pyannote.audio speaker diarization pipeline at *VoxSRC 2022*

Hervé Bredin

IRIT, Université de Toulouse, CNRS, Toulouse, France

herve.bredin@irit.fr

Abstract

This technical report describes the submission of team *pyannote* to the VoxSRC 2022 speaker diarization challenge. It relies on 3 stages: neural speaker segmentation on a 5s sliding window, clustering of neural embedding of each speaker of each window, and final reconstruction. It reaches a diarization error rate (with forgiveness collars) of DER=5.6% on VoxSRC 2022 test set (DER=5.7% on VoxConverse 0.3 test set). In the spirit of reproducible research, the pipeline is readily available in *pyannote.audio* open source library [1].

Index Terms: speaker diarization, reproducible research

1. Description

Figure 1 depicts the manual speaker diarization of a 30s conversation between two speakers that we will use for illustration purposes.

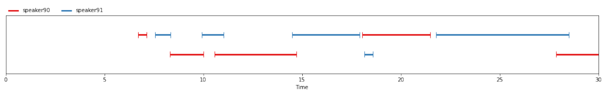

Figure 1: *Reference annotation of the excerpt used throughout this technical report.*

1.1. Neural speaker segmentation

The neural speaker segmentation model introduced in [2] is applied on a 5s sliding window using a 500ms step. Figure 2 illustrates the output of this stage on the excerpt whose manual annotation is depicted in Figure 1.

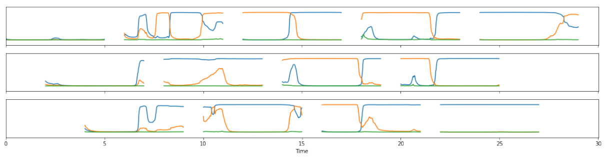

Figure 2: *Output of neural speaker segmentation stage. For each 5s window, the model outputs the probability of activity for up to 3 speakers (blue, orange, and green) using a 16ms temporal resolution. Since the speaker segmentation model has been trained in a permutation-invariant manner, the same speaker might be assigned a different color in two different windows. We use a step of 2s for readability but the actual practical step is 500ms.*

The output of the speaker segmentation model is further binarized using a single threshold $\theta_{\text{segmentation}}$ which is optimized to minimize diarization error rate on VoxConverse 0.3 test set. Figure 3 illustrates the output of this stage.

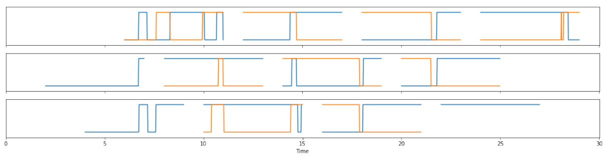

Figure 3: *Binary speaker segmentation. For each 5s window, only speakers whose probability goes above $\theta_{segmentation}$ at least once are kept.*

Two versions of the model were compared: 2022.07 and VoxSRC2022. The former has been trained on the union of AMI (train) [3], AISHELL-4 (train) [4], DIHARD3 (development) [5], REPERE (train) [6] and VoxConverse 0.3 (development) [7]. The latter has been further finetuned using VoxConverse 0.3 (development) only [7].

Table 1 reports the performance of the speaker diarization pipeline, assuming perfect clustering. Based on these numbers, we chose to use version VoxSRC2022 of the model in the rest of the pipeline.

Segmentation	DER%	FA%	MISS%	CONF%
2022.07	8.5	3.4	3.9	1.2
VoxSRC2022	7.9	3.4	3.3	1.2

Table 1: *Performance on VoxConverse 0.3 test set, assuming perfect clustering. DER stands for diarization error rate (without collar), FA for false alarm rate, MISS for missed detection rate, and CONF for speaker confusion rate.*

The binarization step depicted in Figure 3 also allows to compute the instantaneous number of speakers, which will prove very useful in the final reconstruction stage of the speaker diarization pipeline (Section 1.3). For each 16ms frame (the temporal resolution of the speaker segmentation model), this is achieved by averaging the number of active speakers in each overlapping windows: the result is illustrated in Figure 4.

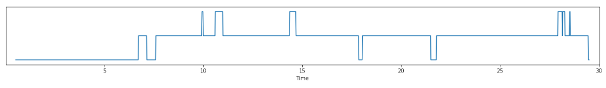

Figure 4: *Instantaneous speaker counting.*

1.2. Clustering with neural speaker embedding

We extract a single neural speaker embedding for each active speaker in each 5s window. For each speaker, we concatenate audio samples during which (1.) they are active and (2.) no other speaker is active ; and pass the resulting audio signal to the neural network in charge of computing speaker embeddings. This concatenation process is depicted in Figure 5.

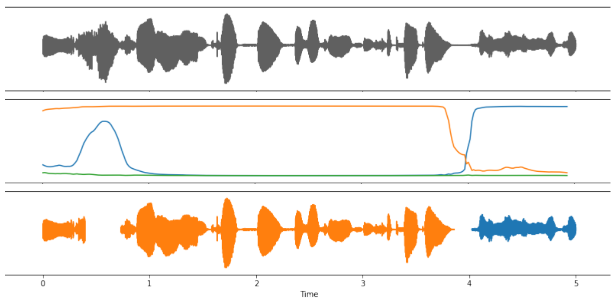

Figure 5: *Speaker embedding. Top row: 5s audio chunk. Middle row: two speakers are active according to the neural speaker segmentation model (the orange one and the blue one). Bottom row: the speaker embedding of the blue speaker is computed using only the blue audio signal, while the concatenation of orange audio signals is used to compute the speaker embedding of the orange speaker. No embedding is extracted for the green speaker as its probability never goes above $\theta_{segmentation}$ threshold.*

Once computed, speaker embeddings are clustered using standard agglomerative hierarchical clustering using a single distance threshold $\theta_{\text{embedding}}$ to decide when to stop merging clusters, and optimized to minimize diarization error rate on VoxConverse 0.3 test set.

A few publicly available speaker embedding models have been compared, listed in Table 2, with the corresponding performance of the resulting pipeline. Based on these numbers, we chose to use speechbrain/spkrec-ecapa-voxceleb [8, 9] in the final pipeline. After trying a bunch of agglomerative clustering algorithms, we eventually

1. switched from *average* linkage to *centroid* linkage (because it performed slightly better empirically)
2. adding a post-processing step that re-assigns small clusters (possible outliers) to the most similar large cluster.

Embedding	DER%	CONF%
pyannote [1]	14.9	8.1
TitaNet [10]	12.0	5.3
RawNet3 [11]	10.7	4.0
ECAPA-TDNN [8, 9]	10.6	3.9

Table 2: *Performance on VoxConverse 0.3 test set, with agglomerative hierarchical clustering and* average *linkage. DER stands for diarization error rate (without collar) and CONF for speaker confusion rate.*

Once speaker embeddings are clustered, we simply assign the original segmentation of each active speaker to the corresponding cluster (Figure 6).

1.3. Final reconstruction

Depicted in Figure 7, the final step aims at combining the instantaneous speaker counts from Section 1.1 and the clustering from Section 1.2 into an actual speaker diarization hypothesis. This pipeline reaches DER = 5.6% on VoxSRC 2022 test set

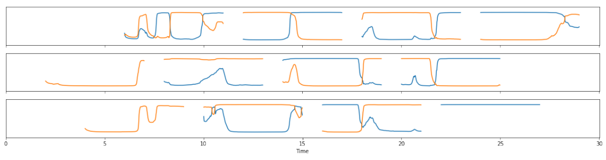

Figure 6: *Clustered active speakers. The same speaker is assigned the same color over different windows.*

and DER = 5.7% on VoxConverse 0.3 test set (both with 250ms forgiveness collars).

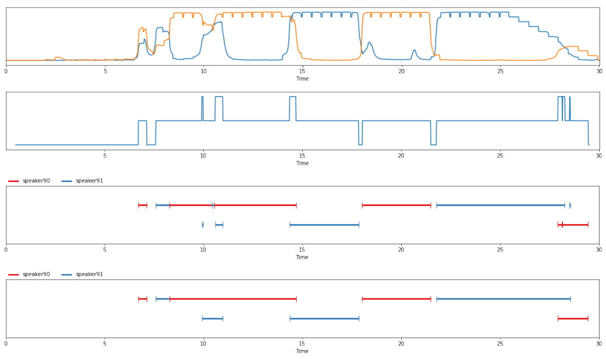

Figure 7: *Final reconstruction. Top row: for each (frame, cluster) pair, we compute the sum of probability over all overlapping 5s windows. Second row: instantaneous speaker counting (same as Figure 4). Third row: for each frame, we select the k clusters with highest sum of probability where k is given by the second row. Fourth row: we fill within-speaker gaps shorter than a few milliseconds.*

2. Reproducible research

The speaker diarization pipeline described in this technical report is readily available for the reader to try:

```
# install pyannote.audio
pip install https://github.com/pyannote/
    pyannote-audio/archive/VoxSRC2022.tar.gz

# load pipeline
from pyannote.audio import Pipeline
pipeline = Pipeline.from_pretrained(
   'pyannote/speaker-diarization@VoxSRC2022')

# apply pipeline
diarization = pipeline('audio.wav')
```

Expected RTTM output on both VoxSRC 2022 test set and VoxConverse 0.3 test set can be downloaded from this link. It takes approximately 67 minutes to process the whole VoxSRC 2022 test set (45 hours), using one Nvidia Tesla V100 SXM2 GPU (for the neural inference part) and one Intel Cascade Lake 6248 CPU (for the clustering/reconstruction part). In other words, processing is 40 times faster than real time.

3. About the VoxSRC 2022 challenge

3.1. Thank you!

I would like to thank the VoxSRC organizers for setting up such a great challenge. Like many others, our (speaker diarization) community struggles to come up with good and reliable ways to measure progress [1] and shared tasks like VoxSRC are definitely a step in this right direction. Thank you!

3.2. Share your RTTMs

Yet, reproducibility still seems out of reach, until we (the community) decide to share the full details of our systems (see Section 2). Though I do understand that it is not always possible to share the code or pretrained models, may I suggest that a condition to participate to future editions of VoxSRC would be to have all submitted RTTM files public right after the challenge deadline? This will not solve the reproducibility problem completely but will be another step in that direction [2].

3.3. I don't like collars!

Why, oh why, are we still using forgiveness collars to compare systems? As speaker diarization systems are getting better and better, it is getting more and more difficult to draw fair and reliable conclusions about which one is better than the other. Overlapping speech is probably one of the main sources of errors. However, using forgiveness collars tend to remove a large portion of overlapping speech from evaluation. I actually computed the numbers myself: the proportion of evaluated overlapping speech decreases from 2.96% (without collar) to 1.55% (with a 250ms collar) on VoxConverse 0.3 development set. That's a 48% relative decrease (while only 10% of *normal* speech is removed by those collars). Collars actually make the most difficult part of speaker diarization easier! May I suggest that we do not use collars in future editions?

4. Acknowledgements

This work was granted access to the HPC resources of IDRIS under the allocation AD011012177R1 made by GENCI.

5. References

[1] 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 *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain, May 2020.

[2] H. Bredin and A. Laurent, "End-to-end speaker segmentation for overlap-aware resegmentation," in *Proc. Interspeech 2021*, Brno, Czech Republic, August 2021.

[3] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal *et al.*, "The ami meetings corpus," in *Proceedings of the Measuring Behavior 2005 symposium on" Annotating and measuring Meeting Behavior*, 2005.

[4] Y. Fu, L. Cheng, S. Lv, Y. Jv, Y. Kong, Z. Chen, Y. Hu, L. Xie, J. Wu, H. Bu, X. Xu, J. Du, and J. Chen, "AISHELL-4: An Open Source Dataset for Speech Enhancement, Separation, Recognition and Speaker Diarization in Conference Scenario," in *Proc. Interspeech 2021*, 2021, pp. 3665–3669.

[5] 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.

[6] J. Kahn, O. Galibert, L. Quintard, M. Carré, A. Giraudel, and P. Joly, "A presentation of the repere challenge," in *2012 10th International Workshop on Content-Based Multimedia Indexing (CBMI)*, 2012, pp. 1–6.

[7] J. S. Chung, J. Huh, A. Nagrani, T. Afouras, and A. Zisserman, "Spot the Conversation: Speaker Diarisation in the Wild," in *Proc. Interspeech 2020*, 2020, pp. 299–303. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2337

[8] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: emphasized channel attention, propagation and aggregation in TDNN based speaker verification," in *Interspeech 2020*, H. Meng, B. Xu, and T. F. Zheng, Eds. ISCA, 2020, pp. 3830–3834.

[9] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio, "SpeechBrain: A general-purpose speech toolkit," 2021, arXiv:2106.04624.

[10] N. R. Koluguri, T. Park, and B. Ginsburg, "Titanet: Neural model for speaker representation with 1d depth-wise separable convolutions and global context," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 8102–8106.

[11] J.-w. Jung, Y. J. Kim, H.-S. Heo, B.-J. Lee, Y. Kwon, and J. S. Chung, "Pushing the limits of raw waveform speaker recognition," *Proc. Interspeech*, 2022.

[1] `https://github.com/BUTSpeechFIT/CALLHOME_sublists/issues/1`

[2] `http://blog.benjaminmarie.com/2/comparing-uncomparable.html`