The JHU-MIT System Description for NIST SRE18

Jesús Villalba[1], *Nanxin Chen*[1], *David Snyder*[1,2], *Daniel Garcia-Romero*[2], *Alan McCree*[2], *Gregory Sell*[2], *Jonas Borgstrom*[3], *Fred Richardson*[3], *Suwon Shon*[4], *François Grondin*[4], *Réda Dehak*[5], *L. Paola García-Perera*[1], *Pedro A. Torres-Carrasquillo*[3], *Najim Dehak*[1]

[1]Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, USA
[2]Human Language Technology Center of Excellence, Johns Hopkins University, Baltimore, MD, USA
[3] MIT Lincoln Laboratory, Lexington, MA, USA
[4] MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA, USA
[5] LSE-EPITA, Villejuif, France

{jvillalba,bobchennan,dsnyder,dgromero,alan.maccree,gsell,paola.garcia,ndehak3}@jhu.edu,
{jonas.borgstrom,frichard,ptorres}@ll.mit.edu,
{swshon,fgrondin}@mit.edu, reda.dehak@gmail.com

Abstract

This document represents the SRE18 system description for the joint effort of the teams at JHU-CLSP, JHU-HLTCOE, MIT Lincoln Labs., MIT CSAIL and LSE-EPITA. All the developed systems consisted of Neural network/i-vector embeddings with some flavor of PLDA back-end. The systems were tailored to the video (VAST) condition or to the telephone condition (CMN2). For VAST, the primary system was a fusion of a 16 kHz TDNN x-vector, 16 kHz factorized TDNN x-vector, 8 kHz TDNN x-vector and 8 kHz ResNet34-Attention embedding. For CMN2, the primary was a fusion of two TDNN x-vectors and ResNet34-Attention embedding. For development in the VAST condition, we used the SITW eval core-multi dataset where we obtained Cprimary=0.105. For telephone, we used the SRE18 dev CMN2 where we obtained Cprimary=0.256. The contrastive submissions included the best single system (JHU-HLTCOE, SITW Cp=0.137, CMN2 Cp=0.312); and the best fusions of 1, 2, 3,... systems from the JHU-CLSP-MIT sub-team.

1. Introduction

The JHU-MIT submission is the joint effort of the teams at Johns Hopkins CLSP and HLTCOE, MIT Lincoln Laboratory (MIT-LL), MIT CSAIL and LSE-EPITA. We worked as two sub-teams until the final part of the evaluation:

- JHU-HLTCOE.
- JHU-CLSP-MITLL: JHU-CLSP, MIT-LL, MIT CSAIL and LSE-EPITA.

All the systems developed for this evaluation consisted of a neural network or i-vector embedding followed by some form of PLDA classifier with or without score normalization. More in detail, all systems followed these steps:

1. Acoustic feature extraction (MFCC).
2. Voice activity detection.
3. Diarization (SITW/VAST data).
4. Embedding extraction.
5. LDA dimensionality reduction.
6. Centering, whitening and length normalization.
7. PLDA log-likelihood ratio evaluation.
8. Adaptive score-normalization (all system used AS-Norm unless said otherwise).
9. Fusion/calibration.

We explored several types of DNN embeddings: deep TDNN x-vectors [1, 2], factorized TDNN [3] x-vector, ResNet with LDE [4] or multi-head attention pooling. We developed systems at 16 kHz and at 8 kHz. We found that fusing 8 kHz and 16 kHz systems was beneficial for the video condition. We adapted the back-ends to the video (VAST) condition or to the telephone condition (CMN2). For VAST, the primary system was a fusion of a 16 kHz TDNN x-vector, 16 kHz factorized TDNN x-vector, 8 kHz TDNN x-vector and 8 kHz ResNet34-attention embedding. For CMN2, the primary was a fusion of two TDNN x-vectors and ResNet34-attention embedding. The contrastive submissions included: the best single system (JHU-HLTCOE); and the best fusion of 1, 2, 3,... systems from the JHU-CLSP-MIT sub-team.

2. Training datasets

2.1. Individual datasets

The datasets used for training included:

- Switchboard phase1-3 and cellular1-2.
- NIST SRE04-10 as prepared by the SRE16 Kaldi recipe[1].
- NIST SRE12 telephone data (SRE12-tel).
- NIST SRE12 phonecalls recorded through far-field microphone (SRE12-micphn). We did not use interviews to avoid dealing with the interviewer removal.
- MIXER6 telephone phonecalls (MX6-tel).
- MIXER6 microphone phonecalls (MX6-micphn).

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

- VoxCeleb 1+2: original distributions of VoxCeleb 1 and 2. Speakers that overlap with SITW were removed.
- VoxCelebCat 1+2: the original distribution of VoxCeleb split each video into multiple short excerpts. We concatenated all excerpts from the same video into one file. This makes the dataset more appropriate for PLDA training and also helps to balance the weight of each video in the embedding training.
- SITW-dev-core: single speaker segments from the Speakers in the Wild development set.
- SITW-dev-test-diarized. Segments obtained from diarizing the SITW dev test set.
- SRE18-dev-unlabeled: This was used for PLDA adaptation and score normalization in the SRE18 CMN2 condition.
- SRE18-dev-VAST-diarized: SRE18 VAST development set. For enrollment segments we used diarization marks provided by the organizer. For the test segments, we used the segments obtained by our diarization system. This was used for centering adaptation and score normalization in the SRE18 VAST condition.

2.2. JHU-CLSP-MITLL Training data

The dataset combinations used in the systems from the JHU-CLSP-MITLL team were:

- CLSP-Train-8k: This was used to train DNN and i-vector embeddings at 8 kHz. This set included Switchboard, SRE04-10, SRE12-tel, SRE12-micphn, MX6-tel, MX6-micphn, VoxCelebCat and SITW-dev-core. Databases originally at 16 kHz were downsampled to 8 kHz. Recordings shorter than 4 seconds and speakers with less than 8 recordings were discarded. This dataset contained 735018 utterances from 12872 speakers.
- CLSP-Train-8k-phn: This was used to train x-vectors at 8 kHz. This set is the same used in the SRE16 Kaldi recipe and includes Switchboard, SRE04-10, MX6-tel, MX6-micphn. It contained 211034 recordings from 5139 speakers.
- CLSP-Train-16k: This was used to train DNN and i-vector embeddings at 16 kHz. This set included SRE12-micphn, MX6-micphn, VoxCelebCat and SITW-dev-core. It contained 436815 recordings from 7936 speakers.
- CLSP-PLDA-tel-8k: This was used to train back-ends for the CMN2 condition with embeddings extracted at 8 kHz. It consisted of SRE04-10-telephone-only, SRE12-tel, and MX6-tel. It contained 175116 recordings from 4585 speakers.
- CLSP-PLDA-vid: This was used to train back-ends for embeddings extracted at 16 kHz for both VAST and CMN2 condition; and for 8 kHz VAST condition. It consisted of VoxCelebCat and SITW-dev-core. It contained 418711 recordings from 7304 speakers.
- SITW-dev-diar: This was used to center the SITW data. It consists of SITW-dev-core and SITW-dev-test-diarized. It consisted of 4447 recordings.
- SITW-SRE18-dev-diar: This was used for score normalization in the VAST condition. It consisted of SITW-dev-core SITW-dev-test-diarized; and SRE18-dev-VAST diarized. It consisted of 4516 recordings.

The *Train* and *PLDA* datasets were augmented with reverberation using impulsional responses from RWCP sound scene database, the 2014 REVERB challenge database and the Aachen impulse response database (AIR)[2]. After that, we added noises from the MUSAN corpus[3]. Thus, augmented signals contained noise and reverberation at the same time. The number of augmented segments was around twice the size of the original dataset. We combined the original data and the augmented data so the total dataset was around $3\times$ the original size.

2.3. JHU-HLTCOE Training data

This section describes the training data used by the JHU-HLTCOE team for the 8 kHz and 16 kHz x-vector systems. The x-vector systems themselves are described in Section 6.2.

2.3.1. 8 kHz System

The 8 kHz x-vector DNN was trained on Switchboard, SRE04-10, and VoxCelebCat. In total, there are 13,136 speakers in this dataset, with 1,314,442 utterances prior to augmentation. After removing utterances with less than 4 seconds of speech (as determined by the Kaldi energy VAD described in Section 5.1) and applying data augmentation, the amount of training data increased to 5,208,831 utterances. All 16 kHz recordings were downsampled to 8 kHz using SoX.

The backend was trained on SRE04-10. This consists of 4,263 speakers with 49,694 utterances before augmentation and 110,944 utterances after augmentation. SRE18-dev-unlabeled was used for adaptation. See Section 7.2.2 for details on the backend.

2.3.2. 16 kHz System

The 16 kHz x-vector DNN was trained on augmented VoxCelebCat. This originally contained 1,236,567 from 7,185 speakers. After augmentation and the removal of any utterance with less than 4 seconds of speech (as determined by the Kaldi energy VAD) the amount of training data increased by 6 times to 7,419,402.

The backend was trained on augmented VoxCeleb. This provides 7,162 speakers. Augmentation doubles the amount of utterances from 155,113 utterances before augmentation to 305,113 afterwards. The SITW-dev and SRE18-dev-VAST-diarized were used for adaptation. See Section 7.2.1 for a description of the backend.

2.3.3. Data Augmentation

To augment an utterance, we randomly pick from one of the following strategies:

- **Reverb:** Artificially reverberate via convolution with simulated RIRs from the AIR dataset
- **Music:** A single music file (without vocals) 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).
- **Babble:** Three to seven speakers are randomly picked from MX6-micphn, summed together, then added to the original signal (13-20dB SNR)

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

- **Codec:** If the file is from VoxCelebCat, simulate GSM AMR phone encoding[4]

2.4. MIT-CSAIL Training data

A total 359,463 utterances from 11,900 speakers are used for the MIT-CSAIL team. At first, 258,655 utterances came from SRE 04, 05, 06, 08, 10 and MIXER6, Voxceleb1 development, Voxceleb2 development and Switchboard dataset for training Neural Network to extract 512 dimensions x-vector. For Voxceleb1 and 2 datasets, utterances from the same audio file were concatenated into a single wav file. Augmentation using MUSAN noise dataset was also done on the training set and randomly selected 150k utterances. The utterances which are shorter than 5 seconds after removing silence and speakers which have fewer than 8 utterances were filtered out from the training .dataset. Finally, total 359,463 utterances from 11,900 speakers are used for training the x-vector network. A different subset of the dataset was used for the back-end of the system.

3. Development datasets

The development datasets were used to train fusion and calibration; and measure performance.

For the VAST condition, we found the development set provided by the organization too small to provide reliable performance estimation (only 270 trials). Also, there were only around 2-3 false alarm errors at the $P_T = 0.05$ operating point so we thougth that calibrating with that data was risky. Thus, we decided to use the SITW eval core-multi condition because we thought that it would be similar to the VAST condition since it also consists of speech from video and it also requires diarization in the test side. However, we found that training calibration on SITW wasn't helpful to obtain good calibration on the VAST dev set. Inspection of the VAST dev score distributions showed that VAST target scores were similar to the SITW target scores, but VAST non-targets were higher than SITW non-targets. We were able to partially fix this problem by using the VAST dev set for embedding centering and adaptive score normalization. Finally, we decided to train fusion and calibration on SITW and didn't risk to do it on VAST.

For the CMN2 condition, we used the dev set provided by the organizers.

4. Acoustic features

Both conditions, VAST and CMN2, were evaluated with 8 kHz and 16 kHz systems. When using 8 kHz systems, 16 kHz datasets were downsampled to 8 kHz. When using 16 kHz systems SRE18 CMN2 were usampled with low-pass filter interpolation using SoX[5].

The JHU-CLSP-MITLL team used 23 dimension MFCC (23 Mel filters) for x-vector systems at 8 kHz; and 40 dimension MFCC (40 filters) for x-vectors at 16 kHz. The JHU-HLTCOE team used 23 dimension MFCCs (with 23 Mel filters) for the 8 kHz x-vector system, and 30 dimension MFCCs (with 30 Mel filters) for the 16 kHz x-vector system. For embeddings based on 2D convolutions (ResNet34), we used 23 log-Mel filter banks and 40 log-Mel filter banks for 8 kHz and 16 kHz respectively. The 8 kHz i-vector system used 23 MFCC with first and second derivatives. Features were short-time centered before silence removal with a 3 seconds sliding window.

5. Voice activity detection

5.1. Kaldi energy VAD

The Kaldi energy VAD makes frame-level decisions, classifying a frame as speech or non-speech based on the average log-energy in a given window. This VAD was used by all JHU-CLSP-MITLL systems for all the training data (8 kHz and 16 kHz); for all the telephone test data; and for the SITW/VAST data with 8 kHz embeddings and the ResNet-LDE 16 kHz system. The JHU-HLTCOE systems also used this VAD for all training and test data.

5.2. MITLL VAD

The speech activity detection (SAD) system was trained as part of the single-channel speech enhancement system in [5]. As discussed in [5], the enhancement system outputs frame-level speech activity posterior probabilities, along with frequency-dependent posteriors. The raw frame-level posteriors were then decoded using the forward-backward algorithm, and a hard threshold was applied. Finally, short durations of active or inactive speech were removed to promote contiguous segments in the final SAD transcription.

The enhancement system was trained by creating a parallel corpus of noisy and reverberant speech, similar to the recipe discussed in [5]. However, only speech from the SRE10 corpus was used to create the clean target signals. The noisy and reverberant signal versions were simulated by applying room impulse responses and noisy signals from the RIR-NOISES and MUSAN corpora.

The JHU-CLSP-MITLL 16 kHz x-vector systems used this VAD on the SITW and VAST data.

5.3. MIT-CSAIL VAD

The Kaldi energy based VAD was used for the telephone speech, CMN2. For the VAST set, we applied unsupervised clustering based VAD to detect musical background noise which confuses the energy based VAD. This VAD approach relies on the hypothesis that speech is dominant on average, but that noise can be dominant sporadically in time. The power spectral density (PSD) frames are first computed with a Short-Time Fourier Transform (STFT) for the entire test utterance. A singular value decomposition (SVD) is then performed on these PSD frames. The PSD frames are then projected on the left singular vectors associated with the largest singular values (which number is set empirically), to produce PPSD, which stands for Projected PSD. The log of the energy of each PPSD frame is then computed, and the resulting observations are split into two classes using k-means. The frames belonging to the class with the highest energy level are then considered as speech. This approach is therefore fully unsupervised, but is only applicable offline as it requires the full test utterance.

6. Embeddings

6.1. JHU-CLSP x-vectors

We tried two different architectures for x-vector embeddings [1, 2]. The first one is the TDNN used in [2], which is also the default in Kaldi recipes. Table 1 summarizes this network architecture. Each feature frame is processed by a sequence of

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

[5] sox.sourceforge.net

Table 1: *Baseline TDNN x-vector architecture*

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

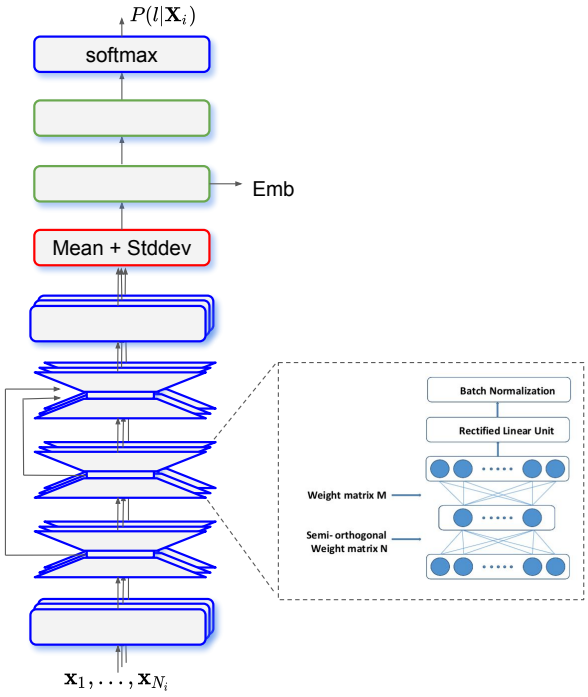

Figure 1: Factorized TDNN x-vector architecture.

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		1024	256
3	F-TDNN-ReLU	t	t		1024	256
4	F-TDNN-ReLU	t-3, t	t, t+3		1024	256
5	F-TDNN-ReLU	t	t	3	1024	256
6	F-TDNN-ReLU	t-3, t	t, t+3		1024	256
7	F-TDNN-ReLU	t-3, t	t, t+3	2, 4	1024	256
8	F-TDNN-ReLU	t-3, t	t, t+3		1024	256
9	F-TDNN-ReLU	t	t	4, 6, 8	1024	256
10	Dense-ReLU	t	t		2048	
11	Pooling (mean+stddev)	full-seq			2x2048	
12	Dense-ReLU				512	
13	Dense-ReLU				512	
14	Dense-Softmax				N. spks.	

Table 3: *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	512
10	Dense-ReLU	t	1500
11	Pooling (mean+stddev)	Full-seq	2x1500
12	Dense(Embedding)-ReLU		512
13	Dense-ReLU		512
14	Dense-Softmax		Num. spks.

time-delay layers. The pooling layer computes mean and standard deviation of the TDNN output over time to obtain a unique representation per recording. The Pooling layer output is projected to a lower dimension to obtain the speaker embedding. The output of the network are posterior probabilities for the training speakers it can be trained by minimizing a categorical cross-entropy objective.

For the second x-vector architecture, we replaced the pre-pooling layers by a factorized TDNN (F-TDNN) with skip connections [3]. Figure 1 depicts this network. 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 [6], we introduced 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 added like in ResNet. This allows to make the network deeper by alleviating the vanishing gradient problem. According to [3], the best option is to create skip connections between the low-rank interior layers of the F-TDNN.

Table 2 summarizes the layers in our F-TDNN x-vector.

In summary, we had five x-vector embeddings:

- x-vector-8k-v1: 8 kHz TDNN x-vector trained on CLSP-Train-8k-phn data.
- x-vector-8k-v3: 8 kHz TDNN x-vector trained on CLSP-Train-8k data.
- x-vector-8k-v4: 8 kHz F-TDNN x-vector trained on CLSP-Train-8k data.
- x-vector-16k-v1: 16 kHz TDNN x-vector trained on CLSP-Train-16k data.

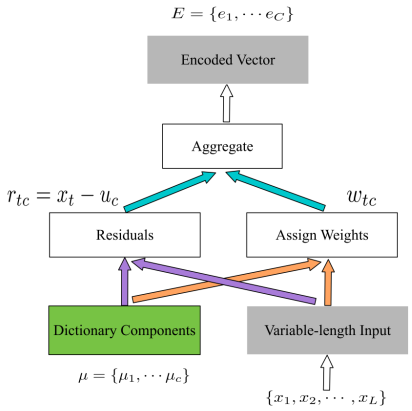

Figure 2: LDE pooling layer [4].

- x-vector-16k-v2: 16 kHz F-TDNN x-vector trained on CLSP-Train-16k data.

6.2. JHU-HLTCOE x-vectors

For the evaluation we used an extended version of the TDNN in [2], which is the default architecture in the public Kaldi recipes. Table 3 summarizes the extended network (E-TDNN) architecture. The two main differences are a slightly wider temporal context of the TDNN (due to the addition of layer 7), and interleaving dense layers in between the convolutional layers (equivalent to the 1x1 convolutions used in computer vision architectures). This architecture has been found to greatly outperform the baseline TDNN in the SITW and SRE16 benchmarks. The network outputs posterior probabilities for the training speakers and it was trained by minimizing a categorical cross-entropy. The x-vector is extracted from layer 12 prior to the ReLU non-linearity.

We trained 2 systems based on this architecture. One using 8 kHz data for the CMN2 task, and another using the 16 kHz data for VAST task. For the 8 KHz system the number of training speakers was 13136 and the total number of parameters was approximately 13 million (only 6 million of them are needed to extract x-vectors). The 16 kHz system was trained with 7185 speakers and the number of parameters was 8 million (4 million for extracting the x-vector).

6.3. JHU-CLSP ResNet-LDE/Att embeddings

The ResNet-LDE is an evolution of the x-vector system where the TDNN layers are replaced by a residual network with 2D convolutions (ResNet34) [6] and the pooling layer is replaced by a Learnable dictionary encoding (LDE) layer [7, 4].

The original x-vector framework assumes that the frame-level TDNN representations before pooling are uni-modal. Thus, to pool those representations, we just compute their mean and standard deviation. Meanwhile, 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 essentially the same as we do in the GMM-i-vector paradigm. The component posteriors are obtained as,

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

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

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

and we concatenate the embeddings for all the components $\mathbf{e} = (\mathbf{e}_1^{\mathrm{T}}, \ldots, \mathbf{e}_C^{\mathrm{T}})^{\mathrm{T}}$. 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 final embedding. This projection has the same role as the total variability matrix in i-vectors.

We also used a variant of LDE that normalizes the component posteriors to sum up to one in the time dimension, instead of doing it in the Gaussian component dimension,

$$w_{t,c} = \frac{\exp(-s_c \left\|\mathbf{x}_t - \boldsymbol{\mu}_c\right\|)}{\sum_{t=1}^{T} \exp(-s_c \left\|\mathbf{x}_t - \boldsymbol{\mu}_c\right\|)} \tag{3}$$

This is known as *multi-head attention*. Euclidean norm is used instead of squared Euclidean distance since sometimes we observed better convergence.

Instead of using categorical cross-entropy for training, we used angular softmax loss [8]. Angular softmax loss has stronger requirements for correct classification when $m \geq 2$ (integer that controls the angular margin), which generates an angular classification margin between embeddings of different classes [4, 9]. For angular softmax training we used pretrained model from the original softmax loss as initialization.

In summary, we had three ResNet-LDE/Att embeddings:

- resnet-8k-v1: 8 kHz ResNet-Att trained on CLSP-Train-8k with angular margin $m = 2$.
- resnet-8k-v2: 8 kHz ResNet-Att trained on CLSP-Train-8k with angular margin $m = 3$.
- resnet-16k-v1: 16 kHz ResNet-LDE trained on CLSP-Train-8k with angular margin $m = 2$.

6.4. LSE-EPITA i-vectors

The LSE-EPITA used a GMM-UBM i-vectors based system. The system is based on a UBM with 2048 full covariance matrix Gaussian components and 600 dimensional i-vectors. It was trained on the CLSP-Train-8k dataset.

6.5. MIT-CSAIL x-vectors

TDNN architecture was slightly modified based on the public Kaldi recipes for training the x-vector system: xvec-8k-CSAIL. First, we removed the Relu activation on the second last layer and extract embeddings from the last hidden layer [10]. Based on this architecture, we trained two different x-vector system by using original segmented Voxceleb dataset and concatenated Voxceleb dataset for the same video. Since the performance is better on the concatenated version, we only contributed the better system. Different VAD was applied to the VAST and SITW dataset as described in section 5.3 but used the same x-vector system to extract speaker embeddings.

7. Back-ends

7.1. JHU-CLSP back-end

The JHU-CLSP back-end consisted of LDA, centering, whitening, length normalization and generative Gaussian SPLDA. We

tuned different back-ends for the SITW/VAST condition and the CMN2 condition.

7.1.1. SITW/VAST

For SITW/VAST, we trained LDA, centering, whitening and SPLDA on the CLSP-Train-vid data. LDA dimension was 200 and SPLDA had 150 eigenvoices. In the dev/eval phase SITW was centered on SITW-dev-diar. Meanwhile, the centering for the VAST was MAP adapted from SITW-dev-diar to SRE18-dev-VAST-diar with relevance factor $r = 14$.

As there may be several speakers in the test segment, we used diarization to obtain several speaker clusters. We scored the enrollment segment against all the test segment clusters and selected the maximum score.

Although, we obtained better results on SITW without score-normalization, we observed better alignment between SITW and VAST dev score distributions by using score normalization. Thus, we thought that, using score-normalization, we would obtain a better calibrated system on the evaluation data also. We used adaptive S-Norm with SITW-SRE18-dev-diar as cohort. For SITW, we selected the 500 top cohort segments. For VAST, we selected the top 120 cohort segments. When applying the score normalization on the VAST dev we have target trials in the cohort score matrices that we don't want to use to compute the normalizing parameters. We assumed that the top 7 segments were target speakers and don't use them to perform the normalization.

7.1.2. CMN2

For CMN2, for systems at 8 kHz we trained LDA, centering, whitening and SPLDA on the CLSP-Train-tel-8k dataset. For systems at 16 kHz, we used the CLSP-PLDA-vid–essentially because we didn't want to upsample all the SRE telephone data–.

When processing CMN2 data SRE18 unlabeled/dev/eval we used the centering computed on the SRE18 unlabeled data. We also adapted the SPLDA to the SRE18 unlabeled data in two steps. First, we adapted SPLDA using the telephone numbers in the meta-data as speaker labels. Second, we used the adapted SPLDA to compute the scores to do agglomerative hierarchical clustering (AHC) of the SRE18 unlabeled segments and obtain new speakers labels. The number of speakers for AHC was tuned based on the SRE18 CMN2 dev Cprimary. In both steps, we adapted from the original out-of-domain SPLDA. The within-class and across-class covariances of the adapted model were a weighted sum of the out-of-domain $\mathbf{S}_{\text{out}}$ and in-domain $\mathbf{S}_{\text{in}}$ covariances,

$$\mathbf{S}_{\text{adapt}} = \alpha\mathbf{S}_{\text{in}} + (1-\alpha)\mathbf{S}_{\text{out}} \tag{4}$$

with $\alpha = 0.6$.

We used adaptive S-Norm using SRE18 unlabeled as cohort. We used the top 400 cohort segments to compute the normalization parameters of each trial.

7.2. JHU-HLTCOE back-end

We used different back-ends for the SITW/VAST condition and the CMN2 condition.

7.2.1. SITW/VAST

For this task, we use a generative PLDA backend. Before scoring, the x-vectors are centered, projected to 200 dimensions using LDA, and length-normalized. LDA and PLDA is estimated using the augmented VoxCeleb described in Section 2.3.2. When scoring against SITW or VAST data, the x-vectors are centered on SRE18-dev-VAST and augmented VoxCeleb (each is equally weighted). Score normalization is performed using the top 10% of cohorts from the combination of SITW-dev-test and SRE18-dev-VAST-diarized. Note that, when we score the SRE18 dev data, we remove the largest 30 scores from the score-normalization, as they are likely from target speakers.

Since there may be multiple speakers in the test recordings, we first perform diarization as described in Section 9.2. For a given trial, we compute the PLDA scores between the enrollment recordings and all speakers discovered in the test recording after diarization. The maximum PLDA score is used as the score for that trial.

7.2.2. CMN2

For this task we used the generative Heavy Tailed PLDA (HT-PLDA) classifier described in [11]. The HT-PLDA training data was pre-processed by centering and whitening, but no length-normalization was applied [12]. We used a speaker subspace of dimension 150 and $\nu = 20$. When processing the CMN2 unlabeled/dev/eval data, we used the centering computed on the SRE18 unlabeled data and the whitening from the HT-PLDA training set.

We performed unsupervised domain adaptation by clustering the CMN2 unlabeled data (we used the clusters estimated by JHU-CLSP) and adapting the within-class and across-class covariances using the interpolation method described in [13]. We set the interpolation parameter $\alpha = 0.2$. We used adaptive S-Norm using the SRE18 unlabeled data as cohort. We used the top 20% segments to compute the normalization parameters of each trial.

7.3. MIT Lincoln Laboratory back-end

For this back-end, i-vector preprocessing (LDA/centering/whitening) was the same as in the JHU-CSLS back-end. Discriminative PLDA was trained using the Newton Method to minimize the log loss of verification trials. By diagonalizing the across-class and within-class covariance matrices as a pre-processing step, the training process only updated the diagonal elements of the covariances matrices, helping to avoid over-fitting. The D-PLDA system performed domain adaptation by extending the cost function to include trails from the CMN2 unlabeled development set. In the absence of true speaker labels for this set, telephone labels were instead assumed to convey accurate speaker information. The D-PLDA training set included data from the NIST SRE04-SRE10, augmented with with noise and reverberation, as well as the SRE18 development set, resulting in approximately 6B training trials.

7.4. MIT-CSAIL back-end

For CMN2, we used SRE18 development unlabeled set for centering. Then centered x-vector projected into 200dimension using LDA trained on SRE telephone (04-10, only telephone part) dataset. PLDA has 150 eigenvoices and trained using same entire dataset for training x-vector(SRE + Voxceleb). Adaptive S-norm was used for score normalization using SRE18 unlabeled set.

For SITW/VAST, we used Voxceleb 1 and 2 for centering the x-vector. LDA and PLDA were trained using SRE(04-

10) plus its Augmentation and Voxceleb concatenated version respectively. Adaptive S-norm was applied using SITW and SRE18 development VAST set.

8. Fusion and Calibration

Fusion and Calibration was performed using linear logistic regression with the Bosaris toolkit [14]. To select the best fusion combination, we implemented 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. Thus, we select the best fusion of two systems. We fix those two system and then add a third system, and so on. To reduce the chances of over-fitting, in each step, we prioritize fusions with only positive weights.

For VAST, we trained fusion/calibration on SITW eval-core multi on operating point $P_{\mathcal{T}} = 0.05$. For CMN2, we trained on SRE18 dev CMN2 set on in operating point $P_{\mathcal{T}} = 0.01$. Although the average operating point for CMN2 is $P_{\mathcal{T}} = 0.075$, we decided to use a higher target prior to have more false alarm errors and obtain a more robust calibration.

9. Diarization

9.1. JHU-CLSP diarization

For diarization of the SITW multi and VAST test data, we used a similar setup to the Kaldi x-vector callhome diarization recipe [6], which is based on [15].

We used of the 16 kHz F-TDNN x-vector (x-vector-16k-v2), to compute embeddings using a sliding window with 1.5 seconds of frame-length and 0.75 seconds of frame-shift. We obtained sliding window embeddings for VAST, SITW and VoxCelebCat without augmentation. We used VoxCeleb x-vectors to train LDA dimensionality reduction to 120, centering and PLDA. We scored all x-vectors in a given recording against each other and applied AHC on the score matrix. We tuned the stopping threshold for AHC to optimize performance SITW eval core and core-multi sets.

We assumed that the target speaker would have a significant amount of speech in the test segment. For that reason, we discarded all the speaker clusters with less than 10 seconds duration unless all clusters in the segment are shorter than that.

9.2. JHU-HLTCOE diarization

We perform speaker diarization on the VAST and SITW test recordings due to the possibility of multiple speakers. The diarization is based on the callhome_diarization Kaldi recipe as well as [16] and uses x-vectors with PLDA and agglomerative hierarchical clustering (AHC).

We use the 16 kHz DNN from Section 6.2 to extract x-vectors. For an utterance in the test data, x-vectors are extracted from 1.5 second segments with a 0.75 second shift. The PLDA backend consists of centering, whitening and length normalization, followed by scoring. All components (mean, whitening, and PLDA model) are trained on 250,000 three second segments extracted from the augmented VoxCeleb recordings. Once an affinity matrix of PLDA scores is obtained for a test recording, the segments are clustered using AHC. In order to eliminate the need for a tuned AHC stopping threshold, we assume that there are never more than K speakers in an utterance, and perform clustering K times, with exactly $k \in \{1, 2, \ldots, K\}$ clusters each time we perform clustering. The product of this diarization strategy is a set of $\frac{K(K+1)}{2}$ ways to partition a recording of up to K speakers. Each of these potential speakers are then scored against the enrollment recordings, as described in Section 7.2.1. In this work, we assume that there are at most $K = 3$ speakers in a recording.

[6] https://github.com/kaldi-asr/kaldi/tree/master/egs/callhome_diarization/v2

10. Individual systems

Table 4 presents the results of the individual systems on SITW and VAST. Table 5 presents the results of the individual systems on SRE18 CMN2.

The HLTCOE systems were the best performers in each one of the conditions.

All the JHU-CLSP-MITLL systems (xvec, resnet, ivec) used Kaldi energy VAD unless they have the label *llvad* which indicates that they used the MITLL VAD. All the JHU-CLSP-MITLL systems used JHU-CLSP back-end unless they have *llbe* label which means that they used MITLL back-end. All systems used AS-Norm unless they have the label *nosn*.

11. Submissions

Table 6 summarizes the fusions in our primary and contrastive submissions. We submitted:

- Primary: Best reasonable fusion of 3-4 DNN embedding systems. We thought that including more systems would lead to over-fitting.
- JHU-HLTCOE: Best single system.
- JHU-CLSP-MIT: Best fusion of 1, 2, 3, ... systems. This will allow us to measure the gain the we obtain each time that we add a new system to the fusion.
- JHU-CLSP-MIT no-vid-snorm: Systems without AS-Norm on the VAST condition. This will allow us to assess whether we were right when we decided that system with AS-Norm would be better calibrated on VAST.

Table 7 presents the results of our submissions on SITW and VAST. Table 8 presents the results of our submissions on SRE18 CMN2.

12. Computation resources

Processing times were measured in Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz. Most of the processing time is dedicated to the embedding extraction. MFCC, VAD and back-end processing time are negligible comparison. For the VAST case, we need to add the time needed to extract x-vector with sliding window for diarization.

13. References

[1] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev 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, aug 2017, pp. 999–1003, ISCA.

[2] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-Vectors : Ro-

Table 4: *Individual systems results on SITW/VAST*

System	SITW EVAL CORE			SITW EVAL CORE-MULTI			SRE18 DEV VAST		
	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp
COE-16k	**1.99**	**0.138**	**0.141**	**2.26**	**0.135**	**0.137**	**3.7**	0.337	0.498
xvec-16k-v1-llvad	3.4	0.185	0.188	3.86	0.191	0.191	**3.7**	0.337	0.424
xvec-16k-v2-llvad	**1.89**	**0.124**	**0.126**	**2.33**	**0.135**	**0.137**	7	0.37	0.498
resnet-16k-v1	2.16	0.136	0.142	2.63	0.145	0.146	**3.7**	**0.226**	0.424
xvec-8k-v1	5.21	0.278	0.284	5.6	0.287	0.287	11.11	0.3	0.691
xvec-8k-v3	3.58	0.197	0.202	3.93	0.206	0.207	7.41	0.296	0.535
xvec-8k-v4	2.6	0.15	0.158	2.94	0.161	0.162	7	0.263	0.42
resnet-8k-v1	2.71	0.154	0.159	3.12	0.164	0.164	5.76	0.148	0.383
resnet-8k-v2	2.69	0.154	0.162	2.99	0.165	0.166	4.12	0.267	**0.267**
xvec-8k-CSAIL	4.6	0.274	0.275	6.18	0.321	0.321	10.7	0.407	0.605
ivec-8k	8.22	0.384	0.393	8.67	0.386	0.387	18.52	0.486	0.568
xvec-16k-v2-llvad-nosn	**1.61**	**0.12**	**0.122**	**2.01**	**0.133**	**0.134**	4.53	0.309	**0.309**
resnet-16k-v1-nosn	1.94	0.129	0.131	2.41	0.141	0.142	**3.7**	**0.267**	0.424
xvec-8k-v4-nosn	2.16	0.146	0.15	2.61	0.157	0.158	**3.7**	0.337	0.815
resnet-8k-v2-nosn	2.3	0.158	0.164	2.7	0.165	0.165	**3.7**	0.416	0.741

Table 5: *Individual system results on CMN2.*

Systems	SRE18 DEV CMN2		
	EER	Min Cp	Act Cp
COE-8k	**4.55**	**0.298**	**0.312**
xvec-16k-v1	12.03	0.719	0.725
xvec-16k-v2	9.13	0.642	0.645
resnet-16k-v1	9.53	0.579	0.601
xvec-8k-v1	7.2	0.505	0.51
xvec-8k-v3	5.76	0.384	0.392
xvec-8k-v4	5.19	0.345	0.357
resnet-8k-v1	4.86	0.351	0.363
resnet-8k-v2	5.46	0.326	0.34
resnet-8k-v2-llbe-nosn	**5.64**	**0.319**	**0.337**
xvec-8k-CSAIL	6.12	0.404	0.42
ivec-8k	10.37	0.664	0.685

bust DNN Embeddings for Speaker Recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, Alberta, Canada, apr 2018, pp. 5329–5333, IEEE.

[3] Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohamadi, and Sanjeev 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.

[4] Weicheng Cai, Jinkun Chen, and Ming 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, jun 2018, pp. 74–81, ISCA.

[5] Bengt J Borgstrom, Michael S Brandstein, and Robert B Dunn, "Improving Statistical Model-Based Speech Enhancement with Deep Neural Networks," in *IWAENC*, 2018.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep Residual Learning for Image Recognition," dec 2015.

[7] Weicheng Cai, Zexin Cai, Xiang Zhang, Xiaoqi Wang, and Ming 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, apr 2018, pp. 5189–5193, IEEE.

[8] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song, "SphereFace: Deep Hypersphere Embedding for Face Recognition," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. jul 2017, vol. 2017-Janua, pp. 6738–6746, IEEE.

[9] Zili Huang, Shuai Wang, and Kai Yu, "Angular Softmax for Short-Duration Text-independent Speaker Verification," in *Interspeech 2018*, Hyderabad, India, sep 2018, pp. 3623–3627, ISCA.

[10] Suwon Shon, Hao Tang, and James Glass, "Frame-level Speaker Embeddings for Text-independent Speaker Recognition and Analysis of End-to-end Model," in *IEEE Spoken Language Technology Workshop (SLT)*, 2018.

[11] Anna Silnova, Niko Brümmer, 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*, Hyderabad, India, 2018, pp. 72–76.

[12] Daniel Garcia-Romero and Carol Y. Espy-Wilson, "Analysis of I-vector Length Normalization in Speaker Recognition Systems," in *Proceedings of the 12th Annual Conference of the International Speech Communication Association, Interspeech 2011*, Florence, Italy, aug 2011, pp. 249–252, ISCA.

[13] Daniel Garcia-Romero, Alan McCree, Stephen H Shum, Niko Brummer, and Carlos Vaquero, "UNSUPERVISED DOMAIN ADAPTATION FOR I-VECTOR SPEAKER RECOGNITION," in *Proceedings of Odyssey 2014 - The Speaker and Language Recognition Workshop*, Joensuu, Finland, jun 2014, number June, pp. 260–264, ISCA.

Table 6: *Submission system fusion summary.*

Submission	VAST	CMN2
Primary	xvec-16k-v2-llvad + xvec-8k-v4 + COE-16k + resnet-8k-v2	COE-8k + resnet-8k-v2-llbe-nosn + xvec-8k-v3
JHU-HLTCOE Best	COE-16k	COE-8k
JHU-CLSP-MIT Best1	xvec-16k-v2-llvad	resnet-8k-v2-llbe-nosn
JHU-CLSP-MIT Best2	+ xvec-8k-v4	+ xvec-8k-v4
JHU-CLSP-MIT Best3	+ resnet-8k-v2	+ resnet-8k-v1
JHU-CLSP-MIT Best4	+ resnet-16k-v1	+ xvec-8k-CSAIL
JHU-CLSP-MIT Best5	+ xvec-8k-v1	+ xvec-8k-v3
JHU-CLSP-MIT Best6	+ xvec-16k-v1-llvad	+ ivec-8k
JHU-CLSP-MIT Best1 no-vid-snorm	xvec-16k-v2-llvad-nosn	resnet-8k-v2-llbe-nosn
JHU-CLSP-MIT Best4 no-vid-snorm	+ resnet-8k-v2-nosn + xvec-8k-v4-nosn + resnet-16k-v1-nosn	+ xvec-8k-v4 + resnet-8k-v1 + xvec-8k-CSAIL

Table 7: *Submission systems results on SITW/VAST*

Submission	SITW EVAL CORE			SITW EVAL CORE-MULTI			SRE18 DEV VAST		
	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp
Primary	**1.53**	**0.097**	**0.098**	**1.82**	**0.105**	**0.105**	**3.7**	**0.305**	0.465
JHU-HLTCOE Best	1.99	0.138	0.141	2.26	0.135	0.137	**3.7**	0.337	0.498
JHU-CLSP-MIT Best1	1.89	0.124	0.126	2.33	0.135	0.137	7	0.37	0.498
JHU-CLSP-MIT Best2	1.75	0.101	0.102	2.03	0.111	0.112	4.12	0.337	0.424
JHU-CLSP-MIT Best3	1.71	0.1	0.1	2	0.109	0.11	**3.7**	**0.305**	0.502
JHU-CLSP-MIT Best4	1.56	**0.098**	**0.099**	1.9	0.107	0.108	**3.7**	**0.3**	**0.387**
JHU-CLSP-MIT Best5	**1.5**	**0.094**	**0.094**	**1.83**	**0.103**	**0.104**	**3.7**	**0.263**	0.461
JHU-CLSP-MIT Best6	**1.5**	**0.09**	**0.091**	**1.79**	**0.1**	**0.101**	**3.7**	**0.222**	0.498
JHU-CLSP-MIT Best1 no-snorm	1.61	0.12	0.122	2.01	0.133	0.134	4.53	**0.309**	**0.309**
JHU-CLSP-MIT Best4 no-snorm	**1.29**	**0.093**	**0.097**	**1.65**	**0.107**	**0.107**	**3.7**	**0.305**	0.465

Table 8: *Submission systems results on CMN2.*

Submission	SRE18 DEV CMN2		
	EER	Min Cp	Act Cp
Primary	**4.09**	**0.249**	**0.256**
JHU-HLTCOE Best	4.55	0.298	0.312
JHU-CLSP-MIT Best1	5.64	0.319	0.334
JHU-CLSP-MIT Best2	4.65	0.281	0.291
JHU-CLSP-MIT Best3	4.4	0.283	0.285
JHU-CLSP-MIT Best4	**4.24**	**0.272**	0.279
JHU-CLSP-MIT Best5	4.25	0.273	**0.276**
JHU-CLSP-MIT Best6	**4.24**	0.275	0.277

Table 9: *Computational resources.*

System	Real time factor	Memory (GB)
TDNN x-vector	16	1.5
F-TDNN x-vector	3.5	3
E-TDNN x-vector	7	2
ResNet34 embedding	26.3	0.2
DNN i-vector	100	1.5
F-TDNN x-vector for diar.	5	3

[14] Niko Brummer and Edward 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.

[15] Daniel Garcia-Romero, David Snyder, Gregory Sell, Daniel Povey, and Alan Mccree, "Speaker Diarization Using Deep Neural Network Embeddings," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017*, New Orleans, LA, USA, mar 2017, pp. 4930–4934, IEEE.

[16] Gregory Sell, David Snyder, Alan Mccree, Daniel Garcia-Romero, Jesús Villalba, Matthew Maciejewski, Vimal Manohar, Najim Dehak, Daniel Povey, Shinji Watanabe, and Sanjeev Khudanpur, "Diarization is Hard: Some Experiences and Lessons Learned for the JHU Team in the Inaugural DIHARD Challenge," in *Proceedings of the 19th Annual Conference of the International Speech Communication Association, INTERSPEECH 2018*, Hyderabad, India, sep 2018, pp. 2808—-2812.