The JHU-MIT System Description for NIST SRE20 CTS Challenge

Jesús Villalba[1,2], *Jonas Borgstrom*[3], *Saurabh Kataria*[1,2], *Jaejin Cho*[1],
Pedro A. Torres-Carrasquillo[3], *Najim Dehak*[1,2]

[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

{jvillalba,ndehak3}@jhu.edu, {jonas.borgstrom,ptorres}@ll.mit.edu

Abstract

This document presents the SRE20 CTS Challenge system description for the joint effort of the teams at JHU-CLSP/HLTCOE, and MIT Lincoln Laboratory. All the developed systems consisted of neural network embeddings with some flavor of PLDA back-end. We evaluated multiple x-vector architectures based on ResNets, Transformer and EfficientNet. We also propose novel PLDA mixture back-end and kNN PLDA back-end that provided improvements with respect to the basic PLDA back-end. Our best fusion achieved Act Cp=0.101 in the SRE20 progress and Act Cp=0.087 in SRE20 Eval set. Our best single system achieved Act Cp=0.110 in the SRE20 progress set.

1. Introduction

This document presents the SRE20 CTS Challenge system description for the joint effort of the teams at JHU-CLSP/HLTCOE, and MIT Lincoln Laboratory.

All the systems developed for this evaluation consisted of a neural network embedding based on ResNet, Transformer or EfficientNet followed by some form of PLDA back-end. We tried PLDA adapted from English to non-English data; a mixture of PLDAs; and training a PLDA adapted to each enrollment side, which is trained on the closest speaker to the enrollment speakers. More in detail, all systems followed these steps:

1. Acoustic feature extraction (MFCC).
2. Voice activity detection.
3. Embedding extraction.
4. Embedding post-processing.
5. PLDA log-likelihood ratio evaluation.
6. Adaptive score-normalization (optional).
7. Fusion/calibration.

2. Training datasets

We used the following training datasets:

- **Switchboard phase1-3 and cellular1-2**.
- **NIST SRE04-10**.
- **NIST SRE12** telephone data (SRE12-tel).
- **MIXER6** telephone phonecalls (MX6-tel).
- **NIST SRE16 Dev:** This is the NIST SRE16 development set. It contains 668 recordings from 10 Mandarin speakers and 659 recordings from 10 Cebuano speakers.
- **NIST SRE16 Eval 60%:** This set contains 60% of the speakers in the NIST SRE16 evaluation set. The remaining 40% was kept for development. This set contains 3299 recordings from 60 Cantonese speakers and 2904 recordings from 61 Tagalog speakers.
- **NIST SRE18 Dev**: This set contains 1741 recordings from 25 Tunisian Arabic speakers.
- **NIST SRE18 Eval**: This set contains 13451 recordings of 188 Tunisian Arabic speakers.
- **Fisher Spanish**: This set contains 1638 recordings from 136 Spanish speakers. Several Spanish accents are included.
- **VoxCeleb 1+2:** This dataset contains 7365 speakers audio from video. The original distribution of VoxCeleb splits 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. After concatenation, we obtain 173088 recordings. We applied GSM and AMR-NB telephone codecs to this data using SoX.
- **NIST LRE:** This set includes telephony samples from NIST LRE11-19 which contain more than 5 seconds of active speech. The set was randomly downsampled to a size of 20k.

We trained our x-vectors on the combination of the datasets above (except LRE) with a total of 304k recordings from 13466 speakers. For x-vector training, we augmented speech on the fly with noise and reverberation. Impulse responses for augmentation were obtained from the Aachen impulse response database (AIR)[1]. The noises were acquired from the MUSAN corpus[2]. We used the same SNR levels as in the Kaldi recipes.

For PLDA back-end training, we used NIST SRE04-18 and Fisher Spanish. We did not use any data augmentation.

We also tried to add other datasets to x-vector and PLDA training, but they did not improve the results on the SRE20 progress set:

- IARPA Babel: We added all available languages and obtained pseudo-speaker labels by clustering.
- NIST LRE17: We added all available languages and obtained pseudo-speaker labels by clustering.

[1] http://www.openslr.org/resources/28

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

- Mozilla Common-Voice: We added all available languages except English, using the provided speaker labels. We only used speakers with more than 30 utterances. Telephone codecs were applied with SoX.
- CN-Celeb: This set includes audio from Video from Chinese Celebrities. Telephone codecs were applied with SoX.
- Multilingual LibriSpeech (MLS). Telephone codecs were applied with SoX.

3. Development datasets

We prepared three datasets for development:

- **NIST SRE16 Eval YUE/TGL40%:** This set contains 40% of the speakers (40 YUE and 40 TGL speakers) in the NIST SRE16 evaluation set. We kept the same trial list as in the original SRE16 but keeping only the trials involving those 80 speakers. In total, there are 158k YUE and 174k TGL trials.
- **NIST SRE19 Eval:** This set contains 2.6M trials from Tunisian Arabic speakers.

These three sets were used for fusion. NIST SRE16 YUE40% was used for individual system calibration and final calibration of fusions. We observed that NIST SRE16 YUE produced the best calibration on the SRE20 progress set.

4. Acoustic features and VAD

The acoustic features were 64 log-Mel-filter banks for all our systems. These features were short-time mean normalized with a 3 seconds window. Silence frames were removed using 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.

5. Audio embeddings

5.1. Architectures

All the x-vector architectures follow the x-vector scheme [1, 2]. In essence, the embedding network consists of an encoder that extracts frame-level discriminant embeddings, a pooling mechanism and a classification head. In our case, we tried several encoder architectures and used either statistics pooling (mean+stddev) [1] or channel-wise attentive statistics pooling [3]. The network is trained to minimize the categorical cross-entropy loss of the predicted speaker posteriors. We used additive angular margin softmax loss [4] in all our networks. We describe the encoder architectures in the following paragraphs.

5.1.1. ResNet34

This encoder is based on the original ResNet34 architecture proposed in [5]. ResNet34 has an input stem layer followed by 16 2D convolutional residual blocks. This architecture downsamples the feature maps 3 times with a stride of 2 (8× total downsampling), at the same time as it multiplies the number of channels in the convolutions.

The output of this network is a four dimensional tensor $(B, C, F/8, T/8)$, where B is batch-size, C is number of channels, F is the number of Mel filters and T is time. Channel and frequency dimensions are flattened to $(B, C \times F/8, T/8)$ before passing the features to the pooling layer [6].

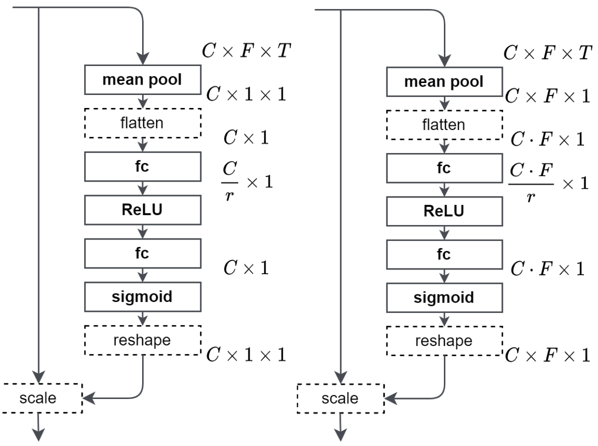

Figure 1: Standard squeeze-excitation (SE) (left) and temporal squeeze-excitation (TSE) (right)

5.1.2. ResNet34-IN

We replaced the ResNet34 Batch-Normalization by Instance Normalization. We also replace the classification head Batch-Normalization by Layer Normalization. Hence, the normalization parameters do not depend on the training batch-size. This enables us to train with smaller batches and longer chunks.

5.1.3. TSE-ResNet34

This encoder adds squeeze-excitation (SE) [7] blocks to ResNet34. The original SE, in Figure 1 (left), in 2D convolutions performs a pooling operation in both time and frequency dimensions (spatial dimensions in image). Then it applies a scaling to the feature maps which is channel dependent but it is the same for all the frequency dimensions. We observed that standard SE does not provide significant gains for speaker recognition. In [8, 9], we proposed temporal squeeze-excitation (TSE), depicted in Figure 1 (right). TSE applies pooling only in the temporal axis and applies a scaling which is different for each channel and frequency dimension.

5.1.4. Transformer

We also tried the Transformer Encoder architecture [10] as an encoder for x-vectors. We used an encoder with 8 self attention blocks. The input stem uses a two 2D Conv layers that downsample time dimension ×4. We also implemented a local attention procedure that limits the self-attention receptive field to 6 time steps (25 msecs) in each layer. This is similar to the Longformer architecture [11].

5.1.5. EfficientNet-b4

EfficientNet architecture was proposed in [12] for images. This is a residual network that used 2D separable convolutions to reduce the number of multiplications of the network. The work in [12] proposes a base architecture, denoted as EfficientNet-b0. Then larger networks EfficientNet-bn are obtained by scaling up the number of channels and network depth in such way that EfficientNet-bn is 2^n times more computationally expensive than b0. We found that b4 was needed to improve ResNet34. We also needed to remove the first two feature map downsamplings from the original EfficientNet architecture.

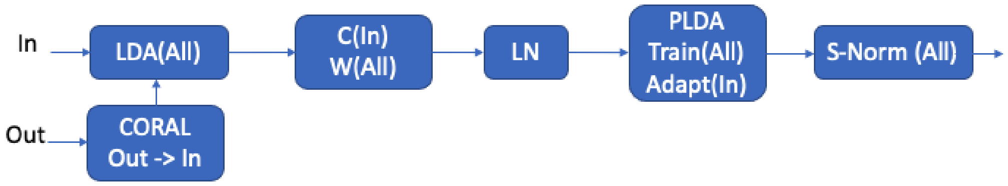

Figure 2: CORAL+LDA+LN+PLDA+S-Norm Back-end. This scheme is used in JHU-PLDA-v4 and MITLL-mix1 back-ends. *In* denotes in-domain data, *Out* denotes out-of-domain data, and *All* denotes pooling both.

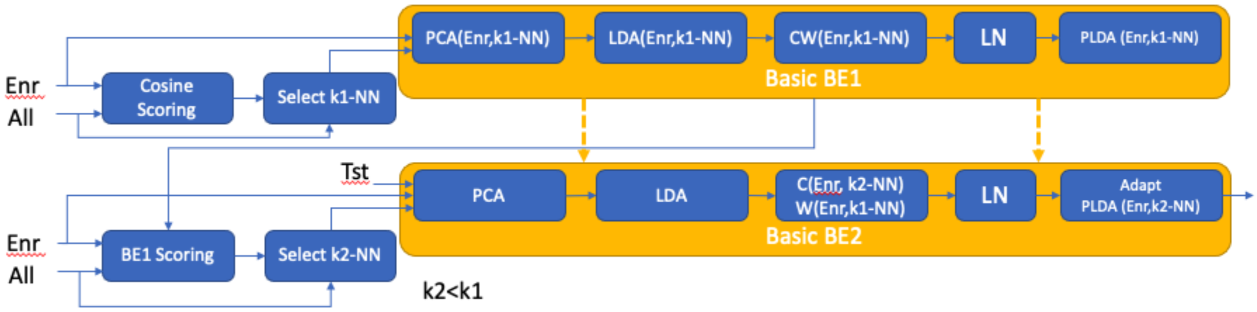

Figure 3: JHU kNN-PLDA-v3 back-end. *Enr* denotes the enrollment segments, *Tst* denotes the test segment, and *All* denotes the full training data.

5.2. Training procedure

All networks were first trained on 4 second chunks using an effective batch-size of 512. The real batch size depended on the GPU memory and network size. Gradient accumulation was used to achieve the desired effective batch size. The learning rate was set to 0.01 and kept constant for 40k model updates. Afterwards, it was divided by two every 10k steps until convergence. Later, the networks were fine-tuned using cyclic cosine learning rate scheduling on longer utterances (10-60 second chunks).

5.3. JHU Embeddings

Here, we summarize the networks included in our fusions. Unless indicated otherwise, all these networks were trained on MIXER-6, SRE04-18, SwitchBoard and VoxCeleb. Unless indicated otherwise, we used statistics pooling.

- **ResNet34**
- **TSE-ResNet34:** ResNet34 with temporal squeeze-excitation.
- **ResNet34-IN:** ResNet34 with instance normalization.
- **ResNet34-IN-chwise-att:** ResNet34 with instance normalization and channel-wise attentive statistics pooling.
- **Transformer**
- **EfficientNet-b4**

5.4. MITLL Embeddings

The MITLL system used speaker embeddings from a **ResNet34** network, which included mean and standard deviation pooling layers to form the extracted embedding.

6. Audio Back-ends

6.1. JHU PLDA-v4

The pipeline for this back-end included CORAL, LDA, centering, whitening, length normalization, generative Gaussian SPLDA and adaptive S-Norm score normalization, as shown in Figure 2. For this back-end, we considered NIST SRE04-12 as out-of-domain (OUT) data (mostly in English); and SRE16-18 and Fisher Spanish as in-domain (IN) data.

The CORAL step computes a rotation that adapts the OUT data to the target domain. Thus, we apply that rotation to the OUT data, while we left the IN data untouched. Next, we pool IN and adapted OUT data to train LDA and the Whitening step. Meanwhile, we compute different centering for OUT and IN data. The latter is the one used on the test data. Next, we apply length normalization.

PLDA is trained on IN+OUT length normalized embeddings. Following, PLDA is adapted to the IN domain data. For PLDA adaptation, the within-class and across-class covariances of the adapted model were a weighted sum of the out-of-domain $\mathbf{S}_{\mathrm{out}}$ and in-domain $\mathbf{S}_{\mathrm{in}}$ covariances,

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

where we set $\alpha = 0.75$

After PLDA scoring, we applied Adaptive S-Norm using all IN+OUT data as cohort. We used the top 500 cohort segments to compute the normalization parameters for each trial.

6.2. JHU kNN-PLDA-v3

The idea of this back-end consists of training a back-end model adapted to each trial. The motivation is that we do not know the number of domains in our eval data and, also, we do not know if all of those domains match any of the domains in our training and adaptation data. Thus, a PLDA mixture may not

Table 1: *Data Used in the MITLL Back-end*

Data Set	CORAL	LDA	Center/ Whiten	PLDA	PLDA Adapt.	S-Norm
SRE04-10				✓		
SRE16 Eval	✓	✓	✓		✓	✓
SRE18 Eval	✓	✓	✓		✓	✓
LRE	✓		✓		✓	✓

Table 2: *Results of Individual systems on SRE16-19 dev sets and SRE20 progress set*

System		SRE19 Eval			SRE16 YUE40%			SRE16 TGL40%			SRE16-9 AVG			SRE20 Prog.		
Embed.	BE	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp
ResNet34-MITLL	MITLL-1mix	3.22	0.179	0.198	2.09	0.133	0.137	5.99	0.372	0.379	3.76	0.228	0.238	2.84	0.120	0.153
ResNet34-MITLL	MITLL-8mix	3.19	0.164	0.206	1.85	0.114	0.115	5.39	0.320	0.363	3.47	0.199	0.228	2.64	0.108	0.145
ResNet34	JHU-v4-SNorm	3.38	0.179	0.22	2.15	0.131	0.134	5.71	0.335	0.361	3.747	0.215	0.238	3.65	0.137	0.161
	JHU-kNN-v3	2.92	0.145	0.153	1.73	0.108	0.11	5.92	0.308	0.36	3.523	0.187	0.208			
ResNet34-IN	JHU-v4-SNorm	3.9	0.194	0.267	1.92	0.126	0.129	5.38	0.338	0.394	3.733	0.219	0.263	3.2	0.144	0.174
	JHU-kNN-v3	2.78	0.14	0.148	1.54	0.093	0.096	5.21	0.277	0.289	3.177	0.170	0.178	**2.52**	0.109	0.113
Transformer	JHU-v4-SNorm	3.55	0.194	0.243	2.41	0.162	0.166	6.55	0.377	0.392	4.170	0.244	0.267	3.25	0.134	0.168
	JHU-kNN-v3	3.08	0.163	0.179	1.96	0.128	0.129	7.51	0.373	0.423	4.183	0.221	0.244			
TSE-ResNet34	JHU-kNN-v3	3.82	0.176	0.184	2.27	0.156	0.156	7.02	0.388	0.408	4.370	0.240	0.249			
EfficientNet-b4	JHU-kNN-v3	2.72	0.136	0.156	1.76	0.1	0.101	5.05	0.279	0.287	3.177	0.172	0.181	2.81	0.12	0.129
ResNet34-IN-chwise-att	JHU-kNN-v3	**2.69**	**0.134**	**0.142**	**1.5**	**0.089**	**0.09**	**4.37**	**0.246**	**0.278**	**2.853**	**0.156**	**0.170**	2.71	**0.108**	**0.11**

work since the eval data may not match any of the components of the mixture.

We simplify the problem by assuming that enrollment and test segments belong to the same domain, as indicated in the eval plan. The method consists of training a back-end (including PCA/LDA/centering/whitening/PLDA) model using the k Nearest training speakers to the enrollment segments (1 or 3) of the trial. The enrollment segments are also included in the back-end training. Hence, even if the trial's domain is not included in the training, the corresponding back-end can be trained using the closest speakers from multiple domains.

We also think that this method can benefit from domain adaptation. The number of in-domain neighbors may be too small to train PLDA. Instead, we can train the back-end on a larger number of speaker neighbors k_1 and adapt to a smallest (closest) number of speakers neighbors k_2.

The procedure is depicted in Figure 3. For each enrollment side, we use cosine scoring to find the k_1 closest training speakers. Then, we pool the enrollment segments and all the recordings from those k_1 speakers and we train a Basic back-end (PCA/LDA/centering/whitening/LNorm), denoted as *BE1*. Then, we score again the enrollment model versus the training speakers, but this time using *BE1* back-end, to find a refined set of in-domain speakers $k_2 < k_1$. Then, we use those speakers to adapt *BE1*'s centering and PLDA and produce *BE2*. In this manner, we train a back-end for each enrollment model, and use that back-end for all the trials that involve that model.

This back-end does not require S-Norm.

6.3. MITLL-1mix

The MITLL-1mix system used a variety of data sets in the back-end. The scoring pipeline was comprised of CORAL feature mapping of the out-of-domain set. LDA dimension reduction to 200 was then applied, followed by global centering and whitening. An out-domain PLDA model was then trained, which was adapted to a partially unlabelled in-domain set. Finally, adaptive S-Norm was applied with a cohort size of 1000. Table 1 outlines the data sets utilized for each component of the back-end scoring system. In order to leverage the large unlabelled **LRE** set during PLDA scoring, semi-supervised adaptation [13] was used to adapt the out-of-domain model to this set, along with the labelled **SRE16 Eval** and **SRE18 Eval** sets.

6.4. MITLL-8mix

The MITLL-8mix system extended the back-end scoring system from Sec. 6.3 to include mixture modelling in the adapted PLDA model. The technique proposed in [13] was generalized to allow for a mixture of PLDA models to be trained with a partially unalabelled data set. In all other respects, the MITLL-8mix was identical to the MITLL-1mix system.

7. Audio Calibration and Fusion

7.1. JHU single system calibration

The JHU systems conditioned score calibration on the number of enrollment cuts (i.e. 1c vs. 3c). A separate logistic regression mapping was trained for each these two conditions on the NIST SRE16 YUE40%, development set. We used a target prior $P_{\mathcal{T}} = 0.05$.

7.2. MIT single system calibration

The MITLL systems conditioned score calibration on gender and the number of enrollment cuts (i.e. 1c vs. 3c). A separate logistic regression mapping was trained for each of the 4 combinations of these attributes, using the **NIST SRE16 YUE40%** data set. The target prior $P_{\mathcal{T}} = 0.05$ was used for each. To obtain gender labels, a gender classifier based on linear discriminant analysis was trained on the **SRE04-10** set.

7.3. Fusion

To select the best fusion combination, we used a greedy fusion scheme as last year [14]. First, we calibrate all the systems and select the best one given the lowest actual cost. We fix that 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. The fusion was trained on SRE16+18 dev sets. Following, we recalibrated the fused scores on NIST SRE16 YUE.

Table 3: *Fusion summary.*

Submission	Date	Num. Sys.	Systems
v1.4-4	2020/10/29	4	ResNet34×MITLL-1mix + (ResNet34-IN + ResNet34+Transformer)×JHU-v4-SNorm
v1.8.1-4	2020/11/16	4	ResNet34×MITLL-1mix + (ResNet34-IN + ResNet34+TSE-ResNet34)×JHU-kNN-v3
v1.17-4	2020/12/18	4	ResNet34×MITLL-8mix + (ResNet34-IN + Transformer + EfficientNet-b4)×JHU-kNN-v3
v1.25-5	2021/04/13	5	ResNet34×MITLL-8mix + (ResNet34-IN + Transformer + EfficientNet-b4 + ResNet34-IN-chwise-att)×JHU-kNN-v3

Table 4: *Results of Submitted fusions on SRE16-19 dev sets and SRE20 progress/eval set*

System	SRE19 Eval			SRE16 YUE40%			SRE16 TGL40%			SRE16-9 AVG			SRE20 Prog.			SRE20 Eval.		
	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp	EER	Min Cp	Act Cp
v1.4-4	2.64	0.146	0.188	1.62	0.112	0.113	4.4	0.291	0.32	2.887	0.183	0.207	2.63	0.107	0.127	**3.16**	0.088	0.097
v1.8.1-4	2.39	0.125	0.137	1.33	0.084	0.084	4.55	0.258	0.26	2.757	0.156	0.160	2.4	0.1	0.108	3.2	0.087	0.09
v1.17-4	2.33	0.121	0.146	1.27	0.077	0.078	4.63	0.258	0.259	2.743	0.152	0.161	**2.28**	**0.089**	0.105	3.2	0.086	0.093
v1.25-5	**2.3**	**0.118**	**0.139**	**1.23**	**0.075**	**0.075**	**4.31**	**0.242**	**0.243**	**2.613**	**0.145**	**0.152**	2.33	0.092	**0.101**	3.19	**0.083**	**0.087**

Table 5: *Computational resources x-vectors. Real-time factor is measured as processing time divided between utterance duration (lower is better). Memory is computed to process 10 seconds of speech.*

System	Real time factor (T_{proc}/T_{dur})	Memory (10secs) (GB)
ResNet34	0.0048	1
ResNet34-IN	0.0048	1
TSE-ResNet34	0.0050	1
Transformer	0.0063	1
EfficientNet-b4	0.008	2

8. Single Systems

Table 2 summarizes the results for the single systems that were part of our fusions. We can see that ResNet34-IN is highly competitive performing better than larget architectures like TSE-ResNet34 or EfficientNet. Adding channel wise attention pooling slightly improves the results. Regarding back-end, we observe that MITLL-mix8 outperforms MITLL-mix1 back-end. Also, the kNN PLDA back-end greatly improves over the single PLDA adapted to non-English data.

9. Submissions

Table 3 summarizes the fusions that appear on the SRE20 Eval leader-board. Table 4 shows the results on the dev, progress and eval set. We observe a significant improvement from our first to our last submission on our dev and SRE20 progress sets. However, the improvement on the SRE20 Eval was less significant.

10. Computation resources

Processing times for x-vectors were measured in a GPU GeForce RTX 2080 Ti. Most of the processing time is dedicated to the embedding extraction. MFCC, VAD and back-end processing time are negligible by comparison. For Audio Embeddings the GPU memory used depends on how many frames (chunk-size) are processed in parallel to compute the hidden features before the x-vector pooling layer. We tune the chunk-size for full memory utilization in a 11 GB GPU. The table shows the memory required to process 10 secs of speech in parallel.

11. 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 : Robust 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] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck, “Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification,” in *Interspeech2020*, 2020, pp. 1–5.

[4] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4685–4694.

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

[6] Hossein Zeinali, Shuai Wang, Anna Silnova, Pavel Matejka, and Oldrich Plchot, “But system description to voxceleb speaker recognition challenge 2019,” in *VoxSRC Challenge workshop*, 2019.

[7] J. Hu, L. Shen, and G. Sun, “Squeeze-and-Excitation Networks,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[8] Saurabh Kataria, Phani Sankar Nidadavolu, Jesús Villalba, Nanxin Chen, Leibny Paola Garcia-Perera, and Najim Dehak, “Feature Enhancement with Deep Feature Losses for Speaker Verification,” in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, may 2020, pp. 7584–7588, IEEE.

[9] Magdalena Rybicka, Jesús Villalba, Piotr Żelasko, Najim Dehak, and Konrad Kowalczyk, “Spine2Net: SpineNet with Res2Net and Time-Squeeze-and-Excitation Blocks

for Speaker Recognition," in *Interspeech 2021*, Brno, Czech Republic, aug 2021, pp. 496–500, ISCA.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[11] Iz Beltagy, Matthew E Peters, and Arman Cohan, "Longformer: The long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020.

[12] Mingxing Tan and Quoc Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, Kamalika Chaudhuri and Ruslan Salakhutdinov, Eds. 09–15 Jun 2019, vol. 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114, PMLR.

[13] Bengt J Borgström and Pedro Torres-Carrasquillo, "Bayesian estimation of plda with noisy training labels, with applications to speaker verification," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7594–7598.

[14] Jesús Villalba, Nanxin Chen, David Snyder, Daniel Garcia-Romero, Alan McCree, Gregory Sell, Jonas Borgstrom, Fred Richardson, Suwon Shon, Francois Grondin, Reda Dehak, Leibny Paola Garcia-Perera, Daniel Povey, Pedro Torres-Carrasquillo, Sanjeev Khudanpur, and Najim Dehak, "State-of-the-art Speaker Recognition for Telephone and Video Speech: the JHU-MIT Submission for NIST SRE18," in *Proceedings of the 20th Annual Conference of the International Speech Communication Association, INTERSPEECH 2019*, Graz, Austria, sep 2019.