The JHU System Description for SDSV2020 Challenge

Jesús Villalba, Najim Dehak

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

{jvillal7,ndehak3}@jhu.edu

Abstract

This document describes the systems submitted by the Johns Hopkins University team to the Short duration speaker verification challenge 2020. All our systems were based on different flavours of x-vectors. x-Vectors networks were pre-trained on VoxCeleb2. Then, they were fine-tuned on the training sets of the corresponding task. A few speakers from the training sets where held out to create dev. trials and measure EER/DCF. For fine-tuning and PLDA in text-dependent task1, we considered the pair speaker-phrase, a class. For text independent task2, the classes were just speakers. x-Vector fine-tuning significantly improved the results. The best systems were ResNet34 x-vectors with squeeze-excitation blocks.

Index Terms: speaker verification, x-vectors, adversarial

1. Introduction

This document describes the systems submitted by the Johns Hopkins University team (team 10) to the Short duration speaker verification challenge 2020 [1]. All our systems were based on different flavours of x-vectors [2]. We used x-vectors based on extended TDNN [3] and ResNet34 [4, 3, 5]. x-Vectors networks were pre-trained on VoxCeleb2. Then, they were fine-tuned on the training sets of the corresponding task. A few speakers from the training sets where held out to create dev. trials and measure EER/DCF. For fine-tuning and PLDA in text-dependent task1, we considered the pair speaker-phrase, a class. For text independent task2, the classes were just speakers.

2. x-Vectors

We used different x-vector versions with different encoders. All versions used mean+stddev pooling, 256 dim. embedding and additive angular softmax objective with $s = 30$ and $m = 0.3$.

2.1. Residual Extended TDNN

Residual Extended TDNN (ResETDNN) used the E-TDNN architecture similar to the ones in [6, 3, 5]. We network encoder consists of 5 E-TDNN blocks with dimension 512. Each block consists of one TDNN layer (1D dilated conv.) followed by a frame-wise full connected layer. In blocks 2 to 5, we added residual connections similar to ResNet [7].

2.2. ResNet34

The ResNet34 followed the configuration similar to [4]. We had a two ResNet34 with 64 to 512 channels in residual blocks as in [5], one was trained with augmentation (ResNet34) and the other without augmentation (ResNet34-no-aug). Also a Thin-ResNet34 with 16 to 128 channels trained with augmentation.

2.3. SE-ResNet34

Here, we added squeeze-excitation blocks (SE-block) [8] to the residual blocks. We have a SE-ResNet34 trained with augmentation. We used a reduction factor of 8 in the bottleneck layer of the SE-block.

2.4. T-SE-ResNet34

Here, we modified the SE-block to compute the SE embedding by averaging just in the time dimension (Standard SE averages in time and freq. dimensions). Thus we obtain an embedding of size $C \times F$ (C is number of channels of the layer, and F is the freq dimension of the layer)–instead of dimension C in Standard SE . Since, this embedding is much larger than in the standard case, we used 16 as reduction factor.

2.5. Training details

The acoustic features employed were 80 dimension log-Mel filter-banks with short-time mean normalization. We pre-trained all the networks in VoxCeleb2 dev+test [9]. augmented $6\times$ with noise from the MUSAN corpus[1] and impulse responses from the AIR dataset[2]. Margin we set to $m = 0$ in the first epoch and linearly increased to $m = 0.3$ in epoch 20. The network was trained on 4 sec. chunks. In each epoch, we used as many chunks as training utterances, and we trained for 70 epochs. We used Adam optimizer with learning rate 0.01 with exponential decay learning rate scheduling.

2.6. Fine-tuning details

The networks were fine-tuned on the training sets of task1 or task2. The fine-tuning data was augmented in the same manner as the training data. First we fine-tuned the last affine layer before embedding extractor and the output layer for a few epochs, while keeping the rest of the network frozen. We denote this by *ft-affine*. Then, we continued fine-tuning the full network. We denote this by *ft-full*. For task1, the fine-tuning classes were speaker+phrase, so we had $10\times$num-spks classes. For task2, the classes were just speakers. For fine-tuning, we randomly sampled chunks between 1 to 6 seconds to match the eval. durations. We used SGD optimizer with exponential decay learning rate scheduling.

3. Back-ends

3.1. Task 1

We used LDA to 200, centering, whitening, length normalization and simplified PLDA with 150 dim. speaker factors. It was trained only on our task1 adaptation set (without augmentation),

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

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

Table 1: *Results on Task1*

	Dev.								Prog.		Eval.	
Non-Targets	All		same-spk+diff-phr		diff-spk+same-phr		diff-spk-diff+phr		All		All	
System	EER	MinDCF	EER	MinDCF	EER	MinDCF	EER	MinDCF	EER	MinDCF	EER	MinDCF
ResNet34	1.13	0.096	21.28	0.914	1.35	0.071	0.64	0.033	10.52	0.698	-	-
ResNet34+ft-affine	0.49	0.029	2.40	0.178	1.31	0.066	0.05	0.002	2.89	0.144	-	-
ResNet34+ft-full	0.33	0.019	0.73	0.063	0.92	0.047	0.04	0.001	2.00	0.088	-	-
ResNet34-no-aug+ft-full	0.30	0.017	0.63	0.051	**0.86**	0.045	**0.01**	**0.000**	-	-	-	-
ThinResNet34+ft-full	0.39	0.022	0.69	0.060	1.12	0.052	0.02	0.001	-	-	-	-
ResETDNN+ft-full	0.61	0.035	0.69	0.061	1.76	0.080	0.03	0.001	-	-	-	-
SE-ResNet34+ft-full	**0.31**	**0.018**	**0.58**	**0.046**	**0.86**	**0.044**	0.03	0.001	**1.94**	**0.085**	**1.99**	**0.086**
TSE-ResNet34+ft-full	0.33	**0.018**	0.61	0.054	0.89	0.045	0.03	0.001	-	-	-	-
(ResNet34+ResETDNN)+ft-full	0.33	0.020	0.68	0.061	0.95	0.042	**0.01**	0.001	1.84	0.078	-	-
(ResNet34+ResNet34-no-aug+ThinResNet34+ResETDNN)+ft-full	0.26	0.015	0.59	0.051	0.77	0.034	**0.01**	**0.000**	1.69	0.069	-	-
(ResNet34+ResNet34-no-aug+ThinResNet34+SE-ResNet34)+ft-full	0.25	**0.013**	**0.50**	**0.047**	0.65	**0.032**	**0.01**	**0.000**	1.62	0.066	-	-
(ResNet34+ResNet34-no-aug+SE-ResNet34+TSE-ResNet34)+ft-full	**0.23**	**0.013**	0.51	**0.047**	**0.64**	**0.032**	**0.01**	0.001	**1.57**	**0.064**	**1.60**	**0.065**

Table 2: *Results on Task2*

	Dev.		Prog.		Eval.	
System	EER	MinDCF	EER	MinDCF	EER	MinDCF
ResNet34+PLDA	2.28	0.083	5.04	0.219	-	-
ResNet34+ft-affine+PLDA	1.88	0.064	4.36	0.191	-	-
ResNet34+ft-full+PLDA	1.60	0.058	3.72	0.169	-	-
ResNet34+ft-full+cos	1.40	0.057	2.96	0.132	-	-
ResNet34-no-aug+ft-full+cos	1.43	0.058	-	-	-	-
ThinResNet34+ft-full+cos	1.76	0.067	-	-	-	-
ResETDNN+ft-full+cos+cos	2.38	0.096	-	-	-	-
SE-ResNet34+ft-full+cos	1.35	0.055	-	-	-	-
TSE-ResNet34+ft-full+cos	**1.28**	**0.050**	**2.64**	**0.116**	**2.62**	**0.116**
(ResNet34+ResETDNN)+ft-full+cos	1.50	0.060	3.28	0.141	-	-
(ResNet34+ResNet34-no-aug)+ft-full+cos	1.36	0.055	2.65	0.119	-	-
(ResNet34+ResNet34-no-aug+SE-ResNet34)+ft-full+cos	1.32	0.052	2.39	0.108	-	-
(ResNet34+SE-ResNet34+T-SE-ResNet34)+ft-full+cos	1.27	**0.050**	2.33	0.105	-	-
(ResNet34+ResNet34-no-aug+SE-ResNet34+TSE-ResNet34)+ft-full+cos	**1.26**	**0.050**	**2.32**	**0.104**	**2.31**	**0.105**

using speaker+phrase as classes. No Voxceleb data was used to train the back-end. We fused the score of different systems using linear logistic regression trained on our task1 dev set, but removing the diff-spk+diff-phr, which are too easy.

3.2. Task 2

We used either LDA+LN+PLDA back-end or cosine scoring. With fine-tuned networks cosine was better. LDA/PLDA were trained on our task2 adaptation set (without augmentation). We fused the score of different systems using linear logistic regression trained on our task2 dev set.

4. Adaptation/Development Data

4.1. Task 1

From the task1 training set, we selected 814 speakers for x-vector/PLDA training and 150 speakers to create dev trials. The new training/adaptation set contained 87k utterances, which became 524k after augmentation. The dev. enrollment set contained 3339 models, each one made out of 3 utterances. The dev. test set contained 3033 utterances producing around 10M trials. The were 11,421 target trials, 93,971 same-spk+diff-phrase non-targets, 1,032,072 diff-spk+same-phrase non-targets, and 8,989,723 diff-spk-phrase non-targets.

4.2. Task 2

From the task2 training set, we selected 500 speakers for x-vector/PLDA training and 88 speakers to create dev trials. The new training/adaptation set contained 73k utterances, which became 439k after augmentation. The dev. enrollment set contained 1349 models. Each model was made with between 2 to 14 utterances. The dev. test set contained 1338 utterances producing around 1.8M trials.

5. Results

5.1. Task1

Table 1 shows results for task1. First line shows results with pre-trained VoxCeleb x-vector (ResNet34). It has high error in same-spk non-targets since x-vector was trained for text-independent task. This error was greatly reduced by just fine-tuning the last affine transform in the x-vector output using the task2 adaptation set with spk-phrase labels (ResNet34+ft-affine). Then, the results were improved further by going on fine-tuning the full network (ResNet34+ft-full). This was confirmed by the results in the progress set. We kept this setup for the rest of neural networks evaluated. ResNet34-no-aug, pre-trained without augmentation, performed similar to ResNet34 pre-trained with augmentation. ThinResNet34 and ResETDNN performed significantly worse than the others. ResNet with SE blocks performed the best on our dev. Our best fusion was a combination of ResNet34 networks. ThinResNet34 and ResETDNN performed significantly worse than the others. TSE-ResNet34 performed the best on our dev. Again, our best fusion was a combination of ResNet34 x-vectors.

5.2. Task2

Table 1 shows results for task2. Here, fine-tuning the full network also performed better than the pre-trained network and finituning the last x-vector layer. Also, using fine-tuned network, cosine scoring performed better than PLDA, since AAM-softmax optimize a metric for cosine scoring. We kept this configuration for the rest of the networks.

6. Conclusions

We presented the JHU systems for SDSV 2020 challenge. We observed that using pre-trained x-vector networks and fine-tuning the full network on the training data of each task provided large improvements. On task1 (test-dep), we fine-tuned using spk-phrase as labels, while in task2 (text-ind), we used speakers as labels. For task1, PLDA was the best back-end, while for task2, cosine scoring was better. In our dev. set, the best single systems were squeeze-excitation ResNets. Our best fusions were combinations of ResNet34 systems.

7. References

[1] K. A. Zeinali, Hossein nad Lee, J. Alam, and L. Burget, "Short-duration speaker verification (sdsv) challenge 2020: the challenge evaluation plan." arXiv preprint arXiv:1912.06311, Tech. Rep., 2020.

[2] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep Neural Network Embeddings for Text-Independent Speaker Verification," in *INTERSPEECH 2017*, Stockholm, Sweden, aug 2017, pp. 999–1003.

[3] J. Villalba, N. Chen, D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, J. Borgstrom, L. P. García-Perera, F. Richardson, R. Dehak, P. A. Torres-Carrasquillo, and N. Dehak, "State-of-the-art speaker recognition with neural network embeddings in NIST SRE18 and Speakers in the Wild evaluations," *Computer Speech & Language*, vol. 60, p. 101026, mar 2020.

[4] H. Zeinali, S. Wang, A. Silnova, P. Matějka, and O. Plchot, "BUT System Description to VoxCeleb Speaker Recognition Challenge 2019," in *The VoxSRC Workhsop 2019*, oct 2019.

[5] J. Villalba, D. Garcia-Romero, N. Chen, G. Sell, J. Borgstrom, A. McCree, L. P. Garcia-Perera, S. Kataria, P. S. Nidadavolu, P. A. Torres-Carrasquillo, and N. Dehak, "Advances in Speaker Recognition for Telephone and Audio-Visual Data : the JHU-MIT Submission for NIST SRE19," in *Odyssey 2020*, Tokyo, Japan, 2020.

[6] J. Villalba, N. Chen, D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, J. Borgstrom, F. Richardson, S. Shon, F. Grondin, R. Dehak, L. P. Garcia-Perera, D. Povey, P. A. Torres-Carrasquillo, S. Khudanpur, and N. Dehak, "State-of-the-art Speaker Recognition for Telephone and Video Speech: the JHU-MIT Submission for NIST SRE18," in *INTERSPEECH 2019*, Graz, Austria, sep 2019.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," dec 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[8] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[9] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech and Language*, vol. 60, 2020.