End-to-End Speaker Diarization System for the Third DIHARD Challenge System Description

Tsun-Yat Leung
Fano Labs, Hong Kong
ty.leung@fano.ai

Lahiru Samarakoon
Fano Labs, Hong Kong
lahiru@fano.ai

***Abstract*—This work aims to improve the recently proposed self-attentive end-to-end diarization model with encoder-decoder based attractors (EDA-EEND) for the third DIHARD Challenge. We propose to (1) replace the transformer encoders with conformer encoders to capture local information; (2) use convolutional upsampling to increase result resolution; (3) incorporate the attention mechanism into the attractor calculation; (4) add the additive margin penalty to increase the robustness; (5) shuffle chunks in each recording to increase combinations. In DIHARD III track 2, our final system achieved 23.86% and 20.05% diarization error rate (DER) on core evaluation set and full evaluation set, respectively, while the strong DIHARD III conventional baseline achieved 27.34% and 25.36% DER.**

***Index Terms*—speaker diarization, end-to-end diarization, DIHARD**

I. Notable Highlights

Our system is based on the recently proposed end-to-end diarization system (EDA-EEND) [1]. We propose to (1) replace the transformer encoders with conformer encoders to capture local information; (2) use convolutional upsampling to increase result resolution; (3) incorporate the attention mechanism into the attractor calculation; (4) add the additive margin penalty to increase the robustness; (5) shuffle chunks in each recording to increase combinations. In DIHARD III track 2, our final system achieved 23.86% and 20.05% diarization error rate (DER) on core evaluation set and full evaluation set, respectively. We focus on track 2 because our end-to-end system is capable of implicit speech activity detection.

II. Data Resources

TABLE I
Training and validation datasets

Dataset	#Mixtures
Pretraining Training Set	
Librispeech Simulated (β=2, 2, 4, 6)	400,000
Pretraining Validation Set	
Librispeech Simulated (β=2, 2, 4, 6)	2000
Fine-Tuning Training Set	
VoxConverse Development	216
DIHARD III Development (Training)	203
DIHARD II Development Clinical	24
Fine-Tuning Validation Set	
DIHARD III Development (Validation)	51

The datasets used are summarized in Table I. We created 1, 2, 3 and 4-speaker simulated mixtures from the Librispeech using the scripts provided in [2] [1]. The simulated training set is created from the Librispeech 960 hour training set, and the simulated validation set is created from the Librispeech dev clean set. For the fine-tuning datasets, we split the DIHARD III development (LDC2020E12) into training and validation sets with a ratio of 80%:20% per domain. We also include the DIHARD II Clinical development set (LDC2019E31) and VoxConverse [3] into the fine-tuning Dataset [2]. Actually, the DIHARD II Clinical development set is a subset of DIHARD III development set, so it could be removed in the future experiments.

III. Detailed Description of algorithm

Our system is based on the recently proposed end-to-end diarization system (EDA-EEND) [1], and our modifications are described in the following subsections. Also, the procedure for getting the track 1 result and the training procedure are described below.

A. Conformer Encoder

Transformers are good at capturing global features of a sequence while lack in capturing fine-grained local features. To address this issue, conformer was proposed, which combines transformers and convolutions networks in a parameter efficient way [4]. Conformer achieved state-of-the-art performance of Librispeech speech recognition task. Therefore, we replace transformer blocks in our model with conformer blocks. In the our conformer setting, relative position encoding and Macaron style feed-forward modules are used. The convolution kernel size is 15 and the number of attention heads is 4. The number of hidden units in feed-forward module is 1024. In our study (Sec. IV), two different settings are explored. One has 4 layers of conformer blocks with 256 hidden units and another has 7 layers of conformer blocks with 128 hidden units.

B. Convolution Subsampling and Upsampling

The input features in the EDA-EEND [1] are subsampled by a factor of ten to lower the computation cost. This low resolution leads to a significant increase in DER when no collar is used in the evaluation. However, removal of subsampling is not an option due to the increased computational cost. To

[1] https://www.openslr.org/12
[2] https://www.robots.ox.ac.uk/~vgg/data/voxconverse/

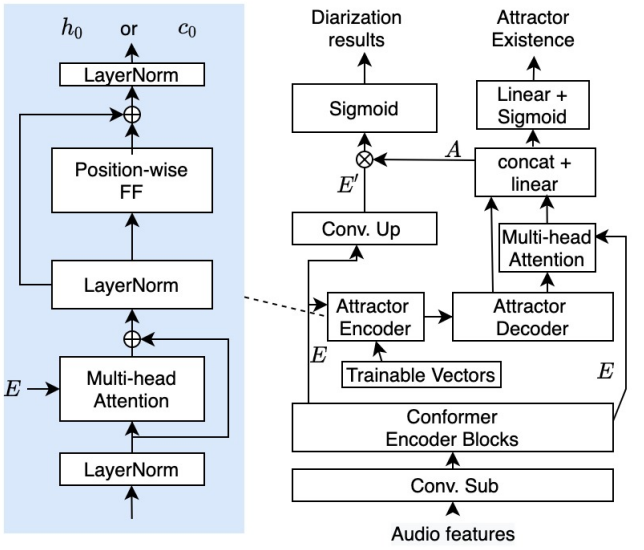

Fig. 1. Proposed system with conformer blocks, convolutional subsampling and upsampling, attractor with multiple attentions.

increase the result resolution without significant overheads, a convolution upsampling module, consisting of transposed convolutional layers, is placed after the conformer encoders to upsample features by a factor of ten. In the meantime, we use convolution subsampling instead of manual subsampling. So, each frame of feature is the 23-dimensional log-Mel filterbanks with a 25ms-frame length and 10-ms frame shift only. Finally, the diarization posterior probabilities are calculated by multiplying the upsampled embeddings $\mathbf{E}'$ with the attractor matrix $\mathbf{A}$ followed by a sigmoid non-linear transformation, as shown on the right of Figure 1.

The subsampling module consists of two convolutional layers. The first layer has 1D filters with a kernel size of 3 and a stride of 2, and the second layer has 1D filters with a kernel size of 5 and a stride of 5. The upsampling module consists of two transposed convolution layers with batch normalization and ReLU activation. The first layer has 1D filters with a kernel size of 3, a stride of 2 and a output padding of 1, and the second layer has 1D filters with a kernel size of 5 and a stride of 5. The number of output channels equals to the number of hidden units in the conformer encoder.

C. Attractor Calculation with Attentions

In EDA-EEND, the attractor LSTM encoder compresses all the essential information into two fixed dimension vectors, i.e. the hidden state $\mathbf{h_0}$ and cell state $\mathbf{c_0}$. However, diarization needs to process long sequences. Consequently, number of time steps in embeddings $\mathbf{E}$ can be considerably large. Therefore, the LSTM attractor encoder-decoder may not be optimal in handling long audio inputs [5]. Instead of using the last timestamp of the encoder outputs and cell states as the $\mathbf{h_0}$ and $\mathbf{c_0}$, two separate networks with multi-head attentions [6] are used as a pooling mechanism to initialize $\mathbf{h_0}$ and $\mathbf{c_0}$. This is illustrated on the left of Figure 1. In details, the number of hidden units in the attractor encoders equals to the number of hidden units in the conformer encoder, the number of attention heads is 4 and the number of hidden units in feed-forward module is 512.

In addition, inspired by the global attentional model in [7], a multi-head attention module [6] is added on the decoder outputs to allow retrieving information from the embeddings at each timestamps. The keys and values in the above attentions are the embeddings $\mathbf{E}$, and the queries in the attractor encoder attentions and the decoder attention are trainable vectors and the hidden state $\mathbf{h_t}$, respectively. The context vector $\hat{\mathbf{c}}_\mathbf{t}$ from the decoder attention is concatenated with the hidden state $\mathbf{h}_t$, and then projected by a linear layer to produce the attractor $\mathbf{a_t}$.

D. Additive Margin Penalty

Angular softmax shows its effectiveness in face recognition and speaker verification [8, 9] by introducing a margin penalty to the target class logit. Here we incorporate the additive margin idea [8] into the speaker diarization result calculation. First, attractors are normalized for the diarization loss calculation [3]. Then, the correct permutation of speaker labels is determined in the same way as [1] using the PIT with the normal diarization loss. Having the correct permutation, we know whom the attractors represent and we can add the margin penalty accordingly. The posterior probability $\hat{y}_{t,s}$ of speaker s at time t becomes:

$$\hat{y}_{t,s} = sigmoid(\gamma(\mathbf{e}_t\mathbf{a}_s - y_{t,s}m + (1 - y_{t,s})m)) \quad (1)$$

where $y_{t,s} \in \{0, 1\}$ is the label of speaker s at t, $\mathbf{e}_t$ is the embedding at time t, $\mathbf{a}_s$ is the attractor of speaker s, γ is the scale factor, m is the additive margin value. γ and m are the hyperparameters. Be noted that $\mathbf{e}_t$ is not normalized in this work. m is set to 0.35 and γ is set to 10.

E. Chunk Shuffling

Similar to [2], our model uses a 50 seconds audio from the original recording as a training sample. To increase the combinations of different audio segments, we divide the original recording into chunks of 10 seconds and shuffle with a probability of 0.5 when generating the training sample of 50 seconds.

F. Producing Track 1 result

End-to-end model is capable of handling the speech activity detection internally. In order to improve the performance of track 1 submission, the segmentation provided by the challenge is used to modify our system outputs. First, we removed the false alarm speeches from our system outputs according to the ground-truth segmentation. Then, the missed speeches in our system output are labeled with the speaker with the highest posterior probability.

[3]The attractor existence calculation still uses the unnormalized attractors.

G. Training Procedure

The training procedure is similar to that in [2]. Our models were pretrained with all Librispeech simulated mixtures together for 35 epochs except that our primary system was trained for 60 epochs. During fine-tuning, for each recording we sampled ten different audio of 50s. Then, they were fine-tuned for 400 epochs with a learning rate of $5e^{-5}$. The weighting parameter mentioned in the total loss in [2] was set to 0.1. In addition, the model was trained only to output four most dominant speakers. Similar to the augmentation procedure in [10], we used a 6-fold augmentation for DIHARD development set that combines the original DIHARD development set with five augmented copies. Each copy was augmented with one of the following:

- babble: randomly picked three to seven speaker from MUSAN and mixed them with the original audio (17-20dB SNR)
- music: randomly selected one or two music audio from MUSAN and added it to the original signal (5-15dB SNR) with repeating or trimming.
- noise: added MUSAN noises at one second intervals to the recording (0-15dB SNR).
- background noise: randomly selected one or two background noise audio from MUSAN and added it to the original signal (5-20dB SNR) with repeating or trimming.
- reverb: : artificially reverberated the recording via convolution with Room Impulse Response and Noise Database.

IV. Results

A. Experiments on Conformer and Resolution

TABLE II
TRACK 2 RESULT OF CONFORMER AND RESOLUTION EXPERIMENTS. "VAL" REFERS TO OUR FINE-TUNING VALIDATION SET.

Part	Conformer	Deep	Conv. Sub & Up	DER (%)		JER (%)	
				Val	Eval	Val	Eval
core	No	No	No	28.29	30.15	54.51	53.20
core	Yes	No	No	27.18	29.03	51.94	**50.86**
core	Yes	Yes	No	25.95	29.05	53.32	52.65
core	Yes	Yes	Yes	**25.06**	**27.90**	**51.85**	52.20
full	No	No	No	26.58	25.70	48.60	46.47
full	Yes	No	No	25.21	24.64	45.82	**44.38**
full	Yes	Yes	No	24.25	24.69	47.28	45.84
full	Yes	Yes	Yes	**22.48**	**23.05**	**45.25**	44.93

Table II reports the track 2 diarization result of using conformer encoders, convolution subsampling and upsampling, and a deeper network architecture with 7 layers of encoders and 128 hidden units. The above 3 modifications significantly improve the DER compared to the EDA-EEND. Therefore, we apply those changes to our systems for the later experiments.

B. Results on attractor with attentions and additive margin penalty

From Table III, it shows that the attractor with attentions improves the system on all sets of data. The system with that and additive margin penalty gives further improvement in the DER on the validation set and the JER on the evaluation set.

TABLE III
TRACK 2 RESULT OF ATTRACTOR WITH ATTENTIONS AND ADDITIVE MARGIN PENALTY.

Part	Attractor with Attention	Additive Margin Penalty	DER (%)		JER (%)	
			Val	Eval	Val	Eval
core	No	No	25.06	27.90	53.32	52.20
core	Yes	No	24.05	**26.08**	52.01	51.73
core	Yes	Yes	**22.66**	26.12	**51.43**	**50.87**
full	No	No	22.48	23.05	47.28	44.93
full	Yes	No	22.25	**21.65**	45.75	44.35
full	Yes	Yes	**21.07**	21.70	**45.17**	**43.86**

As a result, we continue our development on the system with both modifications despite that it performs slightly worse than the system without additive margin penalty in DER on the evaluation set. Be noted that the system with additive margin penalty could be sub-optimal because the pretrained model is not trained with additive margin penalty in this work.

C. Results with chunk shuffling and additional training

TABLE IV
TRACK 2 RESULT OF CHUNK SHUFFLING AND ADDITIONAL TRAINING. "DEV" REFERS TO ORIGINAL DIHARD III DEVELOPMENT SET, AND "LARGER EPOCH" MEANS THAT THE PRETRAINED MODEL IS PRETRAINED WITH MORE NUMBER OF EPOCHS.

Part	Chunk Shuffling	Larger Epoch	DER (%)			JER (%)		
			Val	Dev	Eval	Val	Dev	Eval
core	Yes	No	22.32	**18.33**	24.72	51.44	**42.33**	49.75
core	Yes	Yes	**21.85**	18.64	**23.86**	**50.61**	43.03	**48.85**
full	Yes	No	20.92	**16.36**	20.72	45.21	**36.69**	42.86
full	Yes	Yes	**20.04**	16.53	**20.05**	**44.01**	37.21	**42.06**

Table IV shows the results for the systems where they are fine-tuned with 500 epochs instead of 400 epochs. It can be clearly seen that chunk shuffling and pretraining the pretrained model for a longer time gives the best DER and JER on the validation set and the evaluation set. Our primary system uses chunk shuffling and is pretrained for 60 epochs.

D. Track 1 Result

TABLE V
TRACK 1 RESULT OF THE PRIMARY SYSTEM

Part	DER(%)		JER(%)	
	Dev	Eval	Dev	Eval
core	14.14	18.40	39.90	46.31
full	12.37	15.12	34.13	39.32

The result of track 1 submission is described in Table V.

V. Hardware requirements

The hardware details are shown below:

- Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz is used.
- Total available of RAM = 256 GB.
- One GeForce RTX 2080 is needed for training.
- Pytorch is used as the machine learning framework.

The system execution time for processing the entire development set is 1332 seconds with CPU inference.

REFERENCES

[1] S. Horiguchi, Y. Fujita, S. Watanabe, Y. Xue, and K. Nagamatsu, "End-to-end speaker diarization for an unknown number of speakers with encoder-decoder based attractors," *Proc. Interspeech 2020*, 2020.

[2] Y. Fujita, S. Watanabe, S. Horiguchi, Y. Xue, and K. Nagamatsu, "End-to-end neural diarization: Reformulating speaker diarization as simple multi-label classification," *arXiv preprint arXiv:2003.02966*, 2020.

[3] J. S. Chung, J. Huh, A. Nagrani, T. Afouras, and A. Zisserman, "Spot the conversation: speaker diarisation in the wild," *Proc. Interspeech 2020*, 2020.

[4] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *Proc. Interspeech 2020*, 2020.

[5] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 103–111. [Online]. Available: https://www.aclweb.org/anthology/W14-4012

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, pp. 5998–6008, 2017.

[7] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 1412–1421. [Online]. Available: https://www.aclweb.org/anthology/D15-1166

[8] 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.

[9] D. Garcia-Romero, D. Snyder, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "X-Vector DNN Refinement with Full-Length Recordings for Speaker Recognition," in *Proc. Interspeech 2019*, 2019, pp. 1493–1496. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2205

[10] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.