Team NSYSU-CHT VoxCeleb Speaker Recognition Challenge 2022 System Description

Wei-Ting Lin[1], *Ting-Wei Chen*[1], *Chia-Ping Chen*[1], *Chung-Li Lu*[2], *Bo-Cheng Chan*[2], *Yu-Han Cheng*[2], *Hsiang-Feng Chuang*[2], *Wei-Yu Chen*[2]

[1]National Sun Yat-sen University, Kaohsiung, Taiwan
[2]Chunghwa Telecom Laboratories, Taoyuan, Taiwan

{m093040020, m103040017}@student.nsysu.edu.tw[1], cpchen@cse.nsysu.edu.tw[1], {chungli, cbc, henacheng, gotop, weiweichen}@cht.com.tw[2]

Abstract

This technical report describes team NSYSU-CHT submission for track 1, track 2, and track 3 of the VoxCeleb Speaker Recognition Challenge 2022 (VoxSRC-22). We take multiple measures to enhance ECAPA CNN-TDNN model. In front-end stem structure, we not only change the sequence of residual block so that it becomes a pre-activation block, but also replace attention block form SE to CBAM. In backbone structure, we use SC-block to replace the original SE-Res2Block, and adopted aggregation method of deep layer aggregation. We use three different loss functions for our experiment. Among the three loss functions, the one we proposed, called RAAM-softmax, achieves the highest performance in each experiment. The mean-teacher framework is employed in track 3 to fine-tune the systems from track 1. The final submitted score for track 1&2 is a fusion of four single systems and achieves 0.2288 minDCF and 3.74% EER. The submitted score of track 3 fusion with five single systems is 0.6165 minDCF and 12.36% EER.

Index Terms: front-end module, deep layer aggregation (DLA), Real Additive Angular margin Softmax (RAAM-Sfotmax), VoxSRC-22

1. System Architectures

In this section, we will describe the structure of our speaker embedding extractors in detail. We can divide our extractors into two parts - one is the front-end, and another is the backbone.

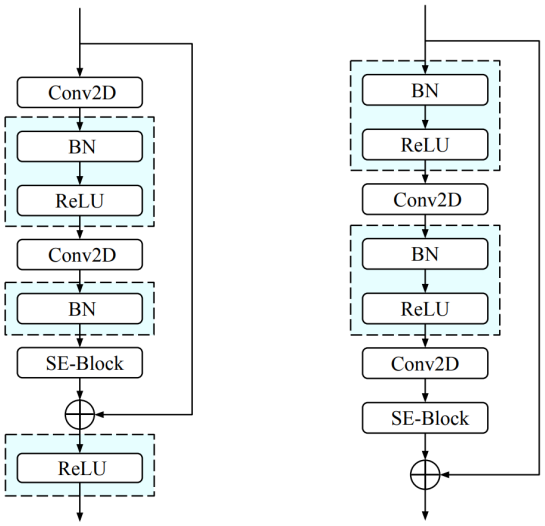
1.1. Front-end Module

Based on the experiment results of ECAPA CNN-TDNN [1], we can observe that adding a front-end feature processing module can boost the performance of the network. Using the research mentioned before as a start point, we used various measures to build the front-end modules for strengthening the performance of our system.

1.1.1. Pre-activation 2D CNN Stem

According to the study in [2], any operation on the shortcut connections of residual blocks would lead to diminished performance of neural networks; on the other hand, switching the neural networks sequence of residual blocks from post-activation to pre-activation could increase generalization of models.

We refer to the modified residual block in [2], and then adjust the neural networks of residual blocks in 2D CNN stem. In order to utilize the ability of attention mechanism, we add a SE-block [3] at the end of each residual block in 2D CNN stem. The comparison of the original structure and the modified structure is shown in Figure 1.



(a) original (b) pre-activation

Figure 1: *Comparison between the original structure and the modified structure of residual block.*

1.1.2. Pre-activation 2D CNN Stem with CBAM

In the original 2D CNN stem, there is a lot of information on the spatial dimension that we have not analyzed because the SE-block only focuses on the channel dimension and assigns weight to each channel. To make full use of the information in the spatial dimension, we use the CBAM-block [4] to replace the SE-block. By integrating channel attention and spatial attention, the 2D CNN stem could reconstruct the audio features in a more detailed way.

1.1.3. MFA Module

This front-end module is a modified version of Res2Block [5]. Research shows that the module is effective on short test utterances, and has enabled MFA-TDNN to achieve state of the art performance [6]. It contains two changes on the basis of Res2Block. First is the dual-pathway multi-scale module (DM). A new TDNN block would add to the end of each convolution branch. The output features of each TDNN block would be sent to the next branch, forming a dual-pathway with the out-

put features sent from the first convolution block at the same branch. Second is the frequency-channel attention module (FA). FA-block is an attention mechanism block that is set inside the TDNN block we added. Though the FA-block has a similar overall concept to the SE-block, it focuses on both channel and frequency dimensions. This would give the MFA module an attention capability similar to the CBAM-block, but with fewer parameters.

1.2. Backbone

We modify the aggregation method and basic block of ECAPA-TDNN [7] and propose ECAPDLA-TDNN. The architecture of ECAPDLA-TDNN is shown in Figure 2. In the ECAPDLA-TDNN model, consider that ECAPA-TDNN uses only the basic residual block of Res2Net. We refer to [8] to replace it with SC block, the convolution block of Self-Calibration Network (SCNet) [9], which integrates the concepts of Res2Net and SKNet [10]. Similar to Res2Net bottleneck residual block (Res2Block), SC block divides input feature maps into several groups. The difference is that SC blocks use self-attention mechanisms instead of only hierarchical residual-like connections to capture spatial information. The mechanism is called self-calibration. The self-calibration mechanism generates additional spaces of different scales through average pooling (AvgPool), convolution, and FC, and multiplies with the original scale space. With self-calibration, the model with SC block can exploit features at multiple scales and avoid spurious signals through the attention mechanism. Based on the above characteristics of SC block, we propose to use it to replace SE-Res2Block.

Furthermore, we adopt DLA to replace the multi-layer aggregation in ECAPA-TDNN. Note that the full DLA architecture is too large to match the size of ECAPA-TDNN. Instead, only the first half of the DLA is implemented. ECAPDLA-TDNN fuses feature information of different depths in a tree structure, and passes the previous aggregation node information to obtain feature information of different time scales. The simplified architecture can achieve the effect of the above two aggregation methods and maintain high training efficiency.

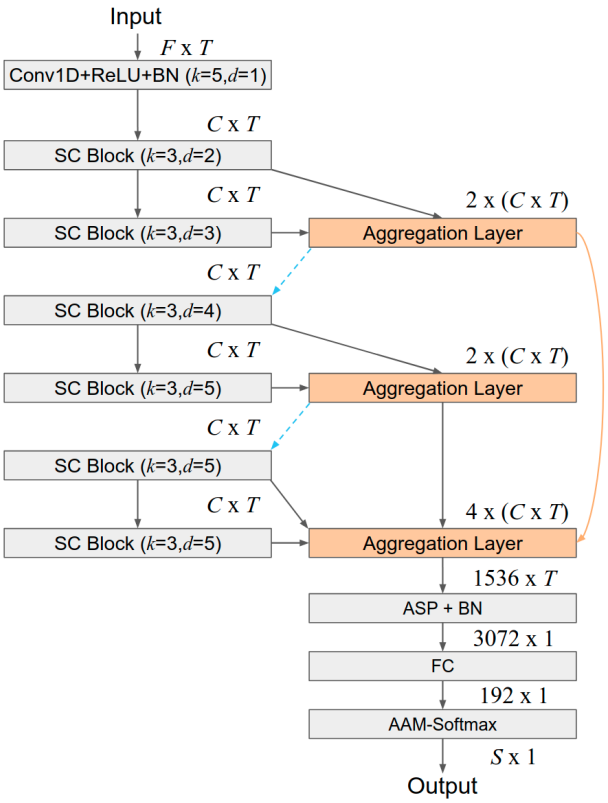


Figure 2: *The architecture of ECAPDLA-TDNN. In the right half, the aggregation method of deep layer aggregation is adopted. We additionally use the SC block as the basic block. The parameter F represents the dimension of acoustic features, T represents the number of input frames, C is the number of convolution channels, k is the size of the convolution kernel, d is the dilation rate, S is the number of speakers. In the experiments in this paper, the parameters C, F, and T are set to 512, 80, and 200, respectively. S is determined by the number of speakers in the training dataset.*

2. Training Procedure

In track 1, there are four speaker embedding extractors composed of different front-end modules and loss functions. We will describe the detail of the loss function later on. One of the extractors mentioned above is also used on the track 2, trained on VoxCeleb1 and VoxCeleb2 corpus. The track 3 uses all of the extractors trained in track 1 as pre-trained models, fine-tuning using data from the target domain with the mean-teacher method. The details of the track 3 will be described later as well.

The input features are 80-dimensional log Mel-filterbank energies with a window length of 25 ms and a 10 ms frame-shift. In addition, the extracted features are mean-normalized without the application of voice activity detection (VAD). We have used the MUSAN [11] corpus (additive speech, music, and noise) and the RIR [12] corpus (reverb) during training steps, each training step would randomly select one method for data augmentation. Track 1&2 training batch size is set to 128, and track 3 fine-tuning batch size is set to 64.

All extractors consistently have 512 channels in convolution frame layers. The channel of 2D CNN stem module is set to 64, and the channel of MFA module is set to 32. The output size of speaker embedding is 192.

2.1. Loss Function

In the loss function, we applied the most popular AAM-Softmax loss in the field of speaker verification. We also use the RAM-Softmax loss proposed in [13]. In addition, we improve AAM-Softmax in the same way and propose a new loss function RAAM-Softmax loss.

2.1.1. AAM-Softmax

Additive angular margin softmax (AAM-Softmax) loss is the most widely used loss function in the field of speaker verification. Similar to other softmax-based loss functions, the margin m enforces the separation of different classes. AAM-Softmax adds the interval m directly to the angle θ in the cosine value. The formula can be expressed as follows:

$$L=-\frac{1}{N}\sum_{i=1}^{N}log\frac{e^{s(cos(\theta_{yi,i}+m))}}{e^{s(cos(\theta_{yi,i}+m))}+\sum_{j\neq y_i}^{c}e^{cos(\theta_{j,i})}}. \quad (1)$$

Where N is the batch size. s and m are the scaling factor and the additive margin, respectively. i is the index of training sam-

ples. y_i is the label of the i-th sample. j is the classes other than y_i. $\theta_{j,i}$ is the angle between the representative vector of the sample x_i and the representative vector of class j. We set the scaling factor and the additive margin as 30.0 and 0.2.

2.1.2. RAM-Softmax

According to the research in [13], the improvement of the additive margin softmax (AM-Softmax) loss can effectively enhance the robustness of the model. The newly proposed loss function by [13] is called real additive margin softmax (RAM-Softmax). The formula of RAM-Softmax is:

$$L=\frac{1}{N}\sum_{i=1}^{N}log\left\{1+\sum_{j\neq y_i}^{c}e^{max\left\{0,-s(cos(\theta_{y_i,i})-cos(\theta_{j,i})-m)\right\}}\right\}. \tag{2}$$

Where N is the batch size. s and m are the scaling factor and the additive margin, respectively. i is the index of training samples. c is the number of classes. y_i is the label of the i-th sample. j is the classes other than y_i. $\theta_{j,i}$ is the angle between the representative vector of the sample x_i and the representative vector of class j. We set the scaling factor and the additive margin as 30.0 and 0.3.

2.1.3. RAAM-Softmax

AAM-Softmax is also a loss function based on Softmax. Therefore, we can refer to the optimization method of RAM-Softmax in [13] to improve AAM-Softmax, and propose a revised loss function real additive angular margin softmax (RAAM-Softmax) loss. Training the model with the RAAM-Softmax loss that focuses on learning difficult situations can improve its performance on the challenging test dataset. The formula of RAAM-Softmax is:

$$L=\frac{1}{N}\sum_{i=1}^{N}\log\left\{1+\sum_{j\neq y_i}^{c}e^{\max\left\{0,-s\left(\cos\left(\theta_{y_i,i}+m\right)-\cos\left(\theta_{j,i}\right)\right)\right\}}\right\}. \tag{3}$$

Where N is the batch size. s and m are the scaling factor and the additive margin, respectively. i is the index of training samples. y_i is the label of the i-th sample. j is the classes other than y_i. $\theta_{j,i}$ is the angle between the representative vector of the sample x_i and the representative vector of class j. We set the scaling factor and the additive margin as 30.0 and 0.3.

2.2. Mean-teacher and Utilization of Target Domain

For the track 3, we fine-tune the extractors from the track 1 using the mean-teacher framework [14]. The Mean Square Error (MSE) loss is used for the consistency cost. Due to the huge disparity of numbers between labeled data and unlabeled data, we experimented with various sampling schemes to seek the most appropriate data mix plan. In the end, we set the batch size to 64, in which the number of labeled data is 10, and the rest are unlabeled data. Training would then proceed to the next epoch once all the labeled data had been used.

In addition to initializing the same pre-trained model for both teacher and student models in the mean-teacher framework, we also experiment with having the student and teacher models use different pre-trained models as initial states. We conducted the experiment because we expect that increasing the difference between the two models in the mean-teacher framework could help use unlabeled data more effectively.

3. Scoring Procedure

Each test file will be extracted into 10 audio segments with a duration of 3 seconds, then input into the extractor to calculate the speaker embedding. The verification trials are scored by calculating the cosine similarity between speaker embeddings of two audio files in test pair. We do not use any score normalization method.

3.1. Score Fusion

We train a linear transformation model to fuse multiple scores into a single score for our submission. This linear transformation model consists of only a single linear layer. The input of the linear layer is the scores to be fused, and the output is a single score. Through this linear transformation model, we can merge the scores calculated by multiple excellent models into a single score and get better results.

We employ score fusion on track1 and track3. On track1, score fusion is performed based on the scores of the five models in Table refTrack12Val. We use the scores of these four models on the validation set to train a linear transformation model, which can be closer to the situation on the test set. Then, the scores of the four models on the test set are fused through this linear transformation model. On track3, score fusion is performed based on the scores of the five models in Table refTrack3val. We also use the scores on the validation set to train the linear transformation model. Finally, the scores of the five models on the test set will be fused through a linear transformation model.

4. Result

Table 1 and Table 3 shows the performance of different single system in VoxSRC-22 validation sets of track 1&2 and track 3. For each track, we select the system with the highest performance and the fusion of every single system to run on the test set. The results of the VoxSRC-22 test sets for track 1&2 and track 3 are shown in Table 2 and Table 4 separately.

Table 1: *System performance on the VoxSRC-22 track 1&2 validation set.*

Architecture	**EER(%)**	**MinDCF**
ECAPDLA CNN-TDNN (RAM)	3.13	0.1951
ECAPDLA CNN-TDNN (RAAM)	2.88	0.1855
ECAPDLA CNN-TDNN (CBAM+RAAM)	**2.86**	**0.1796**
ECAPDLA MFA-TDNN	3.08	0.1948
ECAPDLA CNN-TDNN (RAAM) + trained on Vox1 & Vox2	**1.45**	**0.0962**

Table 2: *System performance on the VoxSRC-22 track 1&2 test set.*

Architecture	**EER(%)**	**MinDCF**
ECAPDLA CNN-TDNN (CBAM+RAAM)	4.28	0.2461
Fusion (close)	**3.74**	**0.2288**
ECAPDLA CNN-TDNN (RAAM) + trained on Vox1 & Vox2 (open)	**3.74**	**0.2297**

Table 3: *System performance on the VoxSRC-22 track 3 validation set.*

Architecture	EER(%)	MinDCF
ECAPDLA CNN-TDNN (RAM)	12.56	0.5263
ECAPDLA CNN-TDNN (RAAM)	12.90	0.5294
ECAPDLA CNN-TDNN (CBAM+RAAM)	**12.44**	**0.5231**
ECAPDLA MFA-TDNN	12.78	0.5283
ECAPDLA CNN-TDNN (CBAM+RAAM) + teacher & student with different initial model	12.59	0.5279

Table 4: *System performance on the VoxSRC-22 track 3 test set.*

Architecture	EER(%)	MinDCF
ECAPDLA CNN-TDNN (CBAM+RAAM)	13.65	0.6936
Fusion	**12.36**	**0.6165**

5. References

[1] J. Thienpondt, B. Desplanques, and K. Demuynck, "Integrating frequency translational invariance in tdnns and frequency positional information in 2d resnets to enhance speaker verification," *arXiv preprint arXiv:2104.02370*, 2021.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[3] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[4] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[5] 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 transactions on pattern analysis and machine intelligence*, vol. 43, no. 2, pp. 652–662, 2019.

[6] T. Liu, R. K. Das, K. A. Lee, and H. Li, "Mfa: Tdnn with multi-scale frequency-channel attention for text-independent speaker verification with short utterances," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7517–7521.

[7] B. Desplanques, J. Thienpondt, and K. Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," *arXiv preprint arXiv:2005.07143*, 2020.

[8] Y.-J. Zhang, Y.-W. Wang, C.-P. Chen, C.-L. Lu, and B.-C. Chan, "Improving time delay neural network based speaker recognition with convolutional block and feature aggregation methods." in *Interspeech*, 2021, pp. 76–80.

[9] J.-J. Liu, Q. Hou, M.-M. Cheng, C. Wang, and J. Feng, "Improving convolutional networks with self-calibrated convolutions," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 096–10 105.

[10] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 510–519.

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

[12] 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.

[13] L. Li, R. Nai, and D. Wang, "Real additive margin softmax for speaker verification," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7527–7531.

[14] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.