INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Intel Far-field Speaker Recognition System for VOiCES Challenge 2019

Jonathan Huang[1], *Tobias Bocklet*[1,2]
[1]Intel Corporation
[2]Technischen Hochschule Nürnberg
jonathan.huang@intel.com, tobias.bocklet@intel.com

Abstract

This paper describes Intel's speaker recognition systems for the VOiCES from a Distance Challenge 2019. Our submission consists of a Resnet50, and four Xvector systems trained with different data augmentation and input features. Our novel contributions include the use of additive margin softmax loss function and the use of invariant representation learning for some of our systems. To our knowledge, this has not been proposed for speaker recognition. We found that such complementary subsystems greatly improved the performance on the development set by late fusion on score level based on linear logistic regression. After fusion our system achieved on the development set EER, minDCF and actDCF of 2.2%, 0.27 and 0.27; and on the evaluation set 6.08%, 0.451 and 0.458, respectively. We discuss our results and give some insight on accuracy with respect to recording distance.

Index Terms: speaker recognition, far-field, VOiCES

1. Introduction

We present the Intel speaker recognition system for the "VOiCES from a Distance Challenge 2019" [1], which is designed to foster research of speaker recognition and automatic speech recognition (ASR) with single-channel far-field audio under noisy conditions. This challenge is based on VOiCES Obscured in Complex Environmental Settings (VOiCES) corpus [2], which we used for the development of our system. Our system was submitted under the fixed condition, which limits the speech training data to the Speakers in the Wild (SITW) [3], VoxCeleb1 [4] and VoxCeleb2 [5].

The rest of the paper is organized as follows. Section 2 provides the overview of every aspect of our system including data augmentation strategy, feature extraction, loss function, training strategies, and neural network architectures. Section 3 presents the results of the system, along with insights into how the system performed on the development and evaluation sets. Finally, section 4 concludes the paper.

2. System Description

This section describes the systems we developed for the fixed condition speaker recognition challenge. Included are details about the data preparation, speech activity detection (SAD), and feature extraction. We explored the use of several loss functions. We propose using a recently develop technique for data augmentation to improve the performance. The neural net architectures, PLDA and fusion systems are also described.

2.1. Training data preparation

In this section we describe the three sets of features used in model training. We include details of data augmentation, SAD, and feature computations.

2.1.1. Training set 1

This training set uses VoxCeleb 1 and 2. Data augmentation, feature computation and SAD follows Kaldi VoxCeleb recipe. We use the room impulse responses from [6] with a 50:50 split from small and medium sized rooms. In addition, we augmented the training data with MUSAN noises [7], combined these 2 sets and limited the amount of utterances to 1M. We then combined with the original data so that this set had in total 2.26M training utterances. We used 30 MFCCs extracted from 25 ms frames, with 15 ms overlap modeling a frequency range from 130-6800Hz. This training set was intended to model low and medium amount of noise and reverb.

2.1.2. Training set 2

We use the same VoxCeleb 1 and 2 data to prepare this training set. In addition to the original 1.26 million training utterances, we generated four times the data through data augmentation. For each augmented speech file we convolve a randomly chosen room impulse response (RIR) from 100 artificially generated by Pyroomacoustics [8] and 100 selected from Aachen Impulse Response Database [9], then mixing in randomly chosen clips from Google's Audioset under Creative Commons [10]. The SNR for mixing was uniformly distributed between 0 and 18 dB. We extract 64-dimensional mel-filterbank features from 25 ms frames with 15 ms overlap. The features are mean and standard deviation normalized by a small subset of the training set. We did not use any SAD. The intention of this dataset was to model medium-to-high noises and more distant speech.

2.1.3. Training set 3

This set is prepared in the same way as set 2. However, the mixing SNR is sampled from range of 5 to 20 dB; 40mel are used as features.

2.2. Loss functions

While the conventional softmax loss works reasonably well for training speaker embeddings, it is specifically designed for classification task. The triplet loss function, which is designed to reduce intra-speaker and increase inter-speaker distance, has shown to be more effective for speaker recognition [11] [12]. Likewise, the end-to-end loss [13] has better performance than softmax. The downside to these kinds of losses is that the training infrastructure is significantly more complicated than one used for supervised learning with softmax. In this paper we explored the use of several recently proposed loss functions that were first introduced in the facial recognition research, Angular Softmax (A-softmax) [14] and Additive Margin Softmax (AM-softmax) [15].

2.2.1. Angular softmax

The key idea behind A-softmax is that it imposes an angular margin between classes, which has the effect of increasing inter-speaker distance in the embedding space. A comparison of A-

http://dx.doi.org/10.21437/Interspeech.2019-2894

softmax for training speaker embeddings shows that it outperforms softmax and triplet losses [16]. The A-softmax loss function is formulated as

$$L_{AS} = \frac{1}{N}\sum_i -log\frac{e^{\|x_i\|cos(m\theta_{y_i,i})}}{e^{\|x_i\|cos(m\theta_{y_i,i})} + \sum_{j\neq y_i} e^{\|x_i\|cos(\theta_{y_j,i})}} \quad (1)$$

where x_i is the i-th embedding in a training set of N examples, y_i is the corresponding label, θ_{i,y_i} is the angle between x_i and column y_i of the fully connected layer weight matrix $\mathbf{W}$, and m is an integer representing the margin between classes.

Besides better recognition accuracy, A-softmax has an advantage over triplet loss in the model training process. Triplet loss function requires hard-negative selection process in order for it to perform well. In contrast A-softmax is essentially a drop-in replacement for softmax, and convergence is relatively easy to achieve.

2.2.2. Additive margin softmax

Motivated by the objective of imposing angular margin between speaker embeddings, AM-softmax takes a slightly different approach by making the margin additive. The loss function is

$$L_{ASM} = \frac{1}{N}\sum_i -log\frac{e^{s\cdot(cos\theta_{y_i}-m)}}{e^{s\cdot(cos\theta_{y_i}-m)} + \sum_{j\neq y_i} e^{s\cdot cos(\theta_{y_j,i})}} \quad (2)$$

where s is a scale factor recommended to be 30 for facial recognition, and the margin m chosen around the range $[0.25, 0.5]$. According to Wang et al. [15], AM-softmax is easier to obtain good results compared to A-softmax because it involves less hyper-parameter tuning in the learning process.

2.3. Invariant representation learning

Invariant representation learning (IRL) was shown to improve speech recognition acoustic model [17] over the standard data augmentation. For three of our systems, we adapted this method for training Xvector embeddings. The IRL training method is illustrated in Figure 1. At each training iteration, clean and noisy features of the same utterance are fed into the network one after another, resulting in two categorical losses. The categorical losses can be softmax, A-softmax or AM-softmax. The two hidden representations after utterance-level stats pooling are further pulled together by imposing the cosine similarity and L2 loss. The net result of these additional constraints is that the noisy speech is encouraged to map to similar embedding space as its clean counterpart. The combined loss function for the clean x and noisy x' pair is represented by

$$L_{IRL}(x,x') = L_c(x) + \alpha L_c(x') - \gamma L_{cos}(x,x') + \lambda L_2(x,x') \quad (3)$$

where L_c is the categorical loss, L_{cos} is the cosine similarity (4), and L_2 is the means square loss (5)

$$L_2(x,x') = \sum_{l=1}^{L} \frac{\phi_l(x)\cdot\phi_l(x')}{\|\phi_l(x)\|\|\phi_l(x')\|} \quad (4)$$

$$L_{cos}(x,x') = \sum_{l=1}^{L} (\phi_l(x) - \phi_l(x'))^2 \quad (5)$$

The hidden layer representation in layer l is denoted by ϕ_l. The parameters α, γ, and λ control the contributions from the losses; in our experiment we set them to 1, 0.001, and 0.001.

2.4. Subsystems

This sections describes the details of each of our subsystems.

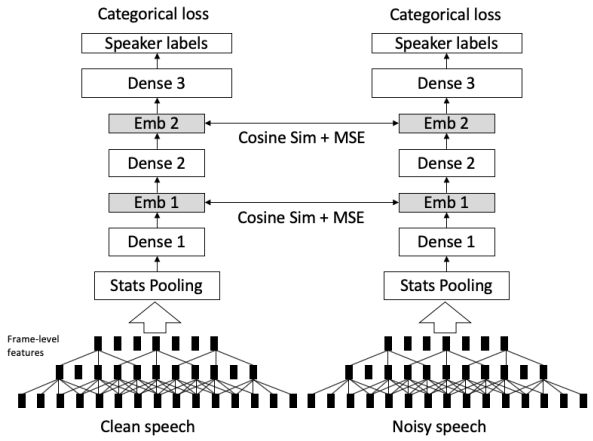

Figure 1: *Invariant representation learning utilizes parallel clean and noisy speech in each training iteration to train the weights of a single network, which we show here as the Xvector architecture. The output of the two forward passes will be used to compute a categorical loss such as softmax. The two embedding layers are constrained with cosine similarity and MSE such that clean and noisy version of the utterance are encouraged to similar representations.*

2.4.1. System 1: Resnet50

This system uses the training set 2. This neural net architecture we used is Resnet50 [18], adapted for variable-length inputs. The input to the network is an arbitrary length sequence of mel-filterbank frames. At the output of the last convolution block, each of the 512 channels are average pooled, producing a 512-dimension vector. Then another fully-connected layer with 256 output dimensions produces the speaker embedding. The embedding layer is attached to another fully-connected layer with 7323 outputs corresponding to the speaker labels. This neural net is trained with softmax loss. It works directly on utterance basis; the input is the mel-spectrogram of the entire utterance.

2.4.2. System 2: Xvector-base

This is the Xvector system from [19] taken from the VoxCeleb-Kaldi recipe with slight modifications in data preparation (see Sec. 2.1.1 for details). We use training set 1 for this system. We selected the output of layer 7 (see Table 1) directly after stats pooling. This system was supposed to act as reference system. A larger Xvector model that we initially planned to submit based on training set 2 data, did not finish in time for the VOiCES submission. Thus, we do not describe it here. This network works on a frame-wise basis and statistically pools the frame-wise statistics to segment-based.

2.4.3. System 3: Xvector-dense

The data set used in this system is training set 3. We have modified the Xvector system, shown on Table 1, to eliminate the dilation in frame layers 2 and 3, thus we call this a "dense" version of Xvector. We also added residual connections for layers 2 and 3. The embedding is extracted at the output of the affine layer in layer 8, which is selected to be 256 dimensions. We use the IRL training method described in Section 2.3. The cosine similarity and L2 losses are imposed at the affine output of layers 7 and 8. For the classification of speaker labels, we use the softmax loss.

2.4.4. System 4: Xvector-Asoftmax

This system is the same as system 3, except that we used A-softmax for the categorical loss.

2.4.5. System 5: Xvector-AMsoftmax

This system is the same as system 3, except that we used AM-softmax for the categorical loss.

Table 1: *XvectorDense architecture.*

Layer	Layer Type	Context	Size	Residual
1	TDNN-ReLU	t-2:t:t+2	512	
2	TDNN-ReLU	t-2:t:t+2	512	yes
3	TDNN-ReLU	t-3:t:t+3	512	yes
4	Dense-ReLU	t	512	
5	Dense-ReLU	t	1500	
6	Stats Pool	T	3000	
7	Dense-ReLU	T	512	
8	Dense-ReLU	T	256	
9	Dense-Softmax	T	# spks	

2.5. PLDA

First, we decrease the dimensions of our mean- and length-normalized embeddings by an LDA to 200. Then, we applied standard PLDA. (P)LDA is trained either on Voxceleb1 or Voxceleb1 + VOiCES Dev set. We did not perform any specific domain-adaptation, noise modeling, or score normaliztion.

2.6. Fusion

We fused our systems either by early or late fusion. Early fusion happened on embedding level where embeddings are concatenated prior to scoring. For late fusion we either used an naive approach where each system is weighted equally or we trained a linear logistic regression (LLR) for calibration and fusion. We used FoCal [20] for LLR calibration and fusion. We decided against fusion with a neural network due to 2 reasons: 1. limited amount of training data in the VOiCES Dev set and 2. adding heterogeneous components to the final system. Early fusion did not bring any improvement compared to late fusion on the Dev set. Thus, all our submissions are based on late fusion.

2.7. Submitted systems

We submitted 3 systems in the fixed condition.

- **System A:** Naive fusion where all sub-systems except Xvector-base have been scored by a PLDA trained on VoxCeleb1 + VOiCES Dev set. For the Xvector-base system we employed a PLDA trained on VoxCeleb1+VoxCeleb2.
- **System B:** LLR-fusion where only the baseline Xvector system has been scored by a PLDA trained on the whole training set. The other systems have been scored by cosine-distance. The fusion was trained on the VOiCES Dev corpus.
- **System C:** LLR-fusion with PLDA scoring on all subsystems. PLDA was trained on VoxCeleb1 for all systems except for the Xvector-base system; for xvector-base we trained the PLDA on VoxCeleb1+VoxCeleb2 ; fusion was trained on the VOiCES Dev set.

3. Results & Analysis

We describe the results of our systems and submissions for stand-alone systems (Sec. 3.1) and fused systems (Sec. 3.2).

Table 2: *Results of the stand-alone subsystems on the Dev set.*

	EER	
System	PLDA vox	PLDA vox+VOiCES Dev
ResNet50	2.96	2.59
Xvector-base	3.22	N/A
Xvector-dense	3.88	3.34
Xvector-Asoftmax	4.48	3.51
Xvector-AMsoftmax	3.86	3.18

In both subsections we present results separated for the Dev and Eval sets. We then conclude this section in Sec.3.3 with an analysis of the system performance with respect to detractor noise and recording distance as defined also in [2]. For the sake of easy readability, we focus on equal error rate (EER) as error measurement in the analysis section. For the 3 fused systems we submitted, we also describe the minDCF, EER, and cLLR numbers.

3.1. Results of Stand-alone Subsystems

The stand-alone results on the Dev set are shown in Table 2. The table shows the results for 2 sets of experiments: 1. with a PLDA trained purely on Voxceleb; 2.with PLDA being trained Voxceleb+VOiCES Dev set. Please note that results for PLDA vox+VOiCES is too optimistic; PLDA training and final testing share the same data. We added these results for the completeness of comparison against the stand-alone results on the Eval set and to show what can be achieved with an "ideal" PLDA. For the Dev set, the best EER was achieved by the ResNet50 system (2.96 %), the Xvector-base achieved (3.22 %) followed by the Xvector-dense systems (3.88 %).

We have done some comparisons to demonstrate the effectiveness of IRL. Using the same configuration as Xvector-dense and cosine distance as scoring metric, we found that the EER on the development set was 6.3 % without and 4.8 % with IRL. While using other categorical losses, we also observed similar improvements by incorporating IRL.

Table 3: *Results of the stand-alone subsystems on the Eval set*

	EER	
System	PLDA vox	PLDA vox+VOiCES
ResNet50	6.84	6.40
Xvector-base	8.79	N/A
Xvector-dense	7.31	6.95
Xvector-Asoftmax	8.27	7.97
Xvector-AMsoftmax	7.48	7.15

The Eval set results are summarized in Table 3. While the EER on the Dev set showed no improvement of the Xvector-dense, the Asoftmax, and the AMSoftmax systems over the Xvector-base, the results on the Eval set are significantly better for these systems; All systems show a significant improvement compared to the Xvector-based. The best system is ResNet50 (6.40 % EER), followed by the Xvector-dense system (6.95 % EER).

When interepreting these results we came to the following conclusions: The lower noise level of the Dev set is a better match for the low amount of data augmentation that was used in the Xvector-base; the other systems were trained with more aggressive data augmentation. They thus produced worse results on the Dev set, but could show their potential in the more noisy Eval set.

Table 4: *Results of the 3 fused submitted systems on the Dev set*

Submission	minDCF	EER	cLLR
System A	0.180	1.36	0.596
System B	0.266	2.20	0.097
System C	0.260	2.22	0.099

3.2. Fusion Results

The results of the three fusion systems we submitted (see Sec.2.6 for details) are summarized in Table 4 for the Dev set and in Table 5 for the Eval set. Results of each fusion system shows a significant improvement of the stans-alone subsystems. On the Dev set, the EER is in the range of 2.2 % for the LLR-fused systems (Systems A and B) and 1.36 % for the naive fusion (System A). The minDCF is 0.26 and 0.18, and cLLR 0.10 and 0.59, respectively. The naive fusion seems to do very well. Please note that these results are highly optimistic; the PLDA being trained on VoxCeleb and VOiCES Dev data for System A. For the LLR-fused systems (B and C) the fusion was calibrated with VOiCES Dev data, PLDA was VOiCES agnostic and trained on pure VoxCeleb data. System C achieved the best results from the LLR-fused systems. However, scoring all systems with a PLDA (System C) shows no real benefit on the Dev set.

Fusing the systems brings also a significant benefit on the Eval set. Our systems achieved a minDCF in the range of 0.45. The most balanced systems according to min- and actDCF is System C. Please note that on the Dev set there was no advantage in system C (i.e., all systems scored by PLDA). On the Eval set the gain by PLDA in LLR is significantly. The LLR-based systems are well calibrated; we had issues with the calibration on the naive system (A) visible but the actDCF of 1.

Table 5: *Results on the Eval set of the 3 fused submitted systems*

Submission	minDCF	actDCF	EER	cLLR
System A	0.449	1	5.69	0.634
System B	0.468	0.469	7.27	0.440
System C	0.451	0.458	6.08	0.453

3.3. Analysis

We started some detailed analysis of our systems and submission with respect to the data structures (e.g., accuracy vs. distractor noises, recording distances, microphone types, enrollment conditions, etc.). We briefly discuss some findings here; we focus on an overall observation followed by some details on results of the recording distance. A more detailed analysis, along with new and improved results will follow in a future article.

As depicted above, there is a vast difference between the Dev and Eval results on the fused systems as well as for the stand-alone systems. The Eval data contains more noisy and problematic conditions, different rooms (L-shaped), additional microphones types, and stronger channel mismatch in enrollment and test utterances than the development set. This results in a significantly lower accuracy of our fused systems on the Eval set. The degradation is strongest for the Xvector-base system when going from Dev to Eval. One reason for that is the relatively low amount of noise augmentation we used for training this system. The Xvector-dense systems (trained with higher noises) performed worse on the Dev set with smaller rooms and different acoustics than the Xvector-base. Due to the mismatch between Dev and Eval, it is far from ideal to train the LLR on the Dev set when scoring on the Eval set.The weights for the sub-systems are estimated for less severe noise conditions. This

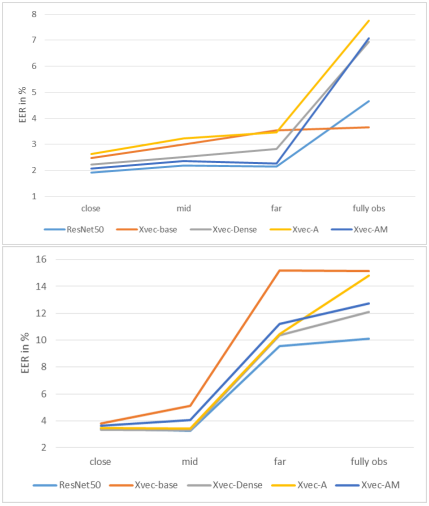

Figure 2: *EER with respect to the distances and fully obstructed microphones for our 5 systems on the Dev set (upper) and the Eval set(lower).*

resulted in over represented weights for the Xvector-base system.

Further analysis by using the same (heavier) noise mixings from the Xvector-dense in the Xvector-base system are currently performed in order to evaluate whether the improvement in the Xvector-dense comes purely from the invariant representation learning or by more aggressive noises in these systems. First results indicate that even with more aggressive noises in the baseline system, IRL has an advantage. Please note, that it was not our main intention to show improvements of IRL in this paper, but to build complementary systems that help in fusion. We wanted to keep the systems as different as possible. Because of that we do not have 1:1 comparable systems in this paper. A comparison between the ResNet50 and the Xvector-base shows a huge improvement by more aggressive noise mixings on the Eval set. While these system are relatively close on the Dev set (8 % relative improvement of ResNet50 compared to Xvector-base), the advantage of the ResNet50 on the Eval set grew to more than 20 %. Figure 2 shows the EER with respect to the recording distance on the Dev and Eval set. We used the same distance notations as [1]. While on the Dev set, all systems are relatively similar in accuracy for close, mid, and far, an advantage is visible for the Xvector-base in the fully-obscured microphones. On the Eval set there is a much stronger degradation when going to far distances. However, close and mid results are very similar across the systems. The degradation in far on the Eval set are tremendous for the Xvector-base system. This can be explained by the light noise and RIR augmentation in that system.

4. Conclusion

This paper described Intel's submission to the VOiCES challenge 2019. We described our stand-alone systems: One based on ResNet50, and four based on the Xvector architecture where we adapted IRL training to the XVector concept. To our knowledge this has not been done before for SID. We discussed the results of our system in detail and also analyzed the results and accuracy difference in Dev and Eval as well as recording distances. In future work we will focus on a more detailed analysis of the accuracy of the systems and an apple-to-apple comparison on architectural network concepts while keeping the training data constant.

5. References

[1] M. K. Nandwana, J. Van Hout, M. McLaren, C. Richey, A. Lawson, and M. A. Barrios, "The voices from a distance challenge 2019 evaluation plan," *arXiv preprint arXiv:1902.10828*, 2019.

[2] C. Richey, M. A. Barrios, Z. Armstrong, C. Bartels, H. Franco, M. Graciarena, A. Lawson, M. K. Nandwana, A. Stauffer, J. van Hout *et al.*, "Voices obscured in complex environmental settings (voices) corpus," *arXiv preprint arXiv:1804.05053*, 2018.

[3] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The speakers in the wild (sitw) speaker recognition database." in *Interspeech*, 2016, pp. 818–822.

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

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

[6] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5220–5224, 2017.

[7] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015, arXiv:1510.08484v1.

[8] R. Scheibler, E. Bezzam, and I. Dokmanić, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 351–355.

[9] M. Jeub, M. Schafer, and P. Vary, "A binaural room impulse response database for the evaluation of dereverberation algorithms," in *2009 16th International Conference on Digital Signal Processing*. IEEE, 2009, pp. 1–5.

[10] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[11] C. Zhang, K. Koishida, and J. H. Hansen, "Text-independent speaker verification based on triplet convolutional neural network embeddings," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 26, no. 9, pp. 1633–1644, 2018.

[12] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep speaker: an end-to-end neural speaker embedding system," *arXiv preprint arXiv:1705.02304*, 2017.

[13] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[14] 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.

[15] 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.

[16] Z. Huang, S. Wang, and K. Yu, "Angular softmax for short-duration text-independent speaker verification," *Proc. Interspeech, Hyderabad*, 2018.

[17] D. Liang, Z. Huang, and Z. C. Lipton, "Learning noise-invariant representations for robust speech recognition," *arXiv preprint arXiv:1807.06610*, 2018.

[18] 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.

[19] 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.

[20] N. Brummer, L. Burget, J. Cernocky, O. Glembek, F. Grezl, M. Karafiat, D. A. van Leeuwen, P. Matejka, P. Schwarz, and A. Strasheim, "Fusion of heterogeneous speaker recognition systems in the stbu submission for the nist speaker recognition evaluation 2006," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2072–2084, 2007.