X-VECTORS: ROBUST DNN EMBEDDINGS FOR SPEAKER RECOGNITION

David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, Sanjeev Khudanpur

Center for Language and Speech Processing & Human Language Technology Center of Excellence
The Johns Hopkins University, Baltimore, MD 21218, USA

ABSTRACT

In this paper, we use data augmentation to improve performance of deep neural network (DNN) embeddings for speaker recognition. The DNN, which is trained to discriminate between speakers, maps variable-length utterances to fixed-dimensional embeddings that we call x-vectors. Prior studies have found that embeddings leverage large-scale training datasets better than i-vectors. However, it can be challenging to collect substantial quantities of labeled data for training. We use data augmentation, consisting of added noise and reverberation, as an inexpensive method to multiply the amount of training data and improve robustness. The x-vectors are compared with i-vector baselines on Speakers in the Wild and NIST SRE 2016 Cantonese. We find that while augmentation is beneficial in the PLDA classifier, it is not helpful in the i-vector extractor. However, the x-vector DNN effectively exploits data augmentation, due to its supervised training. As a result, the x-vectors achieve superior performance on the evaluation datasets.

***Index Terms*—** speaker recognition, deep neural networks, data augmentation, x-vectors

1. INTRODUCTION

Using deep neural networks (DNN) to capture speaker characteristics is currently a very active research area. In our approach, representations called x-vectors are extracted from a DNN and used like i-vectors. This paper builds on our recent DNN embedding architecture [1]. We show that artificially augmenting the training data with noises and reverberation is a highly effective strategy for improving performance in DNN embedding systems.

Most speaker recognition systems are based on i-vectors [2]. The standard approach consists of a universal background model (UBM), and a large projection matrix $\mathbf{T}$ that are learned in an unsupervised way to maximize the data likelihood. The projection maps high-dimensional statistics from the UBM into a low-dimensional representation, known as an i-vector. A probabilistic linear discriminant analysis (PLDA) [3] classifier is used to compare i-vectors, and enable same-or-different speaker decisions [4, 5, 6].

The DNNs most often found in speaker recognition are trained as acoustic models for automatic speech recognition (ASR), and are then used to enhance phonetic modeling in the i-vector UBM: either posteriors from the ASR DNN replace those from a Gaussian mixture model (GMM) [7, 8], or bottleneck features are extracted from the DNN and combined with acoustic features [9]. In either case, if the ASR DNN is trained on in-domain data, the improvement over traditional acoustic i-vectors is substantial [10, 11, 12]. However, this approach introduces the need for transcribed training data and greatly increases computational complexity compared to traditional i-vectors.

Alternatively, neural networks can be directly optimized to discriminate between speakers. This has potential to produce powerful, compact systems [13], that only require speaker labels to train. In early systems, neural networks are trained to separate speakers, and frame-level representations are extracted from the network and used as features for Gaussian speaker models [14, 15, 16]. Heigold et al., introduced an end-to-end system, trained on the phrase "OK Google," that jointly learns an embedding along with a similarity metric to compare pairs of embeddings [13]. Snyder et al., adapted this approach to a text-independent application and inserted a temporal pooling layer into the network to handle variable-length segments [17]. The work in [1] split the end-to-end approach into two parts: a DNN to produce embeddings and a separately trained classifier to compare them. This facilitates the use of all the accumulated backend technology developed over the years for i-vectors, such as length-normalization, PLDA scoring, and domain adaptation techniques.

DNN embedding performance appears to be highly scalable with the amount of training data. As a result, these systems have found success leveraging large proprietary datasets [13, 17, 18]. However, recent systems have shown promising performance trained on only publicly available speaker recognition corpora [1, 19, 20]. This paper is based on the work in [1] and applies data augmentation to the DNN training procedure. This increases the amount and diversity of the existing training data, and achieves a significant improvement for the x-vector system. In comparing with x-vectors, we also contribute a study of augmentation in i-vector systems.

2. SPEAKER RECOGNITION SYSTEMS

This section describes the speaker recognition systems developed for this study, which consist of two i-vector baselines and the DNN x-vector system. All systems are built using the Kaldi speech recognition toolkit [21].

2.1. Acoustic i-vector

A traditional i-vector system based on the GMM-UBM recipe described in [11] serves as our acoustic-feature baseline system. The features are 20 MFCCs with a frame-length of 25ms that are mean-normalized over a sliding window of up to 3 seconds. Delta and acceleration are appended to create 60 dimension feature vectors. An energy-based speech activity detection (SAD) system selects features corresponding to speech frames. The UBM is a 2048 component full-covariance GMM. The system uses a 600 dimensional i-vector extractor and PLDA for scoring (see Section 2.4).

2.2. Phonetic bottleneck i-vector

This i-vector system incorporates phonetic bottleneck features (BNF) from an ASR DNN acoustic model and is similar to [9]. The DNN is a time-delay acoustic model with p-norm nonlinearities. The ASR DNN is trained on the Fisher English corpus and uses the same recipe and architecture as the system described in Section 2.2 of [11], except that the penultimate layer is replaced with a 60 dimensional linear bottleneck layer. Excluding the softmax output layer, which is not needed to compute BNFs, the DNN has 9.2 million parameters.

The BNFs are concatenated with the same 20 dimensional MFCCs described in Section 2.1 plus deltas to create 100 dimensional features. The remaining components of the system (feature processing, UBM, i-vector extractor, and PLDA classifier) are identical to the acoustic system in Section 2.1.

2.3. The x-vector system

This section describes the x-vector system. It is based on the DNN embeddings in [1] and described in greater detail there.

Our software framework has been made available in the Kaldi toolkit. An example recipe is in the main branch of Kaldi at `https://github.com/kaldi-asr/kaldi/tree/master/egs/sre16/v2` and a pretrained x-vector system can be downloaded from `http://kaldi-asr.org/models.html`. The recipe and model are similar to the x-vector system described in Section 4.4.

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	120x512
frame2	$\{t-2, t, t+2\}$	9	1536x512
frame3	$\{t-3, t, t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0, T)$	T	1500Tx3000
segment6	$\{0\}$	T	3000x512
segment7	$\{0\}$	T	512x512
softmax	$\{0\}$	T	512xN

Table 1. The embedding DNN architecture. x-vectors are extracted at layer *segment6*, before the nonlinearity. The N in the softmax layer corresponds to the number of training speakers.

The features are 24 dimensional filterbanks with a frame-length of 25ms, mean-normalized over a sliding window of up to 3 seconds. The same energy SAD as used in the baseline systems filters out nonspeech frames.

The DNN configuration is outlined in Table 1. Suppose an input segment has T frames. The first five layers operate on speech frames, with a small temporal context centered at the current frame t. For example, the input to layer *frame3* is the spliced output of *frame2*, at frames $t-3$, t and $t+3$. This builds on the temporal context of the earlier layers, so that *frame3* sees a total context of 15 frames.

The statistics pooling layer aggregates all T frame-level outputs from layer *frame5* and computes its mean and standard deviation. The statistics are 1500 dimensional vectors, computed once for each input segment. This process aggregates information across the time dimension so that subsequent layers operate on the entire segment. In Table 1, this is denoted by a layer context of $\{0\}$ and a total context of T. The mean and standard deviation are concatenated together and propagated through segment-level layers and finally the softmax output layer. The nonlinearities are all rectified linear units (ReLUs).

The DNN is trained to classify the N speakers in the training data. A training example consists of a chunk of speech features (about 3 seconds average), and the corresponding speaker label. After training, embeddings are extracted from the affine component of layer *segment6*. Excluding the softmax output layer and *segment7* (because they are not needed after training) there is a total of 4.2 million parameters.

2.4. PLDA classifier

The same type of PLDA [3] classifier is used for the x-vector and i-vector systems. The representations (x-vectors or i-vectors) are centered, and projected using LDA. The LDA dimension was tuned on the SITW development set to 200 for i-vectors and 150 for x-vectors. After dimensionality reduction, the representations are length-normalized and modeled by PLDA. The scores are normalized using adaptive s-norm [22].

3. EXPERIMENTAL SETUP

3.1. Training data

The training data consists of both telephone and microphone speech, the bulk of which is in English. All wideband audio is downsampled to 8kHz.

The *SWBD* portion consists of Switchboard 2 Phases 1, 2, and 3 as well as Switchboard Cellular. In total, the SWBD dataset contains about 28k recordings from 2.6k speakers. The *SRE* portion consists of NIST SREs from 2004 to 2010 along with Mixer 6 and contains about 63k recordings from 4.4k speakers. In the experiments in Sections 4.1–4.4 the extractors (UBM/**T** or embedding DNN) are trained on SWBD and SRE and the PLDA classifiers are trained on just SRE. Data augmentation is described in Section 3.3 and is applied to these datasets as explained throughout Section 4.

In the last experiment in Section 4.5 we incorporate audio from the new *VoxCeleb* dataset [19] into both extractor and PLDA training lists. The dataset consists of videos from 1,251 celebrity speakers. Although SITW and VoxCeleb were collected independently, we discovered an overlap of 60 speakers between the two datasets. We removed the overlapping speakers from VoxCeleb prior to using it for training. This reduces the size of the dataset to 1,191 speakers and about 20k recordings.

The ASR DNN used in the i-vector (BNF) system was trained on the Fisher English corpus. To achieve a limited form of domain adaptation, the development data from SITW and SRE16 is pooled and used for centering and score normalization. No augmentation is applied to these lists.

3.2. Evaluation

Our evaluation consists of two distinct datasets: Speakers in the Wild (SITW) Core [23] and the Cantonese portion of the NIST SRE 2016 evaluation (SRE16) [24]. SITW consists of unconstrained video audio of English speakers, with naturally occurring noises, reverberation, as well as device and codec variability. The SRE16 portion consists of Cantonese conversational telephone speech. Both enroll and test SITW utterances vary in length form 6–240 seconds. For SRE16, the enrollment utterances contain about 60 seconds of speech while the test utterances vary from 10–60 seconds.

			SITW Core			SRE16 Cantonese		
			EER(%)	DCF10^{-2}	DCF10^{-3}	EER(%)	DCF10^{-2}	DCF10^{-3}
4.1	Original systems	i-vector (acoustic)	9.29	0.621	0.785	9.23	0.568	0.741
		i-vector (BNF)	**9.10**	**0.558**	**0.719**	9.68	0.574	0.765
		x-vector	9.40	0.632	0.790	**8.00**	**0.491**	**0.697**
4.2	PLDA aug.	i-vector (acoustic)	8.64	0.588	0.755	8.92	0.544	0.717
		i-vector (BNF)	8.00	**0.514**	**0.689**	8.82	0.532	0.726
		x-vector	**7.56**	0.586	0.746	**7.45**	**0.463**	**0.669**
4.3	Extractor aug.	i-vector (acoustic)	8.89	0.626	0.790	9.20	0.575	0.748
		i-vector (BNF)	7.27	**0.533**	0.730	8.89	0.569	0.777
		x-vector	**7.19**	0.535	**0.719**	**6.29**	**0.428**	**0.626**
4.4	PLDA and extractor aug.	i-vector (acoustic)	8.04	0.578	0.752	8.95	0.555	0.720
		i-vector (BNF)	6.49	0.492	0.690	8.29	0.534	0.749
		x-vector	**6.00**	**0.488**	**0.677**	**5.86**	**0.410**	**0.593**
4.5	Incl. VoxCeleb	i-vector (acoustic)	7.45	0.552	0.723	9.23	0.557	0.742
		i-vector (BNF)	6.09	0.472	0.660	8.12	0.523	0.751
		x-vector	**4.16**	**0.393**	**0.606**	**5.71**	**0.399**	**0.569**

Table 2. Results using data augmentation in various systems. "Extractor" refers to either the UBM/**T** or the embedding DNN. For each experiment, the best results are **boldface**.

We report results in terms of equal error-rate (EER) and the minimum of the normalized detection cost function (DCF) at $P_{\text{Target}} = 10^{-2}$ and $P_{\text{Target}} = 10^{-3}$. Note that the SRE16 results have not been "equalized [24]."

3.3. Data augmentation

Augmentation increases the amount and diversity of the existing training data. Our strategy employs additive noises and reverberation. Reverberation involves convolving room impulse responses (RIR) with audio. We use the simulated RIRs described by Ko et al. in [25], and the reverberation itself is performed with the multi-condition training tools in the Kaldi *ASpIRE* recipe [21]. For additive noise, we use the MUSAN dataset, which consists of over 900 noises, 42 hours of music from various genres and 60 hours of speech from twelve languages [26]. Both MUSAN and the RIR datasets are freely available from `http://www.openslr.org`.

We use a 3-fold augmentation that combines the original "clean" training list with two augmented copies. To augment a recording, we choose between one of the following randomly:

- **babble**: Three to seven speakers are randomly picked from MUSAN speech, summed together, then added to the original signal (13-20dB SNR).
- **music**: A single music file is randomly selected from MUSAN, trimmed or repeated as necessary to match duration, and added to the original signal (5-15dB SNR).
- **noise**: MUSAN noises are added at one second intervals throughout the recording (0-15dB SNR).
- **reverb**: The training recording is artificially reverberated via convolution with simulated RIRs.

4. RESULTS

The main results are presented in Table 2 and are referenced throughout Sections 4.1–4.5. We compare performance of two i-vector systems, labeled *i-vector (acoustic)* and *i-vector (BNF)*, with the *x-vector* system. The systems are described in Sections 2.1, 2.2 and 2.3, respectively. Throughout the following sections, we use the term *extractor* to refer to either the UBM/**T** or the embedding DNN.

4.1. Original systems

In this section, we evaluate systems without data augmentation. The extractors are trained on the SWBD and SRE datasets described in Section 3.1. The PLDA classifiers are trained on just the SRE dataset. Without using augmentation, the best results on SITW are obtained by i-vector (BNF), which is 12% better than the x-vector system at DCF10^{-2}. The acoustic i-vector system also achieves slightly lower error-rates than the x-vector system on SITW. However, even without augmentation, the best results for SRE16 Cantonese are obtained by the x-vectors. In terms of DCF10^{-2}, these embeddings are about 14% better than either i-vector system. We observe that i-vector (BNF) has no advantage over i-vector (acoustic) for this Cantonese speech. This echoes recent studies that have found that the large gains achieved by BNFs in English speech are not necessarily transferable to non-English data [27].

4.2. PLDA augmentation

In this experiment, the augmentation strategy described in Section 3.3 is applied to only the PLDA training list. We use the same extractors as the previous section, which were trained on the original datasets. PLDA augmentation results in a clear improvement for all three systems relative to Section 4.1. However, it appears that the x-vectors may benefit from the PLDA augmentation more than the baseline systems. On SITW, the x-vector system achieves slightly lower error-rates than i-vector (acoustic), but continues to lag behind i-vector (BNF) at most operating points. On SRE16, the x-vectors maintain an advantage over the i-vectors by about 14% in DCF10^{-2}.

4.3. Extractor augmentation

We now apply data augmentation to the extractor (UBM/**T** or embedding DNN) training lists but *not* the PLDA list. The effect of augmenting the UBM/**T** is inconsistent in the i-vector system. This observation is supported by prior studies on i-vectors, which have found that augmentation is only effective in the PLDA classifier [28, 29]. On the other hand, augmenting the embedding DNN training list results in a large improvement. In contrast to the i-vector systems, this is considerably more effective than augmenting the PLDA training list. On SITW, the x-vector system achieves lower error-rates than i-vector (acoustic) and has now caught up to the i-vector (BNF) system. On SRE16, the x-vectors are now 25% better than the i-vectors in DCF10^{-2}, which is almost double the improvement the DNN embeddings had with PLDA augmentation alone. The findings in this section indicate that data augmentation is only beneficial for extractors trained with supervision.

4.4. PLDA and extractor augmentation

In the previous sections, we saw that PLDA augmentation was helpful in both i-vector and DNN embedding systems, although extractor augmentation was only clearly beneficial in the embedding system. In this experiment, we apply data augmentation to both the extractor *and* PLDA training lists. We continue to use SWBD and SRE for extractor training and only SRE for PLDA. On SITW the x-vectors are now 10-25% better than i-vector (acoustic) and are slightly better than i-vector (BNF) at all operating points. On SRE16 Cantonese, the x-vectors continue to maintain the large lead over the i-vector systems established in Section 4.3.

4.5. Including VoxCeleb

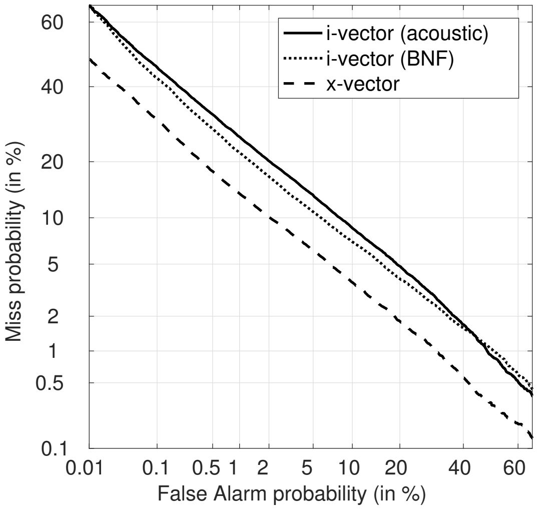

Fig. 1. DET curve for the Cantonese portion of NIST SRE16 using Section 4.5 systems.

The training data in Sections 4.1–4.4 is dominated by telephone speech. In this experiment, we explore the effect of adding a large amount of microphone speech to the systems in Section 4.4. The VoxCeleb dataset [19] is augmented, and added to both the extractor and PLDA lists. As noted in Section 3.1, we found 60 speakers which overlap with SITW; all speech for these speakers was removed from the training lists.

On SITW, both i-vector and x-vector systems improve significantly. However, the x-vector exploits the large increase in the amount of in-domain data better than the i-vector systems. Compared to i-vector (acoustic), the x-vectors are better by 44% in EER and 29% in DCF10^{-2}. Compared to the i-vector (BNF) system, it is now better by 32% in EER and 17% in DCF10^{-2}. On SRE16, the i-vector systems remain roughly the same compared to Section 4.4, but the x-vectors improve on all operating points by a small amount.

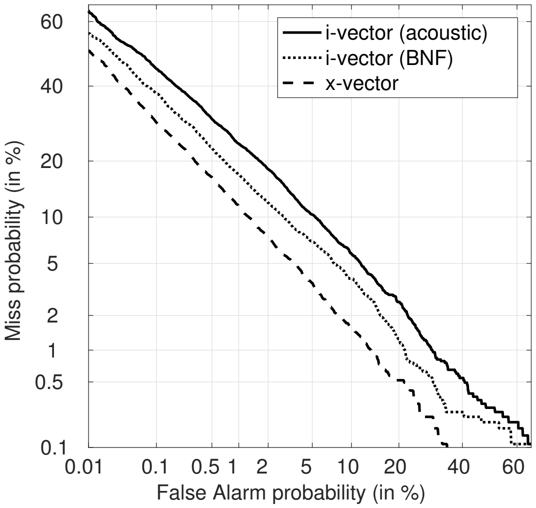

Fig. 2. DET curve for the SITW Core using Section 4.5 systems.

These results are illustrated by detection error tradeoff (DET) plots in Figures 1 and 2.

5. CONCLUSIONS

This paper studied DNN embeddings for speaker recognition. We found that data augmentation is an easily implemented and effective strategy for improving their performance. We also made the x-vector system – our implementation of DNN embeddings – available in the Kaldi toolkit. We found that the x-vector system significantly outperformed two standard i-vector baselines on SRE16 Cantonese. After including a large amount of augmented microphone speech, the x-vectors achieved much lower error-rates than our best baseline on Speakers in the Wild. Bottleneck features from an ASR DNN are used in our best i-vector system, and so it requires transcribed data during training. On the other hand, the x-vector DNN needs only speaker labels to train, making it potentially ideal for domains with little transcribed speech. More generally, it appears that x-vectors are now a strong contender for next-generation representations for speaker recognition.

6. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 1232825. This work was partially supported by NSF Grant No CRI-1513128. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors(s) and do not necessarily reflect the views of the National Science Foundation.

7. REFERENCES

[1] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, “Deep neural network embeddings for text-independent speaker verification,” *Proc. Interspeech*, pp. 999–1003, 2017.

[2] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[3] S. Ioffe, “Probabilistic linear discriminant analysis,” *Computer Vision–ECCV 2006*, pp. 531–542, 2006.

[4] P. Kenny, “Bayesian speaker verification with heavy-tailed priors.,” in *Odyssey*, 2010, p. 14.

[5] N. Brümmer and E. De Villiers, “The speaker partitioning problem.,” in *Odyssey*, 2010, p. 34.

[6] D. Garcia-Romero and C. Espy-Wilson, “Analysis of i-vector length normalization in speaker recognition systems.,” in *Interspeech*, 2011, pp. 249–252.

[7] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, “A novel scheme for speaker recognition using a phonetically-aware deep neural network,” in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1695–1699.

[8] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, “Deep neural networks for extracting Baum-Welch statistics for speaker recognition,” in *Proc. Odyssey*, 2014.

[9] M. McLaren, Y. Lei, and L. Ferrer, “Advances in deep neural network approaches to speaker recognition,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4814–4818.

[10] D. Garcia-Romero, X. Zhang, A. McCree, and D. Povey, “Improving speaker recognition performance in the domain adaptation challenge using deep neural networks,” in *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 2014, pp. 378–383.

[11] D. Snyder, D. Garcia-Romero, and D. Povey, “Time delay deep neural network-based universal background models for speaker recognition,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 92–97.

[12] S. O. Sadjadi, J. Pelecanos, and S. Ganapathy, “The ibm speaker recognition system: Recent advances and error analysis,” *Interspeech*, pp. 3633–3637, 2016.

[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] Y. Konig, L. Heck, M. Weintraub, and K. Sonmez, “Nonlinear discriminant feature extraction for robust text-independent speaker recognition,” in *Proc. RLA2C, ESCA workshop on Speaker Recognition and its Commercial and Forensic Applications*, 1998.

[15] L. Heck, Y. Konig, K. Sonmez, and M. Weintraub, “Robustness to telephone handset distortion in speaker recognition by discriminative feature design,” in *Speech Communication*, 2000, vol. 31, pp. 181–192.

[16] A. Salman, *Learning speaker-specific characteristics with deep neural architecture*, Ph.D. thesis, University of Manchester, 2012.

[17] 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 *Spoken Language Technology Workshop (SLT)*. IEEE, 2016.

[18] S. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, “End-to-end attention based text-dependent speaker verification,” in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 171–178.

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

[20] C. Zhang and K. Koishida, “End-to-end text-independent speaker verification with triplet loss on short utterances,” *Proc. Interspeech*, pp. 1487–1491, 2017.

[21] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, et al., “The Kaldi speech recognition toolkit,” in *Proceedings of the Automatic Speech Recognition & Understanding (ASRU) Workshop*, 2011.

[22] D. Sturim and D. Reynolds, “Speaker adaptive cohort selection for tnorm in text-independent speaker verification,” in *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP’05). IEEE International Conference on*. IEEE, 2005, vol. 1, pp. I–741.

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

[24] “NIST speaker recognition evaluation 2016,” `https://www.nist.gov/itl/iad/mig/speaker-recognition-evaluation-2016/`, 2016.

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

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

[27] O. Novotný, P. Matějka, O. Glembeck, O Plchot, F. Grézl, L. Burget, and J. Černocký, “Analysis of the dnn-based sre systems in multi-language conditions,” in *Spoken Language Technology Workshop (SLT)*. IEEE, 2016.

[28] Y. Lei, L. Burget, L. Ferrer, M. Graciarena, and N. Scheffer, “Towards noise-robust speaker recognition using probabilistic linear discriminant analysis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4253–4256.

[29] D. Garcia-Romero, X. Zhou, and C. Espy-Wilson, “Multicondition training of Gaussian plda models in i-vector space for noise and reverberation robust speaker recognition,” in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 4257–4260.