The NSYSU+CHT System Description for SdSV Challenge 2020

Po-Chin Wang[1], *Chia-Ping Chen*[1], *Chung-Li Lu*[2], *Bo-Cheng Chan*[2], *Shan-Wen Hsiao*[2]

[1]National Sun Yat-Sen University, Taiwan
[2]Chunghwa Telecom Laboratories, Taiwan

m073040069@student.nsysu.edu.tw, cpchen@mail.cse.nsysu.edu.tw
{chungli,cbc,swhsiao}@cht.com.tw

Abstract

In this paper, we describe the system **Team NSYSU+CHT** has implemented for the 2020 Short-duration Speaker Verification Challenge (SdSV 2020). Our single systems are embedding-based speaker verification, with front-end speaker embedding extractors and back-end PLDA verification scorers. The main difference between our system and other well-known systems is that we propose a new architecture, which we call **multi-length context neural network** (MLCNN), for speaker embedding extraction. Essentially, MLCNN combines the bottleneck time-delay neural network (TDNN) blocks and 2D convolution blocks. The motivation is to combine the advantages of both. In our evaluation, MLCNN has outperformed several well-known neural networks. Our best single system achieves an equal error rate (EER) of 4.47% and a minimum detection cost function (minC) of 0.188 on the text-independent track. In addition, our fusion system achieves an EER of 4.21% and 0.1514 of minC on the text-dependent track.

Index Terms: short-duration speaker verification, x-vector, time delay neural network

1. Introduction

In this paper, we present the speaker verification systems developed by the National Sun Yat-sen University and Chunghwa Telecom Laboratories (NSYSU+CHT) team for the Short-duration Speaker Verification (SdSV) challenge 2020 [1]. Our main contribution is to propose and analyze several model architectures with similar costs but better results than conventional methods. The challenge provides benchmarks for text-dependent speaker verification and text-independent speaker verification. It also covers two features. The first feature is that the speaker verification in the challenge focuses on the short duration scenario. The second feature is that there may be different languages between enrollment and test utterances. Therefore, participants need to develop a sufficiently robust system to mitigate the before-mentioned effects.

In recent years, speaker verification systems in many scenarios have adopted embedding-based methods [2, 3, 4] to replace i-vector [5, 6] and become mainstream. The embedding-based method relies on neural networks and a sufficient amount of training data. The embedding neural network architecture can be divided into frame-level layers, pooling, and segment layers. Recently, TDNN-based architecture such as E-TDNN [7] and F-TDNN [8], or well-known ResNet34 and ResNet50 [9] architectures are widely used to extract frame-level information. Many pooling methods [10, 11, 12] and angular based loss functions [13, 14, 15, 16, 17] also help the systems increase performance.. This paper proposes several model architectures. These architectures include residual TDNN using bottleneck TDNN block and residual connection to achieve better accuracy when the number of parameters is close to conventional methods. Furthermore, the multi-length context neural network (MLCNN) we proposed combines the bottleneck TDNN block and the 2D convolution blocks in a novel way, thereby exerting their respective strengths. Our best single system achieves an equal error rate (EER) of 4.47% and a minimum detection cost function (minC) of 0.188 for text-independent tasks, and the fusion system also achieved 4.21% EER and 0.1514 minC for text-dependent tasks.

2. Experimental Settings

2.1. Train and Development data

We use VoxCeleb1, VoxCeleb2 [18, 19, 20], Librispeech [21], and subpart of the DeepMine dataset [22, 23] to develop our systems. The VoxCeleb (VoxCeleb1 and VoxCeleb2) dataset has 7146 speakers and 1.2 million speech segments, and the Libripeech dataset has 2338 speakers and more than 200,000 speech segments. The in-domain DeepMine dataset is subdivided into task 1 training partition and task 2 training partition, which are used for text-depend and text-independent tasks, respectively. Due to the large time cost of training speaker embedding neural networks, we only use VoxCeleb and Librispeech to train speaker embedding neural networks. Since under this setting, two tasks can be performed with the same speaker embedding neural networks. We randomly selected the data from two training partitions in the DeepMine dataset as private trails for the two tasks. The number of speakers in the two private trails is the tail of the number of speakers in the original training partition data. In other words, 63 of the 963 speakers in the text-dependent task and 88 of the 588 speakers in the text-independent task. The remaining data of the two DeepMine training partitions that are not private trails are used for the training of the back-end PLDA and phrase model.

2.2. Evaluation data

The evaluation data for this challenge is a subpart of the DeepMine dataset. Under the text-dependent task, there are a total of 8 million trials. The enrollment consists of three utterances of the same phrase, and there are ten types of phrases. When the enrollment and test of the trials are the same speaker and phrase, they are considered as a target. In the text-independent task, there has a total of 13 million trials. The speech duration of enrollment of the trials is between 3 and 120 seconds, and the language is Persian. Besides, the duration of the test voice is 1 to 8 seconds and the language may not only be Persian but also English.

2.3. Data Preprocessing

Data augmentation enhances the performance of embedding-based speaker verification systems by increasing the amount and diversity of data [4, 24]. Our system uses the data augmentation strategy in the Kaldi recipe, which adds noise and reverberation to the raw speech. Specifically, we randomly select speech, music, noises from the MUSAN dataset [25] to add noise to the speech data. Furthermore, the speech data is also artificially reverberated via convolution with simulated RIRs.

We use 40-dimensional FBank which is mean normalized over a sliding window of up to 3 seconds as the acoustic feature to train the neural networks. The Fbank extracted from 16kHz audio signal with 25 ms frame-length, 10 ms frameshift, and bandwidth is limited to the range of 20-7600 Hz. After acoustic feature extraction, we generate a training archive based on the Kaldi-Tensorflow toolkit [26, 27]. The minimum and the maximum number of frames in each training example is fixed at 200. Additionally, the number of repeats for each speaker is 15.

2.4. Backend Scoring and Phrase modeling

For all our systems, we trained Gaussian PLDA [28] as a backend scoring module. Before training PLDA, LDA will project the dimension of speaker embedding from 512 to 250 and maintain discrimination among speakers. The dimensions of the projection are tuned according to our private trials.

In addition to learning whether each trial is the same speaker, the text-dependent task also needs to determine whether it is the same phrase. Therefore, we have additionally developed a phrase scoring module. Since there are only 10 kinds of phrases, we regard training the phrase model as a classification task.The architecture of the phrase model is a conventional TDNN but halves each layer, and the training data is the same as the PLDA training data of text-dependent tasks.The pre-processing of the data is roughly the same as the speaker embedding neural network training, except that each training example crop or padding the utterances to 1000 frames. Due to the benefits of the lightweight model architecture, the phrase model takes only 4 hours to train on the GTX 1080 Ti to converge, which is much faster than the speaker embedding neural network. During the inference stage, the purpose of the phrase scoring module is to help the original speaker verification system to distinguish wrong trials, especially target-wrong trials. Therefore, when the phrase model predicts that the enrollment and test utterance are the same phrases, we do not change the prediction result of the original system. In contrast, when the prediction is different, we generate the phrase score according to the classification probability of the output of the phrase model. The concept of generating scores is that when predicting different classes, the higher the probability of the class, the more confident the model is that the phrase of utterances is different. Consequently, the penalty is larger.

3. Speaker Embedding Networks

3.1. Training setup

All of our speaker embedding neural networks train 6 epochs on the training archive mentioned in Section 2.3 and the mini-batch size is 32. The E-TDNN baseline system uses Softmax Cross-entropy as the loss function, and the remaining settings are the default values of the Kaldi-Tensorflow toolkit [26]. Our other neural network models use the following training settings. We use Additive Margin Softmax [13] with margin and scaling factor of 0.2 and 50 as the loss function. The learning rate scheduling method is separated into the warm-up stage and decay stage [29]. When the training progress is less than 10%, we increase the learning rate linearly, and then the learning rate decreases exponentially until the end of the training. This kind of scheduling of decreasing the learning rate after going through the warm-up stage is quite important in the training process of some model architectures. The initial learning rate and final learning rate in our experiment are 0.001 and 0.0001, respectively.

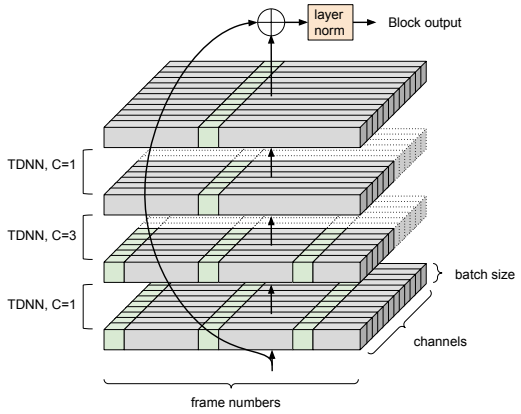

Figure 1: *A bottleneck TDNN block. C represents the number of contexts in the layer. Layers with 3 contexts can have different dilation rates.*

Table 1: *The frame-level layers architecture of the proposed 15-layers residual TDNN. C represents the number of contexts in the layer. K represents the dimension of acoustic features.*

#	Layer type	I/O	bottleneck	dilation
1	TDNN (C=5)	K / 512	-	1
2	TDNN block	512 / 512	256	2
3	TDNN block	512 / 512	256	3
4	TDNN block	512 / 512	256	4
5	TDNN block	512 / 512	256	5
6	TDNN (C=1)	512 / 1536	-	1

In addition, each layer of our network uses batch normalization after the nonlinear unit. When training the network, we also employ Label Smoothing [30, 31] that the parameter for the degree of smoothing α is set to 0.5.

3.2. Network Architectures

The model architecture of the E-TDNN baseline system is the conventional E-TDNN architecture with 12 hidden layers. The first 10 layers learn to extract frame-level information, and then the 1536-dimensional statistical pooling calculates the mean and standard deviation between each output frame of frame-level layers. Finally, the speaker representation of the input speech segment is learned by the 2-layer 512-dimensional fully connected. The other systems not only differ from the training setup with the E-TDNN baseline system, the segment-level has only one layer of 512-dimensional fully connected, which is also the layer to extract embedding during inference. Be-

Table 2: *The frame-level layers architecture of the proposed 27-layers residual TDNN.*

#	Layer type	I/O	bottleneck	dilation
1	TDNN (C=5)	K / 512	-	1
2	TDNN block	256 / 256	128	1
3	TDNN block	256 / 256	128	1
4	TDNN block	256 / 256	128	1
5	TDNN block	512 / 512	256	2
6	TDNN block	512 / 512	256	3
7	TDNN block	512 / 512	256	4
8	TDNN block	512 / 512	256	5
9	TDNN block	512 / 512	256	6
10	TDNN (C=1)	512 / 1536	-	1

Table 3: *The architecture of proposed MLCNN. The dilation rates of 4 TDNN blocks are 2, 3, 4, and 5, respectively.*

#	Layer type	Structure	Output shape	Total context	Input layer
0	-	-	$43 \times 200 \times 1$	1	-
1	Conv2D	$3 \times 3, 64$	$43 \times 200 \times 64$	3	0
2	Conv2D blocks	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 4$	$43 \times 200 \times 64$	19	1
3	Reshape	-	200×2752	19	2
4	Pooling	-	5504	full seq.	3
5	FC	5504×256	256	full seq.	4
6	TDNN	$1, 512$	200×512	19	3
7	TDNN blocks	$\begin{bmatrix} 1, 256 \\ 3, 256 \\ 1, 512 \end{bmatrix} \times 4$	200×512	47	6
8	TDNN	$1, 1536$	200×1536	47	7
9	Pooling	-	3072	full seq.	8
10	FC	3072×256	256	full seq.	9
11	Aggregate	$2 \times 256 \times 256$	512	full seq.	5,10
12	FC	$512 \times \#spks$	$\#spks$	full seq.	11

sides, other systems use 4 heads attentive pooling to integrate the frame-level information instead of statistics pooling. The details of self-attentive pooling are described in Section 3.2.1.

3.2.1. NSYSU E-TDNN

This model is based on the E-TDNN baseline and is modified according to Section 3.1 and Section 3.2. One of the changes to the model architecture is the pooling layer. The attention weight of 4 heads self-attentive pooling we use is shown in Equation 1.

$$\boldsymbol{w}eight = \text{softmax}(tanh(\boldsymbol{H}^T\boldsymbol{W}_1 + \boldsymbol{b})\boldsymbol{W}_2) \quad (1)$$

where $\boldsymbol{H}$ is the output of the frame-level layer, and $tanh(\cdot)$ is the hyperbolic tangent function. In our models, the shapes of the two trainable matrices $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are 1536×512 and 512×4, respectively. In addition, during the training stage. This attention weight will be used to calculate the weighted average and standard deviation after using dropout which keep probability is 0.9.

3.2.2. Residual TDNN

The residual learning framework reduces the difficulty of deeper neural network training because of shortcut connections. Furthermore, because of the use of bottleneck design, it increases the depth and accuracy of the network while keeping time complexity. We follow these design rules and build residual TDNN models of 15 layers and 27 layers, and their parameter amounts are close to conventional TDNN and E-TDNN, respectively. These network architectures are composed of bottleneck TDNN blocks shown in Figure 1. The block contains TDNN layers with contexts 1, 3, and 1, where the first and last layers are responsible for reducing and then increasing dimensions. This allows the middle layer with 3 contexts to have fewer parameters. The reduced input and output dimensions are shown in dotted lines in Figure 1.

The 15 layers residual TDNN is abbreviated as ResTDNN15. Its frame-level architecture is shown in Table 1, and the total context is 33 larger than 23 of E-TDNN. In addition, a block consists of three layers, so the frame-level has a total of 14 layers of the network. But the amount of parameters is only close to the conventional 5-layer TDNN frame-level architecture. The 27 layers residual TDNN is abbreviated as ResTDNN27. Its frame-level architecture is shown in Table 2. The idea of designing the network is to collect information in short contexts using blocks with smaller parameters. Then expand the receptive contexts with other blocks.

3.2.3. Multi-length Context Neural Network

In recent years, in addition to the TDNN-based model architecture widely used for speaker recognition, the well-known ResNet34, ResNet50, etc. [9] are also favorite architectures of the research community. We find that some TDNN-based network architectures perform better in scenarios with longer speech [32], while some in shorter speech scenarios [33] perform better with ResNet based on 2D convolutional networks. Our proposed multi-length context neural network is abbreviated as MLCNN, which combines the advantages of the two architectures. Use a 2D convolutional network structure to learn representations with shorter total context lengths, and stack the TDNN-based structure to enable the network to learn representations with longer total context lengths. After that, we multiply the two representations by a trainable weight and combine the results. The overall network architecture is shown in Table 3.

3.2.4. Residual TDNN and GRU

The RNN structure can bring certain benefits to the speaker recognition model [34]. In addition, we also believe that using a RNN structure to integrate long-term information of speech can reduce the impact of zero paddings in convolutional-based architecture, especially in the case of a very short speech. We modify ResTDNN27. Reduce the dilation rate of the seventh and eighth layers in Table 2 to 1, and replace the block of the ninth layer with the bidirectional GRU. The receptive contexts of the convolutional network under this network is only 20. We abbreviate this model as ResTDNN27-GRU.

The output of the eighth layer network will simultaneously be input into both forward sequence GRU layer and backward sequence GRU layer. The dimension of a GRU layer is 256, and the nonlinear unit is a hyperbolic tangent function. Therefore, the forward-backward GRU pair is 512-dimensional.

3.2.5. Trainable Margin AM-Softmax

We also try to get rid of the situation of adjusting hyperparameters, so that the margin in AM-Softmax can be obtained by training. We add an additional restriction to the margin, as shown in equation 2.

$$margin = softplus\left(tanh\left(m\right) - 0.5\right) \quad (2)$$

Where m is a trainable variable. We use this restriction to prevent the margin from saturating near 0 or 1, which makes

Table 4: *Results of our systems on SdSV challenge 2020 private/progress/evaluation set. Note that the results of two tasks are different PLDA models. In addition, the text-dependent results of each system in the table use the same phrase scoring module.*

Systems	Text-dependent						Text-independent					
	Private		Progress		Evaluation		Private		Progress		Evaluation	
	EER	minC	EER	minC	EER	minC	EER	minC	EER	minC	EER	minC
Clallenge baseline	-	-	9.05	0.529	9.05	0.528	-	-	10.67	0.431	10.67	0.432
E-TDNN baseline	5.12	0.227	5.19	0.192	5.24	0.192	5.75	0.252	7.99	0.347	8.03	0.348
NSYSU E-TDNN	4.97	0.213	4.84	0.191	4.90	0.191	4.94	0.221	5.83	0.242	5.83	0.242
ResTDNN15	4.47	0.184	4.57	0.165	4.69	0.165	4.64	0.203	5.84	0.230	5.63	0.231
ResTDNN27	4.47	0.182	4.60	0.165	4.68	0.165	4.64	0.203	5.65	0.231	5.64	0.231
ResTDNN27-GRU	4.25	0.172	4.48	0.163	4.58	0.164	4.71	0.203	7.07	0.294	7.07	0.293
ResTDNN27-TM	4.71	0.197	4.55	0.175	4.63	0.176	4.81	0.281	5.84	0.237	5.84	0.238
MLCNN	4.72	0.203	4.52	0.171	4.55	0.171	**4.41**	**0.192**	**4.47**	**0.188**	**4.47**	**0.188**
Average fusion	**3.92**	**0.158**	**4.11**	**0.150**	**4.21**	**0.151**						

Table 5: *Computation cost of each model at inference stage.*

Network	#Parameters	Inference time (ms)
E-TDNN baseline	9.4 M	4.05
NSYSU E-TDNN	9.9 M	4.35
ResTDNN15	8.1 M	5.10
ResTDNN27	9.8 M	5.45
ResTDNN27-GRU	9.6 M	54.3
ResTDNN27-TM	9.8 M	7.60
MLCNN	11.6 M	7.79

the margin at least about 0.2 and the maximum value at 0.97. We apply this try to the ResTDNN27 model, so it is called ResTDNN27-TM for short.

4. Results and Analysis

The results of our system in the SdSV challenge 2020 are shown in Table 4. In the table, the minimum detection cost function (minC) and the percentage value of equal error rate (EER) are used as metrics to evaluate each system. The details of how private set divides data are described in Section 2.1. Note that the results of the text-dependent and text-independent tasks are different PLDA models. In addition, the text-dependent results of each system in the table use the same phrase scoring module as described in Section 2.4.

4.1. Text-dependent Speaker Verification

Under the text-dependent task, before fusion the results of different speaker embedding networks, ResTDNN-GRU has the best results on minC. On the contrary, this model architecture has poor results in the evaluation set on the text-independent task. We believe that this difference is the impact of zero paddings in convolutional-based architecture. The shorter the speech duration, the greater the impact (text-dependent speech duration is about 3 seconds, and text-independent is longer). In addition, we average fusion ResTDNN15 with 3 types of ResTDNN27 systems to be our best fusion system. This fusion system reached an EER of 4.21% and minC of 0.151 on the evaluation set, and we also submitted it as a primary system. Due to the challenge rules, our single system cannot use the phrase scoring module. Therefore, we consider the ResTDNN15 system of the text-dependent task without phrase scoring module as a single system, which has an EER greater than 10% and a minC of 0.87 on the progress set.

4.2. Text-independent Speaker Verification

Comparing the E-TDNN baseline and NSYSU E-TDNN can show the effect of different training settings of neural networks. In addition, our proposed ResTDNN27 and ResTDNN15 can have better performance when the amount of parameters is close to or less than that of the E-TDNN architecture. In our training settings, GRU did not play a good role in text-independent tasks, but it required a lot of computing resources. Comparing ResTDNN27 and ResTDNN27-TM shows that the effect of TM is not better than simply setting the margin to 0.2. Although we hope to find the maximum possible margin by training, we think this will make optimization more difficult. The MLCNN architecture we proposed outperforms other models. It achieves an EER of 4.47% and a minC of 0.188 on the evaluation set. We also submitted this system as a primary system and also our single system.

4.3. Inference Computational Resources

Since each part of our system has similar operations except the neural network, we focus on considering the differences between each neural network. Table 5 shows the number of parameters and computation time for each neural network, where the computation time is measured on the GTX 1080 Ti. The computation time indicates the average time for an utterance of about 3 seconds to extract the embedding on the neural network. It can be seen from the results that the use of the RNN structure will greatly increase the computation time.

5. Conclusions

This paper describes the speaker verification systems we developed for SdSVC 2020. Our proposed multi-length context neural network (MLCNN) combines the bottleneck TDNN block with 2D convolution blocks in a novel way and achieves good results on text-independent tasks. The network constructed using the bottleneck TDNN block also shows that the accuracy of the system has improved. In addition, we simply transfer text-independent tasks to text-dependent tasks through an additional phrase modeling module. Moreover, the analysis of each model architecture is also included.

6. References

[1] H. Zeinali, K. A. Lee, J. Alam, and L. Burget, "Short-duration speaker verification (sdsv) challenge 2020: the challenge evaluation plan," *arXiv preprint arXiv:1912.06311*, 2019.

[2] 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.

[3] 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.

[4] 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.

[5] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Twelfth annual conference of the international speech communication association*, 2011.

[6] O. Glembek, L. Burget, P. Matějka, M. Karafiát, and P. Kenny, "Simplification and optimization of i-vector extraction," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 4516–4519.

[7] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5796–5800.

[8] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks." in *Interspeech*, 2018, pp. 3743–3747.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[10] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," *arXiv preprint arXiv:1804.05160*, 2018.

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

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

[13] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, 2018.

[14] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[15] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 212–220.

[16] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5265–5274.

[17] Y. Liu, L. He, and J. Liu, "Large margin softmax loss for speaker verification," *arXiv preprint arXiv:1904.03479*, 2019.

[18] 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.

[19] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.

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

[21] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[22] H. Zeinali, H. Sameti, and T. Stafylakis, "Deepmine speech processing database: Text-dependent and independent speaker verification and speech recognition in persian and english." in *Odyssey*, 2018, pp. 386–392.

[23] H. Zeinali, L. Burget, J. Černockỳ *et al.*, "A multi purpose and large scale speech corpus in persian and english for speaker and speech recognition: the deepmine database," *arXiv preprint arXiv:1912.03627*, 2019.

[24] C.-L. Huang, "Exploring effective data augmentation with tdnn-lstm neural network embedding for speaker recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 291–295.

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

[26] H. Zeinali, L. Burget, J. Rohdin, T. Stafylakis, and J. Cernocky, "How to improve your speaker embeddings extractor in generic toolkits," *arXiv preprint arXiv:1811.02066*, 2018.

[27] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[28] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision*. Springer, 2006, pp. 531–542.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[30] R. Müller, S. Kornblith, and G. E. Hinton, "When does label smoothing help?" in *Advances in Neural Information Processing Systems*, 2019, pp. 4696–4705.

[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[32] S. O. Sadjadi, C. S. Greenberg, E. Singer, D. Reynolds, L. Mason, and J. Hernandez-Cordero, "The 2019 nist speaker recognition evaluation cts challenge," *Proc. Speaker Odyssey (submitted), Tokyo, Japan*, 2020.

[33] J. S. Chung, A. Nagrani, E. Coto, W. Xie, M. McLaren, D. A. Reynolds, and A. Zisserman, "Voxsrc 2019: The first voxceleb speaker recognition challenge," *arXiv preprint arXiv:1912.02522*, 2019.

[34] C.-P. Chen, S.-Y. Zhang, C.-T. Yeh, J.-C. Wang, T. Wang, and C.-L. Huang, "Speaker characterization using tdnn-lstm based speaker embedding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6211–6215.