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

The JHU Speaker Recognition System for the VOiCES 2019 Challenge

David Snyder[1,2], *Jesús Villalba*[1], *Nanxin Chen*[1], *Daniel Povey*[1,2], *Gregory Sell*[2], *Najim Dehak*[1], *Sanjeev Khudanpur*[1,2]

[1]Center for Language and Speech Processing
[2]Human Language Technology Center of Excellence
Johns Hopkins University, Baltimore, MD, USA

`david.ryan.snyder@gmail.com`

Abstract

This paper describes the systems developed by the JHU team for the speaker recognition track of the 2019 VOiCES from a Distance Challenge. On this far-field task, we achieved good performance using systems based on state-of-the-art deep neural network (DNN) embeddings. In this paradigm, a DNN maps variable-length speech segments to speaker embeddings, called x-vectors, that are then classified using probabilistic linear discriminant analysis (PLDA). Our submissions were composed of three x-vector-based systems that differed primarily in the DNN architecture, temporal pooling mechanism, and training objective function. On the evaluation set, our best single-system submission used an extended time-delay architecture, and achieved 0.435 in actual DCF, the primary evaluation metric. A fusion of all three x-vector systems was our primary submission, and it obtained an actual DCF of 0.362.

Index Terms: speaker recognition, VOiCES Challenge 2019

1. Introduction

In this paper, we describe the speaker recognition systems developed by the Johns Hopkins University (JHU) team for the Voices Obscured in Complex Environmental Settings (VOiCES) challenge [1, 2]. The challenge, developed by SRI International and Lab41, aims to measure state-of-the-art speech processing in challenging far-field and noisy conditions. The challenge was split into tracks for automatic speech recognition and speaker recognition, and consists of Librispeech [3] recordings replayed in several differently-sized rooms and recorded by an assortment of microphones. Simultaneously, various distractor noises such as music, babble, or television were played. As a consequence, participants needed to develop systems that were robust to this reverberant and noisy speech. The task was split into a *fixed* condition, where only specified speech material could be used to train systems, and an *open* condition that permitted participants to explore the benefits of more extensive training sets. Our team participated in only the fixed condition of the speaker recognition track.

Recently, the field of text-independent speaker recognition has advanced significantly due to the development of practical neural network-based speaker embeddings [4, 5, 6, 7]. This paradigm, which has now replaced i-vectors [8] as the state-of-the-art, is based on deep neural networks (DNNs) trained to discriminate between speakers at the level of speech segments. To obtain segment-level representations, a temporal pooling layer aggregates across frame-level representations to compute a single set of statistics per input segment. In our early work, the pooling mechanism computed the mean and standard deviation of the frame-level representations [4], but several studies have extended it to include multi-head attention [9, 10] and learnable dictionary layers [11, 12]. Once the statistics are computed, they are mapped to a segment-level representation, which we call an *x-vector*. After the x-vectors are extracted, the backend technology developed for i-vectors can be reused, particularly probabilistic linear discriminant analysis (PLDA) [13, 14] and domain adaptation techniques [15]. The impact of this new framework was clearly demonstrated in the 2018 NIST Speaker Recognition Evaluation (SRE), where top-performing teams adopted DNN embeddings in lieu of traditional systems based on i-vectors. For the VOiCES challenge, we developed three x-vector systems based on systems that achieved excellent results on SRE 2018.

The report is organized as follows. Section 2 lists the resources used for training, augmentation, and evaluation of the performance of the systems. In Section 3, we describe the three types of DNN embedding-based speaker recognition systems we used in the challenge. In Section 4 we report and analyze results on the development and evaluation datasets. Finally, in Section 5 we conclude the paper.

2. Datasets

2.1. VOiCES datasets

The challenge provided a development set *Dev*, with 196 speakers and about 16,000 audio segments, that was used for system development and tuning. To facilitate domain adaptation, we split the development set into two parts, *Dev1*, which is made up of speakers from *Dev* with even-numbered speaker labels, and *Dev2*, which consists of odd-numbered speakers. The evaluation set *Eval*, was only used to evaluate the submitted systems.

2.2. Training datasets

We only trained systems that conform to the requirements of the fixed condition of the challenge [2]. We used the following datasets in our submissions.

- *VoxCeleb-1+2*: the official distributions of VoxCeleb 1 [6] and 2 [16], which consist of video recordings that have been split into short segments.
- *VoxCelebCat-1+2*: segments from VoxCeleb-1+2 have been concatenated together by video ID.
- *SITW*: single-speaker segments from the Speakers in the Wild dataset [17].

2.3. Augmentation datasets

Data augmentation increases the amount and diversity of the training data, and is a crucial step to obtaining good DNN embedding-based speaker recognition systems [7]. In addition, a set of well-chosen augmentations should reduce the mismatch

http://dx.doi.org/10.21437/Interspeech.2019-2979

Table 1: *Extended TDNN x-vector architecture*

Layer	Layer Type	Context	Size
1	TDNN-ReLU	t-2:t+2	512
2	Dense-ReLU	t	512
3	TDNN-ReLU	t-2, t, t+2	512
4	Dense-ReLU	t	512
5	TDNN-ReLU	t-3, t, t+3	512
6	Dense-ReLU	t	512
7	TDNN-ReLU	t-4, t, t+4	512
8	Dense-ReLU	t	512
9	Dense-ReLU	t	1500
10	Pooling (mean+stddev)	Full-seq	2x1500
11	Dense(Embedding)-ReLU		512
12	Dense-ReLU		512
13	Dense-Softmax		Num. spks.

of the domain of the training data with that of the VOiCES data by simulating its noisy and reverberant conditions. We utilized the following augmentation datasets.

- *Babble*: a dataset composed of audio files of 3 to 5 speakers from the microphone portion of Mixer 6 [18] that have been summed together to create babble noise.
- *Music*: the music files from the MUSAN corpus [19][1] that do not contain vocals.
- *Noise*: the noise files from the MUSAN corpus.
- *Reverb*: simulated RIRs from the AIR dataset [2].

3. SID Systems

We developed three systems based on x-vectors [7] with Gaussian PLDA classifiers [13]. The first two are built using the Kaldi speech recognition toolkit [20], and use either a deep time-delay architecture (see Section 3.1) or in conjunction with factorized layers (see Section 3.2). The final system, described in Section 3.3 was built using Pytorch, and is based on ResNet34. All systems were developed using 16 kHz audio.

3.1. Extended TDNN X-vector System

This architecture, abbreviated in Table 3 as *etdnn*, uses the "extended" x-vector architecture described in [21]. Table 1 summarizes the architecture. The two main differences between this architecture and our earlier work (such as [7]) is a slightly wider temporal context of the frame-level layers and the interleaving of dense layers in between the convolutional layers (this is equivalent to the 1x1 convolutions used in computer vision architectures). This architecture has been found to greatly outperform the architecture described in [7] for the SITW and SRE16 benchmarks, and was our best performing system in SRE18.

The features to the network are 30 dimensional MFCCs, which are centered over a 3 second sliding window. After centering, the Kaldi energy VAD was used to remove non-speech based on the average log-energy in a given window. This VAD requires no training data.

The output of the network is the posterior probabilities of the training speakers and it is trained to minimize a categorical cross entropy. After training, the x-vectors are extracted from layer 11 prior to the ReLU non-linearity. The number of parameters in this architecture is 8 million, but only 4 million are used for extracting the x-vector.

[1]http://www.openslr.org/17
[2]http://www.openslr.org/28

Table 2: *Factorized TDNN x-vector architecture*

Layer	Layer Type	Context factor1	Context factor2	Skip conn. from layer	Size	Inner size
1	TDNN-ReLU	t-2:t+2			512	
2	F-TDNN-ReLU	t-2, t	t, t+2		725	180
3	F-TDNN-ReLU	t	t		725	180
4	F-TDNN-ReLU	t-3, t	t, t+3		725	180
5	F-TDNN-ReLU	t	t	3	725	180
6	F-TDNN-ReLU	t-3, t	t, t+3		725	180
7	F-TDNN-ReLU	t-3, t	t, t+3	2, 4	725	180
8	F-TDNN-ReLU	t-3, t	t, t+3		725	180
9	F-TDNN-ReLU	t	t	4, 6, 8	725	180
10	Dense-ReLU	t	t		1500	
11	Pooling (mean+stddev)	full-seq			2x1500	
12	Dense-ReLU				512	
13	Dense-ReLU				512	
14	Dense-Softmax				N. spks.	

The DNN was trained on the 7,185 speakers in the VoxCeleb-1+2 dataset described in Section 2. Using the augmentations described in Section 2 plus simulated GSM AMR phone encodings[3], we multiply the amount of training data by 6 times, which increased the number of segments from 1.2 million to 7.2 million. From these 7.2 million segments, we extract 108 million shorter segments that are between 2 and 3 seconds long, and train the DNN on those for 6 epochs.

The back-end consists of centering, LDA (from 512 to 200 dimensions), length-normalization, and scoring using a Gaussian PLDA model. The training data for the LDA transform and the unadapted PLDA model were the longest 200,000 segments from VoxCeleb-1+2, along with the augmentations drawn from Section 2.3 which increased the total amount to 800,000 segments. Domain adaptation was handled by using Dev1 for centering and PLDA adaptation (with $\alpha = 0.10$ as described in Section 3.4) when scoring Dev2, and Dev2 when scoring Dev1. When scoring on the evaluation data, we adapted to Dev.

3.2. Factorized TDNN X-vector System

For the second x-vector architecture, we replaced the pre-pooling layers by a factorized TDNN (called *ftdnn* in Table 3) with skip connections [22]. Table 2 summarizes the layers in our F-TDNN x-vector. The F-TDNN reduces the number of parameters of the network by factorizing the weight matrix of each TDNN layer into the product of two low-rank matrices. The first of those factors is constrained to be semi-orthogonal. It is assumed that the semi-orthogonal constrain will help to assure that we do not lose information when projecting from the high dimension to the low-rank dimension.

The authors of the original paper found that, instead of factorizing the TDNN layer into a convolution times a feed-forward layer, it is better to factorize the layer into two convolutions with half the kernel size. For example, instead of using a kernel with context (-2, 0, 2) in the first factor of the layer and 0 context in the second factor, it is better to use a kernel with context (-2,0) in the first factor and a kernel with context (0, +2) in the second factor.

As in other architectures like ResNet [23], we incorporate skip connections. This means that some layers receive as input, not only the previous layer but also the output from other prior layers. The prior layers were concatenated to the input of the current layer, instead of summed like in ResNet. This allows us to make the network deeper by alleviating the vanishing gradient problem. According to [22], the best option is to create skip connections between the low-rank interior layers of the F-TDNN.

[3]`http://www.3gpp.org/ftp/Specs/archive/26_series/26.073/26073-800.zip`

The features for this network were 40 dimensional MFCCs short-time centered with a 3 second sliding window. Speech activity detection (SAD) was performed using a pretrained Kaldi SAD model, based on the Aspire recipe[4]. X-vectors were extracted from layer 12 before the ReLU non-linearity. The x-vector network was trained on the VoxCelebCat-1+2 dataset with the augmentations described in Section 2.3. This multiplies the amount of training by 3 times. The network is then trained for three iterations.

The back-end consisted of LDA from 512 to 300 dimensions, centering, whitening, length normalization and Gaussian simplified PLDA model. The training data for the back-end was the same as for the x-vector network. The centering and PLDA within-class covariance matrix were adapted to the voices development set with $\alpha = 1$ and $\alpha = 0.3$ respectively (see Section 3.4). To score the Dev1, we adapted the model using Dev2, and vice-versa. To score the eval set we adapted using the whole development set.

3.3. ResNet34-LDE System

The ResNet-LDE (called *resnet* in Table 3) is a variant of the x-vector system where the TDNN layers are replaced by a residual network with 2D convolutions (ResNet34) [23] and the pooling layer is replaced by a learnable dictionary encoding (LDE) layer [11, 12].

The original x-vector framework assumes that the frame-level TDNN representations before the pooling layer are unimodal. Thus, to pool those representations, we just compute their mean and standard deviation. Instead, the LDE pooling assumes that frame-level representations are distributed in C clusters and it learns a dictionary with the centers of those clusters. This is reminiscent of the GMM-i-vector paradigm. The component posteriors are obtained as,

$$w_{t,c} = \frac{\exp(-s_c \|\mathbf{x}_t - \boldsymbol{\mu}_c\|^2 + b_c)}{\sum_{c=1}^{C} \exp(-s_c \|\mathbf{x}_t - \boldsymbol{\mu}_c\|^2 + b_c)} \tag{1}$$

where s_c is an isotropic precision; and b_c includes the log-weight and log-normalizing constant of the Gaussian. The bias term was not included in the original paper but we found that it slightly improves the results.

Then, we compute an embedding per component

$$\mathbf{e}_c = \frac{\sum_{t=1}^{T} w_{t,c}(\mathbf{x}_t - \boldsymbol{\mu}_c)}{\sum_{t=1}^{T} w_{t,c}} \qquad c = 1, \dots, C \tag{2}$$

and we concatenate together the embeddings for all the components $\mathbf{e} = (\mathbf{e}_1^{\mathrm{T}}, \dots, \mathbf{e}_C^{\mathrm{T}})^{\mathrm{T}}$. This embedding has the same role as the super-vector in GMM-i-vectors. This super-vector is projected to a lower dimension to obtain the x-vector embedding. This projection has the same role as the total variability matrix in i-vectors.

Instead of using the categorical cross-entropy for training, we used the angular softmax loss [24]. The angular softmax loss has stronger requirements for correct classification when $m \geq 2$ (an integer that controls the angular margin), which generates an angular classification margin between embeddings of different classes [12, 25]. Prior to training with the angular softmax loss, we first pre-train with the categorical cross entropy, and use the resulting model for initialization.

This x-vector system uses 40 dimensional log-filter-bank features. The VAD, training data and back-end are the same as in the F-TDNN x-vectors described in Section 3.2.

[4]http://kaldi-asr.org/models/m4

3.4. PLDA Adaptation

In each system, we adapt the original out-of-domain PLDA with the development data, using the parameter interpolation method [15]. When producing scores for Dev1, we use Dev2 for adapation, and vice versa. When producing scores for Eval, we adapt using all of Dev. In the adapted model, the within-class and across-class covariances are a weighted sum of the out-of-domain $\mathbf{S}_{\mathrm{out}}$ and in-domain $\mathbf{S}_{\mathrm{in}}$ covariances,

$$\mathbf{S}_{\mathrm{adapt}} = \alpha \mathbf{S}_{\mathrm{in}} + (1 - \alpha)\mathbf{S}_{\mathrm{out}} \tag{3}$$

3.5. Calibration and fusion

Fusion and Calibration was performed using linear logistic regression with the Bosaris toolkit [26]. To select the best fusion combination, we followed a greedy fusion scheme. First, we calibrate all the systems and select the best one given the lowest actual cost. We fix that as the best system and evaluate all the two system fusions that include the best system, and select this as the best fusion of two systems. We fix those two systems and then add the third system. To reduce the chances of over-fitting, we prioritize fusions with only positive weights.

4. Results

In this section, we report results of the three x-vector-based systems as well as their fusions on the development and evaluation data, abbreviated as *Dev* and *Eval* respectively. In Table 3 the extended TDNN, ResNet34+LDE, and factorized TDNN systems are abbreviated as *etdnn*, *resnet*, and *ftdnn* respectively. The first block of the table shows the results of the individual systems and their fusions on the entire Dev and Eval sets. In the second block, the performance of one representative system is broken down by noise condition. In the third block, we present performance by microphone position; we selected a few representative positions out of a larger set. The final block breaks down the performance of a single system by domain adaptation techniques. Results are reported in terms of the primary evaluation metric used by VOiCES 2019, which was the actual detection cost function with $P_{tar} = 0.01$ and equal costs of misses and false alarms (Act DCF); the minimum of the detection cost function (Min DCF); and the equal error rate (EER).

4.1. Submissions

In Table 3, we see that the etdnn system achieved the best single-system results on Dev, with an Act DCF of 0.17. In comparison, the other two systems obtained slightly over 0.2 Act DCF. The fusion of the etdnn and resnet systems improved over the etdnn system alone, by 23% in Act DCF, and the fusion of all three systems was slightly better. As a result of the observed Dev performance, our submission to the challenge consisted of three systems: a primary system consisting of the fusion of etdnn, resent and ftdnn (system #5 in Table 3); a system using etdnn alone (system #1 in Table 3); and the fusion of the etdnn and resnet systems (system #4 in Table 3).

The last three columns of Table 3 show the performance of our submissions on the Eval data. We observe that calibration on Dev was effective, as the difference between the Min DCF and Act DCF is small. However, error rates in general are much higher on the Eval set. On the Dev set, our primary submission obtained an EER of about 1%. However, on Eval set this was almost 5x higher. Although the etdnn system achieved the best single-system performance on Dev, that was not reflected in the Eval performance. On the Eval data, all three systems

Table 3: *Results on the VOiCES Dev and Eval data, followed by a breakdown by noise condition, mic. position and adaptation type.*

			VOiCES 2019 Dev			VOiCES 2019 Eval		
#	Systems		**EER[%]**	**Min DCF**	**Act DCF**	**EER[%]**	**Min DCF**	**Act DCF**
1	etdnn		1.50	0.169	0.170	5.87	0.432	0.435
2	resnet		1.55	0.209	0.210	5.09	0.417	0.419
3	ftdnn		1.46	0.202	0.203	6.18	0.438	0.439
4	fusion 1,2		1.12	0.131	0.132	4.83	0.364	0.370
5	fusion 1,2,3		1.03	0.128	0.129	4.80	0.355	0.362
	ftdnn	**noise cond**						
		all	1.46	0.202	0.203	6.18	0.438	0.439
		none	1.05	0.173	0.175	3.91	0.349	0.350
		babble	1.56	0.213	0.215	9.24	0.579	0.581
		music	1.61	0.222	0.224	-	-	-
		TV	1.48	0.200	0.200	4.90	0.385	0.386
	ftdnn	**mic position**						
		closest	1.20	0.167	0.168	2.60	0.273	0.275
		mid-distance	1.63	0.204	0.206	2.81	0.261	0.265
		farthest	1.80	0.231	0.233	9.78	0.601	0.603
		wall (fully obstructed)	1.91	0.229	0.230	15.12	0.874	0.876
	ftdnn	**adaptation**						
		none	1.68	0.225	0.227	6.10	0.435	0.439
		s-norm	2.01	0.240	0.241	7.06	0.420	0.445
		plda	1.46	0.202	0.203	6.18	0.438	0.439
		plda + s-norm	1.72	0.208	0.209	7.08	0.418	0.432

performed similarly, with resnet slightly better than the other two. The fusion of etdnn and resnet improved performance over etdnn alone (our single-system submission) by 15% in Act DCF, while the fusion of all three systems was better by 17%.

4.2. Noise Conditions

In an effort to better understand why error rates on the Eval data are so much higher than the Dev data, we've broken down the results for the ftdnn system by noise condition. Note that the trends are quite similar for the other systems. As expected, the condition without any distractor noises (*none* in Table 3) resulted in the best performance for both Dev and Eval at all operating points. Error rates on the Eval data are higher than the Dev data in general, and this is true even for the condition without any noise applied. So it is likely that the higher error rates are due to more challenging rooms or microphone types used in the Eval data, rather than the distractor noises. It is also possible that noise volume was louder in the eval data or that the noise source was closer to the microphones. The babble condition was the most difficult for both Dev and Eval, with an EER of 9.24% for Eval, and only 1.56% for Dev. The music condition did not appear to be present in the Eval data.

4.3. Microphone positions

There were 8 different microphone positions in the Dev data and 11 positions in the Eval data. We selected what we thought as the four most representative positions: closest to the speaker, mid-distance, farthest, and fully obstructed by a wall. Different rooms were used in Dev and Eval–room 1 for Dev enroll, room 2 for Dev test, room 3 for Eval enroll and room 4 for Eval test, so we observed different performance between Dev and Eval for a position with the same name. In Dev, there is small performance degradation as the mic gets farther from the speaker. However in Eval, there is huge degradation for the farthest microphone (EER multiplies by $\sim$ 4) and the wall microphone (EER multiplies by $\sim$ 6) w.r.t. the closest mic. We assume that this is because room 4 has larger reverberation–it may be larger than room 2 or the walls have more reflective materials, or because the noise sources were closest to these two microphones.

4.4. Adaptation

The last four rows of Table 3 break down the results of ftdnn by standard domain adaptation techniques. The other systems (etdnn and resnet) follow similar trends. We found that adapting the PLDA model to Dev using the method described in Section 3.4 achieved better results than either no adaptation or adaptive score normalization. As a result, all of our submitted systems used this form of adaptation only. However, we observe that on Eval, there appears to be no clear performance trend. In terms of Act DCF, the best performance is obtained by PLDA adaptation plus score normalization, and achieved 0.432, while no adaptation and PLDA adaptation both obtained 0.439.

5. Conclusions

This paper described the systems we developed for the speaker recognition track of the VOiCES 2019 challenge, and we presented some cursory post-evaluation analysis. Our systems consisted of three x-vector DNNs with PLDA backends that were closely based on those that obtained excellent results in SRE 2018. We found that, individually, these systems achieved competitive results on the VOiCES evaluation data, and were similar to each other in performance. Our primary submission, which was a fusion of the these three systems, provided a significant improvement over our single-system submission alone, although most of the gains were due to fusing only two of the systems. Finally, we found that the evaluation set was more challenging than the development data, as the very low error-rates we achieved on the development set were not representative of the performance we obtained on the evaluation set. In the future, we plan to better understand this mismatch.

6. Acknowledgements

This work was partially supported by the DARPA MATERIAL and LORELEI programs (FA8650-17-C-9115, HR0011-15-2-0024) and an unrestricted gift from Mobvoi Inc.

7. References

[1] C. Richey, M. Barrios, Z. Armstrong, C. Bartels, H. Franco, M. Graciarena, A. Lawson, A. Nandwana, M. K andStauffer, J. van Hout, P. Gamble, J. Hetherly, C. Stephenson, and K. Ni, "Voices obscured in complex environmental settings (voices) corpus," in *ISCA INTERSPEECH 2018*. ISCA, 2018.

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

[3] 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.

[4] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 165–170.

[5] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep Neural Network Embeddings for Text-Independent Speaker Verification," in *Proceedings of the 18th Annual Conference of the International Speech Communication Association, INTERSPEECH 2017*. Stockholm, Sweden: ISCA, aug 2017, pp. 999–1003.

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

[7] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors : Robust DNN Embeddings for Speaker Recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*. Alberta, Canada: IEEE, apr 2018, pp. 5329–5333.

[8] P. Kenny, "Notes on modeling i-vectors," CRIM, Montreal, Quebec, Canada, Tech. Rep., 2010.

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

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

[11] W. Cai, Z. Cai, X. Zhang, X. Wang, and M. Li, "A Novel Learnable Dictionary Encoding Layer for End-to-End Language Identification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Calgary, Canada: IEEE, apr 2018, pp. 5189–5193.

[12] W. Cai, J. Chen, and M. Li, "Exploring the Encoding Layer and Loss Function in End-to-End Speaker and Language Recognition System," in *Odyssey 2018 The Speaker and Language Recognition Workshop*. Les Sables d'Olonne, France: ISCA, jun 2018, pp. 74–81.

[13] S. Ioffe, "Probabilistic linear discriminant analysis," in *Proceedings of the 9th European Conference on Computer Vision, ECCV 2006*, ser. LNCS, A. Leonardis, H. Bischof, and A. Pinz, Eds., vol. 3954. Graz, Austria: Springer-Verlag Berlin, Heidelberg, may 2006, pp. 531–542.

[14] S. J. Prince and J. H. Elder, "Probabilistic Linear Discriminant Analysis for Inferences About Identity," in *Proceedings of the IEEE International Conference on Computer Vision, ICCV 2007*. Rio de Janeiro, Brazil: IEEE, oct 2007, pp. 1–8.

[15] D. Garcia-Romero and A. McCree, "Supervised domain adaptation for i-vector based speaker recognition," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4047–4051.

[16] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *INTERSPEECH*, 2018.

[17] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The 2016 speakers in the wild speaker recognition evaluation." in *Interspeech*, 2016, pp. 823–827.

[18] L. D. Consortium, "Mixer 6 corpus specification v4.1,," 2013.

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

[20] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU2011*. Waikoloa, HI, USA: IEEE, dec 2011, pp. 1–4.

[21] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, A. McCree, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019*. IEEE, 2019.

[22] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohamadi, and S. Khudanpur, "Semi-Orthogonal Low-Rank Matrix Factorization for Deep Neural Networks," in *Proceedings of the 19th Annual Conference of the International Speech Communication Association, INTERSPEECH 2018*, Hyderabad, India, sep 2018.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," dec 2015.

[24] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "SphereFace: Deep Hypersphere Embedding for Face Recognition," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2017-Janua. IEEE, jul 2017, pp. 6738–6746.

[25] Z. Huang, S. Wang, and K. Yu, "Angular Softmax for Short-Duration Text-independent Speaker Verification," in *Interspeech 2018*. Hyderabad, India: ISCA, sep 2018, pp. 3623–3627.

[26] N. Brummer and E. De Villiers, "The BOSARIS Toolkit: Theory, Algorithms and Code for Surviving the New DCF," in *NIST SRE11 Speaker Recognition Workshop*, Atlanta, Georgia, USA, dec 2011, pp. 1–23.