Interspeech 2022
18-22 September 2022, Incheon, Korea

# Enroll-Aware Attentive Statistics Pooling for Target Speaker Verification

*Leying Zhang*[†], *Zhengyang Chen*[†], *Yanmin Qian*

MoE Key Lab of Artificial Intelligence, AI Institute
X-LANCE Lab, Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
{zhangleying, zhengyang.chen, yanminqian}@sjtu.edu.cn

## Abstract

The well-developed robust speaker verification system can remove the environment noise and retain speaker information automatically. However, when the uttering voice is disturbed by another interfering speaker's voice, the speaker verification system usually cannot selectively extract only the target speaker's information. Some works have been done by introducing a speech separation network to separate the target speaker's speech in advance. However, adding a speech separation network for speaker verification task could be redundant. Here, we proposed enroll-aware attentive statistic pooling (EA-ASP) layer to help the speaker verification system extract specific speaker's information. To evaluate the system, we simulate the multi-speaker evaluation data based on Voxceleb1 data. The results show that our proposed EA-ASP can outperform the baseline system by a large margin and achieved ∼50% relative Equal Error Rate (EER) reduction.

**Index Terms**: target speaker verification, enrollment-aware

## 1. Introduction

Deep learning technology has boosted the performance of speaker verification systems [1, 2, 3, 4] and more and more researchers are dedicated to building robust speaker verification system in more complicated scenarios. The data augmentation method is the most effective and simplest way to improve the robustness of the speaker verification system. In [3], the authors add random noise and reverberation to the original audios to generate more training data. And the authors in [5, 6] directly do random perturbation on the spectrum to augment the training data. Besides, to help the speaker verification system learn to remove the nuisance information explicitly, the researchers in [7, 8, 9] use adversarial technologies to remove the channel information. Similarly, adversarial technology is also used in [10, 11] to help the speaker verification system become more robust to phonetic variability.

All the works mentioned above are intended to remove the information other than the human voice. However, when the uttering voice is disturbed by the voice from other persons, the above system usually cannot selectively remove this interfering voice. Besides, the current popular speaker verification systems always assume that there is only one speaker in the input utterance and map the input utterance to a low-dimensional vector, called speaker embedding, to represent the speaker identity existing in that utterance. It is also unknown whether the speaker embedding can still reflect the information of speaker identity when there are multiple speakers in the input utterance.

[†]:These authors have contributed equally to this work
Yanmin Qian is the corresponding author

The speaker verification involves two stages, enrollment and test. In the test stage, the environment often has various uncertainties, especially when people want to wake up their mobile phones or smart speakers using voice in a crowded environment. Differently, in the enrollment stage, the speakers are required to record their voices in a quieter environment. If the speaker verification system can leverage the enrollment information in the test stage, it could be possible to remove the interfering speaker from the speaker mixture speech. The authors in [12] found that the interfering speaker problem happened frequently in the speaker diarization task and they named the task that used additional enrollment speaker information in the speaker verification process as the target speaker verification (TSV).

To solve the target speaker verification problem, many researchers borrowed techniques from speech separation. In [12, 13, 14], the authors first encode the speaker information of enrollment utterance and then use the encoded information to help the separation network separate the enrollment speaker's speech from test utterance. The separated speech only contains one speaker and can be used in the subsequent speaker verification system. However, leveraging a speech separation network in a speaker verification task could be complex and redundant. Recently, many researchers have introduced the attention mechanism to the pooling operation in the speaker verification system [15] to remove the nuisance information. Inspired from this, we propose enroll-aware attention statistic pooling (EA-ASP) by directly injecting the enrollment speaker's information into the pooling layer, which can help the attention mechanism remove the interfering speaker. Compared with the method using additional speech separation network in [12, 13, 14], our newly proposed EA-ASP layer is more light-weight and can be integrated with most mainstream speaker verification systems. Besides, in the experiment, we found that directly feeding the enrollment embedding into the EA-ASP layer would cause the speaker information leakage in the training process and we designed a bottleneck architecture to solve this problem.

## 2. Method

### 2.1. Enroll-Aware Attentive Statistics Pooling

#### *2.1.1. Network Design*

The left part of Figure 1 is a standard speaker verification system. It consists of a feature extraction network, a pooling layer and an embedding transformation layer. The feature extraction network first maps the input $\mathbf{x}$ to hidden representation sequence $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_t\}$. Then, the pooling layer transforms the variable length representation $\mathbf{H}$ to fix-dimensional representation. Finally, another transformation layer is applied to get speaker embedding $\mathbf{e}$.

To remove the interfering speakers' information, we lever-



10.21437/Interspeech.2022-645

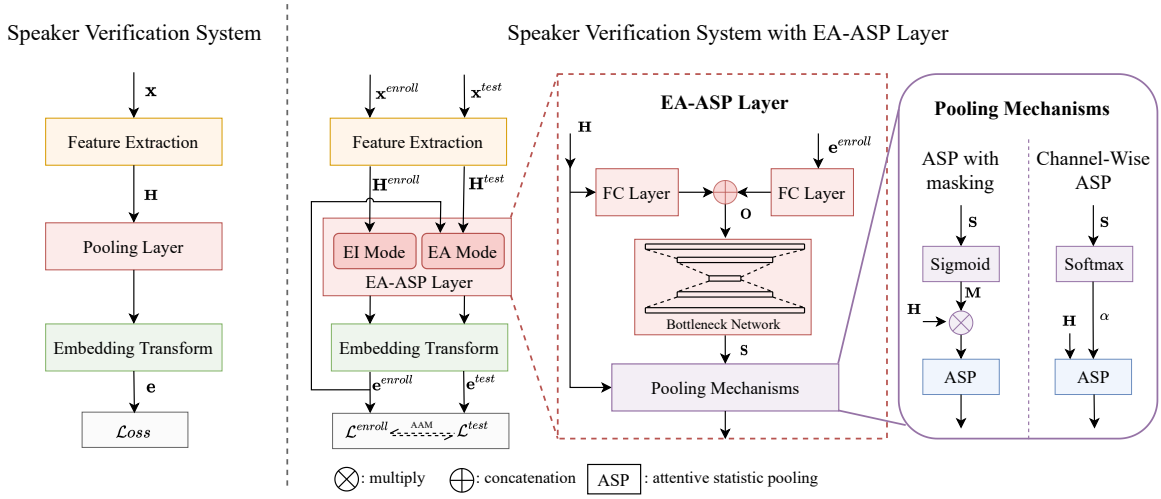



Figure 1: ***System Architecture***. *In **EI Mode**:* $\mathbf{e}^{enroll}$ *is not used in calculation of score* $\mathbf{S}$ *and we directly set score* $\mathbf{S}$ *to all-ones matrix. In **EA Mode**:* $\mathbf{e}^{enroll}$ *is used to calculate the score* $\mathbf{S}$*. In ASP with masking, the attention weight is calculated within the ASP module. In channel-wise ASP, we directly feed attention weight* $\alpha$ *to ASP module.*

age the enrollment speaker as the prior information in the pooling module and propose enroll-aware attentive statistic pooling (EA-ASP). As shown in the right part of Figure 1, our EA-ASP layer takes hidden representation $\mathbf{h}_t \in \mathbb{R}^C$ [1] and enroll embedding $\mathbf{e}^{enroll} \in \mathbb{R}^D$ as inputs. The $\mathbf{h}_t$ and $\mathbf{e}^{enroll}$ are firstly transformed with respective fully-connected layer and then concatenated to $\mathbf{o}_t \in \mathbb{R}^{(C+D)}$,

$$\mathbf{o}_t = \mathbf{W}_h\mathbf{h}_t \oplus \mathbf{W}_e\mathbf{e}^{enroll}$$

Then, $\mathbf{o}_t$ is fed to a bottleneck network to get the intermediate score $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_t \in \mathbb{R}^C\}$. Based on the score $\mathbf{S}$, we proposed two mechanisms to map the hidden representation sequence $\mathbf{H}$ to fixed-dimension representation: channel-wise attentive statistics pooling (EA-ASP-CW) and attentive statistics pooling with masking (EA-ASP-M).

**EA-ASP-CW:** In this mechanism, we will first calculate the channel-wise attention weight $\alpha_t^i$ from $\mathbf{s}_t$ along the time dimension:

$$\alpha_t^i = \frac{\exp\left(s_t^i\right)}{\sum_\tau^T \exp\left(s_\tau^i\right)}$$

where $i$ is the channel index. Then, we calculate the statistics $\mu^i$ and $\sigma^i$ based on the attention weight $\alpha_t^i$ following [15].

$$\mu^i = \sum_t^T \alpha_t \mathbf{h}_t^i, \sigma^i = \sqrt{\sum_t^T \alpha_t \mathbf{h}_t^i \odot \mathbf{h}_t^i - \mu^i \odot \mu^i} \quad (1)$$

It should be noted that, different from [15], we calculate attention weight for each channel in equation 1 to help the pooling layer filter the interfering speaker information when multiple speakers are overlapped.

**EA-ASP-M:** In this mechanism, we first map the score matrix $\mathbf{S}$ to a masking matrix $\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_t \in \mathbb{R}^C\}$ using a sigmoid funtion, and then multiply $\mathbf{H}$ by $\mathbf{M}$ to remove the interfering information.

$$\tilde{\mathbf{H}} = \mathbf{M} \odot \mathbf{H}, \mathbf{M} = sigmoid(\mathbf{S}) \quad (2)$$

Then, we directly feed $\tilde{\mathbf{H}}$ to the attentive statistics pooling module introduced in [15] to get fixed-dimension representation.

[1] For 2D convolutional network that outputs $\mathbf{h}_t \in \mathbb{R}^{C\times L}$, we will first average $\mathbf{h}_t$ along the second dimension to get $\mathbf{h}_t \in \mathbb{R}^C$.

#### 2.1.2. Usage Mode

As introduced in the last section, our proposed EA-ASP layer can help the neural network remove the interfering speaker information with the help of enrollment embedding. However, we cannot get the enrollment embedding at all the time. For example, when we want to extract embedding for enrollment utterance, we cannot get the enrollment embedding in advance. Here, we divide the usage of our proposed EA-ASP layer into two modes so that it can be used in any scenario:

- **Enroll-Ignorant Mode (EI):** In EI mode, we do not use the enrollment embedding as condition by directly setting the score $\mathbf{S}$ in section 2.1.1 to all-ones matrix. In this condition, the EA-ASP-M will degenerate to the normal ASP pooling [15] and EA-ASP-CW will degenerate to normal SP pooling [16].
- **Enroll-Aware Mode (EA):** In EA mode, we will extract embedding just following the pipeline introduced in section 2.1.1.

### 2.2. Training Strategy with EA-ASP Layer

In this section, we introduce the training strategy for speaker verification system with EA-ASP layer. Here, we denote the speaker number in the training set as $N_{spk}$. To simulate the scenario in TSV evaluation process, we split the training set into two parts: enrollment utterances and test utterances. There is only one speaker in the enrollment utterance but the speaker number in the test utterance is unknown. We denote the speaker index within enrollment utterance $\mathbf{x}_i^{enroll}$ as $y_i \in [0, N_{spk})$ and denote the speaker index within test utterance $\mathbf{x}_j^{test}$ as $\mathbb{Y}_j = \{y^1, y^2, ...\} \in [0, N_{spk})$.

As shown in the right part of Figure 1, we construct enrollment-test pair $z = (\mathbf{x}_i^{enroll}, \mathbf{x}_i^{test})$ as input. In each training step, the enrollment utterance $\mathbf{x}_i^{enroll}$ is first fed into the system to get enrollment embedding $\mathbf{e}_i^{enroll}$ with EA-ASP layer in enroll-ignorant mode. Then, the test utterance $\mathbf{x}_i^{test}$ together with enrollment embedding $\mathbf{e}_i^{enroll}$ is fed into the system to extract test embedding $\mathbf{e}_i^{test}$ with EA-ASP layer in enroll-aware mode. Then, we compute loss on $\mathbf{e}_i^{enroll}$ and $\mathbf{e}_i^{test}$ using the

same additive angular margin loss module [17]:

$$\mathcal{L} = \mathcal{L}_{enroll} + \mathcal{L}_{test} \tag{3}$$

$$\mathcal{L}_{enroll} = AAM(\mathbf{e}^{enroll}, l^{enroll}) \tag{4}$$

$$\mathcal{L}^{test} = AAM(\mathbf{e}^{test}, l^{test}) \tag{5}$$

where $l_{enroll}$ and $l_{test}$ are classification label for enrollment and test utterance. Here, we directly use the speaker index as the classification label for enrollment utterance. The classification label for the test utterance is conditioned on enrollment utterance. If the enrollment speaker exists in the test utterance, the test utterance has the same classification label as the enrollment utterance. Otherwise, we assign a unique classification label to test utterance that does not represent any speaker.

$$l_i^{enroll} = y_i \tag{6}$$

$$l_i^{test} = \begin{cases} y_i, & \text{if } y_i \in \mathbb{Y}_i \\ N_{spk}, & \text{otherwise} \end{cases} \tag{7}$$

### 2.3. Speaker Information Leakage Problem

As introduced in section 2.2, when $\mathbf{x}_i^{test}$ contains the same speaker with $\mathbf{x}_i^{enroll}$, we classify $\mathbf{x}_i^{test}$ to the speaker $y_i$ within $\mathbf{x}_i^{enroll}$. However, this can cause some problems. As shown in Figure 1, we fed both $\mathbf{x}_i^{test}$ and $\mathbf{e}_i^{enroll}$ to the system to extract test embedding and the training objective guide the neural network to extract the speaker information of $y_i$ to test embedding. It is obvious that the speaker information of $y_i$ can come from $\mathbf{x}_i^{test}$ or $\mathbf{e}_i^{enroll}$. That is to say, the classification objective may cause the speaker information leakage from $\mathbf{e}_i^{enroll}$. Such speaker information leakage problem may cause our EA-ASP layer not to learn how to remove the interfering information but to retain the speaker information from $\mathbf{e}_i^{enroll}$ better. The experimental results in section 4.3 also confirmed the existence of this problem.

To solve the speaker information leakage problem, we designed a bottleneck network in Figure 1. The bottleneck network consists of three fully connected layers with the shape $((C+D)//2, bottleneck\_dim, C)$ and we insert BatchNorm and ReLU function between different layers. We believe that if the bottleneck dimension is much lower than the speaker embedding dimension, the bottleneck network will constrain the flow of speaker information.

## 3. Experimental setup

### 3.1. System Training

#### *3.1.1. Baseline*

We use the r-vector [4] system as the baseline in our experiment. The development part of Voxceleb2 is used as the training set and we also do data augmentation following [3]. During the training process, we randomly sample 2s segments from utterances to train the baseline system for 165 epochs. Besides, the AAM loss [17] is used for system optimization. We set the scale ratio and margin of AAM to 32 and 0.2 respectively.

#### *3.1.2. System with EA-ASP Layer*

To construct the input pair $(\mathbf{x}_i^{enroll}, \mathbf{x}_i^{test})$ introduced in section 2.2. We simulate the test data $\mathbf{x}_i^{test}$ based on the Voxceleb2 dev set and the simulated test data can be classified into 4 different types:

- Type1: $y_i \in \mathbb{Y}_i$, $\mathbf{x}_i^{test}$ contains one speaker
- Type2: $y_i \in \mathbb{Y}_i$, $\mathbf{x}_i^{test}$ contains two speakers
- Type3: $y_i \notin \mathbb{Y}_i$, $\mathbf{x}_i^{test}$ contains one speaker
- Type4: $y_i \notin \mathbb{Y}_i$, $\mathbf{x}_i^{test}$ contains two speakers

where $y_i$ is the speaker index within $\mathbf{x}_i^{enroll}$ and $\mathbb{Y}_i$ is the speakers index set within $\mathbf{x}_i^{test}$.

The proportion of these four types of data is set to 0.05, 0.05, 0.45, 0.45. During the data simulation, the SNR is randomly sampled from -3 to 3 and the overlap ratio between two speakers is randomly sampled from 0 to 0.5. Besides, $\mathbf{x}_i^{enroll}$ is the 2s chunk randomly sampled from the baseline training set and we also ensure that the duration of simulated $\mathbf{x}_i^{test}$ is 2s.

In the training process, we first initialized the system with EA-ASP layer from the pre-trained baseline system and then trained it for another 66 epochs. We also use AAM loss and the same hyper-parameters mentioned in section 3.1.1.

### 3.2. System Evaluation

In our experiments, we evaluated our system in two different scenarios: single-speaker scenario and multi-speaker scenario. In the single-speaker scenario, we assume there is only one speaker in the test utterance. In the multi-speaker scenario, the speaker number in the test utterance is more than one. We score all the systems based on embedding cosine similarity.

#### *3.2.1. Single-Speaker Scenario*

In the single-speaker scenario, we evaluated the system using Voxceleb1 [18] dataset and reported the results on all three public evaluation trials:Vox_O, Vox_E, Vox_H.

#### *3.2.2. Multi-Speaker Scenario*

In order to better compare with results in single-speaker scenario, we regenerate the evaluation dataset according to the evaluation trial Vox_O, Vox_E, Vox_H. For each trial pair, the enrollment utterance is retained and we simulated a new test utterance. When simulating the new test utterance, we randomly select an utterance of a person who is neither test speaker nor the enrollment speaker as the interfering utterance and mix it with original test utterance. During data mixing, the SNR is randomly sampled from -3 to 3 and the overlap ratio is randomly selected from 0 to 0.5[2]. It should be noted that because the interfering speaker has a different speaker label with the enrollment utterance and original test utterance, the target/non-target label of each trial pair will not change.

## 4. Results

### 4.1. Evaluation of Single-Speaker Verification System in Multi-Speaker Scenario

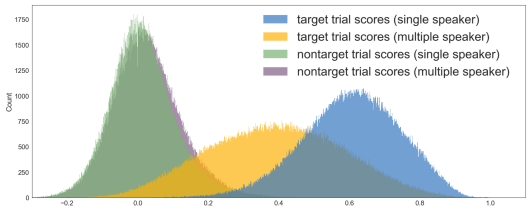
Most speaker verification systems are trained on single-speaker data, and how they would behave in multi-speaker scenario is unknown. Here, we test the r-vector baseline system in the multi-speaker scenario and we plot the score distribution under different conditions in Figure 2a. From the figure, we can see that, in single-speaker scenario, the target score distribution and non-target score distribution are far apart, which means the speaker embedding is very discriminating. When the test utterance is mixed with the other speakers' voice, the target score distribution becomes closer to the non-target score distribution, but they don't overlap. Such a phenomenon indicates that the mixed speakers' voice will obscure the original speaker's information but the embedding can still show the existence of the original speaker. We listed the numerical results in the first line of Table 1. The numerical results show that the ambiguity of

[2]The multi-speaker evaluation data generated in our experiment can be reproduced following https://github.com/vivian556123/EA-ASP-evaluation-dataset

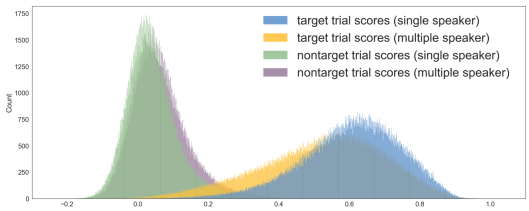
Table 1: ***System Performance.*** *We get the minDCF result with the target probability equal to 0.01. The EI-Mode and EA-Mode in table means that we set the EA-ASP layer to corresponding mode in the test embedding extraction process. We get the ensemble system by maxing the similarity score from the system in EI-Mode and EA-Mode.*

| Model | Single-Speaker Evaluation | | | | | | Multiple-Speaker Evaluation | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Vox_O | | Vox_E | | Vox_H | | Vox_O | | Vox_E | | Vox_H | |
| | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF | EER(%) | minDCF |
| r-vector | **1.128** | **0.118** | 1.240 | 0.150 | 2.385 | 0.228 | 11.16 | 0.574 | 10.54 | 0.571 | 13.81 | 0.651 |
| EA-ASP-CW (EI-Mode) | 1.154 | 0.125 | 1.238 | 0.150 | 2.308 | 0.219 | 10.79 | 0.591 | 10.13 | 0.578 | 13.30 | 0.656 |
| EA-ASP-CW (EA-Mode) | 1.505 | 0.145 | 1.608 | 0.175 | 2.891 | 0.255 | 5.955 | 0.366 | 5.718 | 0.394 | 8.104 | 0.475 |
| EA-ASP-M (EI-Mode) | 1.154 | 0.124 | **1.222** | **0.149** | **2.321** | **0.228** | 11.80 | 0.595 | 10.82 | 0.584 | 14.11 | 0.659 |
| EA-ASP-M (EA-Mode) | 1.601 | 0.148 | 1.736 | 0.192 | 3.221 | 0.280 | **5.212** | 0.335 | **4.988** | **0.342** | **7.067** | **0.435** |
| EA-ASP-CW Ensemble | 1.213 | 0.134 | 1.280 | 0.159 | 2.396 | 0.237 | 6.451 | 0.371 | 6.081 | 0.396 | 8.465 | 0.477 |
| EA-ASP-M Ensemble | 1.148 | 0.128 | 1.317 | 0.158 | 2.517 | 0.241 | 5.377 | **0.332** | 5.019 | 0.345 | 7.127 | 0.436 |

the original speaker information can severely degrade the performance of the system.




(a) *Baseline r-vector system*




(b) *EA-ASP-M in EA mode*

Figure 2: *The Vox_E trial scores distribution in single-speaker and multiple-speaker scenario.*

### 4.2. Evaluation of Enroll-Aware Attentive Statistic Pooling

Here, we evaluated the systems with our proposed enroll-aware attentive statistic pooling. The corresponding results are listed in Table 1. From the results, we found that the EA-ASP-CW and EA-ASP-M systems have comparable performance with the baseline system when no enrollment information is used. When we fed the enrollment embedding into the EA-ASP systems, the EA-ASP-CW and EA-ASP-M both achieved significant improvement on multi-speaker evaluation trials, and EA-ASP-M performed better than the EA-ASP-CW. The EA-ASP-M even achieved ∼50% relative EER reduction on three evaluation trials. However, when enrollment information is added, there is some performance degradation in single-speaker scenario. We calculated the average score of target and non-target trials in single-speaker scenario and found that the averaged non-target score (0.043) of EA-ASP-M (EA-Mode) is a little higher than the averaged non-target score (0.016) of baseline system, which might be the reason for the performance drop. We infer that the rise in score comes from the enrollment speaker information leakage and more detailed analysis is given in section 4.3. To alleviate this issue, we ensemble the systems with and without enrollment conditions by using the decision with higher score and list the result in the bottom part of Table 1. After ensemble, the systems with EA-ASP layer still show great superiority in multi-speaker scenario and have comparable performance with baseline system in single-speaker scenario.

Besides, we also plot the score distribution under different scenarios for EA-ASP-M system in Figure 2b. Compared to the baseline r-vector system, the multi-speaker target score distribution from our EA-ASP system is much closer to the single-speaker target score distribution, which also indicates the EA-ASP-M system can successfully extract enrollment speaker information from the test utterance with interference.

### 4.3. Effect of bottleneck layer

As introduced in section 2.3, the speaker information in the enrollment embedding may leak to the final test embedding and cause the model to be optimized in the wrong direction. In this section, we evaluated the effect of bottleneck dimension on the information leakage. Here, we add a speaker classification head on the statistic pooling result of the intermediate score $\mathbf{s}_t$ in section 2.1.1 and use the speaker classification accuracy to reflect the speaker information existing in the score $\mathbf{s}_t$. It should be mentioned that the gradient is truncated between the score $\mathbf{s}_t$ and the classification head, so that the classification training will not influence the other parts of the system. The corresponding results are listed in Table 2. As expected, as the bottleneck dimension increases, more speaker information will leak to the score $\mathbf{s}_t$. We found that it is best to set the bottleneck dimension to 2, which can avoid too much speaker information leakage and maintain the system performance at the same time.

Table 2: ***Bottleneck Dimension Ablation Study on EA-SAP-M System.***

| BottleNeck Dimension | Speaker Acc (%) on score $\mathbf{S}$ | EER (%) | | |
|---|---|---|---|---|
| | | Vox_O | Vox_E | Vox_H |
| 1 | 0.05 | 6.800 | 6.473 | 8.797 |
| 2 | 0.30 | **5.212** | **4.988** | **7.067** |
| 4 | 2.10 | 6.430 | 5.811 | 7.941 |
| 8 | 15.8 | 6.632 | 6.055 | 8.212 |

## 5. Conclusion

In this paper, we proposed enroll-aware attentive statistic pooling to solve the target speaker verification problem and achieved a great improvement when facing multi-speaker overlapped utterances. Besides, compared to the method which leverages the speech separation technology, our proposed architecture is more light-weight and can be integrated with most mainstream speaker verification systems. To the best of our knowledge, it is the first work using enrollment embedding as a prior information in speaker verification system.

## 6. Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2021ZD0201504, in part by China NSFC projects under Grants 62122050 and 62071288, and in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102.

# 7. References


[1] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2014, pp. 4052–4056.

[2] Y. Liu, T. Fu, Y. Fan, Y. Qian, and K. Yu, "Speaker verification with deep features," in *2014 International joint conference on neural networks (IJCNN)*. IEEE, 2014, pp. 747–753.

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

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

[5] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[6] S. Wang, J. Rohdin, O. Plchot, L. Burget, K. Yu, and J. Černockỳ, "Investigation of specaugment for deep speaker embedding learning," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7139–7143.

[7] Z. Meng, Y. Zhao, J. Li, and Y. Gong, "Adversarial speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6216–6220.

[8] J. Zhou, T. Jiang, L. Li, Q. Hong, Z. Wang, and B. Xia, "Training multi-task adversarial network for extracting noise-robust speaker embedding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6196–6200.

[9] Z. Chen, S. Wang, Y. Qian, and K. Yu, "Channel invariant speaker embedding learning with joint multi-task and adversarial training," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6574–6578.

[10] N. Tawara, A. Ogawa, T. Iwata, M. Delcroix, and T. Ogawa, "Frame-level phoneme-invariant speaker embedding for text-independent speaker recognition on extremely short utterances," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6799–6803.

[11] S. Wang, J. Rohdin, L. Burget, O. Plchot, Y. Qian, K. Yu, and J. Cernockỳ, "On the usage of phonetic information for text-independent speaker embedding extraction." in *Interspeech*, 2019, pp. 1148–1152.

[12] W. Rao, C. Xu, E. S. Chng, and H. Li, "Target speaker extraction for overlapped multi-talker speaker verification," *arXiv preprint arXiv:1902.02546*, 2019.

[13] C. Xu, W. Rao, J. Wu, and H. Li, "Target speaker verification with selective auditory attention for single and multi-talker speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2696–2709, 2021.

[14] C. Zhang, M. Yu, C. Weng, and D. Yu, "Towards robust speaker verification with target speaker enhancement," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6693–6697.

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

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

[17] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699.

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