ABC NIST SRE 2018 SYSTEM DESCRIPTION

Conference Paper · February 2019

CITATION
1

READS
60

21 authors, including:

Md Jahangir Alam
Centre de recherche informatique de Montréal
72 PUBLICATIONS **922** CITATIONS
SEE PROFILE

Niko Brummer
AGNITIO
46 PUBLICATIONS **1,848** CITATIONS
SEE PROFILE

Lukas Burget
Brno University of Technology
163 PUBLICATIONS **6,346** CITATIONS
SEE PROFILE

ABC NIST SRE 2018 SYSTEM DESCRIPTION

Jahangir Alam[5], Niko Brummer[3], Lukáš Burget[1], Mireia Diez[1], Ondřej Glembek[1], Patrick Kenny[5], Michal Klčo[2], Federico Landini[1], Alicia Lozano-Diez[6], Pavel Matějka[1], Gautam Bhattacharya[5], Joao Monteiro[5], Ladislav Mošner[1], Ondřej Novotný[1], Oldřich Plchot[1], Ján Profant[2], Johan Rohdin[1], Anna Silnova[1], Josef Slavíček[2], Themos Stafylakis[4], Hossein Zeinali[1]

[1]Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Brno, Czechia
{matejkap,iplchot,burget,...}@fit.vutbr.cz
[2]Phonexia, Czechia
{profant,slavicek,klco}@phonexia.com
[3]Nuance Communications Inc.
niko.brummer@gmail.com
[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

Index Terms— automatic speaker identification, deep neural networks, bottleneck features, PLDA, snorm

Authors are in alphabetical order.

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, and by Czech Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602". It was also supported by Technology Agency of the Czech Republic project No. TJ01000208 "NOSICI", European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 748097, the Marie Sklodowska-Curie cofinanced by the South Moravian Region under grant agreement No. 665860, and by the U.S. DARPA LORELEI contract No. HR0011-15-C-0115. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

1. INTRODUCTION

This submission is a collaborative/competitive effort of BUT, Phonexia, Omilia, CRIM and UAM. All systems in fusions are based on x-vector paradigm with different features, DNN topologies and backends. No i-vector based system made it to the fusion.

2. TELEPHONE SYSTEMS - CMN2

2.1. Tensorflow x-vector with attention = TFXvecAtt

This system is based on our Tensorflow implementation of the x-vector speaker embedding. The overall topology is the same as the original Kaldi recipe with some modifications which are explained below.

2.1.1. Training data, Augmentation

For training the networks, we used the following:

- SRE 4, 5, 6, 8, 12
- Telephony part of Mixer6

- Fisher English
- All switchboard data
- Voxceleb 1 and 2

For performed the following data augmentations which are the same as in Kaldi recipe except for the compression:

- Reverberated
- Augment with Musan noise
- Augment with Musan music
- Augment with Musan speech
- Compression using ogg and mp3 codecs

After creating a list of utterances for augmentation, a subset of 500K utterances from this list was selected and added to the training data. Afterwards, utterances with less than 500 frames and also speakers with less than 5 training utterance were removed. Finally, the training data for creating training archives contained 17054 speakers.

2.1.2. VAD, Input features

We used MFCC features and energy-based VAD from Kaldi SRE16 recipe without any modification. The features are therefore 23-dimensional MFCCs which are extracted from 25 ms windows with 15 ms overlap. The bandwidth is limited between 20 and 3700 Hz.

2.1.3. Training archives

For creating the training archives, we used Kaldi-like archive generation for our Tensorflow implementation and therefore, for both Kaldi and Tensorflow, the same configuration was used for generating two different sets of archives. Minimum and maximum number of frames in each training example are 200 and 400 respectively. Number of repeats for each speaker is 25 and maximum number of frames per archive is 2 billion. By using this configuration, two sets of 124 archives were generated for Kaldi and Tensorflow versions.

2.1.4. Neural net architecture, training

The overall architecture of the network for x-vector extraction is the same as in Kaldi and is shown in Table 1. The main change is that we use CNN instead of TDNN for second and third frame level layers. Also the nonlinearity here is LReLU instead of ReLU. Based on this topology we designed the self-attention mechanism similar to [1, 2, 3].

Table 1. NN architecture for Tensorflow implementation.

Layer	**Layer context**	**(Input) $\times$ output**
frame1	$[t-2, t+2]$	$(5 \times 23) \times 512$
frame2	$[t-2, t+2]$	$(5 \times 512) \times 512$
frame3	$[t-3, t+3]$	$(7 \times 512) \times 512$
frame4	$[t]$	512×512
frame5	$[t]$	512×1536
stats pooling	$[0, T)$	1536×3072
segment6	0	3072×512
segment7	0	512×512
softmax	0	$512 \times$ N

2.1.5. Backend

We trained HT-PLDA backend [4] on data from Mixer collection (NIST SRE 2004-2010). Training set was expanded by including data augmented with noise, reverberation and music. In total, training set consisted of approximately 280k utterances coming from 5k speakers. Additionally, we applied supervised domain adaptation of model parameters. For this purpose, separate "adaptation" HT-PLDA model was trained on "unlabelled" portion of the development data, where we used phone numbers as the speaker labels. The final adapted HT-PLDA model was derived from the two HT-PLDA models so that the modeled across-speaker covarinace matrix is a weighted combination of the covariance matrices from the constituent models. Similarly the parameters describing within-speaker covariance matrix are also interpolated.

LDA, reducing dimensionality of vectors to 300, centering and length normalization were applied to all x-vectors. Training x-vectors were centered using their own mean and the test x-vectors were centered around the mean computed from unlabelled development data. Size of the speaker subspace was set to 100 for the main model and to 50 for adaptation model.

2.2. Tensorflow x-vector without attention = TFXvec

This system is also a Tensorflow implementation without attention and its architecture is exactly one that is shown in Table 1. We can say that for some of our backends, the network without attention performs better.

2.2.1. Backend

A Gaussian PLDA backend was trained on the augmented MIXER dataset described in the previous section. Similarly to the HT-PLDA backend above (Section 2.1.5), a Gaussian PLDA adaptation model was also trained, us-

ing SRE18 unlabeled data and using the telephone labels. Both main and adaptation PLDA models were given equal weights when estimating the final interpolated model. In this case, the training and test x-vectors were not centered around different means as was the case for the HT-PLDA. Instead, the means from the two PLDA models were also interpolated and the variance corresponding to the difference between the means was added to the within-class covarinace of the final adapted PLDA model.

Length normalization, LDA dimensonality reduction to 160 dimensions followed by another length normalization was applied to xvectors. All data were centered using the training data mean. Speaker subspace size was set to 160 (i.e full rank) for the main model and to 50 for the adaptation model.

We applied SNorm using top scoring 200 files selected from SRE18 unlabeled data.

2.3. KaldiBigXvec

The system was trained in Kaldi toolkit [5] using sre16 recipe with modifications described below.

2.3.1. Training data, Augmentation

Datasets were used from section 2.1.1.

2.3.2. VAD, Input features

We used MFCC features and energy based VAD from Kaldi sre16 recipe.

2.3.3. Neural net architecture, training

We made two modifications of the original Kaldi NN training: We ran the training for 6 epochs (instead of 3) and we slightly altered the NN topology – see Tab. 2.

2.3.4. x-vector extraction

We extracted x-vectors from *segment6* layer before nonlinearity. 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 SRE18eval utterance.

2.3.5. Backend

The backend used was the same as described in Section 2.2.

Table 2. NN architecture for x-vector extraction – KaldiBigXvec. Bold values are our modifications of the original [6] architecture.

Layer	Layer context	Input $\times$ output
frame1	$[t-2, t+2]$	$115 \times$ **256**
frame2	$\{t-2, t, t+2\}$	**768** $\times$ **256**
frame3	$\{t-3, t, t+3\}$	**768** $\times$ **256**
frame4	$\{\mathbf{t-3}, \mathbf{t}, \mathbf{t+3}\}$	**768** $\times$ 512
frame5	$\{\mathbf{t-4}, \mathbf{t}, \mathbf{t+4}\}$	**1536** $\times$ 1500
stats pooling	$[0, T)$	1500 $\times$ 3000
segment6	0	3000 $\times$ 512
segment7	0	512 $\times$ 512
softmax	0	512 $\times$ N

2.4. CRIM_ADAPT_XV_VOX2

2.4.1. Data preparation

At CRIM, to train our speaker verification systems we created following two background data sets:

- Background set 1:This set comprised of recordings from NIST SREs 2004-2010 and Switchboard. There are around 90k recordings from 5.2k speakers. Multi-style data was created by adding a noisy augmented data with the clean data. The augmented data was generated by adding music, babble, reverberation and additive noises to the clean data.
- Background set 2: This set was created by adding a selection of 100k recordings from voxceleb (1 & 2) to the background data 1. This set contains around 190k recordings from 11.5k speakers. Multi-style data was created by adding a noisy augmented data with the clean data. The augmented data was generated by adding music, babble, reverberation and additive noises to the clean data.

We construct an extended unlabeled data by considering the following three sources for better representation of the target domain:

- sre18-unlabeled: 2332 target domain unlabeled recordings (CMN2) provided by NIST.
- vox-unlabeled: 5000 recordings randomly selected from voxceleb data and considered as unlabeled.
- sre-ara-unlabeled: 700 Arabic recordings collected from the previous NIST SREs data (SREs 04-08) and treated as unlabeled.

We used this extended unlabeled data (denoted as sre18-vox-ara-unlabeled) for centering, whitening and

for adaptation of source to the target domain by using unsupervised domain adaptation techniques such as correlation alignment and PLDA adaptation. None of our submitted systems to SRE2018 utilized the CMN2 or VAST labels during systems' development.

2.4.2. Frontends and VAD

We extracted MFCC and PLP features from all the recordings. All features are of 23-dimensional. Features are normalized using short-time mean normalization technique with a window of 300-frames. We used unsupervised GMM- and energy-based voice activity detection for removing non-speech frames [7].

2.4.3. Speaker Embeddings

In this evaluation we made use of i-vector and x-vectors speaker embeddings for speaker recognition task. None of the systems based on i-vector representations were made it to the final fusion. All of our submitted systems use x-vector speaker embeddings [6].

2.4.4. Post-processing

As a post-processing step, we adapt source data (background data) to target domain by applying correlation alignment based domain adaptation technique on speaker embeddings space. This is an asymmetric adaptation method that works by aligning the distributions of source and target features in an unsupervised fashion. This is achieved by aligning second order statistics [8].

2.4.5. Backend

For scoring, we made use of a probabilistic linear discriminant analysis (PLDA) in combination with a linear discriminant analysis (LDA) dimensionality reduction technique. Both PLDA and LDA parameters were learned using sre_combined (multi-style data generated on the top of NIST SREs 2004-2010) data. Unsupervised PLDA adaptation was performed on the unlabeled data (sre18-vox-ara-unlab) for domain adaptation in model domain.

2.4.6. Processing Time

In order to report real time factor we conducted experiments on an Intel(R) Xeon(R) CPU X5650 @ 2.67GHz with a total memory of 94.5GB. The execution time for the extraction of x-vectors (VAD segmentation + Features extraction + extraction of Sufficient statistics + generation of x-vectors + enrollment + scoring in a single thread is of 2 times faster than the real time.

For system CRIM_ADAPT_XV_VOX2, we used background set 2 (as mentioned in 2.4.1) for training the x-vector speaker embeddings network using 23-dimensional MFCC features. This system utilizes both feature-domain and model-domain unsupervised adaptation methods as described in section 2.4.4. For scoring, similar backend was used as mentioned in 2.4.5.

2.5. CRIM_ADAPT_V2_2NDR_LDA150

For system CRIM_ADAPT_V2_2NDR_LDA150, we used background set 1 (as mentioned in 2.4.1) for training the x-vector speaker embeddings network using 23-dimensional MFCC features. Once training was completed we re-trained the network using pre-trained model as initialization. This helped us to improve performance by 0.5% - 1.0% absolute. This system employs both feature-domain and model-domain unsupervised adaptation methods as described in section 2.4.4. x-vectors dimension were reduced to 150 by applying LDA. For scoring, similar backend was used as mentioned in 2.4.5.

2.6. CRIM_ADAPT_V2_PLP

For system CRIM_ADAPT_V2_PLP, we used background set 1 (as mentioned in 2.4.1) for training the x-vector speaker embeddings network using 23-dimensional PLP features. This system employs both feature-domain and model-domain unsupervised adaptation methods as described in section 2.4.4. x-vectors dimension were reduced to 145 by applying LDA. For scoring, similar backend was used as mentioned in 2.4.5.

2.7. KaldiXvecWGAN - x-vectors with adversarial adaptation

In this system we applied adversarial adaptation [9, 10] to compensate for the domain mismatch between the training and the test data. The baseline was an x-vector system trained with the standard Kaldi recipe [6] using the same data as the TFXvec system described in Section 2.1, except that Fisher and voxceleb 1 were not included. The number speakers in the training set were 12170. The adversarial adaptation was applied to the trained baseline model. The main differences compared to the adversarial adaptation scheme applied in [10] were:

- We applied adversarial adaptation to all parameters of an x-vector extractor NN network whereas [10] applied it to a transformation of i-vectors.
- We included language information as side information the network (all layers before the x-vector), making the NN aware of whether it is processing data from the source or target domain. This should

in principle help it to better know how to transform the data.

- We used Wasserstein loss [11, 12] instead of cross-entropy for the discriminator (a.k.a. critic).
- We applied supervised adaptation using the telephone number as "speaker IDs". (Unsupervised adaptation did not work well.)

Results on the development set are provided in Table 3. The adversarial adaptation turned out to be effective and complementary to adaptation of the HTPLDA backend. However, the system was not good enough compared to the systems presented in Section 2.1 to 2.3 (with more training data, attention, score normalization, etc.) and was therefore not included in the final fusion. A more detailed description of these experiments are under preparation for a conference submission.

Table 3. Results for adversarial adaptation. "HP adp" refers to adaptation of the HTPLDA backend and "lan" refers to adding language information to the x-vector NN. Here, we used our standard scoring tool instead of the tool from NIST, and thus all trials are equally weighted. We did not apply score normalization.

	mDCF1	mDCF2	EER(%)
Baseline	0.62	0.68	9.04
Baseline + HP adp.	0.56	0.62	7.35
WGAN	0.55	0.60	8.31
WGAN + lan	0.53	**0.59**	8.05
WGAN + HP adp.	0.54	0.61	7.44
WGAN + lan + HP adp.	**0.52**	**0.59**	**7.18**

3. SYSTEMS FOR AUDIO FROM VIDEO - VAST

3.1. Diarization

The used speaker diarization method is based on the Bayesian Hidden Markov Model described in [13], in which states represent speaker specific distributions and transitions between states represent speaker turns. The transitions probabilities are set to favor staying in the same speakers to avoid too frequent speaker turns. As in the ivector or JFA models, speaker distributions are modeled by GMMs with parameters constrained by eigenvoice priors to facilitate discrimination between speakers.

We used 19 MFCC+Energy coefficients (without any normalization) as features for diarization. We only ran the diarization on segments that contain speech according to our VAD. We used 1024-component, diagonal covariance GMM-UBM, and 400 dimensional i-vectors. The UBM and the total variability matrix matrix were trained on the VoxCeleb datasets [14, 15]. A hierarchical agglomerative clustering (AHC) algorithm based on PLDA scores between i-vectors estimated on 200 ms segments was performed to initialize the assignment of frames to speakers for the VB algorithm.

The processing time of the VB diarization algorithm for 10 minute files (considering AHC from the PLDA scores and VB diarization from pre-extracted features), ranged from 17s to 50s (real time). The processing time difference is because the VB algorithm converging time is dependent on the number of speaker models it is initialized with. On average it is 20 times faster than RT.

Table 4. Configuration of TDNN for x-vector extraction – KaldiXvecVox16k. Bold values are our modifications of the original [6] architecture. X-vectors are extracted at layer *segment6* before the nonlinearity.

Layer	Layer context	Input × output
frame1	$[t-2, t+2]$	115 × 512
frame2	$\{\mathbf{t-4}, t-2, t, t+2, \mathbf{t+4}\}$	1536 × 512
frame3	$\{\mathbf{t-6}, t-3, t, t+3, \mathbf{t+6}\}$	1536 × 512
frame4	$\{t\}$	512 × 512
frame5	$\{t\}$	512 × 1500
stats pooling	$[0, T)$	1500 × 3000
segment6	0	3000 × 512
segment7	0	512 × 512
softmax	0	512 × N

3.2. KaldiXvecVox16k and KaldiXvecVox16k_ADAPT

3.2.1. x-vector extractor

The system was trained in Kaldi toolkit [5] using sre16 recipe. Standard MFCCs were used together with our VAD, which is based on phone recognizer trained on Fisher with 3 variants of Fisher with added noise at different SNR. We dropped all frames that were marked as silence or noise.

X-vector extractor was trained using VoxCeleb1 and VoxCeleb2 development data [14, 15] with 16k Hz sampling rate - all recordings from single session were concatenated into single audio file with one second of silence between each segment. We used 512000 augmentations compared to 128000 in original recipe and ran training for 9 epochs instead of 3. The configuration of the Time Delayed Neural Network (TDNN) used for the x-vector extraction is summarized in Table 4.

Diarization was used for all test files. An x-vector was extracted for each speaker suggested by the diarization algorithm. All test x-vectors were compared with the enrollment x-vector using the PLDA backend described in the next section and maximum score was chosen as the representative score for the given trial.

Processing time for x-vector enrollment is similar to times specified in 2.3.4. For test file it is 5.7 FRT on average because of the diarization.

3.2.2. PLDA backend

A Gaussian PLDA backend was trained on the augmented VoxCeleb1 and VoxCeleb2 development data described in the previous section (around 600k training x-vectors). Because of the lack of adaptation data similar to the VAST development and evaluation data not supervised or unsupervised adaptation was performed for the system denoted as *KaldiXvecVox16k*. For system denoted as *KaldiXvecVox16k_ADAPT*, we attempted to adapt the PLDA model to the very limited amount of VAST development data. Here, we considered the difference between the mean of the training x-vectors and the mean of the VAST development x-vectors and we added the corresponding variance to the within-class covarinace of the PLDA model.

LDA dimensionality reduction to 160 dimensions followed by length normalization was applied to the x-vectors. PLDA with full rank across- and within-speaker covariance matrices was used. We applied Adaptive SNorm where a subset of the PLDA training served as the cohort.

3.3. CRIM_NOADAPT_V2_PLP

This system is similar to the one described in 2.6. The only difference is that in this system we did not apply unsupervised PLDA adaptation.

3.4. CRIM_ADAPT_2NDR_CW_VOX_UNL_V2_NAP

This system is similar to the one described in 2.5 but as an additional post-processing step we applied NAP (before domain adaptation) which tries to make the subsets more similar by projecting away the subspace in which their means differ. The classes for NAP (nuisance attribute projection) were the following subsets: sre18-unlabeled, vox-unlabeled, sre-ara-unlabeled-male, and sre-ara-unlabeled-female, sre-eng (English recordings from NIST SREs 2004-2010). After NAP projection we applied correlation alignment-based unsupervised domain adaptation. The dimension of x-vectors were reduced further to 150 using LDA. For scoring, we made use of same backend as mentioned in 2.4.5 with unsupervised PLDA adaptation.

3.5. CRIM_NOADAPT_V2_2NDR_LDA200

This system is similar to the one described in 2.5. The only difference is that in this system no unsupervised PLDA adaptation was applied. The dimension of x-vectors was reduced to 200 using LDA.

4. OPEN CONDITION

Despite our considerable efforts to prepare a system for the open condition we did not meet the deadline. However, it is worth mentioning them for completeness.

4.1. Additional training data

For open condition, in addition to the training data mentioned in 2.1.1, the following two databases were used:

- Fisher Arabic
- DeepMine speech processing database [16]

After data augmentation and filtering of utterances and speakers like in closed condition, 2139 speakers from Fisher Arabic and 1717 speakers from DeepMine were added to the training data.

4.2. VAD, Features and training archives

Here we used exactly the same VAD and features as in closed condition. For details please see section 2.1.2. For creating training archives, we again followed the same configuration as in closed condition, which resulted in 140 training archives for both Kaldi and Tensorflow.

4.3. Neural net architecture

For open condition we trained three networks: two using Tensorflow and one with Kaldi. The Tensorflow networks are exactly the same as network in Table 1 and its attention based equivalent. However, the Kaldi network slightly differs from network in Table 2 and its architecture is shown in Table 5.

5. CALIBRATION & FUSION

The final submission strategy was one common fusion trained on the labeled development data. 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. We treated

Table 5. NN architecture for x-vector extraction using Kaldi for open condition. Bold values are our modifications of the original [6] architecture.

Layer	Layer context	Input × output
frame1	$[t-2, t+2]$	115×512
frame2	$\{t-2, t, t+2\}$	1536×512
frame3	$\{t-3, t, t+3\}$	1536×512
frame4	$\{\mathbf{t-3}, \mathbf{t}, \mathbf{t+3}\}$	$\mathbf{1536} \times 512$
frame5	$\{\mathbf{t-4}, \mathbf{t}, \mathbf{t+4}\}$	$\mathbf{1536} \times \mathbf{1536}$
stats pooling	$[0, T)$	$\mathbf{1536} \times \mathbf{3072}$
segment6	0	3072×512
segment7	0	512×512
softmax	0	$512 \times$ N

the telephone data (CMN2) and audio from video (VAST) separately and we were also calibrating and fusing separately on the CMN2 or the VAST labeled development set.

Calibration was trained with logistic regression optimizing the cross-entropy between the hypothesized and true labels on a corresponding development set. We chose a generative approach to our fusion which we based on a Multiclass, multivariate, fully Bayesian, generative Gaussian pattern recognizer (MMFBG[1]). Our objective was to improve the error rate on the development set itself, but we were also monitoring error-rate trends on SRE'16 conditions for telephone systems and on Speakers In The Wild and Voxceleb1 for VAST systems.

6. FINAL ABC FUSION & SUBMISSION

We have submitted only to the closed condition. We did try to investigate into the open condition by including other non-English data, but our resources in this directions were limited and our efforts did not bring any improvements over our closed condition systems at the time of submission deadline. We briefly describe our efforts for open condition in 4.

As already stated above, we were treating different domain (VAST and CMN2) separately and we have just pooled the scores from the two systems to form our submissions. We did not produce an universal system that would be tuned for the two domains simultaneously.

6.1. Primary system

For telephone data (CMN2), our primary system is a fusion of three x-vector systems described in sections 2.1, 2.2 and 2.3.

For audio from video (VAST), our primary system is a fusion of three systems: two variants of wide-band x-vector system described in 3.2 and a narrow-band x-vector system described in 3.3.

It is worth noting that all systems utilize x-vector approach to extract embeddings and are based on simple spectral features (MFCCs and PLPs). We have of course produced also generative i-vector baselines, but we have seen substantially worse performance on our development sets. Proper analysis and comparison for SRE18 between different approaches to extract embeddings will be performed once the keys are released.

Processing time for CMN2 is 3 times x-vector based system. Real time factor is 1.3 FRT to compute one trial. But it is faster for more files, because xvectors are reused and not computed separately for each trial. For VAST it is slower because of diarization - it is 1.1FRT in total.

6.2. Contrastive 1

Our contrastive system was the best fusion of the CRIM's systems. For telephone data, it is a fusion of three systems described in sections 2.4, 2.5 and 2.6. For VAST, it is a fusion ot two systems described in sections 3.4 and 3.5.

6.3. Contrastive 2 - single best system

Single best system for telephone data is described in section 2.2. Single best system for VAST is an adapted variant of the wide-band system described in section 3.2. Both systems were included in the primary fusion.

Real time factor is 4 times FRT to compute one trial (2 xvectors each 8FTR). But it is faster for more files, because xvectors are reused and not computed separately for each trial. For VAST it is slower because of diarization (1 enrollment xvector 8FRT + 1 test xvector 5.7FRT)- it is 3.3FRT in total.

6.4. Results on NIST 2018 development set

Table 6. Results for Primary system

	EER(%)	min_C	act_C
CMN2	5.47	0.339	0.348
VAST	7.41	0.152	0.337
Both	–	–	0.343

Results of individual systems entering the primary submission computed on all trials without any weighting.

[1] https://arxiv.org/abs/1307.6143

Table 7. Results for Constrastive system

	EER(%)	min_C	act_C
CMN2	7.02	0.503	0.510
VAST	5.76	0.502	0.716
Both	–	–	0.613

Table 8. Results for Single Best system

	EER(%)	min_C	act_C
CMN2	6.27	0.359	0.367
VAST	7.41	0.333	0.453
Both	–	–	0.410

Note that performance on SRE16 Cantonese is suboptimal as these systems used SRE18 unlabeled dev set for adaptation. We use SRE16 to monitor the robustness of systems on another completely unseen domain.

Table 9. CMN2 - Note that performance on SRE16 Cantonese is suboptimal as these systems used SRE18 unlabeled dev set for adaptation.

system	condition	vastMinDCF	vastActDCF	sre18MinDCF	sre18ActDCF	EER
TFXvecAtt	sre18_dev_cmn2	0.323	0.325	0.510	0.515	0.058
	sre16_evl_yue	0.403	0.972	0.630	1.941	0.064
TFXvec & (SINGLE & BEST)	sre18_dev_cmn2	0.288	0.292	0.428	0.434	0.066
	sre16_evl_yue	0.242	0.243	0.391	0.395	0.048
KaldiBigXvec	sre18_dev_cmn2	0.309	0.311	0.460	0.465	0.069
	sre16_evl_yue	0.255	0.255	0.401	0.403	0.055
PRIMARY & FUSION	sre18_dev_cmn2	0.267	0.268	0.399	0.402	0.058
	sre16_evl_yue	0.231	0.239	0.363	0.379	0.049

Table 10. VAST

system	condition	vastMinDCF	vastActDCF	sre18MinDCF	sre18ActDCF	EER
KaldiXvecVox16k_ADAPT & (SINGLE & BEST)	sitw_core-core_eval	0.145	0.226	0.246	0.426	0.022
	sre18_dev_vast	0.333	0.453	0.333	0.463	0.059
KaldiXvecVox16k	sitw_core-core_eval	0.144	0.453	0.250	0.679	0.023
	sre18_dev_vast	0.416	0.416	0.444	0.500	0.063
CRIM_NOADAPT_V2_PLP	sitw_core-core_eval	0.395	0.701	0.582	0.942	0.069
	sre18$_dev_vast$	0.630	0.831	0.630	0.815	0.058
PRIMARY & FUSION	sitw_core-core_eval	0.153	0.183	0.268	0.349	0.023
	sre18_dev_vast	0.152	0.337	0.370	0.407	0.040

Table 11. Results on pooled SRE16 - adapted with SRE16 unlabeld major data. Fusion and calibration is unoptimal - it is traned on SRE18 CMN2 dev - same as for our primary submission

	EER(%)	min_C	act_C
TFXvec	8.11	0.57	1.13
TFXvecAtt	7.77	0.65	0.66
KaldiBigXvec	8.04	0.58	1.60
Fusion	6.99	0.53	1.13

7. REFERENCES

[1] Yingke Zhu, Tom Ko, David Snyder, Brian Mak, and Daniel Povey, "Self-attentive speaker embeddings for text-independent speaker verification," *Proc. Interspeech 2018*, pp. 3573–3577, 2018.

[2] Koji Okabe, Takafumi Koshinaka, and Koichi Shinoda, "Attentive statistics pooling for deep speaker embedding," *arXiv preprint arXiv:1803.10963*, 2018.

[3] FA Chowdhury, Quan Wang, Ignacio Lopez Moreno, and Li Wan, "Attention-based models for text-dependent speaker verification," *arXiv preprint arXiv:1710.10470*, 2017.

[4] Anna Silnova, Niko Brummer, Daniel Garcia-Romero, David Snyder, and Lukáš Burget, "Fast variational bayes for heavy-tailed plda applied to i-vectors and x-vectors," in *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018.*, 2018.

[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, number EPFL-CONF-192584.

[6] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," *Submitted to ICASSP*, 2018.

[7] Jahangir Alam, Patrick Kenny, Pierre Ouellet, Themos Stafylakis, and Pierre Dumouchel, "Supervised/unsupervised voice activity detectors for text-dependent speaker recognition on the rsr2015 corpus," in *Proc. Odyssey 2014 The Speaker and Language Recognition Workshop*, 2014, pp. 123–130.

[8] 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.

[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, Jan. 2016.

[10] Qing Wang, Wei Rao, Sining Sun, Lei Xie, Eng Siong Chng, and Haizhou Li, "Unsupervised domain adaptation via domain adversarial training for speaker recognition," in *ICASSP*. 2018, pp. 4889–4893, IEEE.

[11] Martín Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017, pp. 214–223.

[12] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu, "Wasserstein distance guided representation learning for domain adaptation," in *AAAI*. 2018, pp. 4058–4065, AAAI Press.

[13] Mireia Diez, Lukáš Burget, and Pavel Matějka, "Speaker diarization based on bayesian hmm with eigenvoice priors," in *Odyssey 2018, The Speaker and Language Recognition Workshop*, 2018.

[14] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman, "Voxceleb: A large-scale speaker identification dataset," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, 2017, pp. 2616–2620.

[15] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman, "Voxceleb2: Deep speaker recognition," in *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018.*, 2018, pp. 1086–1090.

[16] 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.