INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

The SpeeD–ZevoTech submission at DISPLACE 2023

Gabriel Pîrlogeanu, Dan Oneață, Alexandru-Lucian Georgescu, Horia Cucu

Zevo Tech, Romania
Speech and Dialogue Research Laboratory
University Politehnica of Bucharest, Romania
gabriel.pirlogeanu@stud.etti.upb.ro

Abstract

This paper describes our team's collaborative efforts in participating in the Track1 of the Diarization of Speaker and Language in Conversation Environments (DISPLACE) Challenge 2023. Our submission focuses on speaker diarization in multilingual scenarios, dealing with overlapping speech segments with significant noise ratios. To achieve our goal, we fine-tuned the parameters of two speaker diarization toolkits, Pyannote and NeMo, and retrained some components using the DISPLACE development sets and subsets from the MUSAN speech database. The experiments show promising results, we managed to make improvements over the pretrained voice activity detection (VAD) model, as well as training the Multi-scale Speaker Diarization Decoder (MSDD) by using the DISPLACE development datasets. Best systems are combined using DOVER-Lap. Our approach achieves a diarization error rate (DER) of 28.97% on Phase 1 Eval set, compared to the baseline diarization error rate of 40%.
Index Terms: speaker diarization, DISPLACE challenge, NeMo, Pyannote, DOVER-Lap

1. Introduction

Speaker diarization is the process of identifying the different speakers in an audio stream. It is a crucial research topic in computational linguistics, with numerous challenges organized to improve the performance of state-of-the-art systems in complex scenarios. One such challenge, DISPLACE 2023 [1], is focused on conversational environments with multiple speakers who speak different languages, high levels of noise, reverberations, and overlapping speech. Other challenges in recent years, such as DiHARD [2], CHIME [3], Ego4D [4], Fearless Steps [5], Iberspeech RTV [6], and VoxSRC-20 [7], have tackled different aspects of the diarization problem, including audio-visual diarization and recordings performed with far-field microphones.

Conventionally, a diarization system comprises three main blocks: (i) a voice activity detector for speech segmentation, (ii) a speaker representations extractor and (iii) a clustering module. Voice activity detection can be achieved using several methods: time delay neural networks (TDNN) [3] or convolutional neural networks (CNN) combined with different mechanisms, such as self attention (CNN-SA) [8] . The feature extraction block is represented by a speaker embeddings extractor and it is popularly achieved through methods such as: x-vectors [9], TDNN modules [10] or neural networks. Lastly, in order to obtain speaker labels, similarities between speaker embeddings at different timestamps are computed using clustering algorithms such as: agglomerative hierarchical clustering [11] (AHC), k-means or spectral clustering [12].

In this paper we develop a strong baseline system using state of the art components: a voice activity detector based on MarbleNet architecture [13] and a speaker representation extractor based on TitaNet-L [14], a neural network with 1D depth-wise separable convolutions. We employ spectral clustering and implement the whole pipeline using NeMo toolkit [15].

A notable weakness of traditional clustering-only approaches is the fact that only one speaker can be assigned per speech timestamp, meaning that diarization is not overlap-aware. End-to-end solutions try to solve this problem and provide overlap-aware diarization, being designed similarly to a multilabel classifier [16], using self-attention [17] or conformer-transformer based architectures [18].

In our work, we employ a solution that allows for overlapped detections by incorporating an additional neural diarizer module on top of the traditional diarization pipeline. This Multi-scale Speaker Diarization with Dynamic Scale Weighting (MSDD) network [19] uses a dynamic scale weighting approach that can adaptively select the best time scales to represent speaker embeddings. To enhance the performance of speaker diarization, a common approach is to fuse the predictions of multiple diarization systems using late fusion methods like DOVER [20] or DOVER-Lap [21]. Following this approach, we incorporate the segmentation-based diarization approach from Pyannote [22, 23] as a complementary system to our baseline approach. We use Doverlap to integrate the two systems. By doing this, we aim to improve the robustness and accuracy of our diarization system.

The rest of the paper is structured as follows. Section 2 describes our methodology and the used data resources, Section 3 presents the experimental results and, finally, Section 4 lists the conclusions.

2. Methodology

2.1. Data resources

The DISPLACE speech dataset was made available by the organizers of the challenge. We used the entire development set for fine-tuning or training specific parts of our system. For the internal ablation studies, we trained our system on the second and third parts of the development set: `dev2` (7h, 6m) and `dev3` (6h, 33m), respectively. We then evaluated our system's performance on the first part, `dev1` (2h, 5m).

MUSAN [24] is a freely available dataset containing audio partitioned into speech, music and noise. We used a `MUSAN-freesound-background` split (3h, 37m) for non-speech data and a `MUSAN-freesound-noise` split (1h, 15m) to perform noise augmentation when training the voice activity detection system.

10.21437/Interspeech.2023-1839

2.2. Diarization with NeMo

In our experiments, we fine-tuned the pretrained MarbleNet multilingual VAD model and used the pretrained TitaNet-L for speaker embeddings extraction. For clustering, we used the multi-scale spectral clustering module [25] based on Normalized Maximum Eigengap (NME) values to estimate the number of clusters automatically. Finally, we trained from scratch the MSDD neural diarizer on the DISPLACE development in order to obtain overlap-aware diarization.

2.2.1. Voice activity detection: Fine-tuning

In order to improve the performance of the system, we fine-tuned the pretrained VAD Multilingual MarbleNet [13] model on the DISPLACE dataset. We follow the setting of Jia *et al.* [13] and split the audio data into speech segments using a window size of 0.63s, a step of 0.2s and an offset of 0.2s; the offset means that we discard the first 0.2s of each speech segment (this step is done to avoid the boundaries between speech and non-speech segments, which are prone to being mislabeled). The background audios were also split in 0.63s segments, but with a 0.05s step. The train–validation ratio was 80-to-20.

The audio files in the DISPLACE dataset contain mostly speech, resulting in a highly imbalanced dataset. To balance the two classes, we added audios from `MUSAN-freesound-background` as non-speech. The combination of `MUSAN-freesound-background` and `dev2+3` dataset resulted in 150K speech segments and 154K background segments for training. The validation set consists of 37K speech segments and 42K background segments.

For data preprocessing, most of the techniques presented in [13] were kept, with the exception of the input features: we have used log-mel spectograms instead of cepstral coefficients. We have also performed data augmentation using SpecAugment [26] and SpecCutout [27], as well as adding noise from the `MUSAN-freesound-noise` dataset to the speech segments.

We have trained the model for 250 epochs with a batch size of 256, using the SGD optimizer with a learning rate of 10^{-2}, weight decay of 10^{-3}, momentum of 0.9 and a second-order polynomial hold decay annealing scheduler [28].

2.2.2. Voice activity detection: Hyper-parameter tuning

The VAD module assigns a probability of speech presence to each audio frame, which is then binarized into discrete speech or non-speech decisions based on a set of threshold hyper-parameters. In order to fine-tune the VAD module's hyper-parameters, we started from the meeting configuration provided by NeMo for speaker diarization tasks with three to five speakers. We conducted a grid-search in order to find the four main hyper-parameters (`min_duration_off`, `min_duration_on`, `offset` threshold, `onset` threshold) based on the DetER obtained by each setup on the development set, while the rest of hyper-parameters were kept fixed (`overlap_ratio` set to 0.5, `pad_onset` and `pad_offset` set to 0, `filter_speech` set to FALSE).

The `min_duration_off` and `min_duration_on` hyper-parameters refer to the minimum duration that a non-speech or speech segment must have in order to not be removed. The `onset` parameter represents a threshold for detecting the beginning of speech, while the `offset` parameter represents a threshold for detecting the end of a speech segment.

Hyper-parameter tuning on the pretrained VAD model. Figure 1 shows the performance of multiple hyperparamter configurations evaluated for the pretrained VAD system on the development dataset. We highlight the top-performing combinations (corresponding to a DetER lower than 5.5%) and observe that these combinations can be attained with multiple values for three of the fours hyper-parameters, as long as the `offset` threshold is set to 0. This setting means that once speech is detected, the VAD will keep predicting "speech" unless it encounters a frame with a score of zero, resulting in almost the entire audio recording predicted as speech (with the exception of the silence from the beginning and the end of the audio, which is correctly marked as non-speech). As such, the best achieved results indicate an almost null miss error (MISS), at the expense of false alarm error (FA).

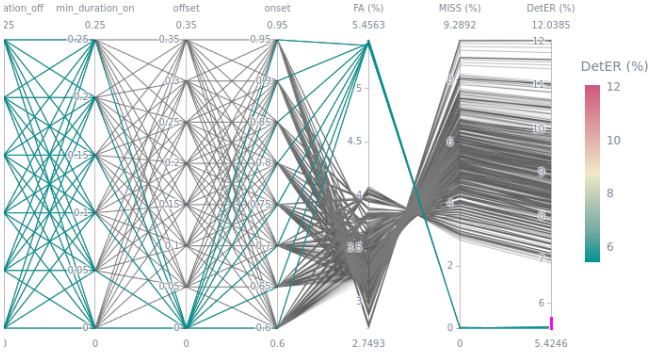

Figure 1: *The impact of the hyper-parameters on the voice activity detection (VAD) pretrained model illustrated as a parallel coordinates plot. We have evaluated over four hyper-parameters (min duration off, min duration on, offset, onset) and monitor the false alarm (FA), miss error (MISS) and detection error (DetER).*

Hyper-parameter tuning on the fine-tuned VAD model. Table 1 compares the hyper-paramter tuning for both the pretrained and the fine-tuned model in terms of multiple performance metrics: false alarm, miss error and detection error. While we observed a behaviour similar to the pretrained model, (namely, that the `offset` threshold is the most sensitive hyper-parameter and that it is important to take low values), we have also noticed that the best-achieving configuration on the development part did not generalize well on the evaluation recordings, which presented longer intervals of non-speech. In the end, we have opted for a hyper-parameter setup that offered a lower false alarm error, to the detriment of the miss error, by slightly increasing the `offset` threshold to 0.05. In the remainder of this paper, we used the fine-tuned VAD model, together with the tuned hyper-parameters, as the speech detector for the NeMo systems, because it performed better in terms of diarization error compared to the pretrained model on the evaluation set.

2.2.3. Multi-scale diarization decoder

The recordings in the development set exhibit a high percentage of overlapped speech. In order to reduce the overlap error, we used the MSDD neural diarizer after the spectral clusterization step in order to generate overlap-aware diarization. The configuration used in all experiments has 6 scales with a temporal resolution ranging from 0.5s to 3s, with a 0.5 segment shift ratio. This configuration is similar to the meeting configuration detailed in the original paper [19].

The model was trained on the entire DISPLACE development set, preparing the recordings with a step of 10×500 ms, batch size of 7 and a train–validation split of 80-to-20. We used

Table 1: *VAD hyper-parameters fine-tuning configuration obtained through grid search. For the pretrained model, the "Data" column refers to the datasets on which we selected the VAD hyper-parameters (not the model weights), while for the fine-tuned model, the "Data" columns refers to the datasets on which we trained both the hyper-parameters and the model weights.*

VAD Model	Data	Min Duration On	Min Duration Off	Onset	Offset	FA	Miss	DetErr
pretrained	dev2+3	0	0	0.85	0	5.42	**0**	**5.42**
	dev1+2+3	0	0	0.90	0	6.58	0.18	6.76
fine-tune	dev2+3	0	0	0.60	0.05	**4.05**	3.03	7.08
	dev1+2+3	0.45	0.40	0.80	0.05	4.97	0.70	5.68

the SGD optimizer with a learning rate of 0.08, weight decay of 2×10^{-4} and cosine annealing [27] with a warm-up ratio of 0.1 as learning rate scheduler for 30 epochs. For inference on the evaluation set, we tested using two sigmoid thresholds τ, one corresponding to the smallest DER on the whole development set (sigmoid threshold of 0.9), and one corresponding to the smallest confusion error (CER) respectively (sigmoid threshold of 0.7). The sigmoid threshold is a value between 0 and 1 that controls the overlap detection sensitivity: a smaller threshold leads to more speaker overlap segments. If the confidence that two segments overlap is greater than the sigmoid threshold, then those two overlapping segments will both appear at that timestamp.

2.3. Diarization with Pyannote

Pyannote [22, 23] is a speaker diarization library that distinguishes itself through the use of a segmentation module. The segmentation module is an end-to-end network that predicts the probability of up to three speakers being active in a five-second window. A threshold hyper-parameter is used to binarize the segmentation predictions, which are further encoded using the speaker representations extracted with the pretrained ECAPA-TDNN [29] model. Finally, Pyannote applies an agglomerative clustering algorithm to link the identities of the speakers across segments and to generate a diarization output for the entire audio. We fine-tuned the pretrained v2.1 model and systematically evaluated over the following components.

Training. Apart from the pretrained model, we fine-tuned the model on the DISPLACE data. We trained for 150 epochs with a learning rate of 10^{-3} and using cosine annealing with warm restarts for learning rate scheduling. Out of the 23 files in the dev2 and dev3, we selected 21 files for training and two for validation (B046 from dev2 and M026 from dev3).

Checkpoint. We monitored the performance of the model on a validation set consisting of two randomly chosen audio files. We keep the best ten models encountered during training and we have experimented with three variants of choosing the final weights: best (uses the weights from the best model), last (uses the weights from the last of the ten models), average (averages the weights of the ten checkpoints).

Threshold. We have tried four thresholds for segmentation $\{0.45, 0.50, 0.55, 0.60\}$; these were selected to be around the values of the pretrained model (0.62) and the best ones obtained on the validation data (0.4–0.5).

3. Experimental results

3.1. Results using NeMo

Table 2 presents results for several configurations and diarization methods. We include the baseline model that uses AHC clustering followed by VB-HMM resegmentation [30] and our own NeMo systems based on either only spectral clustering or spectral clustering followed by the MSDD module. On the dev1 dataset that was initially provided in the challenge, we obtain the best performance when fine-tuning the parameters of the pretrained model on dev2+3, reducing the baseline DER from 27.7% down to 22.61%. In addition, on the eval1 dataset, we are able to improve the performance over the baseline model by fine-tuning the VAD model and training the MSDD model: we lower the baseline system DER from 40.08%, down to 29.07%.

Table 2: *Impact of using the neural diarizer (MSDD) on top of spectral clustering. The parameter τ is the sigmoid threshold. The "data" column lists the datasets used for both selecting the VAD hyper-parameters and training the MSDD network. All configurations use the VAD fine-tuned on the dev2+3 sets.*

Method	Data	dev1	dev1+2+3	eval1
DISPLACE baseline [1]		27.70	32.57	40.08
Spectral clusetring (SC)	dev2+3	**22.61**	26.39	29.22
	dev1+2+3	–	25.91	29.11
SC+MSDD τ: 0.7	dev2+3	22.86	–	-
	dev1+2+3	–	26.08	29.49
SC+MSDD τ: 0.9	dev2+3	23.06	–	–
	dev1+2+3	–	**25.50**	**29.07**

The results across datasets indicate that the dev1 results do not correlate well with the performances on the eval1 set. However, the results on dev1+2+3 are much more in-line with the performance obtained on the evaluation set (we are aware that these latter results might be overly optimistic since these models were fine-tuned on part of this data). This behaviour may be caused by the fact that dev2 and dev3 sets have a distribution that is closer to the one of the eval1 set; so, in future we recommend using the K-fold cross-validation technique to prevent the data mismatch and obtain more accurate results.

3.2. Results with Pyannote

In Table 3 we show the results obtained using Pyannote. First, we note that we are able to improve the performance over the pretrained model: from 23.85% to 21.79% DER on dev1 and from 37.01% to 35.33% DER on eval1.

As observed in the previous subsection, we again see that the performance does not necessarily correlate across various splits. For example the (fine-tuned, average, 0.55) combination obtains a sizeable improvement over (fine-tuned, last, 0.50) on the eval1 split, but the results on the dev1 are much closer.

Finally, we observe that the segmentation threshold can impact the results (three points difference in the fine-tuned, best case), but its value is not as critical when we average the checkpoints (about 0.7 difference for the fine-tuned, average case).

Table 3: *Results with Pyannote library on the* `dev1` *and* `eval1` *sets. We compare across three axes of experimentation: the training procedure, the checkpoint for the model's weights, the segmentation threshold. See the text for more details.*

			DER (%)	
Training	**Checkpoint**	**Thresh**	**dev1**	**eval1**
pretrained	–	0.62	23.85	37.01
fine-tuned	best	0.45	28.70	
fine-tuned	best	0.50	27.86	
fine-tuned	best	0.55	26.91	
fine-tuned	best	0.60	25.75	
fine-tuned	last	0.45	22.87	
fine-tuned	last	0.50	22.24	36.85
fine-tuned	last	0.55	21.98	
fine-tuned	last	0.60	21.79	
fine-tuned	average	0.45	22.88	
fine-tuned	average	0.50	22.24	
fine-tuned	average	0.55	22.16	35.33
fine-tuned	average	0.60	22.17	

Table 4: *DER on* `dev1` *and* `eval1` *for five of our systems (three based on NeMo, two on Pyannote) and their combinations using DOVER-Lap. We also report the average performance of the input systems.*

		DER (%)	
Toolkit	**Configuration**	**dev1**	**eval1**
NeMo	VAD pretrained	21.48	30.20
NeMo	VAD fine-tuned	22.60	29.11
NeMo	VAD fine-tuned + MSDD (τ: 0.7)	22.86	29.49
Pyannote	pretrained	23.85	37.01
Pyannote	avg. thresh. 0.55	22.16	35.33
Systems average (w/ Pyannote)		22.59	32.23
Systems average (w/o Pyannote)		22.31	29.60
DOVER-Lap greedy (w/ Pyannote)		20.44	31.39
DOVER-Lap greedy (w/o Pyannote)		22.51	**28.97**
DOVER-Lap Hungarian (w/ Pyannote)		**20.40**	–
DOVER-Lap Hungarian (w/o Pyannote)		22.53	–

3.3. Model combinations

After evaluating several systems and setups on the development set, we aimed to improve diarization performance by combining their results. We used the official implementation of DOVER-Lap [21] using both the greedy and Hungarian label mapping techniques, in order to reduce the diarization error.

Table 4 presents the results after the DOVER-Lap on both the development and evaluation datasets respectively. We combine five of our systems (three systems based on NeMo and two systems based on Pyannote) in two variants: all five systems and only the three NeMo-based systems. On `dev1`, we decrease the DER using the first variant and both label mapping methods presented in the paper and obtain better results compared to the system average, as well as the best system DER by more than 1%. On `eval1`, we use both varaints with greedy label mapping and manage to obtain our best overall result, 28.97% DER (previously, MSDD with τ of 0.9 achieved the best result of 29.07%), on `eval1` when using only the NeMo systems. Using the MSDD with a sigmoid threshold of 0.7, corresponding to the smallest CER (instead of the sigmoid threshold of 0.9 corresponding to the smallest DER), resulted in better results on DOVER-Lap, as it contained more overlapping segments.

3.4. Hardware requirements

For NeMo, training both the VAD model, as well as the MSDD model, was done on a multi-GPU configuration with 5 Nvidia Tesla T4 GPUs. For VAD fine-tuning on `dev2+3`, an epoch takes approximately 30s. For the MSDD model training on `dev1+2+3`, an epoch takes around 45 minutes. VAD hyper-parameter tuning was performed on CPU and it takes around 15s per iteration. The inference using spectral clustering takes 43 min for 15h and 44 min of recordings from `eval1`, while inference using MSDD takes 46 min, on a single Testa T4 GPU.

4. Conclusions

In this paper we presented our speaker diarization solution for the DISPLACE 2023 challange. We employed two different approaches, namely the multi-scale spectral clustering method from the NeMo library and the neural segmentation approach developed in Pyannote. We conducted a comprehensive study on the impact of training data and various hyper-parameters. Our experiments demonstrated that the pretrained spectral clustering model from NeMo provides robust results.

We showed that incorporating the neural diarizer into the normal clusterization pipeline in NeMo can lead to further improvements in the results, as it enables our system to be aware of overlaps. This component was trained on the development data provided and we observed a decrease in the diarization error even after a few training epochs.

Our results highlight the sensitivity of the voice activity detector (VAD) to its hyperparameters, which persists even after careful fine-tuning on the development set. However, a multi-scale approach was shown to enhance the model's robustness in challenging conversational environments featuring high levels of noise and multiple speakers, by reducing the trade-off between the quality of speaker representations and temporal resolution.

In order to improve over the baseline systems, we combined five configurations of the most promising systems from both toolkits using DOVER-Lap system fusion and reported better results on both `dev1` and `eval1` datasets. We observed that using MSDD with a lower sigmoid threshold as input for the DOVER-Lap algorithm leads to better results, as the MSDD inputs contain more overlapping segments. Our final system achieved a DER value of 28.97%, while the baseline system achieved the best performance of 40.08%.

Finally, for both the individual system performances, as well as the models combination step, the NeMo toolkit obtained better performance than Pyannote. Our two best results were achieved using the MSDD diarizer and the combination of three NeMo systems. Surprisingly, even though NeMo and Pyannote were comparable in terms of performance on the `dev1` data, the results of Pyannote did not translate on the evaluation `eval1` data, indicating again a possible mismatch of their data distributions.

Acknowledgement. This work was partly funded by European Union's HORIZON-CL4-2021-HUMAN-01 research and innovation program under grant agreement No. 101070190 and partly funded by the Competitiveness Operational Programme Romania under project number SMIS 156387 - VOITA.

5. References

[1] S. Baghel, S. Ramoji, Sidharth, R. H, P. Singh, S. Jain, P. R. Chowdhuri, K. Kulkarni, S. Padhi, D. Vijayasenan, and S. Ganapathy, "DISPLACE challenge: DIarization of SPeaker and LAnguage in Conversational Environments," 2023. [Online]. Available: https://arxiv.org/abs/2303.00830

[2] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du, S. Ganapathy, and M. Liberman, "The second DIHARD diarization challenge: Dataset, task, and baselines," in *Interspeech*, 2019, pp. 978–982.

[3] S. Watanabe, M. Mandel, J. Barker, E. Vincent, A. Arora, X. Chang, S. Khudanpur, V. Manohar, D. Povey, D. Raj *et al.*, "CHiME-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," in *Workshop on Speech Processing in Everyday Environments*, 2020.

[4] K. Min, "Intel labs at ego4d challenge 2022: A better baseline for audio-visual diarization," *arXiv preprint arXiv:2210.07764*, 2022.

[5] J. H. Hansen, A. Joglekar, M. C. Shekhar, V. Kothapally, C. Yu, L. Kaushik, and A. Sangwan, "The 2019 inaugural fearless steps challenge: A giant leap for naturalistic audio," *Interspeech*, 2019.

[6] E. Lleida, A. Ortega, A. Miguel, V. Bazán-Gil, C. Pérez, M. Gómez, and A. De Prada, "Albayzin 2018 evaluation: the IberSPEECH-rtve challenge on speech technologies for spanish broadcast media," *Applied sciences*, vol. 9, no. 24, p. 5412, 2019.

[7] A. Nagrani, J. S. Chung, J. Huh, A. Brown, E. Coto, W. Xie, M. McLaren, D. A. Reynolds, and A. Zisserman, "Voxsrc 2020: The second VoxCeleb speaker recognition challenge," *arXiv preprint arXiv:2012.06867*, 2020.

[8] A. Sofer and S. E. Chazan, "CNN self-attention voice activity detector," 2022. [Online]. Available: https://arxiv.org/abs/2203.02944

[9] M. Diez, L. Burget, S. Wang, J. Rohdin, and J. Cernockỳ, "Bayesian HMM Based x-Vector Clustering for Speaker Diarization," in *Interspeech*, 2019, pp. 346–350.

[10] Y. Bai, J. Yi, J. Tao, Z. Wen, and B. Liu, "Voice activity detection based on time-delay neural networks," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 2019, pp. 1173–1178.

[11] W. Zhu and J. Pelecanos, "Online speaker diarization using adapted i-vector transforms," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2016, pp. 5045–5049.

[12] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 14, 2001.

[13] F. Jia, S. Majumdar, and B. Ginsburg, "MarbleNet: Deep 1d time-channel separable convolutional neural network for voice activity detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2021, pp. 6818–6822.

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

[15] O. Kuchaiev, J. Li, H. Nguyen, O. Hrinchuk, R. Leary, B. Ginsburg, S. Kriman, S. Beliaev, V. Lavrukhin, J. Cook, P. Castonguay, M. Popova, J. Huang, and J. M. Cohen, "NeMo: a toolkit for building AI applications using neural modules," *CoRR*, vol. abs/1909.09577, 2019. [Online]. Available: http://arxiv.org/abs/1909.09577

[16] H. Bredin and A. Laurent, "End-to-end speaker segmentation for overlap-aware resegmentation," in *Interspeech*, 2021.

[17] Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu, and S. Watanabe, "End-to-End Neural Speaker Diarization with Self-Attention," in *IEEE Automatic Speech Recognition & Understanding*, 2019, pp. 296–303.

[18] Y. C. Liu, E. Han, C. Lee, and A. Stolcke, "End-to-end neural diarization: From transformer to conformer," in *Interspeech 2021*, aug 2021.

[19] T. J. Park, N. R. Koluguri, J. Balam, and B. Ginsburg, "Multi-scale speaker diarization with dynamic scale weighting," in *Interspeech*, 2022.

[20] A. Stolcke and T. Yoshioka, "DOVER: A method for combining diarization outputs," in *IEEE Automatic Speech Recognition & Understanding*, 2019, pp. 757–763.

[21] D. Raj, L. P. Garcia-Perera, Z. Huang, S. Watanabe, D. Povey, A. Stolcke, and S. Khudanpur, "Dover-lap: A method for combining overlap-aware diarization outputs," in *IEEE Spoken Language Technology Workshop*, 2021, pp. 881–888.

[22] H. Bredin, R. Yin, J. M. Coria, G. Gelly, P. Korshunov, M. Lavechin, D. Fustes, H. Titeux, W. Bouaziz, and M. Gill, "Pyannote.Audio: Neural building blocks for speaker diarization," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 7124–7128.

[23] H. Bredin and A. Laurent, "End-to-end speaker segmentation for overlap-aware resegmentation," in *Interspeech*, 2021.

[24] D. Snyder, G. Chen, and D. Povey, "MUSAN: A music, speech, and noise corpus," *CoRR*, vol. abs/1510.08484, 2015. [Online]. Available: http://arxiv.org/abs/1510.08484

[25] T. J. Park, K. J. Han, M. Kumar, and S. Narayanan, "Auto-tuning spectral clustering for speaker diarization using normalized maximum eigengap," *IEEE Signal Processing Letters*, vol. 27, pp. 381–385, 2019.

[26] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," in *Interspeech*, 2019.

[27] T. Devries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *CoRR*, vol. abs/1708.04552, 2017. [Online]. Available: http://arxiv.org/abs/1708.04552

[28] P. Mishra and K. Sarawadekar, "Polynomial learning rate policy with warm restart for deep neural network," in *IEEE Region 10 International Conference TENCON*, 2019, pp. 2087–2092.

[29] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification," in *Interspeech*, 2020.

[30] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.