ABC System Description for NIST Multimedia Speaker Recognition Evaluation 2019

Jahangir Alam[5], Gilles Boulianne[5], Lukáš Burget[1], Ondřej Glembek[1], Alicia Lozano-Diez[1,6], Pavel Matějka[1,2], Petr Mizera[4], Ladislav Mošner[1], Ondřej Novotný[1], Oldřich Plchot[1], Johan Rohdin[1], Anna Silnova[1], Josef Slavíček[2], Themos Stafylakis[4], Shuai Wang[1,3], Hossein Zeinali[1], Mohamed Dahmane[5], Pierre-Luc St-Charles[5], Marc Lalonde[5], Cédric Noiseux[5], Joao Monteiro[5]

[1]Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Brno, Czechia
{matejkap,iplchot,...}@fit.vutbr.cz
[2]Phonexia, Czechia
slavicek@phonexia.com
[3]Speechlab, Shanghai Jiao Tong University, China
feixiang121976@sjtu.edu.cn
[4]Omilia - Conversational Intelligence, Athens, Greece
tstafylakis@omilia.com
[5]CRIM, Montreal (Quebec), Canada
jahangir.alam@crim.ca
[6]Audias-UAM, Universidad Autonoma de Madrid, Madrid, Spain
alicia.lozano@uam.es

Abstract

In this report, we describe the submission of ABC team to the NIST Multimedia Speaker Recognition Evaluation 2019.

Speaker Recognition Challenge, Deep Neural Networks, ResNet, x-vector, PLDA, Cosine distance

1. BUT

1.1. Experimental Setup

1.1.1. Training data, Augmentations

For ResNet, we used development part of VOXCELEB-2 dataset [1] for training. This set has 5994 speakers spread over 145 thousand sessions (distributed in approx. 1.2 million speech segments). For training DNN based embeddings, we used original speech segments together with their augmentations. The augmentation process was based on the Kaldi recipe[1] and it resulted in additional 5 million segments belonging to the following categories:

- Reverberated using RIRs[2]
- Augmented with Musan[3] noise
- Augmented with Musan music
- Augmented with Musan babel

For TDNN, we tried to add more data to the VoxCeleb-2 development set. We first added the development part of VoxCeleb-1 with around 1152 speakers. The PLP-based systems were trained using this setup (i.e. VoxCeleb 1+2). For other open systems, we also used 2338 speakers from LibriSpeech dataset [2] and 1735 speakers from DeepMine dataset [3]. For all training data, we first discarded utterances with less than 400 frames (measured after applying the VAD). After that, all speakers with less than 8 utterances (including augmentation data) were removed.

Table 1: *x-vector topology proposed in [4]. K in the first layer indicates different feature dimensionalities, T is the number of training segment frames and N in the last row is the number of speakers.*

Layer	Standard DNN	
	Layer context	(Input) $\times$ output
frame1	$[t-2, t-1, t, t+1, t+2]$	$(5 \times K) \times 512$
frame2	$[t]$	512×512
frame3	$[t-2, t, t+2]$	$(3 \times 512) \times 512$
frame4	$[t]$	512×512
frame5	$[t-3, t, t+3]$	$(3 \times 512) \times 512$
frame6	$[t]$	512×512
frame7	$[t-4, t, t+4]$	$(3 \times 512) \times 512$
frame8	$[t]$	512×512
frame9	$[t]$	512×1500
stats pooling	$[0, T]$	1500×3000
segment1	$[0, T]$	3000×512
segment2	$[0, T]$	512×512
softmax	$[0, T]$	$512 \times N$

1.1.2. VAD & Features

Deep Neural Network (DNN) based embeddings used Energy-based VAD from Kaldi SRE16 recipe[4]. We use FBANK features for all BUT systems with this settings: 16kHz, frequency limits 20-7600Hz, 25ms frame length, 40 filter-bank channels and short time mean normalization with a sliding window of 3 seconds.

[1]https://github.com/kaldi-asr/kaldi/tree/master/egs/sre16/v2

[2]http://www.openslr.org/resources/28/rirs_noises.zip

[3]http://www.openslr.org/17/

[4]We did not find a significant impact on performance when using different VAD within the DNN embedding paradigm and it seems that a simple VAD from Kaldi performs very well for DNN embedding in various channel conditions.

Table 2: *ResNet50 architecture, N in the last row is the number of speakers. The first dimension of the input shows number of filter-banks and the second dimension indicates the number of frames.*

Layer name	Structure	Output
Input	–	$40 \times 200 \times 1$
Conv2D-1	3×3, Stride 1	$40 \times 200 \times 32$
ResBlock-1	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 128 \end{bmatrix} \times 3$, Stride 1	$40 \times 200 \times 128$
ResBlock-2	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 4$, Stride 2	$20 \times 100 \times 256$
ResBlock-3	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 6$, Stride 2	$10 \times 50 \times 512$
ResBlock-4	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 3$, Stride 2	$5 \times 25 \times 1024$
StatsPooling	–	10×1024
Flatten	–	10240
Dense1	–	256
Dense2 (Softmax)	–	N

1.2. TDNN x-vectors

The first one is the well-known TDNN based x-vector topology. All its variants were trained with Kaldi toolkit [5] using SRE16 recipe with the following modifications:

- Training networks with 6 epochs (instead of 3). We did not see any considerable difference with more epochs.
- Using modified example generation - we used 200 frames in all training segments instead of randomizing it between 200-400 frames. We have also changed the training examples generation so that it is not random and uses almost all available speech from all training speakers.
- We used a bigger network [6] with more neurons in TDNN layers. Table 1 shows a detailed description of the network.

1.3. ResNet x-vector

ResNet [7] based embeddings are extracted from a standard 50-layer ResNet (ResNet50). This network uses 2-dimensional features as input and processes them using 2-dimensional CNN layers. Inspired by x-vector topology, both mean and standard deviation are used as statistics. The detailed topology of the used ResNet is shown in Table 2. The ResNet was trained using SGD optimizer for 6 epochs.

For this system we used additive angular margin loss (denoted as 'AAM loss') fine-tuning which was proposed for face recognition [8] and introduced to speaker verification in [9]. Instead of training the AAM loss from scratch, we directly fine-tune a well-trained NN supervised by normal Softmax. To be more specific, all the layers after the embedding layer are removed (for both the ResNet and TDNN structure), then the remaining network is be fine-tuned using the AAM loss. For more details about AAM loss, see [8] and [9], s is set to 30 and m is set to 0.2 in all the experiments.

1.3.1. CPU usage

In single threaded setup on Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz, the x-vector extraction time is of 8.0 times faster than real time (FRT) (computed only on detected speech, would be 12.6 FRT computed for whole recordings including silence). Memory consumption is 500 MB for typical utterance. Other computation (PLDA, cosine distance, calibration, fusion) is negligible.

1.4. Backend

1.4.1. General info

For all of backend systems, we followed the same pipeline described here. Particular details and modifications for each system are given in the next sections.

We train the backend on approximately 145k utterances from VoxCeleb 2 (original speech segments corresponding to the same session are concatenated together). For the adaptation, we used 37 utterances of SRE18 VAST development data.

First, training and evaluation data are centered using the training data meanwhile adaptation data are centered with their own mean. Then, we apply feature-distribution adaption (FDA) transformation [10] for the training data. The goal of the transformation is to modify the out-of-domain training data so that their covariance is not lower than the covariance of the in-domain adaptation data in any direction. Here, unlike in the original FDA, we add to the in-domain covariance matrix the variance corresponding to the difference between the training and adaptation mean. After FDA, we apply length normalization, LDA dimensionality reduction followed by another length normalization.

After the preprocessing described above we either train Gaussian PLDA model or use simple cosine scoring to compare the x-vectors. In all cases, we used adaptive symmetric score normalization (adapt S-norm) which computes an average of normalized scores from Z-norm and T-norm [11, 12]. In its adaptive version [12, 13, 14], only part of the cohort is selected to compute mean and variance for normalization. Usually X top scoring or most similar files are selected. As a cohort, we used a subset of the PLDA training data.

All test files were processed by diarization system based on Agglomerative Hierarchical Clustering of x-vectors, which were extracted from input recordings every 0.25 (see [15] for more details). The diarization systems are run to produce output 4 different outputs with 1,2,3 and 4 speakers. Then, an x-vector was extracted for each speaker suggested by the 4 diarization outputs resulting in 10 x-vectors per test file. All 10 test x-vectors were compared with the enrollment x-vector using the backend described in the next sections and maximum score was chosen as the representative score for the given trial.

1.4.2. BUT_ResNet_GPLDA

Here, we set the LDA dimensionality to 200. Gaussian PLDA model is trained with the size of the speaker and channel subspace set to 200 (i.e full-rank). And, we used 100 top scoring files from the cohort (5k x-vectors from the training set) for snorm.

1.4.3. BUT_ResNet_COS

Here, LDA dimensionality is 100. We performed cosine distance scoring on top of 100-dimensional vectors. And, we used 100 top scoring files from the cohort (5k x-vectors from the

training set) for snorm.

1.4.4. BUT_TDNN_GPLDA

For this system, the amount of training data was increased 5 times by including the 4 copies of the data with different augmenattions applied. The LDA dimensionality was 150. Gaussian PLDA model is trained with the size of the speaker and channel subspace set to 150 (i.e full-rank). And, we used 150 top scoring files from the cohort (25k x-vectors from the training set) for snorm.

2. CRIM

At CRIM, we have developed both audio- and video-based systems for NIST Multimedia Speaker Recognition Evaluation 2019 (NIST-MSRE2019). Several speaker verification systems were developed, among them two video-based and three audio-based systems were included in the final ABC submissions.

2.1. Video-based Speaker Verification

We built two video-based speaker verification systems one of them we considered as our baseline system.

2.1.1. Baseline system (CRIM_V_S1PL)

The video baseline system is inspired by the definition of the Face Recognition System in the SRE19 Multimedia Baseline description document (Section 4). First, embeddings are extracted for the enrollment videos using the facial bounding boxes and frame indices provided in the dataset. The corresponding image regions are cropped, normalized, and passed to a Squeeze-Excitation variation of a ResNet-50 [16] pre-trained on VGGFace2 [17] to produce a set of facial embeddings. For each enrollment video, the embeddings are averaged to create a single feature vector that corresponds to a subject. Next, we use the Single-shot Scale-invariant Face Detector (S3FD) of [18] to detect roughly one face per second in the test videos. With those detections' bounding boxes, we extract new facial embeddings using the same approach as before. Finally, in each trial, we compute the cosine similarity between the (averaged) subject embedding and the automatically extracted embeddings. The output score for each video is the maximum similarity found between embedding pairs in that video. No score normalization is performed.

2.1.2. CRIM_V_S2MD

In the literature, different models were proposed to detect a face in images. Among them Dlib [19], SeetaFace [20], FAN [21], and MTCNN [22] process videos in real time with a high accuracy. The multitask CNN (MTCNN) can also locate facial landmarks that could be used for face alignment. Here, we used only the detected bounding boxes (BB) around the faces.

To encode the extract face images we used a SENet50 [16] architecture trained on VGG Face2 [17]. We extracted the learned semantic abstraction of the last pooling layer and applied a global average 2d spatial averaging. The face is then represented by its facial attributes as a vectors of 2048 elements.

Since the videos contain more than one persons, we used Kalman filter to track the extracted BB from frame to frame. An optimization procedure is used to resolve the inter-frame BB association. The cost function is based on a similarity function that considers both the Intersection over Union (IoU) of the bounding boxes and the appearance similarity of the corresponding faces. The cosine similarity is used as a metric of the appearance similarity between each pair of faces.

The tracking leads to several groups of facial attributes corresponding to different tracklets. The track of one person could be possibly represented by a number of tracklets (groups) if the tracking is broken for any reason (e.g., occlusion, leaving the scene, etc.). The Multiple Object Tracking facial features outputs are clustered using the Chinese Whispers algorithm which do not need any prior information about the number of clusters. Since in the same tracklet groups we do not know the number of different appearances the tracked face is exhibiting.

In the verification stage we use the same process as for the tracking. However, the cosine similarity is not performed individually but on the averaged facial attribute per cluster. The maximum will indicate the association target vs. non-target.

2.2. Audio-based Speaker Verification

For NIST-MSRE 2019 audio-based speaker verification task, we developed speaker verification systems following various deep learning architectures on the top of the well known x-vector setting [23] in its Kaldi [5] implementation. Speaker recognition, i.e. multi-class classification over the set of training speakers, has been successfully applied as an auxiliary task for automatic speaker verification (ASV). Outputs of some inner layer of the model trained under that setting can be then used for PLDA training and inference, of for direct scoring using cosine similarity, for instance. To this end, we adopted extended TDNN and factored TDNN (F-TDNN) based x-vector extraction paradigm. As backend, we employed PLDA.

In this case, In order to adapt x-vectors of the out-of-domain data (i.e., PLDA training data) to the in-domain data (i.e., MSRE 2019 or SRE 2018 VAST domain) unsupervised domain adaptation by correlation alignment [24] has been applied. This adaptation technique works by aligning the distributions of out-of-domain and in-domain features in an unsupervised way. This is achieved by aligning second-order statistics, i.e covariances. Training and scoring using PLDA is then performed on the top of CORAL adapted embeddings. Depending on the availability of speaker labels in the in-domain training data we employed either supervised PLDA adaptation (when true speaker labels are present) or unsupervised PLDA adaptation (when there is no speaker labels).

PLDA was employed for scoring dev and test trials after dimensionality reduction of embeddings using linear discriminant analysis (LDA). The dimension of embeddings is reduced to 200. PLDA is trained on embeddings from the train partition with the same augmentation used for training the neural networks. The model adaptation scheme introduced in [25] was further evaluated and utilized for PLDA to help on overcoming any domain shift observed across train and evaluation data due to different recording conditions and language mismatch. To do so, embeddings unlabelled data are then employed for training a second PLDA model. The final back-end is obtained by interpolation of the covariance matrices of the two PLDA models. We found the adaptation described to have different impact in performance depending on the underlying model utilized for generating the embeddings, and in some cases some performance degradation was observed. For supervised PLDA adaptation, we modified the kaldi unsupervised PLDA adaptation code to perform supervised PLDA adaptation using two PLDA models trained on out-of-domain training data and labeled in-domain training data.

2.3. Speech features, VAD and data augmentation

Speech features correspond to either 23 MFCCs (for 8kHz sampling frequency-based systems) or 30 MFCCs (for 16kHz sampling frequency-based systems) obtained with a short-time Fourier transform using a 25ms Hamming window with 10 ms frame shift. For voxceleb, data is down-sampled to 8kHz. An energy-based voice activity detector is employed to filter out non-speech frames. Multi-condition training data is further introduced by augmenting the original train partition with supplementary noisy speech in order to enforce model's robustness across varying conditions. We thus created additional versions of training recordings as similarly done in [23], i.e. by corrupting original samples adding reverberation (reverberation time varies from 0.25s - 0.75s), as well as by adding background noise such as music (signal-to-noise ratio, SNR, within 5-15dB), and babble (SNR varies from 10 to 20dB). Noise signals were selected from the MUSAN corpus [26] and the room impulse responses to simulate reverberation from [27]. We have also developed two systems employing TDNN and F-TDNN on the top of 23-dimensional and 30-dimensional perceptual linear prediction (PLP) features, respectively.

2.3.1. Out-of-domain and in-domain training data

Two out-of-domain training data sets were used for training TDNN and Factored TDNN (F-TDNN) models:

- Data corresponding to SRE's from 04 to 10, Mixer 6, and Switchboard (SWBD) from approximately 5000 speakers.
- Combined voxceleb 1 & 2 (excluding the voxceleb1 test set) corpora, which sums up to approximately 7300 speakers.

As in-domain training data we used the following two sets:

- SRE 2018 VAST portion of development (enroll + test) data and considered it as unlabeled in-domain training data.
- SRE 2018 VAST portion of development (enroll + test, 37 recording from 10 speakers) data + 40 speakers' recordings from OpenSAT (VAST) data. In total, there are approximately 200 recordings from 50 speakers and we considered it as labeled in-domain training data.

2.3.2. Systems included in the final submission

In the section we provide a brief description of audio-based speaker verification systems included in the final ABC submission:

- CRIM_S1: This system is 8kHz sampling frequency-based ASV system where the extended TDNN architectures is trained on multi-style version of SRE's from 04 to 10, Mixer 6, and Switchboard (SWBD) data. Frontend features employed is 23-dimensional PLP features. Embeddings are adapted to in-domain by using only unsupervised adaptation employing correlation alignment.
- CRIM_S5_ADAPT: This system is 16kHz sampling frequency-based ASV system where a factored TDNN architectures is trained on multi-style version of combined voxceleb 1 & 2 (excluding the voxceleb1 test set) data. Frontend features employed is 30-dimensional MFCC features. Embeddings are adapted to in-domain by using unsupervised adaptation employing correlation alignment as well as supervised PLDA adaptation techniques.
- CRIM_S8_ADAPT: This system is similar to CRIM_S5_ADAPT but as frontend features employs 30-dimensional PLP features instead of 30-dimensional MFCCs.

2.4. CPU usage

In single threaded setup on Intel(R) Core(TM) i9-7980XE CPU @ 2.60GHz, the x-vector extraction time is of 6.0 times faster than real time (FRT). Memory consumption is 500 MB for typical utterance. Other computation (PLDA, cosine distance, calibration, fusion) is negligible.

3. ABC submission

3.1. Calibration and Fusion

The final submission strategy was one common fusion trained on the labeled development set. Each system provided log-likelihood ratio scores that could be subjected to score normalization. These scores were first pre-calibrated and then passed into the fusion. The output of the fusion was then again re-calibrated.

Both calibration and fusion was trained with logistic regression optimizing the cross-entropy between the hypothesized and true labels on a corresponding development set. Our objective was to improve the error rates on the NIST SRE 2019 VAST development set.

The results for all tracks (audio only, video only, audio-visual) are listed in 3.

3.2. Audio only

We used the labeled NIST SRE2018 VAST evaluation set to train the calibration and fusion as we were using NIST SRE 2018 VAST development data in some cases to do the system adaptation and we wanted to avoid the overlap which exists between NIST 2018 VAST development set and NIST 2019 VAST development set. We have not split the datasets in any way and we took the risk of having an overlap which exists between NIST SRE2018 VAST evaluation set and SRE19 VAST development set.

3.3. Video and Audio-Visual systems

In this case we had to work only with audio-visual NIST SRE2019 VAST development set which was used to train both calibration and fusion.

4. Acknowledgements

The work was supported by Czech Ministry of Interior project No. VI20152020025 "DRAPAK", Google Faculty Research Award program, Czech Science Foundation under project No. GJ17-23870Y, Czech National Science Foundation (GACR) project NEUREM3 No. 19-26934X, and by Czech Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602".

5. References

[1] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman, "Voxceleb2: Deep speaker recognition," in *Interspeech 2018, 19th*

Table 3: *Results of single systems and submnitted fusions, * denotes single best system.*

#	System	SRE18 VAST eval			SRE19 AV DEV		
		minDCF	actDCF	EER (%)	minDCF	actDCF	EER (%)
	Audio systems						
1	BUT-ResNet_GPLDA	0.328	0.332	8.37	0.228	0.247	4.14
2	BUT-ResNet_COS	0.298	0.409	7.85	0.211	0.233	5.08
3	* BUT-TDNN_GPLDA	0.314	0.329	8.32	0.190	0.204	4.25
4	CRIM_S1	0.397	0.399	0.86	0.487	0.489	11.20
5	CRIM_S5_ADAPT	0.217	0.240	5.98	0.176	0.186	4.80
6	CRIM_S8_ADAPT	0.220	0.226	5.21	0.180	0.205	4.05
	FUSION (PRIMARY) 1+2+3+4	0.277	0.291	6.92	0.149	0.155	3.91
	FUSION (CONTRASTIVE) 2+3+5+6	0.207	0.209	3.39	0.130	0.130	2.66
	Video systems						
7	CRIM_V_S1PL	-	-	-	0.765	0.794	9.17
8	* CRIM_V_S2MD	-	-	-	0.462	0.491	7.24
	FUSION (PRIMARY) 7+8	-	-	-	0.441	0.450	6.66
	Audio-visual systems						
	* FUSION 3+8	-	-	-	0.095	0.102	1.71
	FUSION (PRIMARY) 2+3+7+8	-	-	-	0.070	0.077	1.54
	FUSION (CONTRASTIVE) 2+3+5+6+8	-	-	-	0.048	0.048	0.82

Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018., 2018, pp. 1086–1090.

[2] 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)*, 2015.

[3] Hossein Zeinali, Hossein Sameti, and Themos Stafylakis, "Deepmine speech processing database: Text-dependent and independent speaker verification and speech recognition in persian and english," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 386–392.

[4] David Snyder, Daniel Garcia-Romero, Gregory Sell, Alan McCree, Daniel Povey, and Sanjeev Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP*, 2019.

[5] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.

[6] Jesús Villalba, Nanxin Chen, David Snyder, Daniel Garcia-Romero, Alan McCree, Gregory Sell, Jonas Borgstrom, Fred Richardson, Suwon Shon, François Grondin, et al., "The JHU-MIT system description for NIST SRE18," 2018.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos 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.

[9] Xu Xiang, Shuai Wang, Houjun Huang, Yanmin Qian, and Kai Yu, "Margin matters: Towards more discriminative deep neural network embeddings for speaker recognition," *arXiv preprint arXiv:1906.07317*, 2019.

[10] Pierre-Michel Bousquet and Mickael Rouvier, "On Robustness of Unsupervised Domain Adaptation for Speaker Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2958–2962.

[11] P. Kenny, "Bayesian speaker verification with heavy–tailed priors," keynote presentation, Proc. of Odyssey 2010, June 2010.

[12] Pavel Matějka, Ondřej Novotný, Oldřich Plchot, Lukáš Burget, Mireia Sánchez Diez, and Jan Černocký, "Analysis of score normalization in multilingual speaker recognition," in *Proceedings of Interspeech 2017*. 2017, pp. 1567–1571, International Speech Communication Association.

[13] D. E. Sturim and Douglas A. Reynolds, "Speaker adaptive cohort selection for tnorm in text-independent speaker verification," in *ICASSP*, 2005, pp. 741–744.

[14] Yaniv Zigel and Moshe Wasserblat, "How to deal with multiple-targets in speaker identification systems?," in *Proceedings of the Speaker and Language Recognition Workshop (IEEE-Odyssey 2006)*, San Juan, Puerto Rico, June 2006.

[15] Federico Landini, Shuai Wang, Mireia Diez, Lukáš Burget, Pavel Matějka, Kateřina Žmolíková, Ladislav Mošner, Oldřich Plchot, Ondřej Novotný, Hossein Zeinali, and Johan Rohdin, "BUT System Description for DIHARD Speech Diarization Challenge 2019," *arXiv preprint arXiv:1910.08847*, 2019.

[16] Jie Hu, Li Shen, and Gang Sun, "Squeeze-and-excitation networks," 2018.

[17] Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi, and Andrew Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *13th IEEE International Conference on Automatic Face & Gesture Recognition, FG 2018, Xi'an, China, May 15-19, 2018*, 2018, pp. 67–74.

[18] Shifeng Zhang, Xiangyu Zhu, Zhen Lei, Hailin Shi, Xiaobo Wang, and Stan Z. Li, "S^3fd: Single shot scale-invariant face detector," *CoRR*, vol. abs/1708.05237, 2017.

[19] Pedro F. Felzenszwalb, Ross B. Girshick, and David A. McAllester, "Cascade object detection with deformable part models," in *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, 2010, pp. 2241–2248.

[20] Jie Zhang, Shiguang Shan, Meina Kan, and Xilin Chen, "Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment," in *Computer Vision – ECCV 2014*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, Eds., Cham, 2014, pp. 1–16, Springer International Publishing.

[21] Adrian Bulat and Georgios Tzimiropoulos, "How far are we from solving the 2d & 3d face alignment problem? (and a dataset of

230, 000 3d facial landmarks)," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 1021–1030.

[22] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao, "Joint face detection and alignment using multi-task cascaded convolutional networks.," *CoRR*, vol. abs/1604.02878, 2016.

[23] 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.

[24] Jahangir Alam, Gautam Bhattacharya, and Patrick Kenny, "Speaker verification in mismatched conditions with frustratingly easy domain adaptation," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 176–180.

[25] Daniel Garcia-Romero, Alan McCree, Stephen Shum, Niko Brummer, and Carlos Vaquero, "Unsupervised domain adaptation for i-vector speaker recognition," in *Proceedings of Odyssey: The Speaker and Language Recognition Workshop*, 2014.

[26] David Snyder, Guoguo Chen, and Daniel Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015, arXiv:1510.08484v1.

[27] "Open Speech and Language Resources," 2017, http://www.openslr.org/28/.

6. Recipes at BUT

6.1. ResNet - Shuai Wang

6.2. TDNN - Hossein Zeinali/Pavel Matejka

- system from Voxceleb challenge trained by Hossein
  - PavelM did the forwardpass of the data for NIST

6.3. Diarization - Lukas Burget

/mnt/matylda4/burget/SRE19/DIARIZATION/diarization_AHCxvec2Nspk.sh
Needs *.py files from the same directory and x-vectors extracted every 0.25s (by Shuai) from
/mnt/matylda3/xwangs01/projects/dirazation/DIHARD2019/v4/exp/xvector_nnet_paja/xvectors_${dataset}_25ms/xvector.*.ark

6.4. Backend - Anna Silnova

3 backend recipes are here: `/mnt/matylda5/isilnova/NIST_SRE_2019/VAST/recipes`

All of the needed .m files are in the same directory. The path to x-vectors is specified in the first line of the corresponding script, the directory where to save the models, scores, logs, etc. is defined in the 3rd line.

BUT_ResNet_GPLDA `run_plda_resnet.sh`

BUT_ResNet_COS `run_plda_cos_resnet.sh`

BUT_TDNN_GPLDA `run_plda_tdnn.sh`

6.5. Calibration/Fusion - Oldrich Plchot

6.6. Score format conversion - Pavel Matejka

7. Ideas for Analysis

- Pavel: use of diarization - no diar , diar, 1-4spk, 1-7spk ...
- who:what

8. Retrospective

8.1. What was good and we want to do it next time too

- who:what

8.2. What we can do better next time

- who:what