INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Instance-based Temporal Normalization for Speaker Verification

Thanathai Lertpetchpun, Ekapol Chuangsuwanich

Department of Computer Engineering, Chulalongkorn University, Thailand
`thanathai.lertpetchpun@gmail.com, ekapol.c@chula.ac.th`

Abstract

One of the challenges in speaker verification is domain mismatch and other effects such as language and emotion. Normalization techniques such as Batch Normalization (BN) have been proven effective in improving neural network training and are a popular choice in many speaker verification networks. However, BN may not be able to adequately normalize the feature map for speaker verification. In this work, we investigate several instance-based normalization methods which are more suitable for speaker verification. We propose the Temporal Normalization layer, which normalizes along the time dimension, and show its effectiveness on four different datasets. Experiments on VoxCeleb2 show a relative improvement of 24.3% and 46.15% in terms of EER and DCF over fwSE-ResNet34 in VoxCeleb1-O. Furthermore, we present a systematic evaluation of our networks against three other datasets, namely Thai-Central, THAI-SER, and CREMA-D to show its robustness on language and emotional variants.

Index Terms: Deep Neural Network, Speaker Verification, Normalization Layer

1. Introduction

Speaker verification is the task of validating the identity of a person based on their speech signal. Recent developments in speaker verification have been dominated by deep-learning-based approaches. For example, Emphasized Channel Attention, Propagation and Aggregation in time-delay neural network (ECAPA-TDNN) [1], a network based on one-dimensional convolution, is a popular baseline. More recent architectures incorporate ideas from the computer vision community, such as ResNets [2], and squeeze-and-excitation networks [3, 2].

Normalization layers have been shown to improve the robustness and performance of neural networks in many fields. Batch Normalization (BN) [4] and to a lesser extent Instance Normalization (IN) [5] have been a staple for CNN models especially for computer vision. Layer Normalization (LN) [6] has been shown to improve the performance of networks such as recurrent neural networks and transformers in natural language processing [7]. Recently, [8] proposed Relaxed Instance Frequency-wise Normalization (RFN), a normalization performed on the input layer, to combat domain mismatch for the task of scene classification and speaker verification. However, no systematic evaluation has been performed to measure the effectiveness of different normalization layers on the task of speaker verification and their effect on robustness.

Frame-level feature extraction has been shown to be effective for speaker verification on short utterances [9]. In this work, we introduce a new normalization module called Temporal Normalization (TN), which is an instance normalization that calculates the statistics on the temporal dimension. This can be considered an ensemble of multiple frame-level feature extractors. Although normalization has been common in neural networks, TN has never been considered. The motivation for TN is slightly different from other normalization layers. Temporal normalization tries to decouple the features from different frames in order to create a frame-based ensemble-like model. Applying TN can be done by replacing all the normalization layers with TN and can also be combined with other normalization modules for further improvements.

The proposed methods were evaluated in matched and mismatched conditions using four datasets: VoxCeleb2, Thai-Central, THAI-SER, and CREMA-D. On VoxCeleb2, our methods outperformed state-of-the-art baselines by a relative score of 46.15%. Our model exhibits less degradation when evaluated on mismatch conditions such as emotional speech and different languages. Moreover, further analysis of the model embeddings has shown that TN can help remove emotional information.

2. Background on Normalization Layers

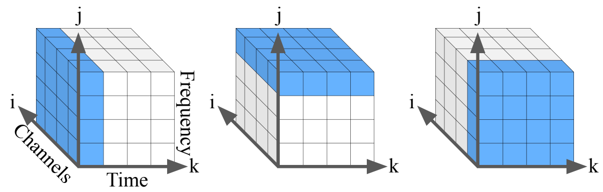
Normalization layers are an essential part of a network that have been shown to help improve convergence and accuracy. Several normalization techniques have been proposed, such as Batch Normalization (BN) [4], Instance Normalization (IN) [5], Group Normalization (GN) [10], and Layer Normalization (LN) [6]. Each differs mostly in how the mean and variance are calculated. For the speaker verification task, instance-based normalization is preferable since it does not require batch processing.

Recently, Kim et al. proposed two normalization layers: Residual Normalization (RN) [11] and Relaxed Instance Frequency-wise Normalization (RFN) [8] for the acoustic scene classification task. RN can be described as $RN(x) = \lambda x + FN(x)$ where x is the input to the layer, FN refers to instance normalization that normalizes along the frequency dimension, and λ is an interpolation weight. RFN is a linear interpolation between features from LN and FN, namely $RFN(x) = \lambda LN(x) + (1 - \lambda)FN(x)$ where LN is Layer Normalization. These normalization techniques were performed on the input layer to help mitigate domain mismatch. The interpolation helps maintain information that might be destroyed when normalized across frequencies.

3. Method

3.1. Temporal Normalization

[8] shows that the feature along the frequency dimension carries more domain-relevant information than the channel dimension. However, it has been shown that for speaker verification, temporal (or frame-level) features also carry speaker-relevant infor-



(a) *TN* (b) *FN* (c) *IN*

Figure 1: *Different **instance-based** normalization methods. Each cube represents a single instance. The highlighted pixels are normalized with the same mean and variance, calculated by aggregating the values of these pixels.*

mation [9].

Temporal Normalization is an instance-based normalization layer that tries to normalize in the temporal dimension. Consider a three dimensional feature map input to the normalization module, $x_{ijk} \in \mathbb{R}^{\mathrm{C}\times\mathrm{F}\times\mathrm{T}}$, with C, F, T denoting the number of channel, frequency, and time dimensions, respectively. TN is computed as follows:

$$\hat{x}_{ijk} = \frac{x_{ijk} - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}} \tag{1}$$

where $\hat{x}_{ijk}$ is the output from the normalization module, μ_k and σ_k^2 is the average and variance of that time dimension, and ϵ is a small number to avoid division by zero. The layer also includes an element-wise affine transform step. Figure 1 illustrates the differences between instance-based normalization methods. For Frequency Normalization (FN), the μ_k and σ_k^2 are replaced with μ_j and σ_j^2, respectively. Note that TN should not be confused with ceptral mean normalization (CMN), which averages the MFCC over the time dimension. This is more analogous to FN than TN. These normalization techniques can also be applied together since they are designed to handle different aspects of speech processing.

3.2. Relaxed Temporal-Frequency Normalization

Similar to the RFN, we can also combine TN with another normalization module to preserve information that has been lost in the temporal dimension. As a result, we propose Relaxed Temporal-Frequency Normalization (RTFN), which can be described as $\hat{x} = \lambda TN(x) + (1-\lambda)FN(x)$, where λ is an interpolation weight hyperparameter. This alleviates the loss of information from the normalization in the temporal and frequency dimensions.

3.3. Incorporating normalization layers

These normalization layers can be included anywhere to replace the typical BN layers in ResNet-like models. For the attention layer in the attentive statistics pooling layer [12], there is a BN layer which cannot be replaced as described above because the input of the layer only has two dimensions, [Channel×Frequency, Time]. However, similar to Equation 1, we can still normalize using the time dimension.

All BN layers in the ResNet model are replaced with our normalization layers, unlike [8] which normalizes only the input. Moreover, we reduce the number of normalization layers and activation layers according to [13] which decreases the training time while maintaining accuracy.

4. Experimental Setups

4.1. Datasets

We performed experiments on four datasets, namely VoxCeleb, Thai-Central, THAI-SER, and CREMA-D. Our network was trained on VoxCeleb2 [14] consisting of 5994 speakers with the provided train list (only the dev partition). VoxCeleb1 [15] (1,251 speakers) was used as a test set with the provided three test lists (original, extended, and hard) [16].

Thai-Central [17] is a read speech dataset for automatic speech recognition collected via crowdsourcing. It was used to test the effectiveness of our methods in a different language and was also used as a cross-language test set. The original training dataset consists of 5,476 speakers which is comparable to VoxCeleb2. However, the total length of the dataset is only 782 hours which is less than half of VoxCeleb2. Only speakers with more than five utterances were included. We randomly split the data into train, development, and test sets, resulting in 4,928, 55, and 493 speakers, respectively.

We also tested our models on two speech emotion datasets. THAI-SER dataset[1] is a Thai dataset consisting of 200 speakers. The dataset was collected by pairing actors together. The actors were tasked with improvising different situations illustrating five emotions (neutral, anger, happiness, sadness, and frustration). Besides improvised situations, there is also a small portion of scripted utterances. There are three recording environments (two studios and a Zoom-based recording), which made for a challenging dataset. We selected only utterances longer than three seconds and treated them as a test set. The other emotional speech dataset is CREMA-D [18] which is a multimodal dataset with multiple emotions per speaker. The dataset comprises 91 actors with six emotions (anger, disgust, fear, happiness, neutral, and sadness) per actor. Only the audio was used to perform speaker verification. Since a speaker with different emotions can have different acoustic parameters such as pitch and fundamental frequencies, the THAI-SER and CREMA-D datasets, which introduce emotion variations, make verifying a speaker more difficult.

4.2. Baselines and base architectures

4.2.1. ECAPA-TDNN

The ECAPA-TDNN network[2] [1, 19] is an improved version of x-vector [20]. The network starts with a 1-dimensional convolution layer, followed by three layers of 1-dimensional Squeeze and Excitation residual blocks. Three output maps from three layers are concatenated before feeding into the Attentive Statistics Pooling layer [12]. The final layers are fully connected layers in order to create an embedding vector. The network comes with BN layers in residual blocks.

4.2.2. ResNet-based Architectures

The ResNet architecture is a popular choice for speaker embedding extraction. In our work, we focus primarily on SE-ResNet34. The network starts with a 2-dimensional convolutional layer. The output feature map is fed through the four stages. In each stage, there are 2-dimensional squeeze and excitation layers [21] with BN [4]. After the last stage, the Attentive Statistics Pooling layer [12] is used to summarize the information.

[1]https://github.com/vistec-AI/dataset-releases/releases/tag/v1
[2]https://github.com/TaoRuijie/ECAPA-TDNN

Table 1: *Evaluation results on four datasets: VoxCeleb1, Thai-Central, THAI-SER, and CREMA-D. The model was trained on VoxCeleb2. Rows with the "+" symbols use fwSE-ResNet34 as the base model.*

	VoxCeleb1-O		**VoxCeleb1-E**		**VoxCeleb1-H**		**Thai-Central**		**THAI-SER**		**CREMA-D**	
	EER	**DCF**	**EER**	**DCF**	**EER**	**DCF**	**EER**	**DCF**	**EER**	**DCF**	**EER**	**DCF**
ECAPA TDNN	1.16	0.151	1.27	0.149	2.42	0.252	9.25	0.540	7.96	0.556	13.67	0.847
SE-ResNet34	1.15	0.120	1.34	0.162	2.53	0.244	8.06	0.498	6.46	0.480	13.55	0.831
fwSE-ResNet34	1.07	0.130	1.23	0.144	2.39	0.230	7.58	0.452	5.64	0.427	13.04	0.831
+ RFN (input) [8]	1.11	0.127	1.19	0.140	2.23	0.219	7.59	0.420	5.74	0.430	12.81	0.812
+ LN	0.97	0.104	1.13	0.133	2.17	0.211	7.38	0.429	5.48	0.406	12.19	0.788
+ FN	0.99	0.110	1.12	0.129	2.15	0.213	7.64	0.440	5.72	0.424	12.72	**0.774**
+ TN (Ours)	0.89	0.079	1.09	**0.120**	1.99	0.199	7.04	0.433	5.48	0.397	12.63	0.800
+ FN + LN	0.96	0.097	1.12	0.128	2.12	0.209	7.64	0.452	5.53	0.407	12.31	0.782
+ FN + TN (Ours)	**0.81**	**0.070**	**1.03**	0.121	**1.92**	**0.188**	**6.99**	**0.418**	**5.36**	**0.381**	**12.16**	0.777

In this work, we implemented two kinds of ResNet architectures as baselines: basic SE-ResNet34 as described above and fwSE-ResNet34 [2]. FwSE-ResNet34 adds a learnable frequency positional encoding into the input of each residual block and incorporates frequency-wise squeeze and excitation, which re-scale the feature along the frequency dimension instead of the channel dimension.

4.3. Training Details and Evaluation Measures

Our training setup followed [19]. The models were trained on two-second segments randomly extracted from the data. We used Kaldi's Voice Activity Detection based on the Kaldi Aspire recipe ("egs/aspire/s5") to segment the data. The input features to the neural network are 80-dimensional log mel spectrogram with mean normalization, a window size of 25 ms and a frame shift of 10 ms. Following [2], we trained the network with the Adam optimizer [22] and used the cyclic learning rate scheduler (triangle2 policy) with a base learning rate of 1e-8 and a maximum learning rate of 1e-3. We trained the model with a cycle of 30 and 10 epochs for VoxCeleb2 and Thai-Central, respectively. The networks were trained for three full cycles with a mini-batch size of 128. Additive angular margin loss [23] was used to train the network with scale s_{AAM} = 30 and m_{AAM} = 0.2. Moreover, we augmented each chunk of utterance with five types of noise (music, babble, noise, television noise, and reverberation) by combining all the noises in the MUSAN dataset [24] and the RIR dataset [25]. The SpecAugment [26] algorithm was applied to all log mel spectrograms fed into the network and randomly masked 0 to 8 bins in the frequency domain and 0 to 10 frames in the time domain. All systems were implemented in Pytorch [27].

For evaluation, we extracted the log mel spectrogram from the entire utterance and sampled five 3-second equally-spaced segments. If any utterance had less than three seconds, it was padded using 'wrap' mode until it reached three seconds. Cosine similarity was used as the distance metric to compute the score between each embedding pair. The final score was a mean score between chunks of the utterance and the entire utterance. Equal Error Rate (EER) and the minimum decision cost function (DCF) [28] with $P_{target} = 1e{-}2$ with $C_{FA} = C_{miss} = 1$ were used as evaluation metrics.

As the Thai-Central, THAI-SER, and CREMA-D datasets are not standard speaker verification datasets, they do not come with test lists for speaker verification. Thus, we randomly paired the speakers with 4 positive and 4 negative samples for each utterance. Each utterance cannot be paired with itself, and each pair cannot be duplicated. As a result, there are 246,045, 353,480, and 59,040 pairs of speakers, and the average length of each utterance in the test list are 5.84, 4.68, and 1.84 seconds for Thai-Central, THAI-SER, and CREMA-D, respectively. The models were trained on one A100 GPU.

5. Results and discussion

We present the effectiveness of our method through several training and testing scenarios. The first and last subsections use VoxCeleb2 as the training data and evaluate the model in various settings, including other out-of-domain datasets. The second part uses Thai Central, which contains more uniform data compared to VoxCeleb2.

5.1. VoxCeleb and out-of-domain evaluation

The results of the model trained using VoxCeleb2 data are summarized in Table 1. Popular baselines such as ECAPA-TDNN and SE-ResNet34 are included. Another baseline is RFN (input), which is a fwSE-ResNet34 model with RFN applied only to the input instead of a mean normalization. This helps improve the performance on out-of-domain data as claimed in [8]. However, it lowers the performance on the easiest test set (VoxCeleb1-O). Unexpectedly, there is a large drop in performance when evaluating on the Thai datasets. However, the largest performance gap is on CREMA-D because the utterances are much shorter.

The rows with "+" prefixes are fwSE-ResNet34 models with different instance-level normalization techniques. FN+LN refers to the relaxed version of FN and LN with $\lambda = 0.5$ [8] while FN+TN refers to those of FN and TN with $\lambda = 0.7$. As described in Section 3.3, the normalization in the attention pooling layer is always fixed to TN. Using any normalization layer across the entire network yields a noticeable gain over the baselines. Using just TN can improve the performance over strong baselines that use BN or FN proposed recently. On VoxCeleb1-O, TN has a relative improvement of 39.2% and 28.2% in terms of DCF over BN and FN, respectively. This is considered a large improvement over a strong baseline. Moreover, the best performing instance-based normalization is FN+TN, which consistently outperforms on most datasets. Combining multiple normalization techniques provides better results than those with single normalization. Note that FN+LN is essentially RFN, highlighting the impact of applying the normalization throughout the network.

5.2. Thai-Central

Table 2 shows the results of the models trained on Thai-Central. The results show that TN and FN+TN yield are the best performing models. FN+TN performs better on Thai-Central, while TN performs better on THAI-SER. Further investigation revealed that FN+TN performs better on utterances with recording artifacts such as bad microphones or encoding. Thai-Central is collected via crowdsourcing which has some low-quality recordings. Note that the results on THAI-SER are worse than those in Table 1. Since Thai-Central is read speech, the model lacks the robustness to generalize well to emotional recordings in THAI-SER.

Table 2: *Results for models trained on Thai-Central.*

	Thai-Central		**THAI-SER**	
	EER	DCF	EER	DCF
ECAPA TDNN	2.18	0.168	15.46	0.811
SE-ResNet34	2.17	0.158	15.38	0.789
fwSE-ResNet34	2.05	0.165	15.21	0.806
RFN (input) [8]	2.31	0.179	17.62	0.859
+ TN (Ours)	1.91	0.149	**14.06**	**0.787**
+ FN + LN	1.94	0.147	14.88	0.788
+ FN + TN (Ours)	**1.83**	**0.144**	14.49	0.790

5.3. Analysis of the effect on the embeddings

We have conducted experiments using the embedding from various layers to perform speaker verification. Figure 2 shows that the model with Relaxed Temporal-Frequency Normalization is superior to that of Batch Normalization in terms of performance. Moreover, the relative improvement increases as embeddings from higher layers are used. This indicates that FN+TN is better than BN at extracting speaker information.

To quantify the amount of non-speaker related information still presented in the embeddings, we also trained an emotion classifier using logistic regression with the learned speaker embeddings as input features. We split the CREMA-D and the THAI-SER data into training and test sets and created models for each dataset. Classifiers with features from the BN model got accuracy values of 59.61% and 57.57% on THAI-SER and CREMA-D, respectively. On the other hand, their FN+TN counterparts got lower accuracy scores of 54.93% and 55.47%. Thus, the embeddings from BN seem to have more emotional information present. These results corroborate the usefulness of instance-based normalization over Batch Normalization for speaker verification.

6. Ablation Studies

6.1. Attentive Statistics Pooling layer

The effect of replacing the BN in the attentive statistics pooling layer with TN on the model trained on VoxCeleb2 is shown in Table 3. We set $\lambda = 0.5$ for this experiment. In the VoxCeleb1-E, VoxCeleb1-H, Thai-Central, and CREMA-D, TN outperforms BN. However, the results of both models are comparable in VoxCeleb1-O and THAI-SER. Thus, using TN in the pooling layer may improve the robustness in hard conditions.

6.2. Effect of the λ parameter

Figure 3 shows the effect of different λ values on the performance of different datasets. $\lambda = 0$ means that we only use FN.

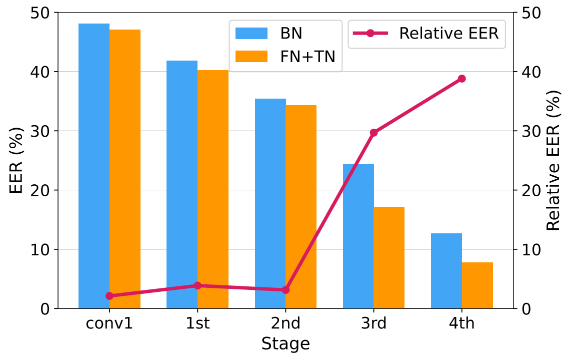


Figure 2: *Evaluation of feature maps from different stages of fwSE-ResNet34 in terms of EER on VoxCeleb1-O. The red line illustrates the relative performance gain of FN+TN over BN.*

Table 3: *The effect of using TN in the attention pooling layer of fwSE-ResNet34+FN+TN.*

	BN		**1d TN**	
	EER	DCF	EER	DCF
VoxCeleb1-O	**0.85**	0.103	0.89	**0.090**
VoxCeleb1-E	1.10	0.124	**1.08**	**0.122**
VoxCeleb1-H	2.10	0.208	**2.01**	**0.201**
Thai-Central	7.71	0.463	**6.98**	**0.408**
THAI-SER	5.71	**0.414**	**5.60**	0.429
CREMA-D	12.19	0.794	**12.17**	**0.775**

We report performance values relative to $\lambda = 0$. The results show that tuning only the λ hyperparameter can improve the performance up to 17.34% in terms of EER. Note that we chose $\lambda = 0.7$ to treat VoxCeleb1-O as the dev set for tuning.

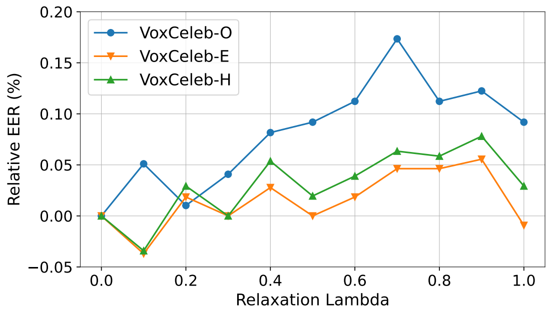


Figure 3: *Effect of the λ parameter in Relaxed Temporal-Frequency Normalization.*

7. Conclusion

This paper investigates the usage of the instance-based normalization layer for the speaker verification task. We introduce Temporal Normalization, Relaxed Temporal-Frequency Normalization, and a strategy to incorporate them in any ResNet-based architectures. The results of the experiments showed that our proposed methods improved the performance and provided additional robustness in mismatch conditions such as language and emotion.

8. References

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

[2] 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 *Interspeech*, 2021, pp. 2302–2306.

[3] 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 *Interspeech*, 2021, pp. 496–500.

[4] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*. PMLR, 2015, pp. 448–456.

[5] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv e-prints*, pp. arXiv–1607, 2016.

[6] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *stat*, vol. 1050, p. 21, 2016.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, vol. 30, 2017.

[8] B. Kim, S. Yang, J. Kim, H. Park, J. Lee, and S. Chang, "Domain Generalization with Relaxed Instance Frequency-wise Normalization for Multi-device Acoustic Scene Classification," in *Interspeech*, 2022, pp. 2393–2397.

[9] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, "Deep speaker feature learning for text-independent speaker verification," in *Interspeech*, 2017, pp. 1542–1546.

[10] Y. Wu and K. He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[11] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design," DCASE2021 Challenge, Tech. Rep., June 2021.

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

[13] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 976–11 986.

[14] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Interspeech*, 2018, pp. 1086–1090.

[15] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Interspeech*, 2017, pp. 2616–2620.

[16] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, vol. 60, p. 101027, 2020.

[17] A. Suwanbandit, B. Naowarat, O. Sangpetch, and E. Chuangsuwanich, "Thai dialect corpus and transfer-based curriculum learning investigation for dialect automatic speech recognition," in *Interspeech*, 2023.

[18] H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova, and R. Verma, "Crema-d: Crowd-sourced emotional multimodal actors dataset," *IEEE transactions on affective computing*, vol. 5, no. 4, pp. 377–390, 2014.

[19] R. K. Das, R. Tao, and H. Li, "HLT-NUS SUBMISSION FOR 2020 NIST conversational telephone speech SRE," *arXiv e-prints*, pp. arXiv–2111, 2021.

[20] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *ICASSP*. IEEE, 2018, pp. 5329–5333.

[21] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *CVPR*, 2018, pp. 7132–7141.

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[23] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *CVPR*, 2019, pp. 4690–4699.

[24] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv e-prints*, pp. arXiv–1510, 2015.

[25] 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 *ICASSP*. IEEE, 2017, pp. 5220–5224.

[26] 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," in *Interspeech*, 2019, pp. 2613–2617.

[27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *NeurIPS*. Curran Associates, Inc., 2019, pp. 8024–8035.

[28] NIST 2018 speaker recognition evaluation plan. https://www.nist.gov/system/files/documents/2018/08/17/sre18_eval_plan_2018-05-31_v6.pdf. Accessed: 2023-03-05.