Journal of Computer Science and Cybernetics, V.40, N.1 (2024), 67–77
DOI no 10.15625/1813-9663/18248

THE NO_TRAIN_NO_GAIN SYSTEM FOR O-COCOSDA AND VLSP 2022 - A-MSV SHARED TASK: ASIAN MULTILINGUAL SPEAKER VERIFICATION

NGOC-DUNG NGUYEN*, NHAT-NAM LY, TRONG-KHANH LE

School of Information and Communication Technology
Hanoi University of Science and Technology, Ha Noi, Viet Nam

Abstract. This paper proposes a semi-supervised multilingual speaker verification (MSV) system submitted for the 2 tasks, MSV for the Asian language inside the training set (T01) and outside the training set (T02) in O-COCOSDA and VLSP challenge 2022. To solve the problem, our strategy is training a baseline acoustic model with given labeled data (MSV CommonVoice) and fine-tuning the trained acoustic model with both given labeled data and given unlabeled data (MSV Youtube). To achieve the fine-tuning step, the unlabeled data is converted to labeled data by pseudo labeling technique using the clustering method with the embedding vectors extracted from the trained acoustic model. Besides, we also apply test-time augmentation, back-end scoring, and score normalization with the AS-Norm technique to improve the result. When evaluated on the VLSP 2022 challenge's given test set, our best system with baseline ECAPA-TDNN achieves an equal error rate (EER) of 2.296% in T01 and 3.3296% in T02, which ranks second rank in both two tasks.

Keywords. Speaker verification; ECAPA-TDNN; GMM; Fine-tuning; Score normalization.

Abbreviations

MSV	Multilingual speaker verification
VLSP	Vietnamese language and speech processing
O-COCOSDA	Oriental chapter of coordination and standardization of speech databases and assessment techniques
ECAPA	Emphasized channel attention, propagation, and aggregation
TDNN	Time delay neural network
CNN	Convolutional neural network
EER	Equal error rate
MUSAN	Music, speech, and noise
RIRs	Room impulse response
AAM-softmax	Additive angular margin softmax
LMCL	Large margin cosine loss
GMM	Gaussian mixture model
PLDA	Probabilistic linear discriminant analysis
AS-Norm	Adaptive symmetric normalization
MFCC	Mel-frequency cepstral coefficient

*Corresponding author.
E-mail addresses: dungnasa10@gmail.com (N.D. Nguyen); nhatnam04042002@gmail.com (N.N. Ly); khanh.lt2669@gmail.com (T.K.Le)

1. INTRODUCTION

Speaker verification is an important bio-metric problem attracting significant attention from the research community and industry due to its urgent applications in practice. The objective of speaker verification is the authentication of a claimed identity from measurements on the voice signal. Therefore, a general speaker verification system consists of two main steps: registration and verification. First, the user registers one or several voice samples. A background model must be created to capture the speaker-related information, also known as the speaker's acoustic features. These features are considered the "voice signature" of each person and will be saved in the database. In the verification process, the user's voice sample will be provided as query utterances and extracted features by the system. The user is identified by comparing the features with the stored existing acoustic feature.

Recently, with the development of deep learning, many neural network-based speaker models have been proposed for effective feature learning for verifying speaker's utterances. Deep architecture has mostly been treated as black boxes, some approaches have been presented for feature extraction and demonstrated promising results.

In this work, we use ECAPA-TDNN and ECAPA CNN-TDNN as a backbone feature extracting model. Experiments are conducted on the labeled MSV Common Voice dataset and the unlabeled MSV YouTube dataset. Our method is first trained on the Common Voice dataset and further fine-tuned on the YouTube dataset. Augmentation will be applied with the MUSAN corpus for noise and sample simulated filters for RIRs. The results are calculated by the cosine similarity method along with the AS-Norm technique which is also reviewed. The rest of the paper is organized as follows. Section 2. gives information about the speech corpus. Section 3. describes our proposed system. The experimental details are shown in Section 4. while the experimental results are outlined in Section 5. Finally, conclusions are presented in Section 6.

2. DATASETS AND DATA AUGMENTATION

2.1. Provided datasets

The provided training set consists of two datasets: MSV Common Voice data and MSV YouTube data. Both datasets contain utterances from 9 languages, of which 7 languages are from Asia: English, French, Uzbekistan, Hindi, Tamil, Chinese, Japanese, Vietnamese, and Thai. In terms of MSV CommonVoice data, after removing some non-existing audios from the Tamil metadata file, we obtained 592840 utterances and 17714 speakers in total. The number of utterances per speaker ranges from 1 to over 40000. Meanwhile, the MSV YouTube data set contains 73542 utterances without speaker ID. For both public and private tests, T01 consists of utterance pairs of 4 languages (French, Japanese, Thai, and Vietnamese) while T02 contains test pairs of 3 languages (Arabic, Indonesian, and Mongolian). For detailed statistics about each dataset, please refer to Table 1, 2, 3, and 4.

2.2. Data augmentation

Augmentation increases the amount and diversity of the training data, which helps reduce overfitting. We employ two of the popular augmentation methods in speech processing –

Table 1: MSV Common Voice data statistics

Language	# Speaker	# Utterances
English	3975	100991
French	2495	90035
Hindi	226	9189
Japanese	465	29447
Tamil	325	108397
Thai	5515	126058
Uzbekistan	901	79704
Vietnamese	96	3859
Chinese	3716	45160

Table 2: MSV YouTube data statistics

Language	# Utterances
English	7116
French	7643
Hindi	7999
Japanese	8413
Tamil	8522
Thai	7439
Uzbekistan	8405
Vietnamese	8894
Chinese	9111

Table 3: Public_test statistics

Task	# Utterances	# Test pairs
French	2896	15337
Japanese	4431	15488
Thai	2257	15352
Vietnamese	3771	15339
Arabic	2301	15444
Indonesian	1471	15304
Mongolian	3203	15298

Table 4: Private_test statistics

Task	# Utterances	# Test pairs
French	3022	15337
Japanese	4766	15488
Thai	2381	15352
Vietnamese	4000	15339
Arabic	2602	15444
Indonesian	1411	15304
Mongolian	3431	15298

additive noise and room impulse response (RIR) simulation. For additive noise, we use the additive noise and music subsets of the MUSAN corpus [14]; for RIRs, we sample the simulated filters of small and medium rooms released in [9].

2.2.1. Offline data augmentation

For each speaker in MSV Common Voice data having less than 6 utterances, we use RIRs and MUSAN to create extra augmented copies of his/her randomly selected utterances. After this augmentation, 620363 utterances from 17714 speakers are generated to extract acoustic features.

2.2.2. Online data augmentation

We apply an efficient implementation of the augmentation methods so that they can be performed online in the data loader. Because the augmented version of the dataset does not need to be stored, this allows the models to be trained without the need for a large amount of storage. Moreover, this allows different noises and RIR filters to be applied at every epoch, therefore, allowing the creation of unlimited variations of the utterances during the training model. SpecAugment, which is introduced in [11], is also applied in this implementation.

It is worth mentioning that offline augmented data is also adopted in online augmentation but with a smaller chance of getting augmentation. To increase the diversity of training data as well as the balance between the augmentation data and the original data, after trial and error, we chose the ratio for offline data augmentation and online data augmentation to be 30% and 65%, respectively.

3. SYSTEM DESCRIPTION

We adapt the same system settings for both T01 and T02. Figure 1 and 3 describe the overview of our solution to these challenges.

3.1. Network structures

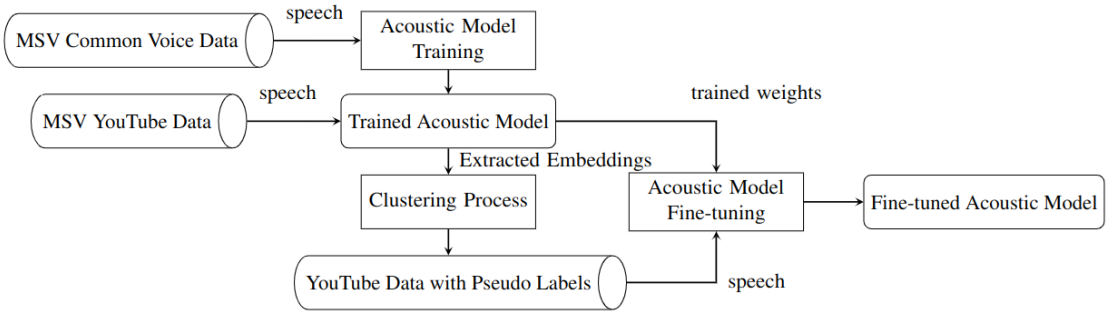

Figure 1: Training protocol

3.1.1. Backbone (Acoustic model)

We use ECAPA-TDNN as the backbone model in our experiments [3]. This architecture is an enhanced version of the popular x-vector topology [15]. The use of hierarchi-

cally grouped convolutions [4] reduces the model parameter count. It also introduces 1-dimensional TDNN-specific Squeeze-and-Excitation-blocks [6] which rescale the intermediate time context-bound frame-level features per channel according to global utterance properties. The pooling layer uses a channel and context-dependent self-attention mechanism to attend to different speaker-characterizing properties at different time steps for each feature map. Finally, Multi-layer Feature Aggregation [5] provides additional complementary information for the statistics pooling by concatenating the final frame-level features with the intermediate features of previous layers.

3.1.2. Loss function

- **Additive angular margin softmax (AAM-softmax)** [2]. Introduce a concept of margin between classes to increase inter-class variance.
- **Large margin cosine loss (LMCL)** [17]. Reformulate the softmax loss as a cosine loss by L2 normalizing both features and weight vectors to remove radial variations, based on which a cosine margin term is introduced to further maximize the decision margin in the angular space. As a result, minimum intra-class variance and maximum inter-class variance are achieved by virtue of normalization and cosine decision margin maximization.

3.2. Training protocol

All our models are trained through three stages. Firstly, we train our models on MSV Common Voice data. After that, the trained models are applied to MSV YouTube data to extract embeddings, do clustering, and get pseudo labels. Then, we fine-tune our models with these pseudo labels.

3.2.1. Training with MSV common voice data

In the first stage, we train the model ECAPA-TDNN on the MSV Common Voice data using AAM-softmax or LMCL as the loss function. The detail of our training configurations is presented in Subsection 4.3. and Table 5.

3.2.2. Clustering with MSV YouTube data

To use YouTube data to fine-tune the model, we decided to apply a clustering method to convert this unlabeled data to the labeled ones which was in the same form as the input of MSV Common Voice data. The overview of our clustering process is presented in Figure 2.

First, in each folder which is named after the URL of the YouTube video containing the preprocessed audios of speakers, we use the Gaussian mixture model (GMM) [12] as an approach to divide similar audios into clusters that are considered as the speakers and we do this for each language separately. By experiment and experience, we realize that clustering for each YouTube video brings a better result than clustering for combining all the videos of each language due to observing directly the number of speakers obtained in both two methods. The number of clusters parameter of the GMM algorithm is set to 3 based on observing some samples from the data set.

However, some clusters may have noisy audios as well as can be overlapping with other clusters since a speaker might appear in more than one video. To deal with the first problem,

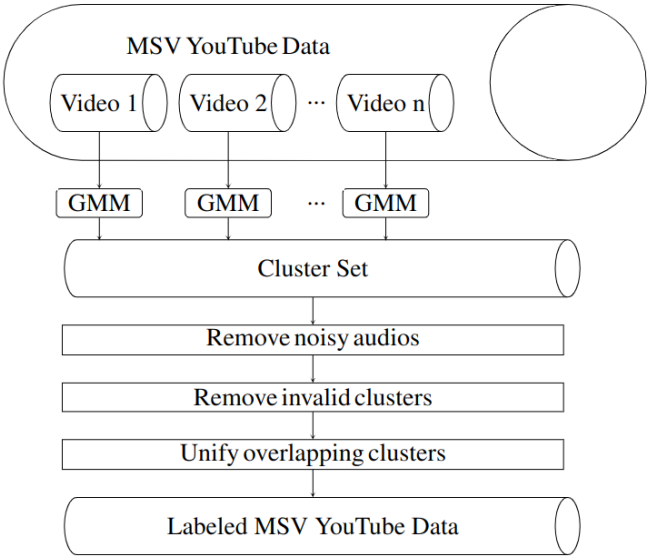

Figure 2: Clustering process

every utterance having more than one cosine score with other utterances inside this cluster smaller than a threshold of 0.3 is considered noisy audio and will be marked to be removed. After that, every cluster having the mean of its cosine similarity matrix lower than the threshold (default 0.45) will be also removed. Finally, we use a cosine score matrix belonging to different clusters to compare with each other. The score of two speakers is the average of the cosine similarity matrix between them which is evaluated by a threshold given by plotting the distribution of the cosine matrix. If the average cosine score of 2 clusters is higher than a threshold of 0.7, they are combined into one speaker. All the above threshold is decided by trial and error based on hearing some utterances of the obtained clusters. After doing all these steps, we have 3206 speakers with 44638 utterances in total and the number of utterances per speaker ranges from 1 to 933. Our idea is inspired by this paper [16].

3.2.3. Fine-tuning with MSV YouTube data

After the clustering process, we obtain the pseudo-label for the MSV YouTube data. We use this data to fine-tune the pre-trained model in the first stage to obtain the final model which is used for testing. The details of training setups are shown in Section 4.3.

Table 5: Sub-system configurations

Index	Configuration
S1	ECAPA-TDNN + AAM-softmax + Cosine
S2	ECAPA-TDNN + AAM-softmax + PLDA
S3	ECAPA-TDNN + AAM-softmax + Cosine + AS-Norm
S4	ECAPA-TDNN + LMCL + Cosine
S5	ECAPA-TDNN + LMCL + Cosine + AS-Norm
S6	ECAPA-TDNN + LMCL + Fine-tune + Cosine + AS-Norm

Table 6: EER(%) of our submissions to O-COCOSDA and VLSP 2022 - A-MSV shared task.

Index	T01 Public	T02 Public	T01 Private	T02 Private
S1	1.6041	2.814	-	-
S2	2.074	2.834	-	-
S3	1.3875	2.6577	-	-
S4	0.7547	1.3858	2.6328	3.7529
S5	0.6832	1.1944	2.5886	3.3908
S6	**0.6047**	**0.9835**	**2.296**	**3.3296**

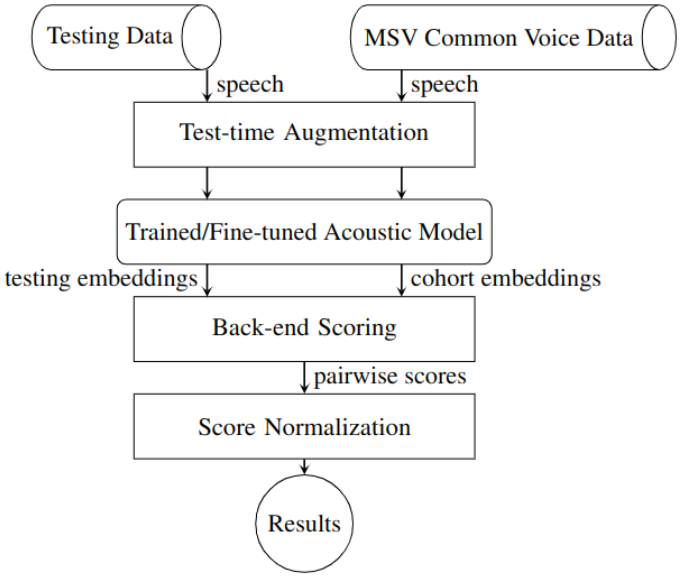

Figure 3: Testing protocol

3.3. Testing protocol

3.3.1. Test-time augmentation

We first extract the embedding vectors of full audio and calculate the similarity score between the two vectors for each test pair. Moreover, we sample ten 3-second temporal segments at regular intervals from every segment in the test set and compute the $10 \times 10 = 100$ similarities for every pair using all possible combinations of segments. The average of the full audio similarity and the mean of the 100 similarities is used as the final pairwise score.

3.3.2. Back-end scoring

Two different methods are applied as a back-end for two tasks. Firstly, we utilize the Cosine similarity score to compute the distance between the two embedding vectors. Besides that, we also use the PLDA [8] classifier for both models. We use LDA as a dimensionality reduction technique and the LDA dimension was set to 100. After that, the vector representations are length-normalized and modeled by PLDA.

3.3.3. Score normalization

To achieve better results, score normalization is applied to reduce within-trial variability. This section describes the techniques in which the score between enrollment utterance e and test utterance t is denoted $s(e,t)$.

- Z-norm [13]: Zero score normalization employs impostor score distribution for enrollment files. N speakers $\mathcal{E} = \{\varepsilon_i\}_{i=1}^{N}$ to be different from e and t will be used as cohort samples for calculation. The cohort scores are formed by scoring utterance e and all utterances in the cohort set. The formula is

$$S_e = \{s(e,\varepsilon_i)\}_{i=1}^{N}. \tag{1}$$

The normalized score is then

$$s(e,t)_{z-norm} = \frac{s(e,t) - \mu(S_e)}{\sigma(S_e)}. \tag{2}$$

- T-norm [1]: Test score normalization is different from the Z-score as it normalizes the impostor score distribution for the test utterance. The scores can be computed by the same formula as the Z-score

$$S_t = s(t,\varepsilon_i), \tag{3}$$

$$s(e,t)_{t-norm} = \frac{s(e,t) - \mu(S_t)}{\sigma(S_t)}. \tag{4}$$

- S-norm [7]: The symmetric normalization computes an average of normalized score from Z-norm and T-norm. Z-norm and T-norm depend on the order of e and t while the S-norm function normalized with symmetric normalization recipes $s(e,t)$ and $s(t,e)$. The S-norm score is formed as

$$s(e,t)_{s-norm} = \frac{1}{2}\left(\frac{s(e,t) - \mu(S_e)}{\sigma(S_e)} + \frac{s(e,t) - \mu(S_t)}{\sigma(S_t)}\right). \tag{5}$$

- Adaptive score normalization [10]: In adaptive score normalization techniques, the normalization score is computed depending on part of the cohort. Z-norm, T-norm, and S-norm are approached with the same cohort selection while the selected cohort might change in adaptive techniques. There are ways to select an adaptive cohort: either selected to be X most positive scores samples to the enrollment utterance or the test utterance. Therefore, each enrollment utterance e and test utterance t have different cohort sets, and scores based on cohorts for the enrollment can be computed as follows

$$S_e(\mathcal{E}_e^{top}) = \{s(e,\epsilon)\}_{\forall\epsilon\in\mathcal{E}_e^{top}}, S_e(\mathcal{E}_t^{top}) = \{s(e,\epsilon)\}_{\forall\epsilon\in\mathcal{E}_t^{top}} \tag{6}$$

and correspondingly for the test utterance t. The adaptive S-norm score for the first is

$$s(e,t)_{s-norm} = \frac{1}{2}\left(\frac{s(e,t) - \mu(S_e(\mathcal{E}_e^{top}))}{\sigma(S_e(\mathcal{E}_e^{top}))} + \frac{s(e,t) - \mu(S_t(\mathcal{E}_t^{top}))}{\sigma(S_t(\mathcal{E}_t^{top}))}\right). \tag{7}$$

We utilize an Adaptive S-norm for the normalization of $s(e,t)$ which is computed by cosine similarity score. The imposter cohort includes more than 200000 utterances from the training dataset which will be selected from the top 1000 scores to compute mean and standard deviation for each enrollment and test utterance in a trial pair.

4. EXPERIMENTAL DETAILS

4.1. Backbone model and loss function

We conduct all of our experiments using ECAPA-TDNN as the backbone model. There are two variants proposed in the paper, but we use the larger model with $C = 1024$.

Regarding the loss function, we select a margin of 0.2 and a scale of 30 for AAM-softmax, since these values give the best results on the VoxCeleb1 test set as shown in the paper. We use a margin of 0.35 and a scale of 64 for LMCM in this work.

4.2. Input representations

During training, we use random 2-second temporal segments extracted from each utterance. Pre-emphasis with a coefficient of 0.97 is applied to the input signals. The spectrograms are extracted with a hamming window of 25ms width and 10ms step size. The input features are 80-dimensional Mel-frequency cepstral coefficients (MFCCs) extracted from the spectrograms. The features are mean normalized at the input to the network.

4.3. Training details

As mentioned in Subsection 3.2.1, when training with the MSV Common Voice data), the Adam optimizer with weight decay of 2e-5, the exponential decay rate for the 1^{st}-moment estimates of 0.9 and 2^{nd}-moment estimates of 0.999 are used. We use a single GPU GTX 2080 Ti with 64 mini-batch and an initial learning rate of 0.001 to train all of our models. 200 frames of each sample in one batch are adopted to avoid over-fitting and speed up training. The learning rate is reduced by 5% every 2 epochs. The network has been trained for 100 epochs. Finally, we set the maximum utterances per speaker to 200 when training to avoid much imbalance between speakers and also speed up the training process.

As described in Subsection 3.2.3, the settings for the final stage in our proposed method are slightly different from the first stage. We first let the last fully connected layer warm up by freezing all previous layers for 5-10 epochs and then trained the entire network with an initial small learning rate of 5e-5 with a decay factor of 0.1. Since the amount of data and the number of utterances per speaker are not too large, we did not set the maximum utterances per speaker when training with MSV Common Voice data.

5. EXPERIMENT RESULTS

In this section, we show our submissions to the AMSV-VLSP 2022 Challenge in two tasks: T01 and T02. The baseline system is an ECAPA-TDNN backbone followed by AAM-Softmax. The performance is evaluated using the equal error rate (EER). As Table 6 shows, our baseline system performs better when using the Cosine back-end compared to use PLDA back-end, and its performance is improved significantly with the help of AS-Norm. The EER is also decreased considerably when we use ECAPA CNN-TDNN backbone and it gets the most promising result with ECAPA-TDNN backbone followed by LMCL, Cosine back-end, and AS-Norm. By fine-tuning with MSV YouTube data, our best system achieves EER of 2.296% in T01 and 3.3296% in T02, which ranks second rank in both two tasks.

6. CONCLUSION

In this paper, we describe our method submitted to the O-COCOSDA and VLSP 2022 - A-MSV Shared task: Asian Multilingual Speaker Verification. ECAPA-TDNN has been an outperformed backbone for embedding extractors. It is trained on MSV Common Voice data and fine-tuned on MSV YouTube data, both based on the LMCL. We also apply the test-time augmentation technique, Cosine similarity back-end, and AS-Norm for score normalization. The system shows superiority and results in 2.296% and 3.330% for EER respectively which ranks second place on both T01 and T02 of the O-COCOSDA and VLSP 2022 - A-MSV Shared task: Asian Multilingual Speaker Verification.

REFERENCES

[1] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, "Score normalization for text-independent speaker verification systems," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 42–54, 2000.

[2] 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.

[3] 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.

[4] 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.

[5] Z. Gao, Y. Song, I. McLoughlin, P. Li, Y. Jiang, and L.-R. Dai, "Improving aggregation and loss function for better embedding learning in end-to-end speaker verification system." in *INTERSPEECH*, 2019, pp. 361–365.

[6] 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.

[7] P. Kenny, "Bayesian speaker verification with, heavy tailed priors," *Proc. Odyssey 2010*, 2010.

[8] P. Kenny, T. Stafylakis, P. Ouellet, M. J. Alam, and P. Dumouchel, "Plda for speaker verification with utterances of arbitrary duration," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7649–7653.

[9] 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.

[10] P. Matejka, O. Novotnỳ, O. Plchot, L. Burget, M. D. Sánchez, and J. Cernockỳ, "Analysis of score normalization in multilingual speaker recognition," in *Interspeech*, 2017, pp. 1567–1571.

[11] 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.

[12] D. A. Reynolds, "Gaussian mixture models." *Encyclopedia of Biometrics*, vol. 741, no. 659-663, 2009.

[13] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

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

[15] 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.

[16] D. V. Thanh, T. P. Viet, and T. N. T. Thu, "Deep speaker verification model for low-resource languages and vietnamese dataset," in *Proceedings of the 35th Pacific Asia Conference on Language, Information and Computation*, 2021, pp. 445–454.

[17] Q. Wang, K. A. Lee, and T. Liu, "Scoring of large-margin embeddings for speaker verification: Cosine or PLDA?" *arXiv preprint arXiv:2204.03965*, 2022.

Received on December 25, 2023
Accepted on February 19, 2024