The UPC Speaker Verification System Submitted to VoxCeleb Speaker Recognition Challenge 2020 (VoxSRC-20)

Umair Khan and Javier Hernando

TALP Research Center, Department of Signal Theory and Communications,
Universitat Politecnica de Catalunya Barcelona, Spain

{umair.khan, javier.hernando}@upc.edu

Abstract

This report describes the submission from Technical University of Catalonia (UPC) to the VoxCeleb Speaker Recognition Challenge (VoxSRC-20) at Interspeech 2020. The final submission is a combination of three systems. System-1 is an autoencoder based approach which tries to reconstruct similar i-vectors, whereas System-2 and -3 are Convolutional Neural Network (CNN) based siamese architectures. The siamese networks have two and three branches, respectively, where each branch is a CNN encoder. The double-branch siamese performs binary classification using cross entropy loss during training. Whereas, our triple-branch siamese is trained to learn speaker embeddings using triplet loss. We provide results of our systems on VoxCeleb-1 test, VoxSRC-20 validation and test sets.

Index Terms: VoxSRC-20, speaker verification

1. Introduction

This report explains the submission from Technical University of Catalonia (UPC) to the VoxCeleb Speaker Recognition Challenge (VoxSRC-20), which is held in conjunction with Interspeech 2020. The VoxSRC-20 is the second edition of the speaker recognition challenge held by VoxCeleb team. The challenge has four separate tracks, three of them are dedicated to speaker verification task and one to speaker diarization. The UPC system was submitted to Track 3 which is a closed self-supervised speaker verification task.

The submitted result is a combination of three systems. In system-1 we trained an autoencoder to reconstruct neighbor i-vectors, rather than the same training i-vectors. After the training, we extract speaker vectors for the testing i-vectors, which are referred to as autoencoder vectors or shortly ae-vectors [1]. In system-2 and -3 we trained two siamese networks [2], i.e., double-branch and triple-branch, which consist of two and three branches, respectively [3]. Each branch is composed of a Convolutional Neural Network (CNN) encoder, inspired by the VGG architecture [4, 5, 6]. The double-branch network is trained as a binary classifier by minimizing binary cross entropy loss. After training, we obtain decision scores for the speaker verification trials from the output of the network. The triple-branch network is trained by minimizing triplet loss [7, 8]. In the testing phase, we extract speaker embeddings for the test data using any branch of the triple-branch network, which are scored using cosine scoring.

The rest of the report is organized as follows. Section 2 explains the three systems in detail. Section 3 describes the experimental setup and results. Finally, conclusions are drawn in Section 4.
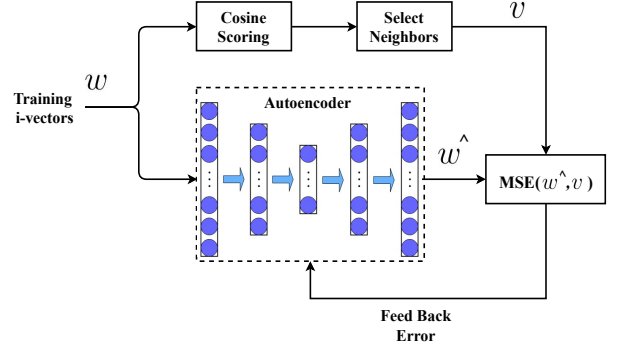

Figure 1: *Architecture of System-1: Autoencoder*

2. Proposed method

2.1. System-1: Nearest Neighbor Autoencoder

System-1 is an autoencoder which is trained by minimizing the loss function : $MSE(w\hat{}, v)$, as shown in Figure 1, where v is a similar i-vector to w and $w\hat{} = decoder(encoder(w))$. For every i-vector we select a set of similar i-vectors as neighbor i-vectors. All the training i-vectors are scored among each other using cosine scoring. We select the top k i-vectors with highest scores as neighbor i-vectors. In the testing phase, we transform the test i-vectors into new speaker vectors by extracting at the output of the autoencoder. These are referred to as autoencoder vectors or shortly ae-vectors. Details can be found in [1, 9].

2.2. System-2: Double-branch siamese network

Figure 2 shows the architecture of our double-branch siamese network. There are two identical branches, i.e., the CNN encoder. Mel-spectrogrm features of a training pair of an anchor along with a client or an impostor sample is fed into the network. The two branches share weights and biases with each other. The outputs of the two branches are concatenated which is followed by five Fully Connected (FC) layers. The last layer is connected to the binary class labels, i.e., 1/0, indicating if the anchor sample is paired by a client/impostor sample, respectively. During training, binary cross-entropy loss is minimized. A pair of reference and test utterances, which is involved in an experimental trial, is fed into the network. The decision scores are obtained directly from the output [3].

2.3. System-2: Triple-branch siamese network

A block diagram of our triple-branch siamese network is shown in Figure 3. As indicated by the name, there are three branches, each of which is the CNN encoder followed by l2-normalization
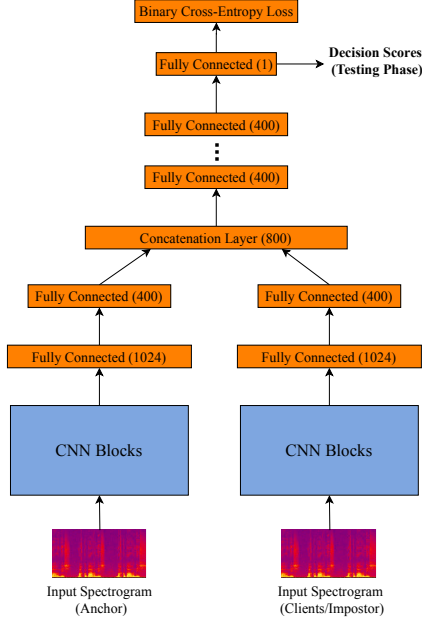
Figure 2: *Architecture of System-2: Double-branch siamese.*

Figure 3: *Architecture of System-3: Triple-branch siamese.*

Table 1: *Architecture of the VGG based CNN Encoder.*

Layer	Size	In dim	Out dim	Stride	Feat size
conv1-1	3x3	1	128	1x1	80xN
conv1-2	3x3	128	128	1x1	80xN
mpool-1	2x2	-	-	2x2	40xN/2
conv2-1	3x3	128	256	1x1	40xN/2
conv2-2	3x3	256	256	1x1	40xN/2
mpool-2	2x2	-	-	2x2	20xN/4
conv3-1	3x3	256	512	1x1	20xN/4
conv3-2	3x3	512	512	1x1	20xN/4
mpool-3	2x2	-	-	2x2	10xN/8
SAP	-	N/8	1	-	512x10
fc-1	-	1	1	-	1024
fc-2	-	1	1	-	400

layer. Each branch is fed by the Mel-spectrogrm features of a training pair of an anchor along with a client and an impostor sample. The CNN encoder encodes the Mel-spectrogram inputs into a vector based representation i.e., speaker embeddings. After the l2-normalization layer, triplet loss [7] is computed between the embeddings of anchor, client and impostor samples. Once the network is trained, we extract speaker embeddings using any CNN encoder branch of the network [3].

2.4. Selection of clients and impostor samples

Our siamese networks are trained using pairs of training samples, i.e., anchor, client and impostor. Since we do not use speaker labels, we generate the training pairs in a self-supervised manner [3]. The selection process of client and impostor samples is carried out in the i-vector space using two non-overlapping subsets of VoxCeleb-2 database, i.e., A and B. We assume that the speakers in one subset do not appear in the other. We extract i-vectors for all the utterances in both the subsets. Then, all the i-vectors in A are scored among each other using cosine scoring. For every i-vector in A we select k number of similar i-vectors as potential client i-vectors. The potential clients are subjected to a $threshold$ and final client i-vectors are obtained.

After this, we score all the i-vectors in A with those in B, using cosine scoring. For every i-vectors in A, we select k number of i-vectors from B according to the cosine scores. Since, we assume that the speakers in A does not appear in B, the selected i-vectors are assumed as the potential impostors, which are subjected to a threshold. In this way, every i-vector in A has been assigned k client and k impostor i-vectors. For the double-branch network we make training pairs of two, i.e., [anchor, client] and [anchor, impostor], for which the binary labels are 1 and 0 respectively. Whereas for the triple-branch network we make tuples of three samples, i.e., [anchor, client, impostor]. This process is carried out in i-vector space while the actual inputs to the networks are Mel-spectrogram features [3].
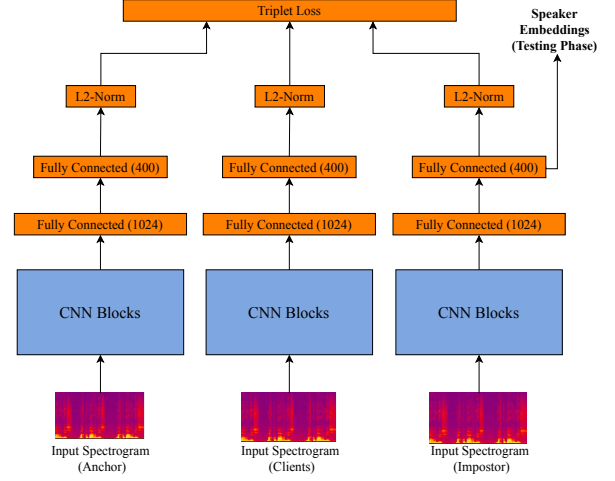
2.5. CNN encoder

The CNN encoder is inspired by the VGG architecture [4], recently adapted for speaker verification in [5, 6, 3]. It consists of three blocks, each containing two convolutional and one max-pooling layer as shown in Table 1. The blocks are followed by self attention pooling (SAP) layer [10], and two FC layers.

3. Experiments and results

The training was performed on the development partition of VoxCeleb-2 database [6]. The evaluation was performed on the test partition of VoxCeleb-1 [5], and the VoxSRC-20 validation and test sets. The autoencoder of System-1, is a fully connected feed forward network which consists of 3 hidden layers. The encoder and decoder parts are symmetrical. The hidden layer 1 and 3 have 300 neurons each, while hidden layer 2 consists of 200 neurons. The input and output layers consist of 400 neurons each. The autoencoder training was carried out for 100 epochs using Stochastic Gradient Descent (SGD) optimizer. All the layers of the autoencoder used ReLU activation except the last layer which used linear activation. The learning rate was set to 0.01 with a decay of 0.0002 and the batch size was set to 100.

For the clients and impostor selection in Systems-2 and System-3, (since we participated in a closed track of the challenge) we split VoxCeleb-2 into two parts to generate subsets

Table 2: *EER in % and DCF obtained for best values of k, on VoxCeleb-1 test set.*

Approach	k	EER(%)	DCF
System-1: Autoencoder [1]	15	10.20	0.8066
System-2: Double-branch [3]	10	6.90	0.7681
System-3: Triple-branch [3]	10	6.95	0.6606
Fusion of the above	-	5.58	0.5452

Table 3: *EER in % and DCF on VoxCeleb-1 test, VoxSRC-20 validation and test sets, using fusion of the three systems.*

Approach	EER(%)	DCF
VoxCeleb-1 (test set)	5.58	0.5452
VoxSRC-20 (validation set)	14.30	0.6921
VoxSRC-20 (test set)	14.71	0.7514

A and B as discussed in Section 2.4. (For an open track, one could use two separate datasets, for instance, VoxCeleb-1 as A and VoxCeleb-2 as B). Mel-spectrograms of 80 dimensions were computed, of which a randomly selected window of length $N = 350$ was input to the networks. All the CNN and fully connected layers were activated using ReLU function, whereas the last layer of the double-branch network used sigmoid activation. The training was carried out using Adam optimizer. The fusion scores were obtained using the following expression:

$$Scores = ((S_1 \times \alpha) + (S_2 \times (1-\alpha)) \times \beta) + (S_3 \times (1-\beta)) \quad (1)$$

Where S_1, S_2 and S_3 are the individual scores obtained using System-1, 2 and 3, respectively. The values of α and β were tuned on the VoxSRC-20 validation set, and were set to 0.30 and 0.79, respectively.

Table 2 shows the EER in % and DCF obtained for the best values of k on VoxCeleb-1 test set (also reported in [1, 3]). Each row contains the results obtained using the individual systems. From the Table we can see that the double-branch siamese network has achieved the best EER of 6.90% which is almost similar to that of the triple-branch network. Moreover, if we perform a score level fusion of all the three systems, we obtain an EER and DCF of 5.58% and 0.5452, respectively.

Table 3 shows the EER and DCF obtained on VoxSRC-20 validation and test sets. The results were obtained using the fusion of the individual scores of the three systems. Our submission ranked 3rd in the VoxSRC-20 competition (Track 3).

4. Conclusions

This report describes the submission of Technical University of Catalonia (UPC) to the VoxCeleb Speaker Recognition Challenge (VoxSRC-20) at Interspeech 2020. The submitted result is a combination of three systems, i.e., autoencoder, double- and triple-branch siamese networks based on Convolutional Neural Network (CNN). Brief analysis of the proposed systems (individual and in fusion) is provided on VoxCeleb-1 test set. The fusion results for VoxSRC-20 validation and test sets have also been reported.

6. References

[1] U. Khan, M. India, and J. Hernando, "Auto-encoding nearest neighbor i-vectors for speaker verification," *Proc. Interspeech 2019*, pp. 4060–4064, 2019.

[2] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in neural information processing systems*, 1994, pp. 737–744.

[3] U. Khan and J. Hernando, "Unsupervised training of siamese networks for speaker verification," *Proc. Interspeech 2020*, 2020.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[5] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Interspeech*, 2017.

[6] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Interspeech*, 2018.

[7] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.

[8] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[9] U. Khan, M. India, and J. Hernando, "i-vector transformation using k-nearest neighbors for speaker verification," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7574–7578.

[10] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 74–81.