PolyU Submission to the Third DIHARD Challenge

Weiwei Lin
Dept. of Electronic and Information Engineering
The Hong Kong Polytechnic University
Hong Kong SAR
weiwei.lin@connect.polyu.hk

Man-Wai Mak
Dept. of Electronic and Information Engineering
The Hong Kong Polytechnic University
Hong Kong SAR
man.wai.mak@polyu.edu.hk

***Abstract*—This paper describes the system developed by HK PolyU for the Third DIHARD Speech Diarization challenge. Unlike the official baseline, which employs a very sophisticated pipeline including probabilistic linear discriminant analysis (PLDA) and variational Bayes hidden Markov model (VB-HMM) re-segmentation, our system relies entirely on the speaker embeddings obtained from a DenseNet. For each fixed-length speech segment, we computed the cosine distance scores between its speaker embedding and the speaker embeddings of the adjacent speech segments to identify the speaker turns. Then we used cosine distance again as a metric for agglomerative hierarchical clustering (AHC). Despite the straightforward approach, we produce competitive results.**

***Index Terms*—Speaker diarization, DIHARD, Speaker change detection, DenseNet, Speaker embedding**

I. Highlights

The most distinct feature of our system is the use of simple cosine metric in both speaker change detection and clustering. Thanks to the speaker change detection module, we are able to use a shorter segment than widely seen in the literature [1]. Unlike the unbounded PLDA scores [2], [3], cosine distance is bound between 0 and 2, which makes setting the thresholds for speaker change detection and speaker clustering easier.

II. Data Resources and Training Procedures

The training data include the VoxCeleb1 development set and the VoxCeleb2 development set [4], [5]. We followed the data augmentation strategy in the Kaldi SRE16 recipe. The training data were augmented by adding noise, music, reverb, and babble to the original speech files in the datasets. After filtering out the utterances shorter than 4 seconds and the speakers with less than 8 utterances, we are left with 7,302 speakers. We used the filter-bank features implemented in Kaldi. We used a frame length of 25ms. The number of mel-scale filters is 40, and the lower and upper cutoff frequencies covered by the triangular filters are 20Hz and 7,600Hz, respectively.

III. System Details

A. Speech Activity Detection

We used the pre-trained model[1] from Pyannote [6] for speaker activity detection (SAD). The network comprises two bi-directional long short-term memory (B-LSTM) layers and three fully-connected layers [7]. The network was trained with 2-second chunks from the single-channel subset of DIHARD [1].

B. DenseNet Architecture for Speaker Embedding

DenseNets were proposed in [8] for computer vision. A DenseNet comprises two block types, namely, dense block and transition block. In a dense block, each layer is connected by all the output from the previous layers. To prevent the number of feature maps from growing excessively, a transition block is introduced to reduce the feature map size. Suppose each convolutional layer produces k feature maps, then the l-th layer inside the block has $k_0 + k \times (l-1)$ feature maps, where k_0 is the number of channels in the input layer. The parameter k is referred to as the growth rate. In this work, we used a dense network composed of 1-dimensional convolution instead of 2D convolution [9], [10]. We used the same statistics pooling layer as that of the x-vector network. Because max-pooling and average pooling do not work well in speaker recognition, we replaced the max-pooling by stride 2 convolution layers. Table I shows our network architecture.

C. Additive Margin Softmax

Margin-based loss has been very successful in face recognition and speaker recognition [11]. Additive margin loss enforces a minimum margin m between the target class and non-target classes:

$$\begin{aligned}\mathcal{L}_{AMS} &= -\frac{1}{n}\sum_{i=1}^{n}\log\frac{e^{s\cdot\left(\cos\theta_{y_i}-m\right)}}{e^{s\cdot\left(\cos\theta_{y_i}-m\right)}+\sum_{j=1,j\neq y_i}^{c}e^{s\cdot\cos\theta_j}}\\ &= -\frac{1}{n}\sum_{i=1}^{n}\log\frac{e^{s\cdot\left(\mathbf{W}_{y_i}^{\intercal}\mathbf{x}_i-m\right)}}{e^{s\cdot\left(\mathbf{W}_{y_i}^{\intercal}\mathbf{x}_i-m\right)}+\sum_{j=1,j\neq y_i}^{c}e^{s\mathbf{W}_j^{\intercal}\mathbf{x}_i}},\end{aligned} \tag{1}$$

where $\mathbf{W}$ is a weight matrix ($\mathbf{W}_j$ is the j-th column of $\mathbf{W}$) and $\mathbf{x}$ is an embedding vector, both of which are normalized to have unit length. s is a scaling constant.

D. Speaker Change Detection

After obtaining the speech segments through SAD or using oracle SAD, we divided the segments into 1-second chunks without overlapping. Then, starting from the first chuck, we scored it with the next chunk using cosine similarity. The score was used to decide whether to merge the two segments. If the

[1] https://github.com/pyannote/pyannote-audio-hub/tree/master/models/sad_dihard.zip

Layers	Output Size	DenseNet-121
Convolution	400 × 40	conv 3
Dense Block (1)	400 × 80	$\begin{bmatrix}\text{conv 1}\\\text{conv 3}\end{bmatrix} \times 6$
Transition Layer (1)	200 × 320	conv 2 stride 2
Dense Block (2)	200 × 320	$\begin{bmatrix}\text{conv 1}\\\text{conv 3}\end{bmatrix} \times 12$
Transition Layer (2)	100 × 640	conv 2 stride 2
Dense Block (3)	100 × 640	$\begin{bmatrix}\text{conv 1}\\\text{conv 3}\end{bmatrix} \times 24$
Transition Layer (3)	50 × 1280	conv 2 stride 2
Dense Block (4)	50 × 1280	$\begin{bmatrix}\text{conv 1}\\\text{conv 3}\end{bmatrix} \times 16$
Stats-pooling Layer	50 × 2560	-
FC	1	2560 × 256 Linear
Classification Layer	1	256 × # of classes AM-Softmax

TABLE I

Densenet architecture for speaker embedding. The growth rate for the networks is 40. Note that each "conv" layer shown in the table corresponds to the sequence BN-ReLU-Conv.

Dataset	DER [%]	JER
Development	20.07	46.21
Evaluation	19.51	45.11

TABLE II

Performance of Our Track1 system on full set.

Dataset	DER [%]	JER
Development	21.09	50.84
Evaluation	21.53	51.26

TABLE III

Performance of our Track 1 system on core set.

Dataset	DER [%]	JER
Development	27.33	53.23
Evaluation	26.91	51.09

TABLE IV

Performance of our Track 2 system on full set.

Dataset	DER [%]	JER
Development	29.54	57.34
Evaluation	28.57	56.25

TABLE V

Performance of our Track 2 system on core set.

score is greater than 0.65, the two chunk were merged into a single chunk. The merged chunk was scored against the next chunk, and the procedure was repeated until the last chunk was reached. The goal of our speaker change detection is not to accurately identify all speaker changes as in [12], but to have a very low false positive rate.

E. Agglomerative Hierarchical Clustering

After obtaining the segments from speaker change detection, we clustered the segments using agglomerative hierarchical clustering (AHC). We chose Ward's method as linkage criterion to merge the clusters. We swept the threshold from 0 to 9 with an interval of 0.5 and chose the one that optimizes the performance on the development set.

F. Results

The performance of our system in Track 1 is presented in Table II and Table III. The performance of our system in Track 2 is presented in Table IV and Table V.

IV. Hardware Requirements

All experiments were run on a computer with two Intel Xeon Silver 10 core CPUs, 64GB RAM, and two Nvidia 2080TI graphic cards. The memory consumption and processing time are summarized in Table VI.

References

[1] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du, S. Ganapathy, and M. Liberman, "The second DIHARD diarization challenge: Dataset, task, and baselines," *arXiv preprint arXiv:1906.07839*, 2019.

[2] P. Singh, H. Vardhan, S. Ganapathy, and A. Kanagasundaram, "Leap diarization system for the second DIHARD challenge," *Proc. Interspeech 2019*, pp. 983–987, 2019.

[3] N. Ryant, P. Singh, V. Krishnamohan, R. Varma, K. Church, C. Cieri, J. Du, S. Ganapathy, and M. Liberman, "The third DIHARD diarization challenge," *arXiv preprint arXiv:2012.01477*, 2020.

[4] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A large-scale speaker identification dataset," *Proc. Interspeech 2017*, pp. 2616–2620, 2017.

[5] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *Proc. Interspeech 2018*, pp. 1086–1090, 2018.

[6] H. Bredin, R. Yin, J. M. Coria, G. Gelly, P. Korshunov, M. Lavechin, D. Fustes, H. Titeux, W. Bouaziz, and M.-P. Gill, "Pyannote. audio: Neural building blocks for speaker diarization," in *Proc. ICASSP*, 2020, pp. 7124–7128.

[7] M. Lavechin, M.-P. Gill, R. Bousbib, H. Bredin, and L. P. Garcia-Perera, "End-to-end Domain-Adversarial Voice Activity Detection," 2020. [Online]. Available: https://arxiv.org/abs/1910.10655

Task	Time (sec.)	Memory (Mb)
SAD	0.3	3.95
Extract embedding	0.63	94.50

TABLE VI

Processing time and memory consumption of the SAD and speaker embedding extractor.

[8] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4700–4708.

[9] W. Lin, M. W. Mak, and L. Yi, “Learning mixture representation for deep speaker embedding using attention,” in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, 2020, pp. 210–214.

[10] W. Lin, M.-W. Mak, N. Li, D. Su, and D. Yu, “A framework for adapting dnn speaker embedding across languages,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2810–2822, 2020.

[11] F. Wang, J. Cheng, W. Liu, and H. Liu, “Additive margin softmax for face verification,” *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, 2018.

[12] R. Yin, H. Bredin, and C. Barras, “Speaker change detection in broadcast TV using bidirectional long short-term memory networks,” *Proc. Interspeech 2017*, pp. 3827–3831, 2017.