idea, we propose bottleneck-like SE-Res2Block and further increase the number of blocks, through which DF-ECAPA52 can be easily scaled up to DF-ECAPA244. It outperforms ECAPA(C=1024) even with 25% fewer parameters and 26% fewer FLOPs. The above analyses reveal that our proposed design decisions tailored for ECAPA-TDNN are more efficient than the original ones. Meanwhile, it re-confirms our statement that depth is more important than width for the SV task.

TABLE VI
RESULTS COMPARISON BETWEEN THE ORIGINAL ECAPA-TDNNS AND PROPOSED DF-ECAPAS

System	# Params	FLOPs	EER (%)		
			Vox1-O	Vox1-E	Vox1-H
ECAPA(C=512)	6.39M	1.05G	0.97	1.22	2.31
ECAPA(C=1024)	14.85M	2.67G	0.81	1.01	2.04
DF-ECAPA52	5.64M	1.05G	0.83	1.01	1.87
DF-ECAPA244	11.03M	1.98G	0.71	0.89	1.72

B. Evaluation on Attentive Feature Fusion Scheme

1) ResNet/18/34/DF-ResNet56: To examine the effect of attentive feature fusion scheme on small models for low computation condition, we first apply it to ResNet18, 34 and DF-ResNet56. As stated in Section IV-C1, the element-wise addition in residual block can be replaced by our proposed AFF module. Specifically, we implement both sequential and parallel AFF based on MS-CAM and CA respectively, which provides four different configurations: S-AFF(MS-CAM), S-AFF(CA), P-AFF(MS-CAM) and P-AFF(CA).

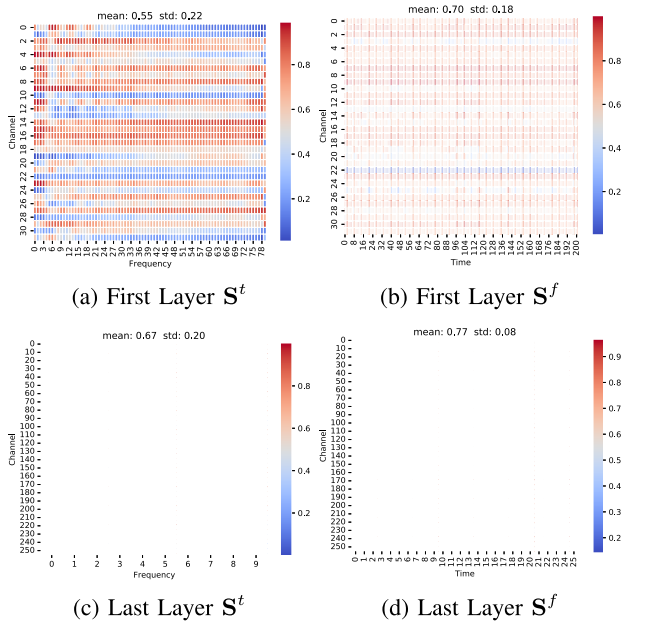
From Table VII, we can see that both S-AFF and P-AFF can bring great performance improvements with negligible computational overhead compared to the baselines for both ResNet and DF-ResNet. For S-AFF, MS-CAM based and CA based modules are both light-weight and powerful. The advantage of attentive feature fusion originates from the ability of dynamically learning and generating fusion weights according to the features' contents. Compared to the conventional fix-weighted fusion methods, AFF scheme enjoys the benefit of focusing more on speaker-related information in intermediate features. Interestingly, CA based S-AFF can perform much better with fewer parameters and FLOPs compared to MS-CAM based S-AFF. This phenomenon can be attributed to the characteristic of CA where frequency and temporal information in features are independently modeled. Prior studies [37], [62] have unveiled

TABLE VII
RESULTS OF THE PROPOSED AFF SCHEME ON RESNET18, 34 AND DF-RESNET56. **S-AFF**: SEQUENTIAL AFF. **P-AFF**: PARALLEL AFF

System	# Params	FLOPs	EER (%)		
			Vox1-O	Vox1-E	Vox1-H
ResNet18	4.11M	2.22G	1.48	1.52	2.72
+S-AFF(MS-CAM)	+0.18M	+0.07G	1.29	1.36	2.49
+S-AFF(CA)	+0.13M	+0.01G	0.93	1.05	1.94
+P-AFF(MS-CAM)	+0.36M	+0.15G	1.19	1.29	2.37
+P-AFF(CA)	+0.26M	+0.01G	0.86	0.99	1.82
ResNet34	6.63M	4.63G	0.96	1.01	1.86
+S-AFF(MS-CAM)	+0.33M	+0.15G	0.79	0.89	1.71
+S-AFF(CA)	+0.24M	+0.01G	0.65	0.82	1.59
+P-AFF(MS-CAM)	+0.66M	+0.30G	0.75	0.84	1.68
+P-AFF(CA)	+0.48M	+0.03G	0.62	0.79	1.57
DF-ResNet56	4.49M	2.66G	0.96	1.09	1.99
+S-AFF(MS-CAM)	+0.38M	+0.17G	0.82	0.96	1.82
+S-AFF(CA)	+0.28M	+0.02G	0.73	0.89	1.71
+P-AFF(MS-CAM)	+0.76M	+0.34G	0.79	0.93	1.79
+P-AFF(CA)	+0.56M	+0.03G	0.71	0.86	1.65

TABLE VIII
RESULTS OF THE PROPOSED AFF SCHEME ON ECAPA(C = 512) AND DF-ECAPA52. "+AFF" STANDS FOR S-AFF FOR BINARY FUSION IN RESIDUAL BLOCK AND P-AFF FOR MULTIPLE FUSION IN MULTI-LAYER FEATURE AGGREGATION MODULE

System	# Params	FLOPs	EER (%)		
			Vox1-O	Vox1-E	Vox1-H
ECAPA(C=512)	6.39M	1.05G	0.97	1.22	2.31
+AFF	+0.02M	-0.16G	0.92	1.10	2.11
DF-ECAPA52	5.64M	1.05G	0.83	1.01	1.87
+AFF	+1.63M	+0.16G	0.76	0.95	1.81



(a) First Layer $\mathbf{S}^t$ (b) First Layer $\mathbf{S}^f$

(c) Last Layer $\mathbf{S}^t$ (d) Last Layer $\mathbf{S}^f$

Fig. 6. Visualization of the learned fusion weights in proposed AFF module.

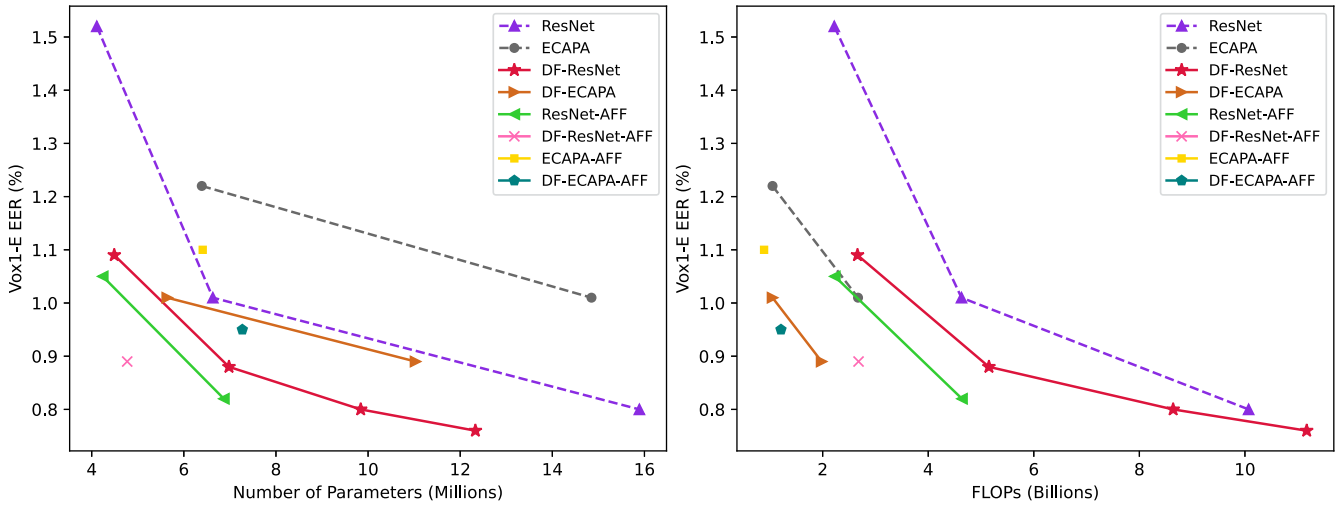
the importance of processing the spectrogram's frequency and temporal dimension separately rather than regarding them as a whole for the SV task. On the other hand, P-AFF can lead to the double increase in parameters and FLOPs than S-AFF. Accordingly, better performance can be achieved by P-AFF. In addition, similar trend exists between MS-CAM based and CA based P-AFF modules. The above analyses demonstrate the effectiveness and superiority of our proposed AFF scheme over the conventional fusion methods.

2) ECAPA(C = 512)/DF-ECAPA52: Then, AFF scheme is applied to ECAPA(C = 512) and DF-ECAPA52. According to Section IV-C2, there exist two places where AFF scheme can be adopted for ECAPA and DF-ECAPA models. Specifically, we implement binary AFF scheme to replace the residual addition in SE-Res2Block and multiple AFF scheme to substitute the concatenation in multi-layer feature aggregation module. CA is specially designed for 2D convolution, which can not be directly applied to 1D ECAPA/DF-ECAPAs. In addition, we notice that P-AFF for binary fusion will lead to significant increase in parameter with similar performance gains. Therefore, we finally adopt the combination of S-AFF based on 1D convolution MS-CAM for binary fusion and P-AFF based on 1D convolution MS-CAM for multiple fusion.

As shown in Table VIII, for ECAPA(C = 512), the parameter almost stays the same and FLOPs decreases by 0.16 G after using the above mentioned AFF modules. Surprisingly, the performance becomes better than the baseline. This reveals that the proposed AFF scheme is also beneficial to ECAPA and DF-ECAPA models. Compared to the original feature fusion methods, AFF scheme is not only more powerful to extract speaker-related information, but also more efficient in computation. On the other hand, after applying AFF to DF-ECAPA52, although the increase in parameters is 1.63 M, the FLOPs increase is still small, merely 0.16 G. Compared to the baseline, around 10% relative improvements are achieved, which re-verifies the effectiveness of our attentive feature fusion scheme.

3) Visualization of the Learned Fusion Weights: To further verify the effectiveness of the proposed AFF scheme, we visualize and analyse the distribution of the learned feature fusion weights in this part. Take CA-based AFF module as an example, we randomly sample several utterances from one speaker in Voxceleb1 test dataset and feed them into the pre-trained ResNet18-AFF model to calculate the corresponding attention map $\mathbf{S}^t$ and $\mathbf{S}^f$ on average. Specifically, the fusion weights generated by AFF module in the first and last layer are illustrated in Fig. 6.

As Fig. 6 displays, AFF module exhibits the capability to produce speaker-specific features throughout various network layers. In the first layer, the weight distribution is more diverse with low mean value. By comparison, the variance of fusion weights in the last layer is much lower and the weight values become larger or even close to 1. This phenomenon is reasonable because features in shallow layers of neural network contain much rawer and coarser speech information, which means that speaker-related and non speaker-related information co-exist. AFF module has the ability to focus more on speaker information



(a) Vox1-E EER vs. # Params (b) Vox1-E EER vs. FLOPs

Fig. 7. The comparison of our proposed models and the baseline systems in terms of performance and complexity.

and suppress speaker-unrelated one. On the contrary, much denser and more speaker-specific information exists in deep layers. Accordingly, the weight values generated by AFF module are becoming much larger or even close to 1, and the distribution is more even. This demonstrates that compared to the traditional fix-weighted feature fusion methods, our proposed AFF module can dynamically generate fusion weights based on the contents of features and effectively attach more importance to speaker-related information, yielding more robust and discriminative speaker representation.

C. Analysis of Performance and Complexity

In this section, we present a detailed analysis of our proposed models and the baseline systems from the perspective of the trade-off on performance and complexity. Fig. 7 summarizes the comparison in terms of performance-parameter and performance-FLOPs trade-off.

1) Performance vs. # Params: Firstly, we examine the proposed depth-first model families. From Fig. 7(a), it can be observed that DF-ResNets significantly exceed the corresponding ResNets in both low and high parameter regimes. Similarly, DF-ECAPAs achieve better performance than the original ECAPA-TDNNs with roughly the same or even fewer parameters. For example, 28% relative performance gains are obtained by DF-ResNet56 compared to ResNet18 with similar parameters. Also, DF-ResNet110 exhibits much better results than ResNet34 and ECAPA(C = 512). Plus, our DF-ResNet179 cuts down 38% parameters while maintaining almost the same EER as ResNet101. Likewise, DF-ECAPA244 obtains better results with 25% fewer parameters compared with ECAPA(C=1024). The above results demonstrate the superiority and efficiency of our depth-first version of ResNet and ECAPA over the original ones.

In addition, in low computation scenario, our proposed AFF scheme can further improve the performance of small models with a slight increase in parameters for ResNet, ECAPA, DF-ResNet and DF-ECAPA. Specifically, up to 40% relative improvements can be obtained after applying AFF to ResNet18, ResNet34 and DF-ResNet56. In particular, ResNet34-AFF has similar performance compared to ResNet101 with 2.3x fewer parameters. And DF-ResNet56-AFF achieves better results than ECAPA(C=1024) with 3.1x fewer parameters. This illustrates that our newly-designed AFF module can significantly bridge the performance gap between small and large models.

2) Performance vs. FLOPs: Regarding FLOPs, Fig. 7(b) shows that both DF-ResNets and DF-ECAPAs possess more superior performances over ResNet and ECAPA across the full range from low FLOPs region to high one. Specifically, DF-ResNet110 has similar performance to ResNet101 while containing 50% fewer FLOPs. For DF-ECAPA244, 12% relative performance improvements are obtained than ECAPA(C=1024) with 26% fewer FLOPs. In the low FLOPs condition, AFF scheme can further boost the performance of ResNet18/34, DF-ResNet56, ECAPA(C = 512) and DF-ECAPA52 with negligible FLOPs overhead. For example, ResNet34-AFF achieves roughly the same results as ResNet101 with 2.2x fewer FLOPs. DF-ResNet56-AFF can obtain 42% performance gains over ResNet18 under similar FLOPs. These results confirm that significant reduction in EER can be achieved by equipping small models with AFF module at the negligible cost of FLOPs.

In summary, a much better trade-off on performance and complexity is achieved through our proposed depth-first design rule and attentive feature fusion scheme over the baseline models in both low and high computation scenarios.

D. Comparison With Other Systems

In this section, we present a comprehensive comparison between our proposed models and other advanced SV systems published recently. Specifically, various systems including CNN-based, TDNN-based, Transformer-based, MLP-based and Pretrain-based, are thoroughly listed and analysed.

From Table IX, it can be obviously observed that our proposed models outperform previous published SV systems across

TABLE IX
PERFORMANCE COMPARISON BETWEEN OUR PROPOSED MODELS AND OTHER ADVANCED SV SYSTEMS ON VOX1-O, VOX1-E AND VOX1-H. THE SYSTEM, ARCHITECTURE TYPE, PARAMETER NUMBER, FLOPS AND THE CORRESPONDING RATIO ARE PRESENTED IN DETAILS

System	Architecture	# Params	Ratio	FLOPs	Ratio	Vox1-O	Vox1-E	Vox1-H
DF-ResNet56	CNN	**4.5M**	**1x**	**2.7G**	**1x**	**0.96**	**1.09**	**1.99**
DF-ResNet56-AFF	CNN	**5.0M**	**1.1x**	**2.7G**	**1x**	**0.71**	**0.86**	**1.65**
E-TDNN [7]	TDNN	6.8M	1.5x	-	-	1.49	1.61	2.69
ResNet34 [9]	CNN	6.0M	1.3x	3.3G	1.2x	1.46	1.55	2.76
MLP-SVNet [15]	MLP	15.2M	3.4x	4.4G	1.6x	1.36	1.46	2.49
PRN-50v2 [62]	CNN	4.7M	1.1x	-	-	1.08	1.43	2.67
ECAPA(C=512) [11]	TDNN	6.2M	1.4x	-	-	1.01	1.24	2.32
DF-ECAPA244	TDNN	**11.0M**	**1x**	**2.0G**	**1x**	**0.71**	**0.89**	**1.72**
SAEP [13]	Transformer	20.5M	1.9x	5.9G	3.0x	2.91	2.87	4.75
E-TDNN(large) [7]	TDNN	20.4M	1.9x	-	-	1.26	1.37	2.35
ECAPA(C=1024) [11]	TDNN	14.7M	1.3x	-	-	0.87	1.12	2.12
ECAPA(C=2048) [30]	TDNN	56.2M	5.1x	10.7G	5.4x	0.86	1.08	2.01
DF-ResNet179	CNN	**9.8M**	**1x**	**8.6G**	**1x**	**0.62**	**0.80**	**1.51**
GCSA [14]	Transformer	47.2M	4.8x	13.4G	1.6x	1.96	2.07	3.65
ResNet34-ISKConv [35]	CNN	10.1M	1.1x	-	-	1.26	1.32	2.47
SpineNet-49 [52]	CNN	28.6M	2.9x	26.0G	3.0x	1.11	1.17	2.14
T-SE-Spine2Net-49 [52]	CNN	58.0M	5.9x	26.2G	3.0x	0.92	0.99	1.95
SimAM-ResNet34 [29]	CNN	21.5M	2.2x	18.5G	2.2x	0.72	0.99	1.65
DF-ResNet233	CNN	**12.3M**	**1x**	**11.2G**	**1x**	**0.58**	0.76	1.44
Wav2Vec2.0(Large) [63]	Pre-train	∼320M	26.0x	∼26G	2.4x	0.80	0.73	1.39
HuBERT(Large) [63]	Pre-train	∼320M	26.0x	∼26G	2.4x	0.81	0.78	1.51
UniSpeech-SAT(Large) [63]	Pre-train	∼320M	26.0x	∼26G	2.4x	0.70	0.69	1.43
WavLM(Large) [64]	Pre-train	∼320M	26.0x	∼26G	2.4x	0.62	**0.66**	**1.32**

the full scope of computation regimes. DF-ResNet56 and DF-ResNet56-AFF achieve promising results among other systems under the constraint of small parameter number and FLOPs. Particularly, ∼50% relative EER improvements are obtained but with 3.4x fewer parameters and 1.6x fewer FLOPs compared to MLP-SVNet. In addition, in the line of TDNN-based models, our DF-ECAPA244 achieves the best performance among E-TDNN, ECAPA and its variants. Notably, the performance improvements are very limited by simply doubling the channel number of traditional ECAPA from 1024 to 2048. In contrast, our depth-first version ECAPA can obtain 18% relative EER improvement but with 5.1x fewer parameters and 5.4x fewer FLOPs, which reflects the superiority of depth-first design rule again.

In addition, we also include the results of recently published pre-train based systems for speaker verification. It is widely known that pre-trained models achieve the state-of-the-art results across various downstream speech tasks including speaker verification [63], [64], benefiting from large-scale architectures and training datasets. For example, WavLM(Large) contains 24 Transformer layers with around 320 M parameters which is pre-trained on 94,000 hours speech data. Surprisingly, our best model DF-ResNet233 is on a par with all the pre-train based SV systems, including Wav2Vec2.0(Large), HuBERT(Large), UniSpeech-SAT(Large) and WavLM(Large), however just with roughly **26x** fewer parameters and **2.4x** fewer FLOPs.

Moreover, doing the comparison between the proposed DF-ResNet56-AFF system (the second line of Table IX) and the other ones, it is observed that our proposed depth-first neural architecture with attentive feature fusion is very efficient, compact and high-performance. It only demands 5.0 M parameters with 2.7 G FLOPs, which is obviously much smaller than all the others, however it still owns the strong ability on speaker modeling, which approaches the current state-of-the-art on Voxceleb speaker verification.

VII. CONCLUSION

In this work, we explore efficient architecture design for speaker verification. Firstly, the effect of depth and width on the performance of SV system is investigated, and we empirically conclude that depth is more important than width of networks for the SV task. Then two new model families of much deeper networks named DF-ResNets and DF-ECAPAs are constructed according to the depth-first design rule. To further boost the performance of small models in low computation condition, attentive feature fusion (AFF) scheme is introduced to replace the conventional feature fusion methods. Specifically, two different fusion strategies are proposed including sequential AFF (S-AFF) and parallel AFF (P-AFF). Experiments on the Voxceleb dataset demonstrate that the newly proposed DF-ResNets and DF-ECAPAs can achieve a much better trade-off on performance and complexity than the original ResNet and ECAPA-TDNN for speaker verification. Besides, AFF scheme can further significantly boost small models' performance with negligible computational overhead. Comparison with other published SV systems confirms that our proposed methods achieve the best performance-complexity trade-off in both low and high computation scenarios.

REFERENCES

[1] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19, no. 4, pp. 788–798, May 2011.

[2] S. Ioffe, "Probabilistic linear discriminant analysis," in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 531–542.

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

[4] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 4052–4056.

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

[6] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors: Robust DNN embeddings for speaker recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5329–5333.

[7] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using X-vectors," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 5796–5800.

[8] D. Garcia-Romero, A. McCree, D. Snyder, and G. Sell, "Jhu-HLTCOE system for the voxsrc speaker recognition challenge," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7559–7563.

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

[10] Y. Yu and W. Li, "Densely connected time delay neural network for speaker verification," in *Proc. Interspeech*, 2020, pp. 921–925.

[11] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification," in *Proc. Interspeech*, 2020, pp. 3830–3834.

[12] B. Liu, Z. Chen, S. Wang, H. Wang, B. Han, and Y. Qian, "DF-RESNet: Boosting speaker verification performance with depth-first design," in *Proc. Interspeech*, 2022, pp. 296–300.

[13] P. Safari, M. India, and J. Hernando, "Self-attention encoding and pooling for speaker recognition," in *Proc. Interspeech*, 2020, pp. 941–945.

[14] B. Han, Z. Chen, and Y. Qian, "Local information modeling with self-attention for speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 6727–6731.

[15] B. Han, Z. Chen, B. Liu, and Y. Qian, "MLP-SVNET: A multi-layer perceptrons based network for speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 7522–7526.

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

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

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

[19] S. Wang, Y. Yang, Y. Qian, and K. Yu, "Revisiting the statistics pooling layer in deep speaker embedding learning," in *Proc. IEEE 12th Int. Symp. Chin. Spoken Lang. Process.*2021, pp. 1–5.

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

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

[22] X. Xiang, S. Wang, H. Huang, Y. Qian, and K. Yu, "Margin matters: Towards more discriminative deep neural network embeddings for speaker recognition," *in Proc. IEEE Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.,* 2019, pp. 1652–1656.

[23] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Interspeech*, 2015, pp. 3214–3218.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[25] A. Vaswaniet al., "Attention is all you need," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[26] A. Dosovitskiyet al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Representations*, 2021.

[27] A. Gulatiet al., "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.

[28] W. Xie, A. Nagrani, J. S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 5791–5795.

[29] X. Qin, N. Li, C. Weng, D. Su, and M. Li, "Simple attention module based speaker verification with iterative noisy label detection," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 6722–6726.

[30] J. Thienpondt, B. Desplanques, and K. Demuynck, "The IDLAB VoxSRC-20 submission: Large margin fine-tuning and quality-aware score calibration in DNN based speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5799–5803.

[31] S. Wang, Y. Yang, T. Wang, Y. Qian, and K. Yu, "Knowledge distillation for small foot-print deep speaker embedding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6021–6025.

[32] B. Liu, H. Wang, Z. Chen, S. Wang, and Y. Qian, "Self-knowledge distillation via feature enhancement for speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022 pp. 7542–7546.

[33] S.-H. Kim and Y.-H. Park, "Adaptive convolutional neural network for text-independent speaker recognition," in *Proc. Interspeech*, 2021, pp. 66–70.

[34] J. Qi, W. Guo, and B. Gu, "Bidirectional multiscale feature aggregation for speaker verification," in *Proc. Interspeech*, 2021, pp. 71–75.

[35] Y. Wu, J. Zhao, C. Guo, and J. Xu, "Improving deep CNN architectures with variable-length training samples for text-independent speaker verification," in *Proc. Interspeech*, 2021, pp. 81–85.

[36] Y. Liu, Y. Song, I. McLoughlin, L. Liu, and L. Dai, "An effective deep embedding learning method based on dense-residual networks for speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 6668–6672.

[37] J. Thienpondt, B. Desplanques, and K. Demuynck, "Integrating frequency translational invariance in TDNNs and frequency positional information in 2D resnets to enhance speaker verification," in *Proc. Interspeech*, 2021, pp. 2302–2306.

[38] M. Zhao, Y. Ma, M. Liu, and M. Xu, "The speakin system for voxceleb speaker recognition challange 2021," 2021, *arXiv:2109.01989*.

[39] L. Zhang, Q. Wang, and L. Xie, "Duality temporal-channel-frequency attention enhanced speaker representation learning," *in Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, *pp. 206–213.*

[40] J. Thienpondt, B. Desplanques, and K. Demuynck, "The IDLAB VoxCeleb speaker recognition challenge 2021 system description," 2021, *arXiv:2109.04070*.

[41] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2Net: A new multi-scale backbone architecture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 652–662, Feb. 2021.

[42] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[43] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[44] K. Han, Y. Wang, Q. Zhang, W. Zhang, C. Xu, and T. Zhang, "Model Rubik's cube: Twisting resolution, depth and width for tinynets," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2020, pp. 19353–19364.

[45] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf.*, 2016, pp. 87.1–87.12.

[46] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4700–4708.

[47] A. G. Howardet al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[48] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetv2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.

[49] A. Howardet al., "Searching for mobilenetv3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1314–1324.

[50] M. Tanet al., "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE /CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2820–2828.

[51] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020 s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11976–11986.

[52] M. Rybicka, J. Villalba, P. Zelasko, N. Dehak, and K. Kowalczyk, "Spine2Net: Spinenet with res2net and time-squeeze-and-excitation blocks for speaker recognition," in *Proc. Interspeech*, 2021, pp. 496–500.

[53] Y. Dai, F. Gieseke, S. Oehmcke, Y. Wu, and K. Barnard, "Attentional feature fusion," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2021, pp. 3559–3568.
[54] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 13713–13722.
[55] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Netw.*, vol. 107, pp. 3–11, 2018, doi: 10.1016/j.neunet.2017.12.012.
[56] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A large-scale speaker identification dataset," in *Proc. Interspeech*, 2017, pp. 2616–2620.
[57] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," in *Proc. Interspeech*, 2018, pp. 1086–1090.
[58] W. Wang, D. Cai, X. Qin, and M. Li, "The DKU-DukeECE systems for VoxCeleb speaker recognition challenge 2020," 2020, *arXiv:2010.12731*.
[59] W. Cai, J. Chen, J. Zhang, and M. Li, "On-the-fly data loader and utterance-level aggregation for speaker and language recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 1038–1051, 2020.
[60] D. S. Parket al., "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019, pp. 2613–2617.
[61] Z. N. Karam, W. M. Campbell, and N. Dehak, "Towards reduced false-alarms using cohorts," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2011, pp. 4512–4515.
[62] S. Yadav and A. Rai, "Frequency and temporal convolutional attention for text-independent speaker recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6789–6793.
[63] Z. Chenet al., "Large-scale self-supervised speech representation learning for automatic speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 6147–6151.
[64] S. Chenet al., "WavLM: Large-scale self-supervised pre-traininfor full stack speech processing," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 6, pp. 1505–1518, Oct. 2022.

Bei Liu (Student Member, IEEE) received the B.Eng. degree from the School of Aerospace Engineering and Applied Mechanics, Tongji University, Shanghai, China, in 2016, and the M.Sc. degree from the Department of Computer Science, University of Southern California, Los Angeles, CA, USA, in 2019. He is currently working toward the Ph.D. degree with the X-LANCE Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, under the supervision of Yanmin Qian. His research focuses on speaker recognition.

Zhengyang Chen (Student Member, IEEE) received the B.Eng. degree from the Department of Electronic and Information Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2019. He is currently working toward the Ph.D. degree with the X-LANCE Lab, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, under the supervision of Yanmin Qian. His research interests include speaker recognition and speaker diarization.

Yanmin Qian (Senior Member, IEEE) received the B.S. degree from the Department of Electronic and Information Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2007, and the Ph.D. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2012. Since 2013, he has been with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, where he is currently a Full Professor. From 2015 to 2016, he was an Associate Research with the Speech Group, Cambridge University Engineering Department, Cambridge, U.K. He has authored or coauthored more than 200 papers in peer-reviewed journals and conferences His research interests include automatic speech recognition, speaker and language recognition, speech enhancement and separation, key word spotting, and multimedia signal processing. He was the recipient of several awards including the Best Paper Award from IEEE ASRU in 2019. He was also the Member of IEEE Signal Processing Society Speech and Language Technical Committee.