MSc in NLP Supervised Project

Université de Lorraine

IDMC

Speaker diarization with overlapped speech
Realization report

Authors:
Justine Diliberto
Cindy Pereira
Anna Nikiforovskaja

Supervisor:
Md Sahidullah, MULTISPEECH
Reviewer:
Imran Sheikh, MULTISPEECH

June 21, 2021

Abstract

Our project aims at improving speaker diarization with overlapped speech. This report describes the second part of our project, the experimentation phase, whereas the first phase was focused on the comprehension of the subject and the bibliographical research. We first discuss the speaker diarization in general, and speaker diarization with overlapped speech in particular. We also analyze the acoustic characteristics of overlapped speech and the impact of overlapped speech on speaker diarization. Then, we investigate this impact both in terms of performance and influence on acoustic features. Finally, several models based on x-vector analysis are explored for identification of overlapped speech. We state the problem through classification and regression, and we find out that in the case of our imbalanced data, classification methods work better and can be further improved to have better results.

Acknowledgement

We would like to express our gratitude to our supervisor for his assistance at every stage of the project.

Experiments presented in this paper were partially carried out using the Grid'5000 testbed[1], supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations.

[1] https://www.grid5000.fr

Contents

List of Abbreviations

BLSTM	Bidirectional long short-term memory
CNN	Convolutional neural network
DER	Diarization error rate
ERROR	Speaker error
FA	False alarm
GRU	Gated recurrent unit
JER	Jaccard error rate
MFCC	Mel-frequency cepstral coefficients
MISS	Missed speech
NOV	Non-overlapped speech
OV	Overlapped speech
SAD	Speech activity detection
SD	Speaker diarization
SGD	Stochastic gradient descent
SVC	Support vector machine classifier
TDNN	Time delay neural networks
UAR	Unweighted average recall

Chapter 1

Introduction

The aim of this report is to experiment with overlapping speech, a recurrent issue in *speaker diarization*, by exploring the consequences on performance, on acoustic features, and by evaluating overlap detector methods with x-vectors (Snyder et al., 2018).

This first chapter recalls the definitions of speech signal and speaker diarization (SD). Then the components, applications, and issues of SD are exposed. Finally, the contribution, scope, and organization of this report are presented.

1.1 Speech signal and speech technology

The sound is a sequence of vibrations (Diliberto, Pereira & Nikiforovskaja, 2021). Sound coming from our phonatory system constitutes the speech. The speech signal is stored as a sequence of samples, encoded in different formats. The number of samples per second is known as the sampling rate.

Speech technology involves the processing of speech signal by a machine (Rudnicky, Hauptmann & Lee, 1994). Speech sounds are analyzed by computing short-term characteristics which represent acoustic and prosodic information. These components are then compared to stored patterns to recognize spoken words, speakers, emotions, and language.

1.2 What is speaker diarization?

Speaker diarization designates the task of finding *who spoke when* in an audio recording containing several speakers' voices. This is the unsupervised identification of each speaker within an audio stream and the durations during which each speaker is speaking (Anguera et al., 2012).

Speaker diarization is a relatively new field and thus is still in need of research and improvement. Some competitions such as DIHARD (Ryant et al., 2019b) and the Rich Transcription Evaluation by the American National Institute of Standards and Technologies (Sadjadi et al., 2017) are organized to promote research in this field.
[Paragraph taken from our previous report (Diliberto, Pereira & Nikiforovskaja, 2021)]

1.3 Components of speaker diarization system

As a reminder, the components of a typical speaker diarization system are shown in Fig. 1.1. The main steps are: preprocessing, segmentation, embedding extraction, cluster initialization, splitting or merging tools & cluster distance calculation, and stopping criterion. Each step is explained in more details in our previous report (Diliberto, Pereira & Nikiforovskaja, 2021).

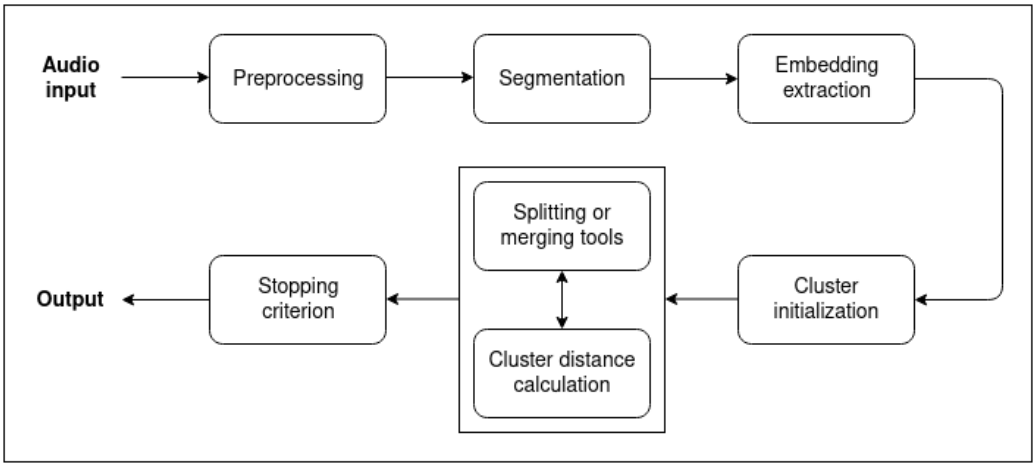

Figure 1.1: Components of a typical speaker diarization system.
[Figure taken from our previous report (Diliberto, Pereira & Nikiforovskaja, 2021)]

The uniform segmentation for state-of-the-art speaker diarization systems is followed by speaker embedding extractions. Commonly, x-vector embeddings are extracted and they are used with a clustering technique called agglomerative hierarchical clustering. In addition, re-segmentation is often applied for frame-level refinements of results.

1.4 Applications of speaker diarization technology

Speaker diarization is a useful tool and has many applications, such as (Tranter & Reynolds, 2006):

- enabling automatic speaker-attributed speech-to-text transcription for interviews, meetings, conferences or courtroom audiences;
- ameliorating the task of searching and indexing audio archives;
- improving accuracy and reducing computational cost of automatic speech recognition, when used as a preprocessing step;
- speaker spotting in voice assistant technology.

[Paragraph taken from our previous report (Diliberto, Pereira & Nikiforovskaja, 2021)]

1.5 Issues and Challenges

The state-of-art speaker diarization systems show reasonably good results in controlled conditions. However, the performance is degraded in realistic conditions due to the following reasons:

- overlapping speech;
- background noise;
- distance variations between speakers and microphones.

[Paragraph taken from our previous report (Diliberto, Pereira & Nikiforovskaja, 2021)]

1.6 Scope of the project

This project aims at studying SD with the particular issue of overlapped speech. We provide performance analyses to determine the effects of speech overlaps, and acoustic analyses to understand audio features and find discriminating ones. We also develop methods to detect overlapping speech through classification and regression.

1.7 Contributions

The experiments in the following chapters of this report are contributions to the scientific community, as they shed a new light on the old problematic of speech overlap.

The performance experiment deals with the impact of overlap on performance based on the DIHARD II dataset. After removing audio segments containing overlap, the baseline is run and its performances are calculated. This experiment shows the impact of overlap on non-overlapping segments on SD performance.

We investigate the impact of overlap on acoustic features that can weaken diarization results. 90 different acoustic features are computed on the DIHARD II development dataset files, which we divided between overlap and non-overlap segments. The study confirms that some features are useful to discriminate overlap from non overlap speech.

The performance of overlap detection methods based on x-vectors is studied. We build a system for training and testing them and implement both classical machine learning and deep learning-based methods. We apply both these methods to the classification and the regression interpretation of the problem. We show that classification interpretation of the problem works better, and x-vectors can display some information for overlap detection.

1.8 Organization of the project

The second chapter, Chapter 2, focuses on the method, aiming at describing the dataset, the metrics, the system, and the tools used for our study. Then, the performance analysis, consisting in the removal of overlapping segments, is described in Chapter 3. The acoustic impact experiments follow, aiming at identifying acoustic features, in Chapter 4. Chapter 5 deals with the development of overlap detectors based on x-vectors. Lastly, a conclusion is given in Chapter 6 with a summary and limitations of this report, as well as future work suggestions.

Chapter 2

Experimental setup

[Part of this chapter has been taken from our previous report (Diliberto, Pereira & Nikiforovskaja, 2021)]

2.1 Dataset description

2.1.1 Source of the dataset

The dataset used for our experimentations is the Second DIHARD Diarization Challenge dataset (Ryant et al., 2019a; Sahidullah et al., 2019). The DIHARD Speech Diarization Challenge is a series of yearly challenges on speaker diarization. To be more precise, the task is to automatically determine *who spoke when* in a multi-speaker environment and using only audio recordings.

2.1.2 Types of track conditions

The tracks used as input can be single channels or multi-channels. More information on how these channel types are recorded can be found in our previous report (Diliberto, Pereira & Nikiforovskaja, 2021).

Two different speech activity detection (SAD) are included in the dataset: reference SAD and system SAD. The reference SAD condition means that a speech segmentation is supplied, whereas system SAD stands for unprocessed audio.

These four conditions result in four different evaluation tracks: single channel using reference SAD; single-channel using system SAD; multichannel using reference SAD; multichannel using system SAD.

2.1.3 Origins of the tracks

Both the training and evaluation data for single-channel tracks are taken from eleven different domains such as audiobooks, broadcast interviews, child language, clinical, courtroom, map task, meeting, restaurant, socio-linguistic field and lab, and web videos. The combination of the tracks belonging to each domain is approximately two hours long.

The multichannel data comes from the CHiME-5 dinner party corpus. This corpus is composed of real conversational speech, recorded in the homes of the participants during dinner parties. Twenty parties were organized, each lasting 2 to 3 hours and to which attended 2 hosts and 2 guests. The recordings were performed by Microsoft Kinect devices (producing 4 channel linear arrays). The locations were divided into three areas, and each had two of these devices, which produces 24 channels in total.

Every segment containing personal identifying information was removed before the publishing of the dataset. The files are 16 bit FLAC type for single-channel and WAV type for multichannel, sampled at 16 kHz. Concerning the reference SAD files for the development set, they are given as rich transcription time marked files.

2.2 Evaluation metrics

The results of the diarization are compared to those of a human segmentation, which is called *ground truth*. When the results are different from the ground truth, an error is identified. Three kinds of error can occur: speaker error, false alarm, and missed speech.

Speaker error (ERROR) refers to the assignment of a segment to the wrong speaker. A false alarm (FA) occurs when a segment has been assigned to a speaker but actually contains no speech. Missed speech (MISS) is the term for a segment of speech that has not been assigned to any speaker.

Two kinds of error rates are usually computed to consider the results of a diarization task. Diarization error rate (DER) is the most famous one and is used to determine the proportion of reference speaker time that is not correctly attributed to a speaker. It is obtained by adding the segments having one of the three kinds of errors (false alarm, missed speech, and speaker error) and dividing their result by the total speaker time.

$$\text{DER} = \frac{\text{FA} + \text{MISS} + \text{ERROR}}{\text{TOTAL}}$$

Jaccard error rate (JER) is based on the Jaccard index, which aims at computing the optimal mapping between a reference and a system speaker pair. For each reference speaker, a specific JER can be drawn by dividing the sum of false alarms and missed speeches by the union of reference and system speaker segments. The JER is simply the average of every specific JERs.

$$\text{JER}_{ref} = \frac{\text{FA} + \text{MISS}}{\text{TOTAL}} \qquad \text{JER} = \frac{1}{N}\sum_{ref} \text{JER}_{ref}$$

2.3 Description of the speaker diarization system

2.3.1 State-of-the-art systems

The current state-of-the-art for speaker diarization systems is turning away from previously used i-vectors to obtain speaker characteristics for the embedding extraction step (Snyder et al., 2017). This new kind of system is focusing on the use of deep neural network embeddings to distinguish speaker differences, by mapping variable-length utterances to fixed-dimensional embeddings called x-vectors; however, the challenge is to gather enough training data.

2.3.2 Baseline system

The system we use is the baseline system supplied by the Second DIHARD Diarization Challenge (Ryant et al., 2019b). Four different tasks are performed, that is to say speech enhancement, beamforming, speech activity detection, and diarization.

Firstly, a model is trained to forecast the ideal ratio masks from log-power spectra features using a densely connected long short-term memory architecture, which is a model of Deep Neural Network particularly useful to make predictions.

Then, weighted delay-and-sum beamforming — a mathematical technique to identify the distance and orientation of sound waves caught by a microphone — is carried out.

After that, speech activity detection for tracks 2 and 4 is completed thanks to WebRTC's SAD, as found in the *Py-webrtc* Python package (see 2.1.2).

Finally, the diarization is achieved by isolating each recording into small overlapping segments, extracting x-vectors, scoring using probabilistic linear discriminant analysis, and clustering with agglomerative hierarchical clustering (see 1.3).

2.4 Libraries and tools

This section presents the libraries and tools used in the development of this project.

We use AudioSegment from the *Pydub* library (Robert, Webbie, et al., 2018) to arrange the files for the acoustic analysis.

Audiofile library (Wierstorf, 2019) enables us to read audio files in Python.

The features we analyze are computed with the libraries *Parselmouth* (Jadoul, Thompson & de Boer, 2018) and *Opensmile* (Eyben, Wöllmer & Schuller, 2010).

The plots are built using the library *Matplotlib* (Hunter, 2007) with data in *Pandas* (McKinney, 2010) format.

Basic statistics are calculated by the *Numpy* library (Harris et al., 2020).

Sklearn is used to work with data for machine learning, to implement classical machine learning methods, and to train them to predict speech overlap (Pedregosa et al., 2011).

Finally, *PyTorch* enables us to implement and train deep learning methods (Paszke et al., 2019).

Chapter 3

Performance impact

Different experiments are performed to understand how overlapped speech impacts the results of diarization. This chapter focuses on the overlapped speech removal experiment. More precisely, the segments containing overlap are removed to measure the performance of the baseline on the "cleaned" dataset. This is done using the development dataset from DIHARD II, presented in 2.1 (Ryant et al., 2019b).

3.1 Finding and removing overlapped speech segments

The first step of the experiment was to identify the segments containing overlapped speech. There is a .rttm file for each audio track, which contains the precise time of beginning and the length of speech for each of the speakers. These .rttm files are provided as a part of the corpus, because the first track condition is a reference SAD (see 2.1.2). By comparing these files, the overlapping segments can be computed.

Then, the objective was to discard the segments containing overlap. The .uem files, containing the beginning and end of each audio track, were modified to keep only segments without any overlapping speech.
After that, .rttm and .sad files, which are files containing beginning and end of speech segments for each speaker, were adapted according to the new .uem files. If a .rttm segment did not belong to the new .uem segments, even partially, it was removed. The same logic was applied to .sad files.

The new .uem segments were used to cut the .flac files and remove any speech overlap.

Using these newly created .rttm, .uem, .sad and .flac files, we were finally able to run the baseline.

3.2 Results on the new data

The results we obtained thanks to the baseline were quite surprising, as the median DER did not decrease in most cases. The dataset contains 12 audio categories, as said in 2.1.3 which are "audiobooks", "broadcast interview", "child", "clinical", "court", "maptask", "meeting", "restaurant", "socio field", "socio lab" and "webvideo".

Among these categories, only two have an unchanged or reduced error rate. The first one, "audiobooks", contains no overlap so it is logical that the DER did not change after removing overlap segments. The second one, "webvideo", has a slightly better error rate when compared to the category results with overlap segments.

The performances in terms of overall DER by category, on original data and data with overlap removed, as well as the average percentage of overlap are presented in the following Table 3.1.

Category	DER on original data	DER on overlap removed	Percent of overlap
Audiobooks	4	1.3	0
Broadcast interview	9	14.1	0.9
Child	31.7	37.5	7.5
Clinical	18.5	40.5	2.4
Court	16.3	29.3	1.6
Maptask	6.7	28.2	2
Meeting	34.1	49	21.3
Restaurant	50.5	59	21.4
Socio field	14.7	35.4	5.7
Socio lab	10.4	29.7	3.7
Webvideo	38.1	35.3	17.7

Table 3.1: Average DER score by category on original data and dataset with overlap removed (DIHARD II development dataset) and average percentage of overlap by category. DER worsening does not seem to depend on the percentage of overlap.

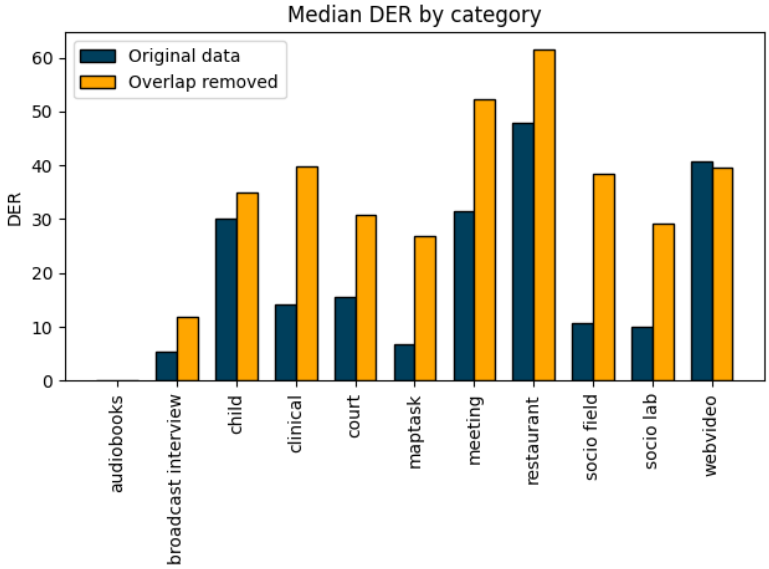

Figure 3.1: Median DER for each audio category. Most DER did not improve after having removed overlapped segments.

This table shows no correlation between the average percentage of overlap and the evolution of average DER for a category. For example, the category "maptask" with 2% of average overlap obtained an average of 28.2 DER after removing overlap, whereas it scored only an average of 6.7 DER with the original data. Another example, with a high percentage of overlap such as the category "meeting" which has more than 20% of overlap, the average DER worsened by 15 points.

To visualize how each average error rate evolves, Fig. 3.1 shows the median of each category before and after removing overlap.

3.3 Understanding the results

With such surprising results, the codes and process were reviewed to look for any mistake.

We made a graph to find any correlation between the length of a segment and its performance (see Fig. 3.2). For each file, we computed the length of the audio segments that have been removed and compared it to the difference of DER from original data to non-overlap. There seems to be no real correlation in the graph, so a shorter audio cannot explain why the new performance measures are so low.

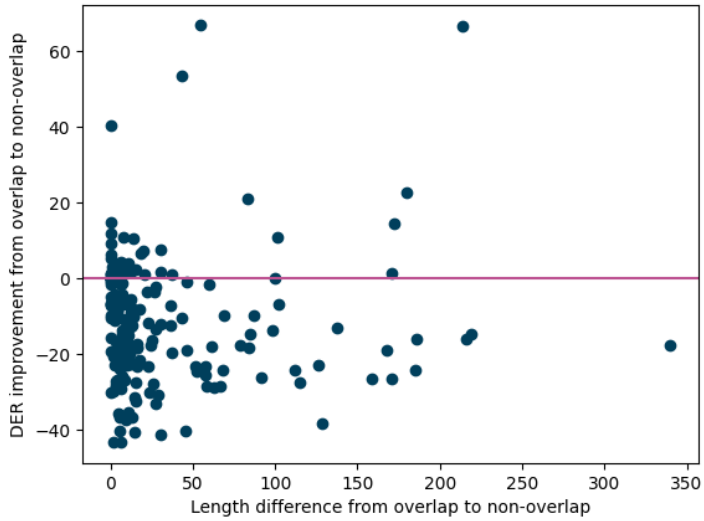

Figure 3.2: Mapping of the DER improvement and the length difference for each file (from the unchanged dataset to the segments without overlap). It is not possible to identify a clear correlation between the amount of speech removed and the worsening of the performance.

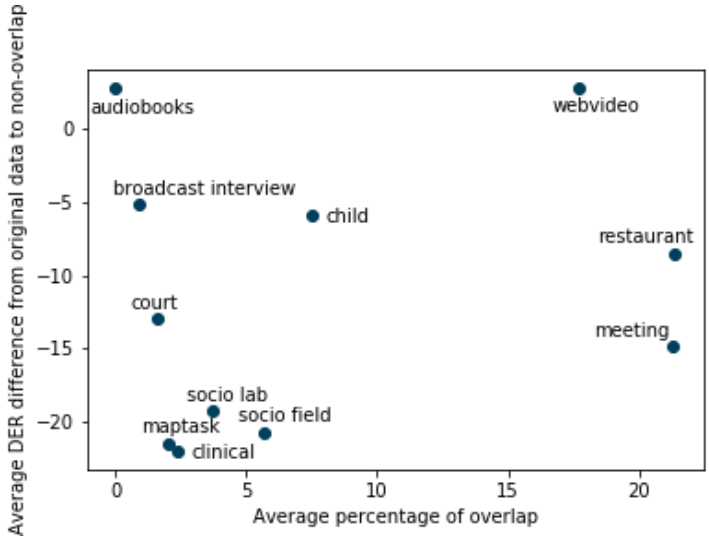

Figure 3.3: Mapping of the average DER difference and the average percentage of overlap by category. We cannot observe a clear correlation between the percentage of overlap and the DER worsening.

In addition to this, we created Fig. 3.3 to compare the average percentage of overlap on the original data to the average DER difference between original data and data without overlap for each category. First, we computed the average DER per category, then we subtracted the scores from non-overlap DER to the scores from original DER. Finally, this difference is compared to the average percentage of overlap. Once more, we do not observe a clear correlation which could explain the DER worsening.

3.4 Summary

This performance experiment resulted differently from what we expected, as we thought removing overlap regions should reduce diarization errors. Different studies have been made to understand why the performance was so low, but no correlation was found to explain it.

The cause for poor diarization results can be a combination of different elements, such as the length of audio removed and the percentage of overlap, or it can be a more complex one.

As said in our previous report, "when an amount of overlap is big, it makes it harder for the model even to perform diarization on the non-overlap regions, even though those regions are usually easier for the model", which means the overlap has an influence on the whole dataset (Diliberto, Pereira & Nikiforovskaja, 2021).

In addition to this, it needs to be recalled that overlap regions have been removed but noises, such as background noise, are still present and can make the task more difficult.

From these results, we can infer that removing overlap regions and retraining the model on the dataset without overlap is not a viable approach.

This experiment differs from the results shown in the tables from our last report, as these results were obtained without running the baseline again; in other words, without retraining the model on the newly obtained data without overlapping segments (Diliberto, Pereira & Nikiforovskaja, 2021).

To further explain the results from this chapter, the acoustic features are explored in the next part. We identify features impacted by overlap and having a direct link with diarization errors.

Chapter 4

Acoustic impact

Acoustic features can be impacted by overlapping segments and reduce the performance of diarization (Diliberto, Pereira & Nikiforovskaja, 2021).

Thus we have decided to identify which acoustic features are impacted by overlapping speech. It will help to understand why it is so difficult to apply a speech diarization method on overlap speech samples. Experiments are run on the DIHARD II development dataset to compare many features computed on the overlap and non-overlap samples.

In this section, features will be called discriminative when they are impacted by overlap.

4.1 Splitting audio files

The first step was to split the audio files. We needed to have separated files with or without overlapping speech. Using .rttm files, we found the overlap or non-overlap segments in single audio channels by calculating the periods in which more than one speaker was active. Then, we joined all non-overlapped segments together and exported them using AudioSegment from the *Pydub* library, and we did the same thing for overlapped segments. From one audio, we obtained three files: the complete audio, with both overlap and non-overlap segments, the non-overlapping audio, and the overlapping one.

With these files, we could then compute a few acoustic features and compare the results between the overlap or non-overlap versions of the same file. This would allow us to understand if, with the same parameters (speakers, environment, etc.), there is a difference between the segments when only one speaker is talking, and the ones when at least two speakers are talking at the same time.

At first, we used every single channel audio file from the development part of the dataset to get a large amount of data (192 audio files in total). However, we observed that the "audiobook" files were composed of only one speaker and that we wouldn't find any overlap segment in this category. We decided to remove these files because it was not possible to compare the overlap and non-overlap versions of the same audio.

4.2 Calculated features

We used two different libraries to compute the features. With *Parselmouth* library we computed the pitch, which was identified in our previous report as being one of the main features impacted by overlapping speech (Diliberto, Pereira & Nikiforovskaja, 2021). *OpenSmile* library enabled us to compute 89 different features on our audio files (see the list of features in Fig. 4.1).

Category	Detailed feature
F0 semitone from 27.5Hz	amean stddev Norm percentile 20.0 percentile 50.0 percentile 80.0 pctl range 0-2 mean Rising Slope stddev Rising Slope mean Falling Slope stddev Falling Slope
Loudness	amean stddev Norm percentile 20.0 percentile 50.0 percentile 80.0 pctl range 0-2 mean Rising Slope stddev Rising Slope mean Falling Slope stddev Falling Slope peaks per sec
Spectral Flux	amean stddev Norm V amean V stddev Norm UV amean
MFCC	MFCC1 amean MFCC1 stddev Norm MFCC2 amean MFCC2 stddev Norm MFCC3 amean MFCC3 stddev Norm MFCC4 amean MFCC4 stddev Norm MFCC1V amean MFCC1V stddev Norm MFCC2V amean MFCC2V stddev Norm MFCC3V amean MFCC3V stddev Norm MFCC4V amean MFCC4V stddev Norm
Jitter Local	amean stddev Norm
Shimmer Local	amean stddev Norm
HNRdBACF	amean stddev Norm
logRel-F0-H1	H2 amean H2 stddev Norm A3 amean A3 stddev Norm

F1	Frequency amean Frequency stddev Norm Bandwidth amean Bandwidth stddev Norm Amplitude logRelF0 amean Amplitude logRelF0 stddev Norm
F2	Frequency amean Frequency stddev Norm Bandwidth amean Bandwidth stddev Norm Amplitude logRelF0 amean Amplitude logRelF0 stddev Norm
F3	Frequency amean Frequency stddev Norm Bandwidth amean Bandwidth stddev Norm Amplitude logRelF0 amean Amplitude logRelF0 stddev Norm
Alpha Ratio	V amean V stddev Norm UV amean
Hammarberg Index	V amean V stddev Norm UV amean
Slope	V0-500 amean V0-500 stddev Norm V500-1500 amean V500-1500 stddev Norm UV0-500 amean UV500-1500 amean
Voicing	Voiced segments per sec mean voiced segment length per sec stddev voiced segment length per sec mean unvoiced segment length per sec stddev unvoiced segment lenght per sec
Equivalent sound level	
Pitch	

Table 4.1: Acoustic features computed.

4.3 Visual results

These 90 features were computed on each audio (complete, overlap, and non-overlap) of each category, and visualized with histograms to get a first opinion on what are the most discriminative features. For this analysis, we also removed the "broadcast interview" audio files, because the mean of overlapped percentage in this category is 0.86%.

As can be seen in Fig. 4.1, the values of pitch look very different when dealing with overlapped speech and with non-overlapped. Pitch results from the tension of the vibration of the vocal folds and is closely related to the fundamental frequency F0 (Aung & Puts, 2020). For each file, the pitch value of overlapped segments is at least 100 Hz higher than the one for non-overlapped segments. We can assume this disparity is one of the reasons why it is so difficult to apply diarization on overlapped speech.

As for the pitch, we can see in Fig. 4.2 a large variation in loudness peaks per second for overlapped and non-overlapped. When multiple speakers are talking at the same time, it seems there are more loudness peaks in the speech.

Similarly, Fig. 4.3 represents the comparison between the voiced segments per second in overlapped and non-overlapped speech samples. The number of voiced segments looks higher for each overlapped audio file than the number for the corresponding non-overlapped file.

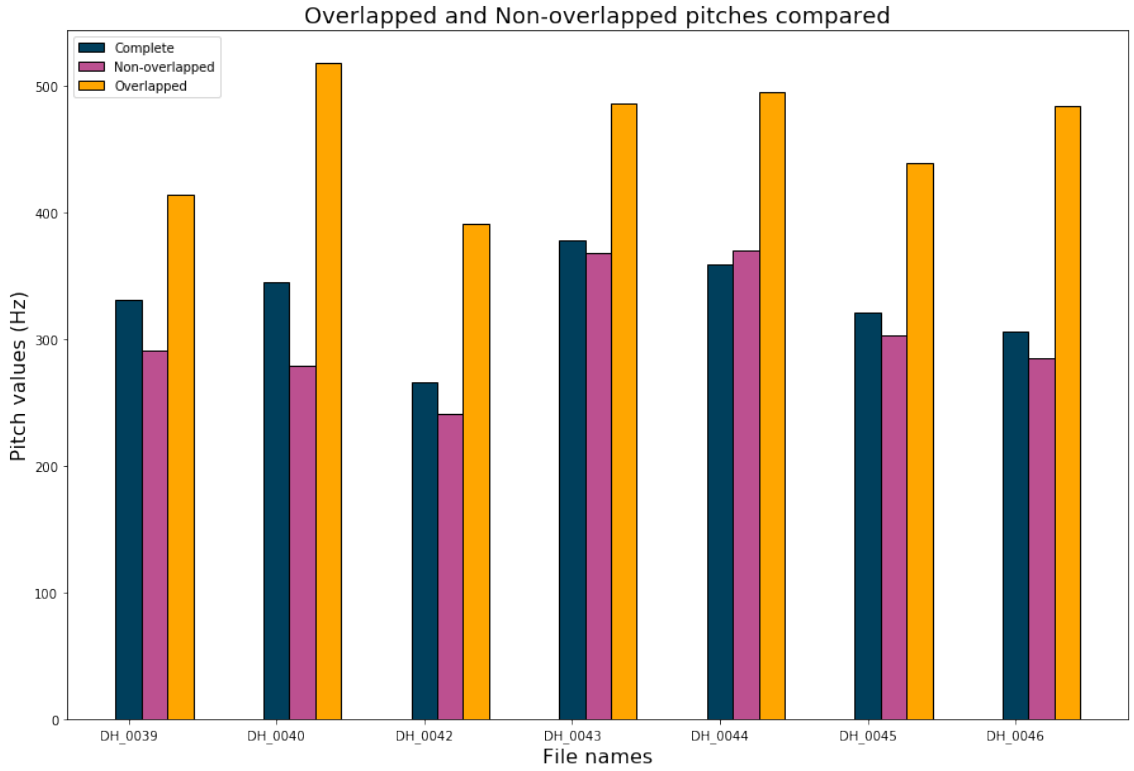

Figure 4.1: Pitch values for overlapped and non-overlapped speech samples in the category "restaurant".

4.4 Statistical results

For the statistical analysis of features, we decided to keep only the files with a percentage of overlap higher than 20% to be sure our results were consistent. According to this condition, the analysis was performed on 26 files from the following categories: 3 from "child", 6 from "meeting", 7 from "restaurant", 10 from "webvideo". Table 4.2 shows the statistical values of some features we selected using these 26 files.

In our table, the ratio corresponds to the quotient of the non-overlap mean (or median) over the overlap mean (or median). When this ratio is higher than 1, it means that the values increase when there is overlap, in comparison to non-overlap. The higher the ratio is, the more discriminative the feature is (i.e. values differ a lot from "normal values" when there is overlapping speech). Oppositely, if it is lower than 1, it means that the values decrease, and the lower the ratio is, the more discriminative the feature is. NOV refers to non-overlapped speech while OV refers to overlapped speech. We listed here the most discriminative features.

4.5 Discriminative features

One can see in Table 4.2 that the pitch value differs a lot between overlap and non-overlap segments. There is an increase of 46% in the mean of overlap in comparison to the non-overlap one

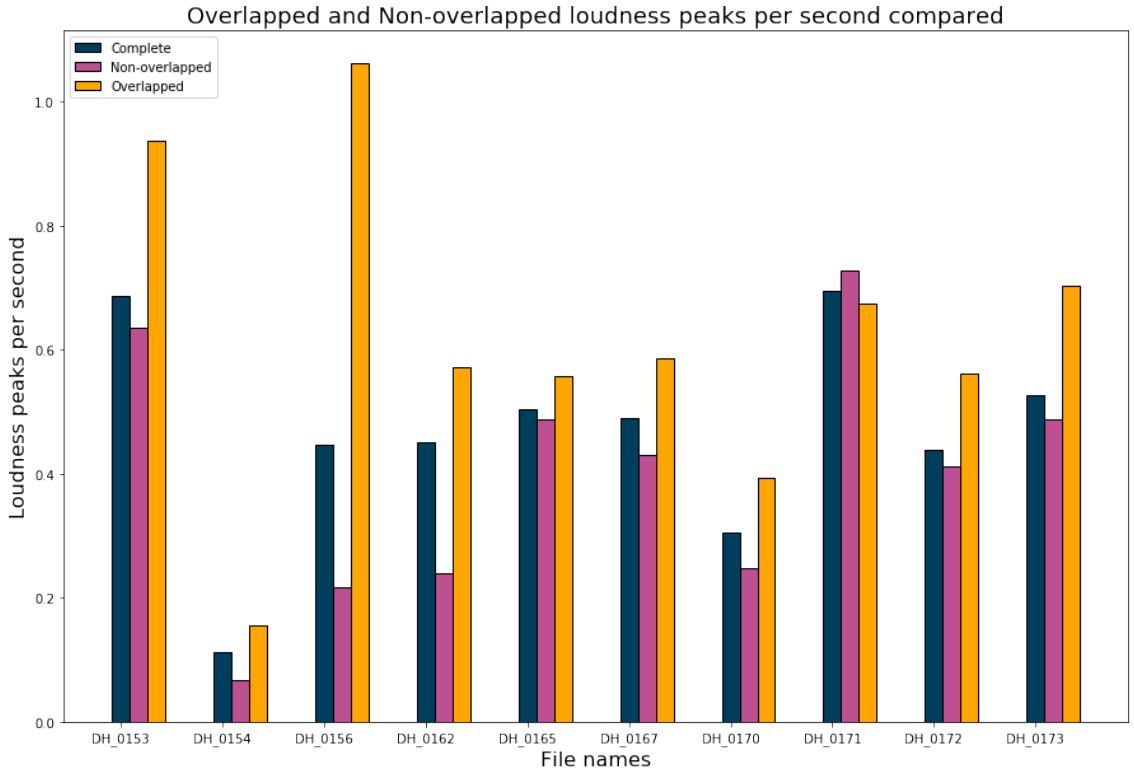

Figure 4.2: Loudness peaks per second for overlapped and non-overlapped speech samples in the category "webvideo".

and 71% for the median. However, the standard deviation remains similar which means that the variation of the values doesn't increase.

The Spectral Flux measures the degree of variation in the spectrum across time (Sadjadi & Hansen, 2013). It is very discriminative for overlap as we can see: the mean increases by 47% and the median by 60%.

Similarly, the overlap values of the Spectral Slope — the logarithmic power of the Mel band — are really distinct from the non-overlap ones (increase of 39% for mean and 65% for median), as well as for the fundamental frequency F0 (increase of 35% for mean and 44% for median) (Zheng, Wang & Jia, 2020).

As could be supposed, the loudness, the length of unvoiced segments, and the rate of voiced segments are also impacted by overlap. When many people talk at the same time, the ambient is noisier. The voicing (voiced and unvoiced segments) corresponds to the vibration (or not) of the vocal folds. The more people are talking, the more voiced speech is detected, which explains why the mean length of unvoiced segments decreases with overlap while the number of voiced segments per second increases.

4.6 Summary

At this stage, we identified six features that are impacted a lot by overlap: pitch, spectral flux, fundamental frequency, loudness, spectral slope, and unvoiced or voiced segments.

These features compute values that are dissimilar to non-overlap ones, as they are greatly higher or lower. From this analysis, we can conclude they certainly impact the performance of speaker diarization methods applied on overlapped speech. To improve state-of-the-art, a new perspective could be to base a model on non discriminative features.

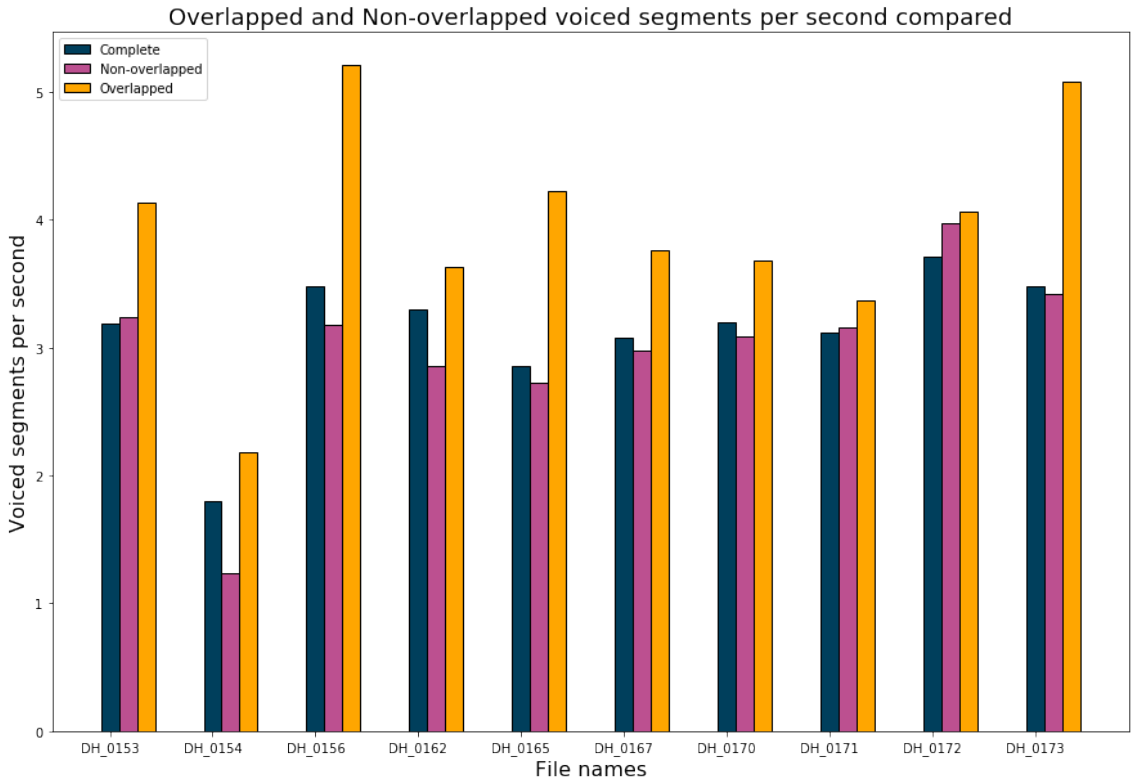

Figure 4.3: Voiced segments per second for overlapped and non-overlapped speech samples in the category "webvideo".

Feature	Mean			Median			Std Dev	
	NOV	OV	Ratio	NOV	OV	Ratio	NOV	OV
Pitch	439	641	1.46	367	628	1.71	218	233
SpectralFlux: amean	0.32	0.47	1.46	0.23	0.36	1.60	0.22	0.30
F0: meanFallingSlope	98	132	1.35	88	127	1.44	43	56
Loudness: amean	0.73	0.99	1.34	0.59	0.82	1.39	0.46	0.56
SlopeV0-500: amean	0.015	0.02	1.39	0.008	0.01	1.65	0.03	0.03
Unvoiced seg len: mean	0.41	0.24	0.59	0.31	0.19	0.61	0.25	0.13
Voiced seg/sec	1.93	2.78	1.44	2.01	2.62	1.32	0.56	0.76

Table 4.2: Statistical results of some features based on the study of 26 files.

Chapter 5

Overlap detectors

5.1 Introduction

Overlapping segments can drastically reduce the quality of performance (Diliberto, Pereira & Nikiforovskaja, 2021). Thus we have decided to experiment on overlap detection to improve the performance.

The main idea is to develop a classifier which would detect if a segment of the recording contains overlap or not.

Overlap detection is usually performed in one of these three ways: with signal processing, statistical methods or deep learning. The most popular and effective classifiers currently are deep learning based (Diliberto, Pereira & Nikiforovskaja, 2021). The LSTM-based (Bullock, Bredin & Garcia-Perera, 2020; Yoshioka et al., 2018) and CNN-based methods (Kunešová et al., 2019; Andrei, Cucu & Burileanu, 2019; Málek & Žďánsky, 2020) are especially popular.

X-vectors are embeddings for recording segments which were trained by CNN. It was shown that usage of x-vectors can improve the performance, as they might contain more information than only about single speakers (Málek & Žďánsky, 2020). We want to further explore their usage for overlap detection with different models.

In this chapter we describe the prepared data, the models used around x-vectors, the testing system architecture, and finally the evaluation of the chosen methods.

5.2 Classification methods idea

The main idea was to use x-vectors as the embedding vectors of the audio segments to classify those segments into overlap and non overlap. However, as shown in Fig. 5.1 the percentage of overlap in the segment is quite often not 100%. That is why we have decided to divide the segments not into two groups for classification, but into several, so that we could have more information about the segments automatically.

5.3 Experimental setup

5.3.1 Data organization

We exploited already computed x-vectors on development data as a source material. We used the following categories of recordings from the dataset to train and evaluate our methods: "webvideo", "meeting", and "restaurant". These categories contain quite a lot of overlap segments which are of a decent quality, this explains why we used them.

For each segment for which we had an x-vector, we calculated the ratio of the overlap part on

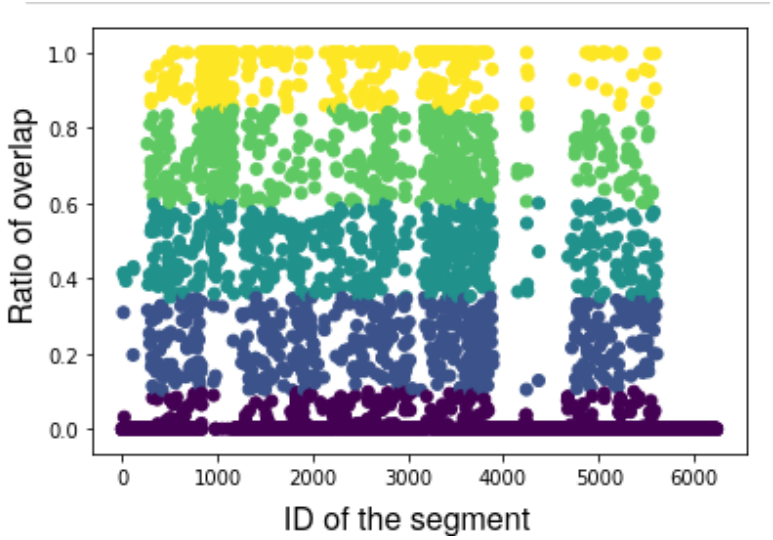

Figure 5.1: Distribution of ratio of overlap. Colors show the way we have divided the ratios into classes.

this segment. And therefore we produced the ratio which can be predicted.

Also, for each ratio we decided to which class it belongs, so that we had a classification problem, as our final goal is to predict if there is or is not an overlap.

Afterwards, we built a dataset with the following parameters: number of segments information to take before the goal segment, number of segments information to take after the goal segment, and the identification numbers of the file where the segments are taken from. Then we had two variants: either to have classes to predict or the ratio itself. The ratio intervals of the classes are calculated using the following formula. If the ratio is smaller than 0.1 then the class is 0. Otherwise the classes are equally distributed on the segment [0.1, 1] and are calculated with the formula $1 + \min(3, \lfloor (r - 0.1) \cdot 4 \rfloor)$, where r is the overlap ratio. Therefore we have 5 classes of overlap.

The development data is divided into train and test parts in the proportion of 7:3. We divided the segments into these two groups such that all the segments of one file belong to the same group.

5.3.2 Architecture

We used Object Oriented Programming to organize our code so that it was easy to modify. There are base classes for regression and classification methods, which contain several evaluation methods inside.

From each base class, classes from *Sklearn* (Pedregosa et al., 2011) and *Pytorch* (Paszke et al., 2019) algorithms are inherited. The class for *Sklearn* algorithm allows to run all the needed functionalities, by passing the name of the *Sklearn* method and its parameters. The class for *Pytorch* algorithms allows to run *Pytorch* models by passing the base model itself, optimizer, epochs and the device to run on.

The described above scheme of the classes can be seen on Fig. 5.2.

All the introduced algorithms take x-vectors as an input as can be seen in Fig. 5.3. However, classification algorithms return the class of overlap, while regression ones return the ratio of overlap.

5.3.3 Evaluation methods

As we had quite imbalanced classes, we have decided to use UAR for evaluation of classification results. UAR stands for Unweighted Average Recall and is an unweighted mean value of recall for

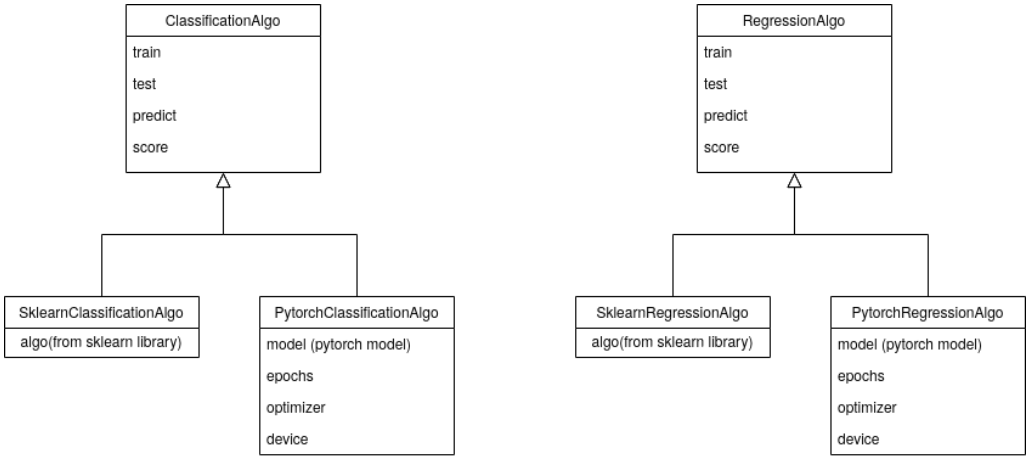

Figure 5.2: Classes structure diagram of the project.

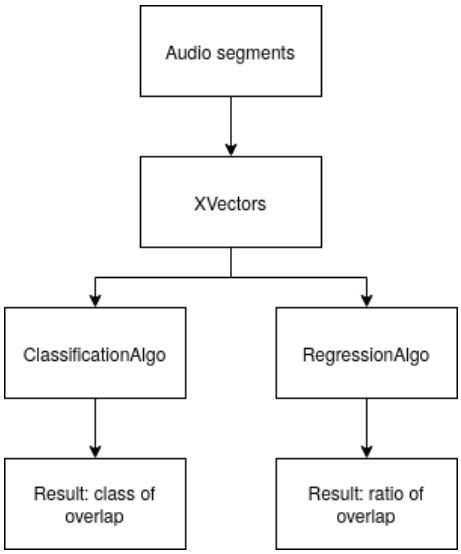

Figure 5.3: Block scheme of usage of the x-vectors.

each class. This evaluation methods helps to check that each class is predicted well enough, not only the majoring one.

For evaluation of regression results we used R2 score, which is commonly used for regression tasks and is a coefficient of determination. R2 score is a score based on the proportion of the variance.

5.4 Models tested

First we used some simple methods as a baseline for classification on x-vectors. They are machine learning methods from *Sklearn* library such as RidgeClassifier, SVC, SGDClassifier and DecisionTreeClassifier.

As for *Pytorch* based models, we created a simple linear network for another baseline solution. Besides we used the TDNN model introduced in a previous research as it was already designed for x-vectors (Málek & Žďánsky, 2020). We call this model TDNNBasedModel, as it mainly consists of TDNN layers which are CNN-based layers applied to plain vectors (Krizhevsky, Sutskever & Hinton, 2012). The main idea is to gather a context of the value in the vector with some weights for each value and create the next vector this way. If the context is symmetrical it can be done with a one-dimensional convolution layer.

As we have also seen, BLSTM usually work well; however, they have never been applied to x-vectors. So we tried to apply the model from a previous research, which consists of BLSTM layers (Bullock, Bredin & Garcia-Perera, 2020). LSTM layers are recurrent neural network layers which iterate through the sequence and on each iteration take the previous output and use it as a new input. Bidirectional LSTM or BLSTM repeats this process in the other direction. We also tried a similar model with GRU units instead of BLSTM units. GRU stands for Gated Recurrent Unit, and is a simpler version of LSTMs. We call these models BLSTMBasedModel and GRUBasedModel respectively.

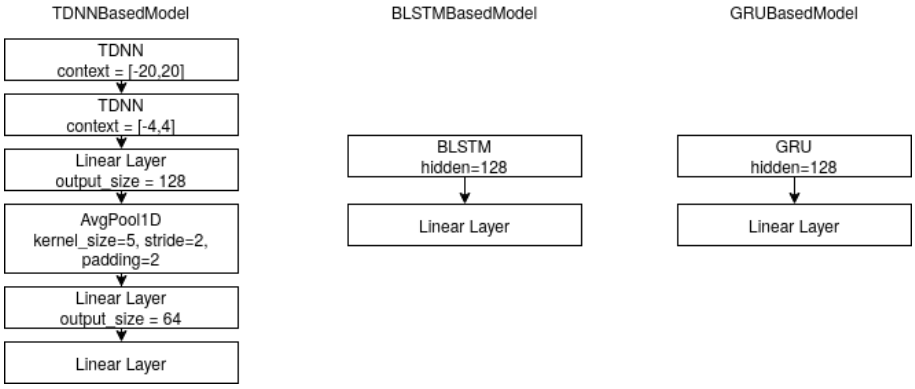

Figure 5.4: Used models parameters and layer structure.

The *Pytorch* models used are described by the layer structure in Fig. 5.4. The last layer in each model outputs a tensor, which length is either the number of classes if we perform classification, or 1 if we perform regression. BLSTM and GRU based models iterate over different x-vectors, which are taken as a context.

When used for classification, *Pytorch* models are trained with cross-entropy loss, while when used for regression they are trained with mean squared error loss. The parameters of optimizers were tuned for each model separately.

As classes were not balanced well (60% of the data corresponds to class 0, when there are 5 classes), during training we extracted segments from the classes, so that the probability to take a segment from one class was equal to the probability to take a segment from another class. This way we improved the performance for classification methods.

5.5 Evaluation results

Current results for classification experiments are shown in the Table 5.1. Table 5.2 shows the results for regression experiments. We ran those algorithms on bigger contexts consisting of 3 vectors of information for previous segments, 1 vector of information for the following segment, and smaller contexts which consist of 2 vectors of information for previous segments of a current one.

Classification results show that the deep learning based model performs much better with the increase of the context, while there might not be such a big difference for the simple machine learning methods. The top performers are RidgeClassifier and TDNNBasedModel; they do not have a big difference.

Regression results show that TDNNBasedModel improves with the increase of the context; however, the overall performance for all methods is low. The leaders here are Lasso and SVR. We believe the results here might be that low because of the imbalance of data, which was fixed for classification part but was not fixed for regression.

Method	UAR score on bigger context	UAR score on smaller context
RidgeClassifier	0.26	0.23
SVC	0.20	0.20
SGDClassifier	0.24	0.23
DecisionTreeClassifier	0.22	0.22
LinearNet	0.24	0.24
TDNNBasedModel	0.25	0.23
BLSTMBasedModel	0.23	0.20
GRUBasedModel	0.22	0.19

Table 5.1: Evaluation results for classification methods.

Method	R2 score on bigger context	R2 score on smaller context
Lasso	0.006	-0.051
SVR	0.070	0.052
SGDRegressor	-2e27	-1.5e27
DecisionTreeRegressor	-0.79	-0.808
LinearNet	-0.34	-0.333
TDNNBasedModel	-0.065	-0.124
BLSTMBasedModel	-0.498	-0.252
GRUBasedModel	-1.207	-0.551

Table 5.2: Evaluation results for regression methods.

5.6 Summary

We built a convenient system for training and testing models for overlap detection based on x-vectors. To that end, we implemented several methods; both classical machine learning methods and deep learning methods.

The results of the evaluation show us that classification-based methods work better for overlap prediction. We can also see that among deep learning methods, the TDNN-based is the best one and shows an improvement with the increase of the context. We acknowledge that x-vectors contain some information which can be used for overlap detection.

We have some more ideas on how to increase the performance of the introduced methods. The deep learning models can be increased in size. It also makes sense to take more data for training and to possibly perform some kind of data augmentation before. It would also be good to balance data via adding new overlapped segments, which could be achieved by data augmentation or an artificial audio recordings combination.

Chapter 6

Conclusion

6.1 Summary

The intent of this project was to come up with and test suggestions to improve speaker diarization with overlapped speech.

The first stage of this project was a bibliographic research, as described in our previous article (Diliberto, Pereira & Nikiforovskaja, 2021). This study of the state-of-the-art of diarization methods enabled us to comprehend the three main kinds of approaches: using signal processing, statistics, or more recently deep neural networks. Even if the new methods have good efficiency, the performance is reduced with overlapped speech, and there is still space for improvement.

This part of the project aimed at running experiments to further analyze and understand how speaker diarization is impacted by overlap, and finding suitable approaches to deal with overlapped speech.

To determine the impact on performance of an overlapped speech, we tried to remove overlapping segments and to run the system baseline. Unfortunately, the experiment did not perform as well as we expected. This can be explained by the fact that overlap has an influence even on non-overlap regions. Other causes for diarization errors can be found such as the presence of background noise.

We investigated the impact of overlapped speech on 90 acoustic features to determine if they can be a cause for the loss of performance in speaker diarization. We identified six features that have unexpected values when computed on overlap segments. This can explain the low efficiency results for speaker diarization. As they are trained with non-overlap values, models won't perform well on speech with features having these disparate values.

We introduced the possibility of using x-vectors to train overlap detectors and implemented several models to do that. We found out that classification interpretation of the problem of speech overlap detection allows to have better results than regression interpretation. We also showed that with the increase of segments in context we obtain higher results with deep learning methods. Finally, we introduced several possible improvements, as we consider the usage of x-vectors as promising.

In this report, we provided some suggestions about the improvement of speaker diarization with overlapped speech. We identified the reasons of the low performance of actual systems by exploring the results of overlap removal, and by computing and examining acoustic features. We finally built a convenient system of overlap detection by comparing different deep neural network models.

6.2 Limitations

The experiments were conducted on the DIHARD II dataset, which was designed for the purposes of the challenge on speaker diarization. Thus, the corpus is not representative of a real world situation, as the overlap amount has been chosen and the speaker number has been controlled.

The results of our experiments are impacted by background noise. The outcome could be better without any background noise; however, it is a normal phenomenon, which happens regularly in real world situations.

The performance of our overlap detector is reduced by the small size of our dataset. Moreover, this dataset contains a relatively low percentage of overlap (see Table 3.1), which can prejudice the training of the model.

6.3 Future work

The same experiments using a real world corpus need to be explored as an extension of this work.

In addition, using tools to remove background noise as a preprocessing task might be useful.

For further development of overlap detectors, it is desirable to test our experiments on larger corpora with a higher percentage of overlap.

Finally, the findings from our acoustic experiments can guide future works on speaker diarization with overlapped speech.

Bibliography

Andrei, V., Cucu, H. & Burileanu, C. (2019) Overlapped Speech Detection and Competing Speaker Counting – Humans Versus Deep Learning. *IEEE Journal of Selected Topics in Signal Processing*. 13, 850–862.

Anguera, X., Bozonnet, S., Evans, N., Fredouille, C., Friedland, G. & Vinyals, O. (2012) Speaker Diarization: A Review of Recent Research. *IEEE Transactions on Audio, Speech, and Language Processing*. 20 (2), 356–370.

Aung, T. & Puts, D. (2020) Voice pitch: a window into the communication of social power. *Current Opinion in Psychology*. 33, 154–161.

Bullock, L., Bredin, H. & Garcia-Perera, L. P. (2020) Overlap-aware diarization: Resegmentation using neural end-to-end overlapped speech detection. *2020 IEEE International Conference on Acoustics, Speech and Signal Processing*. Barcelona, Spain, 7114–7118.

Diliberto, J., Pereira, C. & Nikiforovskaja, A. (2021) Speaker diarization with overlapped speech; Bibliographical report.

Eyben, F., Wöllmer, M. & Schuller, B. (2010) Opensmile: The Munich Versatile and Fast Open-Source Audio Feature Extractor. *Proceedings of the 18th ACM International Conference on Multimedia*, 1459–1462.

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C. & Oliphant, T. E. (2020) Array programming with NumPy. *Nature*. 585 (7825), 357–362.

Hunter, J. D. (2007) Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*. 9 (3), 90–95.

Jadoul, Y., Thompson, B. & de Boer, B. (2018) Introducing Parselmouth: A Python interface to Praat. *Journal of Phonetics*. 71, 1–15.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012) Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*. 25, 1097–1105.

Kunešová, M., Hrúz, M., Zajíc, Z. & Radová, V. (2019) Detection of overlapping speech for the purposes of speaker diarization. *International Conference on Speech and Computer*, 247–257.

Málek, J. & Žďánsky, J. (2020) Voice-Activity and Overlapped Speech Detection Using x-Vectors. *International Conference on Text, Speech, and Dialogue*, 366–376.

McKinney, W. (2010) Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*. Ed. by S. van der Walt & J. Millman, 56–61.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. & Chintala, S. (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox & R. Garnett. Curran Associates, Inc., 8024–8035.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011) Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 12, 2825–2830.

Robert, J., Webbie, M., et al. (2018) Pydub.

Rudnicky, A. I., Hauptmann, A. G. & Lee, K.-F. (1994) Survey of current speech technology. *Communications of the ACM*. 37 (3), 52–57.

Ryant, N., Church, K., Cieri, C., Cristia, A., Du, J., Ganapathy, S. & Liberman, M. (2019a) Second dihard challenge evaluation plan. *Linguistic Data Consortium, Tech. Rep*.

— (2019b) The Second DIHARD Diarization Challenge: Dataset, Task, and Baselines. *20th Annual Conference of the International Speech Communication Association*, 978–982.

Sadjadi, S. O. & Hansen, J. H. L. (2013) Unsupervised Speech Activity Detection Using Voicing Measures and Perceptual Spectral Flux. *IEEE Signal Processing Letters*. 20 (3), 197–200.

Sadjadi, S. O., Kheyrkhah, T., Tong, A., Greenberg, C. S., Reynolds, D. A., Singer, E., Mason, L. P. & Hernandez-Cordero, J. (2017) The 2016 NIST Speaker Recognition Evaluation. *18th Annual Conference of the International Speech Communication Association*, 1353–1357.

Sahidullah, M. et al. (2019) The Speed Submission to DIHARD II: Contributions & Lessons Learned. *arXiv preprint arXiv:1911.02388*.

Snyder, D., Garcia-Romero, D., Povey, D. & Khudanpur, S. (2017) Deep Neural Network Embeddings for Text-Independent Speaker Verification. *18th Annual Conference of the International Speech Communication Association*, 999–1003.

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

Tranter, S. E. & Reynolds, D. A. (2006) An overview of automatic speaker diarization systems. *IEEE Transactions on audio, speech, and language processing*. 14 (5), 1557–1565.

Wierstorf, H. (2019) Audiofile.

Yoshioka, T., Erdogan, H., Chen, Z., Xiao, X. & Alleva, F. (2018) Recognizing Overlapped Speech in Meetings: A Multichannel Separation Approach Using Neural Networks. *19th Annual Conference of the International Speech Communication Association*, 3038–3042.

Zheng, C., Wang, C. & Jia, N. (2020) An Ensemble Model for Multi-Level Speech Emotion Recognition. *Applied Sciences*. 10 (1).