



# Overlapped Speech Detection in Broadcast Streams Using X-vectors

Lukas Mateju, Frantisek Kynych, Petr Cerva, Jiri Malek, Jindrich Zdansky

Faculty of Mechatronics, Informatics and Interdisciplinary Studies,  
Technical University of Liberec, Studentska 2, 461 17 Liberec, Czech Republic

lukas.mateju@tul.cz

## Abstract

A new approach to overlapped speech detection (OSD) is introduced in this work. It is designed for real-time processing of streamed data and utilizes x-vectors as its input features. It thus allows us to reduce computational demands within the entire streaming data processing chain, where the same x-vectors can also be used for the related task of speaker diarization. Within our method, the x-vectors are extracted using a feed-forward sequential memory network (FSMN) and then fed into a simple neural classifier (speech or cross-talk), whose output is smoothed by a decoder based on weighted finite-state transducers (WFSTs). The evaluation is done on a Czech/Slovak broadcast dataset (we make this data public) and on the AMI meeting corpus. Our online method yields a solid performance while operating with a 2-second latency.

**Index Terms:** overlapped speech detection, x-vectors, stream data processing, deep neural networks

## 1. Introduction

In natural human conversation, moments occur when two or more people speak at the same time. These overlapped speech segments are often referred to as cross-talks, and are commonly encountered in, e.g., interviews, talk shows, debates, as well as in broadcast news. Cross-talks can have various forms, including interruptions, back-channel responses (“yes”), or premature turn-taking [1]. They can also be formed by numerous speaker sounds, such as hesitation, laughter, or coughing.

All these types of cross-talk are known to deteriorate the performance of various speech processing tasks, e.g., speech recognition [2] or speaker diarization [3]. The best approach to this problem is to detect overlapped speech and then omit the corresponding audio segments in subsequent processing steps.

In this work, we extend our previous investigations on the use of x-vectors [4] for speech activity (and overlapped speech) detection [5] by presenting a complete online overlapped speech detection scheme, which is suitable for frame-wise data processing. That means that our scheme takes in a sequence (stream) of speech frames on its input and provides a stream of labels identifying the presence of overlapped speech instances in every frame on its output. Our target system is a platform for 24/7 monitoring of various TV/R streams, namely in Slavic languages (see our multilingual radio monitoring application<sup>1</sup>).

The utilization of x-vectors for this task stems from the fact that these features are employed in subsequent blocks of the speech processing pipeline – in particular for speaker diarization. The use of the same features within both of these interrelated tasks thus allows us to reduce the computational demands and the complexity of the system, which is a crucial factor for processing of streamed data.

Experimental results obtained on real-world streams of broadcast data show that our approach is capable of detecting overlapped speech with a frame F-measure of 64.9% while operating with a latency of around 2 seconds. Our additional contribution to the speech research community is that we make public<sup>2</sup> the set of recordings with hand-annotated segments of overlapped speech that we used for evaluation.

## 2. Related works

In most cases, overlapped speech detection is carried out in two consecutive steps; namely, feature extraction is followed by classification. In the first step, the most commonly utilized features are the standard Mel-frequency cepstral coefficients (MFCCs) [6, 7] and log filter bank coefficients (FBCs) [8, 9]. However, various acoustic features, such as spectral flatness [10], linear predictive coding residual energy [11], kurtosis [12], convolutive sparse coding [13], or pyknoogram [14], have been used with varying success. A popular option remains to hand-craft various combinations of the aforementioned features to further improve the performance [15, 16].

For the second step, hidden Markov models and Gaussian mixture models [10, 17] were widely employed in the early years. With the advances in deep learning, deep neural networks (DNNs) have been heavily applied to OSD as well. This approach has naturally led to notable improvements in performance. Various neural network architectures, such as fully connected (FC) feed-forward DNNs [5, 7] or convolutional neural networks (CNNs) [16, 18], have been studied. The modeling power of recurrent neural networks (RNNs) was explored in the form of long short-term memory (LSTM) RNNs in [15]. In [8], a comparison of FC feed-forward DNNs, CNNs, unidirectional and bidirectional LSTM RNNs was shown. Moreover, bidirectional LSTM RNNs implemented in pyannote [19] delivered excellent performance [20] on the AMI meeting corpus [21]. More complex architectures, e.g., temporal CNNs [9], block-based CNNs [22], time-convolving NNs [23], or the architecture of Deep Speech 2 [24], have been exploited as well. Recently, a convolutional RNN [25] produced state-of-the-art results using a three-class classifier (speech, overlapped speech, non-speech) on DIHARD II [26] evaluation. Finally, approaches based on end-to-end neural speaker diarization [27] started to deal with overlapped speech during diarization [28]. Specifically, state-of-the-art results were achieved on DIHARD III [29] evaluation and AMI corpus in [30].

Improvements in the performance of OSD can also be achieved by smoothing the outputs of a given classifier. For this purpose, several techniques, such as moving average [7], Hamming window [31] or Viterbi decoding [8], have been successfully deployed.

<sup>1</sup><https://tul-speechlab.gitlab.io/>

<sup>2</sup><https://owncloud.cesnet.cz/index.php/s/toPSfajLobR1usV>

### 3. Proposed approach

The proposed approach to overlapped speech detection can be divided into three consecutive steps:

1. Extraction of x-vectors.
2. Classification of x-vectors.
3. Output smoothing.

#### 3.1. Extraction of x-vectors

In the first step, the x-vectors are extracted using a DNN; in this case, a vectorized version of FSMN [32] is used. Similar to RNNs, this neural architecture is suitable for modeling long-time dependencies in the input signal. However, it eliminates the recursion by adding several memory blocks with trainable weight coefficients into each layer of a classic FC feed-forward DNN. These memory blocks use a tapped-delay line structure to encode the long context information into a fixed-size representation. That means that the FSMN layers are built on top of the context of the previous layers making the final context a sum of the partial ones.

Table 1 describes the structure of the utilized FSMN. The symbol  $\ell$  marks the current frame, on which the temporal context is centered. On the input, each frame of the signal is represented by 40-dimensional FBCs computed from 25-ms-long frames with frame shifts of 12.5 ms each. Within the context of the first layer (i.e., 161 frames), cepstral mean subtraction is applied. In the pooling layer, only the means of the frames (without the variances) are computed in the context of 41 consecutive frames. As the activation function, exponential linear unit (ELU) is used in all neurons. In the final layer, the number of output neurons corresponds to the number of unique speakers (7,238) in the training data. Finally, a multiclass cross-entropy objective function is used during the training.

Table 1: The structure of the FSMN-based x-vector extractor.

layer	layer context	total context	input x output
FSMN 1	$\ell \pm 80$	161	$40 \times 1024$
FSMN 2	$\ell \pm 4$	169	$1024 \times 768$
FSMN 3	$\ell \pm 4$	177	$768 \times 512$
FSMN 4	$\ell \pm 4$	185	$512 \times 384$
FSMN 5	$\ell \pm 4$	193	$384 \times 256$
FSMN 6	$\ell \pm 4$	201	$256 \times 128$
FC 1	$\ell$	201	$128 \times 128$
Pooling	$\ell \pm 20$	241	$(41 \cdot 128) \times 128$
FC 2	$\ell$	241	$128 \times 128$
Softmax	—	241	$128 \times 7,238$ speakers

After the extractor is trained, the x-vectors are computed after the pooling layer (i.e., the dimension of the final x-vectors is 128) with shifts of one frame. The resulting sequence then forms the input to the classifier in the second step. Note that the extractor operates with a total context of 241 frames as shown in Table 1, which corresponds to 3.0125 seconds.

#### 3.2. Classification of x-vectors

The extracted x-vectors are fed to a binary DNN-based classifier, generating probabilities for the respective speech and overlapped speech classes in the second step. This simple classifier consists of two FC feed-forward hidden layers with 128 and

64 neurons, respectively, and the ELU activation function is used in each neuron. The network is trained within 50 epochs with a learning rate set to 0.08 and using mini-batches of size 1,024. Note that no additional input context is provided during the training.

#### 3.3. Output smoothing

In the last step, the outputs from the classifier are smoothed using an online decoder based on WFSTs. The main advantage of this approach is that it represents a general smoothing concept, which allows us to model various smoothing approaches by merely choosing the corresponding transduction model and the respective semi-ring.

The decoding scheme consists of two transducers depicted in Figure 1. The upper transducer models the input signal. The other one is the transduction model and represents the smoothing algorithm. It consists of three states, where the first state, denoted as 0, is the initial state. The transitions between states 1 and 2 emit the speech (S), and overlapped speech (OS) labels, respectively. These transitions are also weighted by penalty factors P1 and P2, whose values have been fined-tuned on the development set.

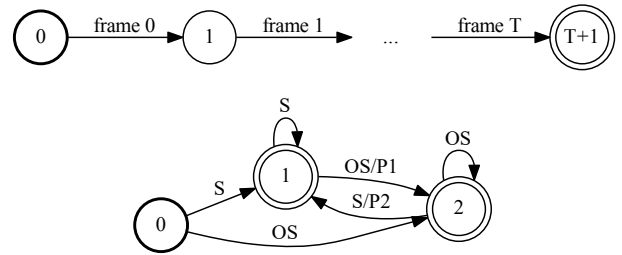


Figure 1: The transducers representing the input signal (upper) and the transduction model (lower) used for OSD.

Given the two transducers described above, the decoding process is performed using the on-the-fly composition of the transduction and the input model of an unknown size. That method is feasible since the input is considered to be a linear-topology, unweighted, epsilon-free acceptor. After each composition step, the shortest path (considering the tropical semi-ring) determined in the resulting model is compared with all other alternative hypotheses. When a common path (i.e., with identical output labels) is found among these hypotheses, the corresponding concatenated output labels are marked as the resulting fixed output. Since the rest of the best path is not known with certainty, it is denoted as a temporary output (i.e., it can be further refined).

## 4. Data and metrics

#### 4.1. Training data for x-vector extractor

The proposed x-vector extractor is trained using data from several sources, including VoxCeleb2 [33], LibriSpeech (“train-360-clean”) [34], and Czech microphone recordings. The training set of VoxCeleb2 provides 5,994 speakers, while the used subset of the Librispeech adds 921 of them. Furthermore, the Czech dataset includes an additional 922 speakers. Lastly, the set of target speakers is extended by the noise class defined over the training part of the CHiME-4 dataset [35]. In total, there are 7,238 unique speakers and 3,000 hours of speech recordings.

## 4.2. Training data for x-vector classifier

The amount and availability of annotated datasets suitable for training and evaluation of OSD, especially in a broadcast environment, are severely limited. For this reason, we decided to prepare an artificial training set. It is designed in a way similar to that in [8] as described in detail in our previous work [5].

The basis is that of clean Czech recordings (containing a single annotated speaker) from various TV/R broadcasts originally used for speech recognition. For each recording, two new ones are created. The first is unchanged and represents the speech class. The other one is summed together with a randomly selected second recording of a different speaker. A random shift is introduced at the beginning of the summed recording, such that a minimum length of overlapped speech is one second. Moreover, noise augmentation using the CHiME-4 dataset [35] is also done as specified in [5]. Finally, an additional 100 frames of non-speech are added to the beginnings and the ends of the final recordings to provide more context for the extracted x-vectors. In total, a 60-hour corpus of training data has been created this way, out of which 10 hours are labeled as overlapped speech.

## 4.3. Development and test data

Due to the aforementioned limitations of available annotated data for OSD, we have also prepared a new dataset for evaluation purposes. It is comprised of several Czech and Slovak TV, radio, and internet shows. In total, we have gathered over 10 hours of data which we have divided into two subsets of an equal size – development (dev) and test – and we annotated the data by hand. We labeled four different acoustic events – speech, non-speech (ns), overlapped speech (os), and “cross-noise” (cn). The first three labels are self-explanatory, while the latter class covers events when one of the speakers is talking, and the other is making noises (e.g., hesitation – “uh-huh”, “ehm”; coughing or heavy breathing). During this evaluation, cross-noise can be considered either as speech or overlapped speech (in our case), depending on the application. Detailed information about the datasets is listed in Table 2.

Table 2: An overview of the development and test datasets.

data	count	length	os [min]	cn [min]	ns [min]
dev	16	5 h	6:41	1:34	5:51
test	13	5.2 h	12:29	1:26	3:08

To further stimulate the research of OSD, we have decided to release the data with manual annotations to the general public. For each segment of every recording, start and end times are given together with the label of the corresponding acoustic event. Each recording is thus provided with an XML reference file, which can easily be viewed in NanoTrans [36].

## 4.4. Evaluation metrics

Within the experimental evaluation, standard metrics – frame precision (P), frame recall (R), and frame F-measure (F) [22] – are used to evaluate the quality of OSD on the frame level without any forgiveness collar. The overall frame error rate (FER) is also presented. Lastly, overlap detection error (ODE), defined as a ratio between summed missed and false-positive overlapped speech frames and the total number of overlapped speech frames in reference [15], is also shown.

# 5. Experimental evaluation

## 5.1. The proposed approach

The baseline experiment follows exactly the proposed approach as described in Sec. 3. The x-vector extractor was trained using the data detailed in Sec. 4.2, while the data highlighted in Sec. 4.3 was used to train the DNN-based classifier. The penalty weights of the WFST-based decoder were tuned on the development set to equalize the frame precision and frame recall.

The results are in the first row of Table 3 (best-achieved values are in bold). They collectively show that the performance of the proposed approach is solid. The FER of 1.9% signals that the accuracy of OSD is relatively high, while the balanced F-measure of 65% suggests only a small number of false negatives and false positives are made. Furthermore, we have also examined the specific errors, and there are two main issues causing problems. The first is implied by omitted short (<0.5 seconds) back-channel responses, resulting in false negatives. The other one follows from non-speech events (mostly loud laughter) during the talk of the other speaker, producing false positives.

Table 3: Results [%] of the proposed approach on the dev set.

approach	FER	P	R	F	ODE
baseline	1.9	64.9	65.2	65.0	70.1
3 classes	1.9	64.7	64.7	64.7	70.7
data × 0.5	2.0	63.2	63.7	63.4	73.5
data × 2	1.9	66.2	66.5	66.3	67.5
data × 3	<b>1.8</b>	<b>67.0</b>	<b>67.6</b>	<b>67.3</b>	<b>65.8</b>
data × 4	<b>1.8</b>	66.9	67.1	67.0	66.1
data × 5	1.9	66.0	66.0	66.0	67.9

## 5.2. Ternary classification

As suggested in [25], we have also tried to replace the binary classifier with a ternary one by splitting the non-overlapped speech class into two subclasses – speech and non-speech. This allowed the original authors to improve their results on the DI-HARD II challenge. We have also been able to do this since the training and evaluation data are annotated for non-speech as well. The rest of the experiment remains unchanged.

However, as summarized in the second row of Table 3, there is no similar improvement on the broadcast development set. The results, however, show similar, although slightly worse, performance when compared with the binary classifier.

## 5.3. The effect of an increasing amount of training data

To cope with the issues outlined in Sec. 5.1, a significant extension to the 60-hour training dataset was made. Another 240 hours of data has been prepared in the same way as previously (see Sec. 4.3). The experiment has been conducted with various amounts of data added (i.e., in total, 120, 180, 240, and 300 hours). Additionally, a smaller 30-hour set has also been evaluated for comparison.

The results are grouped up in the bottom part of Table 3. They show a clear trend that more training data is beneficial up to 180 hours where the performance starts to plateau, even slightly worsen, and the additional information is no longer helpful. In summary, the F-measure has been improved by 2.3% up to 67.3%, and the ODE has been reduced to 65.8%. Lastly, the smallest dataset has led to a worse performance level.

#### 5.4. The effect of smoothing

To demonstrate the importance of the WFST-based smoothing, we compare it with no smoothing and moving-average (MA) smoothing with different window lengths. Specifically, three window lengths are explored – 0.5, 1, and 1.5 seconds.

The results are presented in Table 4, where the first row clearly manifests that smoothing is indeed necessary. Without it, all the monitored metrics are notably worsened, e.g., the achieved FER has been degraded by more than three times. The main issue, in this case, is implied by rapid and short changes between the speech and overlapped speech classes. This issue is resolved by applying a selected smoothing method, say, the moving-average one (where the 1-second window has performed the best), which offers improvements in almost all metrics. However, the WFST-based decoder is able to find an even better solution and boosts all the main metrics (FER, F-measure, and ODE) even further (see the rest of Table 4).

Table 4: Results [%] of the proposed approach using various smoothing techniques on the development set.

smoothing	FER	P	R	F	ODE
none	5.9	28.5	<b>74.5</b>	41.2	212.8
MA 0.5 s	2.6	52.3	72.9	60.9	93.7
MA 1 s	2.2	61.7	56.5	59.0	78.6
MA 1.5 s	2.2	65.1	42.9	51.7	80.1
WFST	<b>1.8</b>	<b>67.0</b>	67.6	<b>67.3</b>	<b>65.8</b>

#### 5.5. Final performance

The final approach (i.e., trained on 180 hours of data) has also been evaluated on the test set. This data has turned out to be slightly more complex than the development one. This may be caused by almost six more minutes of overlapped speech segments as presented in Table 2. The results are shown in Table 5. Specifically, the proposed approach has scored an F-measure of 64.9% (without a forgiveness collar) and has achieved a FER of 3.1%, setting a new baseline for future research.

Table 5: Results [%] of the proposed approach on the test set.

approach	FER	P	R	F	ODE
final	<b>3.1</b>	<b>66.4</b>	<b>63.5</b>	<b>64.9</b>	<b>68.6</b>

Moreover, the performance in the streamed processing environment has been evaluated. For this purpose, two metrics are monitored – real-time factor (RTF)<sup>3</sup>, a ratio of processing time to the duration of the input recording, and latency, an average time between a class change and the moment the decoder outputs the corresponding label. The proposed approach operates with an RTF value of 0.1 and yields an average latency of 2 seconds, out of which 0.5 seconds is caused by the WFST-based decoder, making it perfectly viable for use in stream processing.

### 6. Evaluation on AMI meeting corpus

To compare the proposed approach with other OSD techniques available in the literature, the AMI meeting corpus [21] has been selected. It is a multi-modal 100-hour long dataset consisting of

<sup>3</sup>measured by processor Intel Core i7-3770K @ 3.50GHz

meeting recordings, and it has been widely used for solving several speech processing tasks, such as speaker diarization. During the evaluation, we have followed the newly standardized AMI evaluation protocol, which was introduced in [37], and only the Mix-Headset subset is used.

The proposed OSD approach is applied as described in Sec. 3. First, the x-vectors for the training portion of the AMI corpus are extracted and next used to train the simple binary DNN classifier. After that, the classifier’s outputs for the test data are smoothed by the WFST-based decoder (whose penalty weights are fine-tuned on the development set), and final labels are produced. Note that no additional training data or data augmentation techniques are incorporated.

The results are shown in the first row of Table 6. While the obtained precision is high, the achieved recall can be improved. Note that FER and ODE are 7.9% and 60.6%, respectively.

Table 6: A comparison of OSD approaches on AMI corpus.

approach	P [%]	R [%]	F [%]
proposed (2 classes)	85.5	47.4	61.0
<i>Kunesova et al.</i> [18, 30]	71.5	46.1	56.0
<i>pyannote 1.1 (MFCC)</i> [20]	<b>91.9</b>	48.4	63.4
<i>pyannote 1.1 (wave)</i> [20]	86.8	65.8	73.9
<i>pyannote 2.0</i> [30]	80.7	<b>70.5</b>	<b>75.3</b>

The bottom part of Table 6 provides a comparison with several approaches found in the literature. Here, our method is outperformed, namely in values of recall by two configurations of the pyannote toolkit (the last two rows of Table 6). The reasons are two-fold. First, the pyannote toolkit is designed primarily for overlap-aware speaker diarization, and it thus combines several training datasets (from the meeting domain) and implements various data augmentation techniques to further boost the results on each individual dataset. Second, it works in an offline mode and requires GPU for processing in real-time: the used models can thus be more complex than in the case of our method designed for streamed data processing using just a CPU.

## 7. Conclusions

In this paper, a new approach to frame-wise OSD has been proposed. It is composed of three consecutive steps; at first, an FSMN-based x-vector extractor produces speaker embeddings for artificially mixed training data. These embeddings are then used to train a computationally undemanding binary DNN-based classifier. Finally, the output of the classifier is smoothed by a WFST-based decoder to produce the final segmentation.

The performance of the proposed approach has been evaluated using standard metrics on the dataset of broadcast news, whose annotations are released to the speech research community, and on the standardized AMI meeting corpus. The achieved results show a solid performance (and provide a starting baseline for future research), while the proposed approach is computationally undemanding and operates online with low latency and real-time factor.

## 8. Acknowledgements

This work was supported by the Technology Agency of the Czech Republic (Project No. TO0100027), and by the Student Grant Competition of the Technical University of Liberec under project No. SGS-2022-3052.

## 9. References

- [1] I. Siegert, R. Bock, A. Wendemuth, B. Vlasenko, and K. Ohnemus, "Overlapping speech, utterance duration and effective content in HHI and HCI – an comparison," in *CogInfoCom 2015, Gyor, Hungary*, 2015, pp. 83–88.
- [2] F. Brugnara, D. Falavigna, D. Giuliani, and R. Gretter, "Analysis of the characteristics of talk-show TV programs," in *Interspeech 2012, Portland, USA*, 2012, pp. 1388–1391.
- [3] M. Huijbregts, D. A. van Leeuwen, and C. Wooters, "Speaker diarization error analysis using oracle components," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, pp. 393–403, 2012.
- [4] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *ICASSP 2018, Calgary, Canada*, 2018, pp. 5329–5333.
- [5] J. Malek and J. Zdansky, "Voice-activity and overlapped speech detection using x-vectors," in *TSD 2020, Brno, Czech Republic*, 2020, pp. 366–376.
- [6] K. Boakye, B. Trueba-Hornero, O. Vinyals, and G. Friedland, "Overlapped speech detection for improved speaker diarization in multiparty meetings," in *ICASSP 2008, Las Vegas, USA*, 2008, pp. 4353–4356.
- [7] M. Diez, F. Landini, L. Burget, J. Rohdin, A. Silnova, K. Zmolkova, O. Novotny, K. Vesely, O. Glembek, O. Plchot, L. Mosner, and P. Matejka, "BUT system for DIHARD speech diarization challenge 2018," in *Interspeech 2018, Hyderabad, India*, 2018, pp. 2798–2802.
- [8] N. Sajjan, S. Ganesh, N. Sharma, S. Ganapathy, and N. Ryant, "Leveraging LSTM models for overlap detection in multi-party meetings," in *ICASSP 2018, Calgary, Canada*, 2018, pp. 5249–5253.
- [9] S. Cornell, M. Omologo, S. Squartini, and E. Vincent, "Detecting and counting overlapping speakers in distant speech scenarios," in *Interspeech 2020, Shanghai, China*, 2020, pp. 3107–3111.
- [10] K. Boakye, O. Vinyals, and G. Friedland, "Two's a crowd: improving speaker diarization by automatically identifying and excluding overlapped speech," in *Interspeech 2008, Brisbane, Australia*, 2008, pp. 32–35.
- [11] M. Zelenak and J. Hernando, "The detection of overlapping speech with prosodic features for speaker diarization," in *Interspeech 2011, Florence, Italy*, 2011, pp. 1041–1044.
- [12] K. Boakye, O. Vinyals, and G. Friedland, "Improved overlapped speech handling for speaker diarization," in *Interspeech 2011, Florence, Italy*, 2011, pp. 941–944.
- [13] J. T. Geiger, R. Vippera, S. Bozonnet, N. W. D. Evans, B. W. Schuller, and G. Rigoll, "Convolutional non-negative sparse coding and new features for speech overlap handling in speaker diarization," in *Interspeech 2012, Portland, USA*, 2012, pp. 2154–2157.
- [14] N. Shokouhi and J. H. L. Hansen, "Teager-Kaiser energy operators for overlapped speech detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, pp. 1035–1047, 2017.
- [15] J. T. Geiger, F. Eyben, B. W. Schuller, and G. Rigoll, "Detecting overlapping speech with long short-term memory recurrent neural networks," in *Interspeech 2013, Lyon, France*, 2013, pp. 1668–1672.
- [16] M. Yousefi and J. H. L. Hansen, "Frame-based overlapping speech detection using convolutional neural networks," in *ICASSP 2020, Barcelona, Spain*, 2020, pp. 6744–6748.
- [17] S. H. Yella and H. Bourlard, "Overlapping speech detection using long-term conversational features for speaker diarization in meeting room conversations," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, pp. 1688–1700, 2014.
- [18] M. Kunesova, M. Hruz, Z. Zajic, and V. Radova, "Detection of overlapping speech for the purposes of speaker diarization," in *SPECOM 2019, Istanbul, Turkey*, 2019, pp. 247–257.
- [19] H. Bredin, R. Yin, J. M. Coria, G. Gelly, P. Korshunov, M. Lavachin, D. Fustes, H. Titeux, W. Bouaziz, and M.-P. Gill, "Pyanote.audio: Neural building blocks for speaker diarization," in *ICASSP 2020, Barcelona, Spain*, 2020, pp. 7124–7128.
- [20] L. Bullock, H. Bredin, and L. P. Garcia-Perera, "Overlap-aware diarization: Resegmentation using neural end-to-end overlapped speech detection," in *ICASSP 2020, Barcelona, Spain*, 2020, pp. 7114–7118.
- [21] J. Carletta, "Unleashing the killer corpus: experiences in creating the multi-everything AMI meeting corpus," *Language Resources and Evaluation*, vol. 41, pp. 181–190, 2007.
- [22] M. Yousefi and J. H. L. Hansen, "Block-based high performance CNN architectures for frame-level overlapping speech detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 28–40, 2021.
- [23] V. A. M. Filho, D. A. Silva, and L. G. D. Cuozzo, "Joint discriminative embedding learning, speech activity and overlap detection for the DIHARD speaker diarization challenge," in *Interspeech 2018, Hyderabad, India*, 2018, pp. 2818–2822.
- [24] V. Andrei, H. Cucu, and C. Burileanu, "Detecting overlapped speech on short timeframes using deep learning," in *Interspeech 2017, Stockholm, Sweden*, 2017, pp. 1198–1202.
- [25] J.-W. Jung, H.-S. Heo, Y. Kwon, J. S. Chung, and B.-J. Lee, "Three-class overlapped speech detection using a convolutional recurrent neural network," in *Interspeech 2021, Brno, Czech Republic*, 2021, pp. 3086–3090.
- [26] 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, Graz, Austria*, 2019, pp. 978–982.
- [27] Y. Fujita, N. Kanda, S. Horiguchi, K. Nagamatsu, and S. Watanabe, "End-to-end neural speaker diarization with permutation-free objectives," in *Interspeech 2019, Graz, Austria*, 2019, pp. 4300–4304.
- [28] Y. Takashima, Y. Fujita, S. Watanabe, S. Horiguchi, P. Garcia, and K. Nagamatsu, "End-to-end speaker diarization conditioned on speech activity and overlap detection," in *SLT 2021, Shenzhen, China*, 2021, pp. 849–856.
- [29] N. Ryant, P. Singh, V. Krishnamohan, R. Varma, K. Church, C. Cieri, J. Du, S. Ganapathy, and M. Liberman, "The third DIHARD diarization challenge," in *Interspeech 2021, Brno, Czech Republic*, 2021, pp. 3570–3574.
- [30] H. Bredin and A. Laurent, "End-to-end speaker segmentation for overlap-aware resegmentation," in *Interspeech 2021, Brno, Czech Republic*, 2021, pp. 3111–3115.
- [31] D. Charlet, C. Barras, and J. Lienard, "Impact of overlapping speech detection on speaker diarization for broadcast news and debates," in *ICASSP 2013, Vancouver, Canada*, 2013, pp. 7707–7711.
- [32] S. Zhang, C. Liu, H. Jiang, S. Wei, L. Dai, and Y. Hu, "Feed-forward sequential memory networks: A new structure to learn long-term dependency," *CoRR*, vol. abs/1512.08301, 2015.
- [33] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," in *Interspeech 2018, Hyderabad, India*, 2018, pp. 1086–1090.
- [34] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," in *ICASSP 2015, South Brisbane, Australia*, 2015, pp. 5206–5210.
- [35] E. Vincent, S. Watanabe, A. A. Nugraha, J. Barker, and R. Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech & Language*, vol. 46, pp. 535–557, 2017.
- [36] L. Seps, "NanoTrans - editor for orthographic and phonetic transcriptions," in *TSP 2013, Rome, Italy*, 2013, pp. 479–483.
- [37] F. Landini, J. Profant, M. Diez, and L. Burget, "Bayesian HMM clustering of x-vector sequences (VBx) in speaker diarization: Theory, implementation and analysis on standard tasks," *Computer Speech & Language*, vol. 71, 2022.