UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC
DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dept. of Computer Systems and Computation

Automatic speaker diarization based on deep learning and its application to audiovisual subtitling

Master's Thesis

Master's Degree in Artificial Intelligence, Pattern Recognition and Digital Imaging

AUTHOR: Cano Caravaca, Vicent

Tutor: Juan Císcar, Alfonso

Cotutor: Silvestre Cerdà, Joan Albert

ACADEMIC YEAR: 2022/2023

Abstract / Resum / Resumen

Abstract

Speaker Diarization is a constantly evolving research field. It focuses on the development of automatic systems capable of segmenting acoustic signals according to the speakers who intervene in them. This task is commonly explained, in a simple way, as being able to respond to the question: "Who spoke when?". This field has been based, for many years, on the use of classical audio processing techniques to perform the subtasks which composed Speaker Diarization. This fact has been changing during these last years due to the rise in popularity experienced by neural networks and, nowadays, Speaker Diarization systems are mainly based on deep learning techniques [1], such as Recurrent Neural Networks.

This work aims to explore the state of the art of Speaker Diarization in order to select some of the most promising techniques and adapt them to audiovisual media subtitling in Valencian and Spanish. The experimental evaluation is based on tasks and data used by the Machine Learning and Language Processing Group (MLLP) in recent projects and challenges. In particular, it is conducted with data from *Radio y Televisión Española (RTVE)* and *Corporació Valenciana de Mitjans de Comunicació (CVMC)*.

Resum

La diferenciació automàtica de locutors (SD, per la seua denominació en anglés: *Speaker Diarization*) és una àrea de recerca en constant evolució. Es centra en el desenvolupament de sistemes de segmentació automàtica de senyals acústics en funció dels locutors que hi intervenen. Esta tasca s'explica habitualment, de manera simple, en ser capaç de respondre a la pregunta: "Qui ha parlat en cada moment?". Durant anys aquest camp s'ha basat en la utilització de tècniques clàssiques de processament d'àudio per a les diferents subtasques que componien la diferenciació automàtica de locutors. Açò ha anat canviant durant els últims anys amb l'augment de la popularitat de les xarxes neuronals i, actualment, els sistemes de diferenciació automàtica de locutors es basen en tècniques d'aprenentatge profund [1], com ara xarxes neuronals recurrents.

En aquest treball es proposa fer una revisió de l'estat de l'art en diferenciació automàtica de locutors per tal de seleccionar algunes de les millors tècniques actuals i adaptar-les a la subtitulació de mitjans audiovisuals en valencià i castellà. L'avaluació experimental es basa en tasques i dades en les quals ha treballat recentment el Machine Learning and Language Processing Group (MLLP). En particular, es fa amb dades de *Radio y Televisión Española (RTVE)* i de la *Corporació Valenciana de Mitjans de Comunicació (CVMC).*

Resumen

La diarización de locutores (SD, por su denominación en inglés: *Speaker Diarization*) es una área de investigación en constante evolución. Se centra en el desarrollo de sistemas de segmentación automática de señales acústicas en función de los locutores que intervienen en ellas. Esta tarea se explica habitualmente, de manera simple, en ser capaz de responder a la pregunta: "¿Quién ha hablado en cada momento?". Durante años este campo se ha basado en la utilización de técnicas clásicas de procesamiento de audio para las distintas subtareas que componían la diarización de locutores. Esto ha ido cambiando durante los últimos años con el aumento de la popularidad de las redes neuronales y, actualmente, los sistemas de diarización de locutores se basan en técnicas de aprendizaje profundo [1], como redes neuronales recurrentes.

En este trabajo se propone hacer una revisión del estado del arte en diarización de locutores para así seleccionar algunas de las mejores técnicas actuales y adaptarlas a la subtitulación de medios audiovisuales en valenciano y castellano. La evaluación experimental se basa en tareas y datos en las cuales ha trabajado recientemente el Machine Learning and Language Processing Group (MLLP). En concreto, se hace con datos de *Radio y Televisión Española (RTVE)* y de la *Corporació Valenciana de Mitjans de Comunicació (CVMC)*.

Contents

Acronyms

AI Artificial Intelligence. 25

ASR Automatic Speech Recognition. 1–3, 5, 7, 13, 19, 25, 50

BLSTM Bidirectional Long Short-Term Memory. 11, 19, 30

CNN Convolutional Neural Network. 11, 17

CV Computer Vision. 11, 17, 50

CVMC Corporació Valenciana de Mitjans de Comunicació. iii–v, 3, 4, 25, 29, 49, 50

DER Diarization Error Rate. 14, 15, 33–36, 38–44, 46–50

DL Deep Learning. 10

DNN Deep Neural Network. 9, 16, 30

ECAPA Emphasized Channel Attention, Propagation and Aggregation. 17, 30, 49, 50

EEND End-to-End Neural Speaker Diarization. 30, 49, 50

FA False Alarm Rate. 14, 15

FNN Feed Forward Neural Network. 9, 10

ID Identification. 1, 4

JER Jaccard Error Rate. 14

LSTM Long Short-Term Memory. xi, 11, 19, 30

MFA Multi-layer Feature Aggregation. 17

ML Machine Learning. 1, 3, 5, 7–10, 13, 19, 25, 50

MLE Maximum Likelihood Estimation. 8

MLLP Machine Learning and Language Processing Group. iii–v, 3, 4, 24–26, 49, 50

MT Machine Translation. 2, 3, 50

NLL Negative Log Likelihood. 8

NLP Natural Language Processing. 1, 15

NN Neural Network. iii, xii, 5, 9–11, 13, 16, 17, 19

ResNet Residual Neural Network. 17

RNN Recurrent Neural Network. iii, 10, 11

RTTM Rich Transcription Time Marked. 15, 21–24, 31

RTVE Radio y Televisión Española. iii–v, 24

SA Self-Attention. 17, 19, 50

SAD Speech Activity Detection. 15, 29

SD Speaker Diarization. iii–v, 1–5, 7, 13–16, 19, 21, 24–27, 29, 30, 43, 50, 51

SE Squeeze-Excitation. 17

TDNN Time Delay Neural Network. 16, 17, 30, 49, 50

UEM Unpartitioned Evaluation Map. 22, 23

UPV Universitat Politècnica de València. 3

WDER Word Diarization Error Rate. 14

WER Word Error Rate. 13, 14

Chapter 1

Introduction

1.1 What is Speaker Diarization?

The understanding of the title of this master's thesis is fundamental, so before deepening any further into this work, a few paragraphs will be dedicated to this purpose.

The vast majority of words in the title of this work can be easily understood by everyone. However, two words in it, "Speaker Diarization", will be unfamiliar even for readers with a relatively broad knowledge of Machine Learning and even Natural Language Processing.

To diarize is to take note of something in a diary. When talking about Speaker Diarization, the thing that gets "recorded in the diary" is who is speaking at each moment. This is why SD is informally defined as the task of asking the question: "Who spoke when?"

Speaker Diarization has been defined more formally in many papers, one of the most popular definitions being: "The task of labeling audio or video recordings with classes that correspond to speaker identity" [1].

One vital nuance to clarify the given definition is that the concept "speaker identity" does not usually refer to a specific physical person's identity with name and surname but to an ID given by the SD systems to each speaker in an audio or video track that is used to distinguish them from the others appearing in it, without storing any more information about the speaker. This is the case of all the Speaker Diarization systems that will be presented in this work.

Although Speaker Diarization was first seen as just a way to improve Automatic Speech Recognition and Natural Language Processing systems, SD has much value on its own and many applications. In this work, we explore the application of SD to TV audiovisual productions.

1.2 Motivation

1.2.1 The importance of subtitles

According to the World Health Organization (WHO), in 2021, more than 1.5 billion people were affected by hearing loss. Furthermore, these statistics are expected not to improve in the future but to worsen with demographic changes, with predictions by WHO expecting hearing loss to affect ∼2.5 billion people in 2050 [2].

Adapting audiovisual content is essential for the inclusion of people suffering from hearing loss, and audiovisual subtitling is vital to do this. This is why developing new techniques using the latest technologies is imperative to make subtitling viable and easy for any company or person producing audiovisual content.

Moreover, subtitles are helpful not only to those affected by hearing loss but, combined with translation, are also a significant asset for making content understandable, relatively cheaply, for people who do not understand the language in which media content was initially produced, without the need to dub this content, which would be more complex and expensive.

1.2.2 The need of SD in audiovisual production

To understand the need of SD in audiovisual production, it is necessary first to understand that a subtitle, to have good quality and to be actually helpful for people with hearing loss, needs more than just the transcription of what is said in the audiovisual content. It is also crucial to let the viewer know who is saying each word or phrase when multiple speakers appear on the screen.

To achieve this, audiovisual subtitles must have colors. There are multiple laws and standards that regulate how these colors should be. In this work, we will use as reference the "Norma Española de subtitulado para sordos" (UNE-153010:2012) by "Asociación Española de Normalización".

However, to apply colors to subtitles, it is first necessary to know who is speaking at each fragment of time, and, as it was explained in Section 1.1, this is just what Speaker Diarization tries to answer.

To understand the need of Speaker Diarization in this context, it is also essential to consider the massive rise in audiovisual production. While some decades ago this activity was limited to TV channels, film corporations, and other big companies due to the technical requirements it had, nowadays, this has changed radically. Anyone with a smartphone and basic notions about audiovisual production can do it at home.

With this quantity of media produced daily, it is nearly impossible to make adequate subtitles for every piece. Even only considering the content offered by big media corporations, doing this with a traditional approach, that is, manually, is extremely difficult and costly.

For all these reasons, Automatic Speech Recognition, Machine Translation and, what is most relevant to this work, Speaker Diarization are becoming essential tools that help to handle this problem.

1.2.3 The MLLP(UPV)-CVMC agreement

Once the motivation for working in SD in general has been explained in subsections 1.2.1 and 1.2.2, it is also important to explain the specific context that motivated this work. This master's thesis has been done while working at the Machine Learning and Language Processing Group (MLLP), a research group of the Universitat Politècnica de València (UPV), and, specifically for the MLLP-Corporació Valenciana de Mitjans de Comunicació (CVMC) agreement: "Subtitulacio Assistida Per Ordinador en Temps Real i Basada en la Intel·ligencia Artificial, de Continguts Audiovisuals".

This agreement, as his name suggests, aims to develop Machine Learning systems to provide CVMC with subtitles for their audiovisual content. Until this work was started, these systems were Automatic Speech Recognition and Machine Translation systems. With these ML systems, the CVMC was being provided with subtitles and its translations, but not with colors to differentiate speakers.

Being this as relevant as explained in the previous subsections, both parties agreed on the importance of developing a Speaker Diarization system in the context of this agreement.

This was the final motivation to aim this master's thesis to the development of Speaker Diarization systems for "À Punt" (the main TV channel and webpage of CVMC).

1.3 Problem Description

As it was stated in Section 1.2, the problem to solve in this master's thesis is the study of different Speaker Diarization techniques based on Machine Learning and the posterior implementations of some of them, creating one or more SD systems able to obtain good results determining "Who spoke when?" in the context of audiovisual production, and, more specifically, that focus on the performance on "À Punt" (CVMC) media.

After reviewing the state of the art, it can be thought that experimentation with the different systems is the next step to take. However, an essential part of the problem to solve before doing this is to have good input data and to have it formatted the way the developed systems require.

Considering this, to tackle the problem, the next thing to do will be analyzing how the chosen techniques need to have the input data formatted and labeled.

Studying whether there are already available datasets with the required format within the application field of audiovisual production and, if this is the case, performing some experiments before starting to work on the "À punt" data can also be part of dividing the problem into smaller steps to make it more approachable. It can be an excellent way to get familiarized with the systems.

However, the truly crucial step regarding data will be to pre-process the "À punt" data to meet the requirements of the SD systems.

After reviewing the state of the art and pre-processing the data, the step to solve the problem is to design the baseline system. It has already been explained what is,

in general, Speaker Diarization, but to profile the problem correctly, it is necessary to specify in an exact way the desired output.

The designed baseline system to solve the problem should be able to give as output a file containing, for each intervention of any speaker in the media file:

- A speaker ID (which does not contain any information about the specific speaker, only to link it with their other interventions)
- The starting time of the intervention
- The duration of the intervention

To solve the problem in the most effective way possible, it is also necessary for the developed system to be able to detect overlapping interventions. Figure 1.1 shows a graphical representation of an output from a SD system.

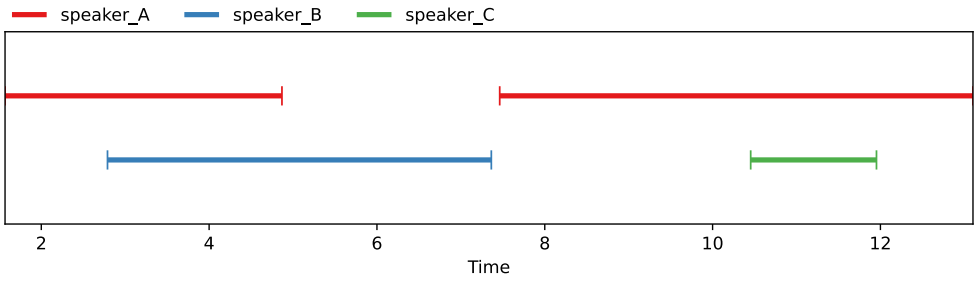


Figure 1.1: Example of output timeline, representing the interventions of three speakers: A(—), B(—) and C(—).

1.4 Goals

Having in mind the motivation and the problem which will be addressed in this master's thesis, the following goals were set and summarized on the two following points:

- Implementing a baseline system from the state of the art and applying it to the MLLP-CVMC context.
- Adapting the baseline system to different tasks of the MLLP-CVMC context in order to get improved systems with better results.

1.5 Structure of the work

This master's thesis will be structured in the following way:

In Chapter 2 the fundamentals of the technology used in this work will be reviewed. A particular emphasis will be put on Machine Learning (ML), Neural Network (NN), Automatic Speech Recognition (ASR) and Speaker Diarization (SD). Of this last concept, an overview of its state-of-the-art will be presented.

Chapter 3 is the one that explains the different datasets used in the development of this work, as well as the standard file formats employed in Speaker Diarization in which those datasets are based.

In Chapter 4 the implemented baseline system is explained, presenting a review of its fundamentals and features. This chapter tries to tackle the first goal presented in Section 1.4.

Following, Chapter 5 presents the experiments performed in order to achieve the second goal presented in Section 1.4. It includes explanations of the ideas behind those experiments as well as their results.

To conclude, Chapter 6 reviews again the goals proposed in Section 1.4 to check whether they have been reached in the previous two chapters and draw conclusions from this. Moreover, future work is proposed based on these conclusions and potential improvement ideas.

Chapter 2

Fundamentals of Speaker Diarization

2.1 Machine Learning

To be able to understand the fundamentals of ASR and SD, it is essential to have a clear idea of what is Machine Learning. For this reason, a brief review of this basic concept is provided in what follows.

Machine Learning can be defined as a set of methods that can automatically detect patterns in data and then use the uncovered patterns to predict future data or to perform other kinds of decision-making under uncertainty [3].

However, if an even more formal definition is wanted, ML can also be defined as:

> A computer program is said to learn from experience E with respect to some class of tasks T, and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E [4].

There are many different Machine Learning algorithms, but they are all divided into two big groups: supervised learning and unsupervised learning. Both will be explained in Subsections 2.1.1 and 2.1.2 since the two of them will be used in this master's thesis. There is another group of ML algorithms, Reinforcement Learning, but it will not be reviewed since it is irrelevant to this work.

2.1.1 Supervised Machine Learning

Supervised ML algorithms are those based on learning the relationship between some input and some output from previously labeled data.

That relationship can be formalized as a mapping f from the inputs $x \in X$ to the outputs $y \in Y$ [5] and has to be learned.

The algorithm will be able to learn this relationship from a "training set", that is, a set of data in which input is labeled with its corresponding output. That is, a set of N input-output pairs $D = \{(x_n, y_n)\}_{n=1}^{N}$ [5].

2.1.1.1 Classification

One kind of supervised Machine Learning is classification, also known as pattern recognition.

Suppose the term classification is used without further detail. In that case, the task is to assign one class label $c \in C$, with $C > 2$, to each input x, which is known in the bibliography as multiclass classification. However, other kinds of classification exist, such as binary classification, in which the task is the same but with $C = 2$, and multi-label classification, in which more than one $c \in C$ can be assigned to each input x [3].

Classifiers are trained to perform this task in supervised learning, adjusting their parameters. The set of parameters of a classifier can be defined as θ.

To train the model, that is, to adjust θ, the empirical risk has to be minimized. The classifier's goal is to minimize errors, giving particular attention to those with a higher degree of negative impact. Hence, the empirical risk can be defined as seen in Equation 2.1, where ℓ is a loss function that penalizes errors according to their negative impact.

$$\mathcal{L}(\theta) \triangleq \frac{1}{N} \sum_{n=1}^{N} \ell(c_n, f(x_n; \theta)) \tag{2.1}$$

It is common to use the negative log probability as the loss function, as shown in Equation 2.2:

$$\ell(c, f(x; \theta)) = -\log(p(c|f(x; \theta))) \tag{2.2}$$

The average negative log probability of the training set is the Negative Log Likelihood (NLL), defined in Equation 2.3

$$\text{NLL}(\theta) = -\frac{1}{N} \sum_{n=1}^{N} \log(p(c_n|f(x_n; \theta))) \tag{2.3}$$

Minimizing it, as shown in Equation 2.4, will get the Maximum Likelihood Estimation (MLE).

$$\hat{\theta}_{mle} = \arg\min_{\theta} NLL(\theta) \tag{2.4}$$

This is a prevalent way to train classifiers [5].

2.1.2 Unsupervised Machine Learning

Nowadays, when someone is speaking about Machine Learning, everyone, but especially those with a shallower knowledge of ML, will tend to think about Supervised Machine Learning. They will not be wrong often since it is now the most popular type of ML.

However, Unsupervised Machine Learning not only exists but is very useful and relevant, as will be seen later in this master's thesis.

Contrary to what happens in Supervised ML, in Unsupervised Machine Learning, there is no training set formed by input-output pairs $D = \{(x_n, y_n)\}_{n=1}^{N}$ to learn a mapping from. In this case, the goal is also to learn a mapping from the inputs to some output. However, it will be done having only the inputs $D = \{x_n : n = 1 : N\}$, without any corresponding output y_n [5].

This can be seen as a more difficult task. Nonetheless, it is also fascinating since it makes the application of ML algorithms without the need to collect and label massive training datasets. That is usually very costly and sometimes even impossible.

There are many types of Unsupervised Machine Learning. Nevertheless, the one that will be used in this master's thesis will be clustering.

2.2 Neural Networks

As stated before, the most common form of ML is supervised learning. Within this scope, the most popular techniques are those based on Deep Neural Network (DNN).

Artificial Neural Networks, as their name suggests, are supervised Machine Learning models inspired by the human brain's neuronal structure. They consist of a bunch of artificial networks organized into layers.

2.2.1 Perceptron

Artificial neurons are based on the Perceptron [6]. The perceptron is a deterministic linear binary classifier. The perceptron concept mathematically represents a function that maps an input x to an output $f(x)$. That function can be seen in Equation 2.5.

$$f(x) = H(w^T x + b > 0) \tag{2.5}$$

In Equation 2.5, w represents a vector of learnable weights, x is the input, a feature vector, and b is a learned bias. Finally, $H(x)$ represents the Heaviside step function, shown in Equation 2.6 [5].

A diagram of the perceptron can be seen in Figure 2.1.

$$H(a) \triangleq \mathbb{I}(a > 0) \triangleq \begin{cases} 1 & \text{if } a > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

2.2.2 Feed Forward Neural Networks

Feed Forward Neural Networks are the most basic type of Neural Networks. They are just a bunch of perceptrons organized in layers. These layers, except the first and last one, always get the previous layer's output as input.

At the output of each neuron, as perceptrons are linear functions, a non-linearity or activation function is applied to its output before passing it to the next layer in order to be able to handle problems that are not linearly separable.

Some of the most popular activation functions are ReLU, Sigmoid, and Softmax [7].

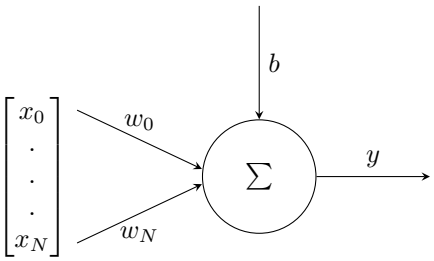


Figure 2.1: Perceptron diagram

The first layer is called the input layer and is the one getting the feature vector. The last one is known as the output layer. To conclude, the rest of the layers are named hidden layers.

This type of network can solve many problems, including non-linearly-separable binary and multiclass classification.

A diagram of a Feed Forward Neural Network (FNN) can be seen in Figure 2.2.

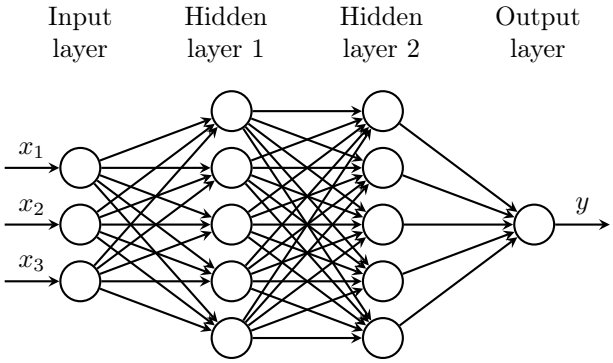


Figure 2.2: Example of Feed Forward Neural Network with an input layer of three neurons, two hidden layers of five neurons each and an output layer with one neuron.

2.2.3 Recurrent Neural Networks

Recurrent Neural Networks [8] are a type of artificial neural network used in Machine Learning and Deep Learning. They are specifically designed to handle sequential data, such as time series, text, speech, and more. The main advantage of RNNs over Feed Forward Neural Networks lies in their ability to capture and work with sequential dependencies in data.

To be able to do this, the basic idea is that their prediction y does not only depend on the input x, but also in a hidden state h, able to store information from previous steps.

2.2.4 Long Short-Term Memory Networks

Despite the fact that original Recurrent Neural Networks handled sequential data, they were pretty limited, struggling when processing long sequences of data. For this reason, an improved recurrent neural architecture was designed, the Long Short-Term Memory (LSTM) [9].

This new architecture was able to handle long sequences way better. The main idea of this new architecture was to add three types of gates to control the propagation of the information in the network:

- **Forget gates**: Determine which previous information should be kept and which one should be discarded from the cell memory.
- **Input gate**: Decides which new input information should be stored in the cell memory.
- **Output gate**: Decides which stored information should be used to compute the output.

2.2.5 Bidirectional Long Short-Term Memory Networks

Long Short-Term Memories solved the problem of long sequence handling. However, the power of understanding the context was not being used entirely because they were only capable of storing past information.

Bidirectional Long Short-Term Memorys [10] solved this problem by joining two different simple LSTMs, one processing data in the forward direction and another one doing so in the backward direction.

2.2.6 Convolutional Neural Networks

Convolutional Neural Networks are a type of Neural Network initially developed for processing grid-like data, such as images. Their popularity in Computer Vision (CV) made them quickly and widely applied to other fields.

Their main contribution is the convolutional layers. These layers apply small filters, also called kernels, to local regions of their input data and slide these kernels across the entire input to learn features.

An example of convolution can be seen in Figure 2.3.

Another relevant feature of CNNs are the pooling layers, which aggregate information from the neighboring regions of structured inputs. The most common pooling operations are average pooling and max pooling.

2.3 Clustering

Clustering is a type of unsupervised machine learning (explained in Section 2.1.2) in which, as its name suggests, the goal is to find clusters of similar data. That is, to divide the k-dimensional space into regions containing seemingly related points.

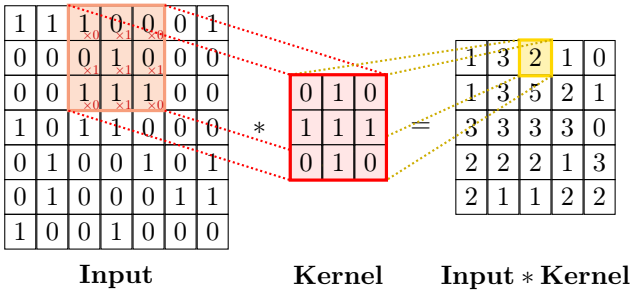
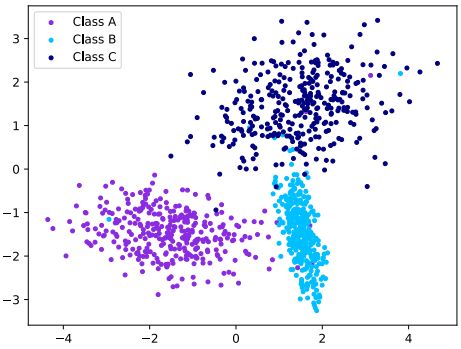


Figure 2.3: Convolution example over 2D data.

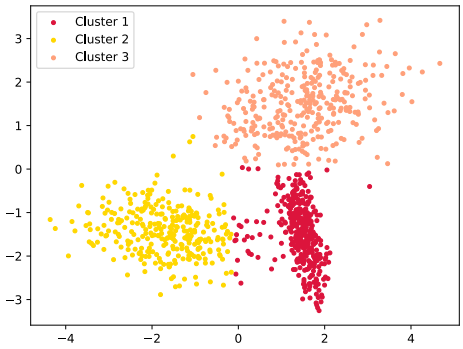
The clustering technique itself does not have a "correct" number of clusters since it only tries to find similar points, and the definition of similar can be more or less strict, allowing to increase or reduce the complexity of the model. However, a certain number of clusters can be desired in specific tasks, and clustering algorithms can be adapted to find that number of clusters.

Many clustering algorithms exist, including k-means, spectral clustering, hierarchical agglomerative clustering [5], Bayesian hidden Markov model clustering, etc.

An example of k-means clustering can be seen in Figure 2.4 where k-means is applied to find three clusters in a two-dimensional space. It is essential to notice that even though we can see that the points in Figure 2.4a belong to three different classes, clustering algorithms do not classify. However, the formed clusters can be classified after the clustering ends. That is one of the cases where there exists a desired number of clusters.



(a) 3 classes forming clusters in two dimensions: Class A(●), Class B(●) and Class C(●).



(b) Result of k-means clustering algorithm which predicted: Cluster 1(●), Cluster 2(●) and Cluster 3(●).

Figure 2.4: Example of clustering using k-means over two dimensions.

2.4 Automatic Speech Recognition

Although Automatic Speech Recognition is not the primary field of study for this master's thesis, it must be briefly introduced in it for a few reasons.

The first reason is that Speaker Diarization was born as a way to improve Automatic Speech Recognition. Now, SD constitutes its own field of research with its standalone systems, but the trace and essence of ASR can still be seen in many aspects of it, with these two fields having a lot in common.

On top of that, in this work, every goal is centered just in the development of a Speaker Diarization system, but as it has been explained in Section 1.2, the developed system will later be deployed together with a large ASR system.

2.4.1 Definition

Automatic Speech Recognition (ASR) is a field inside Machine Learning (ML) and Natural Language Processing, aiming to convert spoken language into text automatically.

However, it is essential to provide a more formal definition:

Let $X = \{x_1, x_2, ..., x_T\}$ be an acoustic signal.

From a probabilistic point of view, an ASR system is a system which, given the acoustic signal X tries to obtain the most probable sequence of words $\hat{W}$. That can be approached using the posterior probability as seen in Equation 2.7, where L is the vocabulary of the system and L^* the set of all possible sequences of words (sentences) of L.

$$\hat{W} = \underset{W \in L^*}{\arg\max} P(W|X) \tag{2.7}$$

In the latest approaches, Equation 2.7 is directly modeled using end-to-end Neural Network (NN). However, the traditional approach has separated the ASR task into two different models by applying Bayes Theorem as shown in Equation 2.8.

$$\hat{W} = \underset{W \in L^*}{\arg\max} P(W|X) = \underset{W \in L^*}{\arg\max} \frac{P(X|W)P(W)}{P(X)} = \underset{W \in L^*}{\arg\max} P(X|W)P(W) \tag{2.8}$$

In the last step of Equation 2.8, $P(X)$ has been removed from the equation since it is a constant. After that, the equation has two terms $P(X|W)$, modeled by the acoustic model, and $P(W)$, modeled by the language model.

2.4.2 Evaluation

The standard metric used to evaluate Automatic Speech Recognition systems is the Word Error Rate (WER). The WER is a distance metric, derived from the Levenshtein distance [11], and calculated as shown in Equation 2.9.

$$WER = \frac{I + D + S}{R} * 100 \tag{2.9}$$

In Equation 2.9, I, D, and S are, respectively, the number of word insertions, deletions, and substitutions that have to be made in order to make the predicted sequence of words $\hat{W}$ equal to the pronounced sequence of words. Also, R is the number of pronounced words.

2.5 Speaker Diarization

With the formal definition of Speaker Diarization presented in Section 1.1, and after reading Section 2.1.1.1, it is clear that Speaker Diarization is indeed a classification problem, even though it involves other subtasks which make the problem more complex than a simple classification.

Depending on the needs and resources available in each case, diarization systems can or can not handle speaker overlaps, that is, two or more people speaking simultaneously and being recognized by the diarization system.

When there is no overlap, the classification task is a multiclass classification. Nonetheless, when diarizing overlapping speakers is part of the task, the problem becomes more challenging and is a multi-label classification problem.

2.5.1 Evaluation

Before explaining different kinds of Speaker Diarization systems, it is crucial to have a clear view of how they will be evaluated.

In the literature, different metrics are used to evaluate SD like Jaccard Error Rate (JER), Word Diarization Error Rate (WDER), and Diarization Error Rate (DER). The latter is the most popular one and the one that will be used in this master's thesis [1].

Diarization Error Rate is the standard metric for evaluating SD systems and is an edit distance metric. It is highly similar to Word Error Rate, but it uses spoken time instead of using words as base units. Both are metrics derived from the Levenshtein distance [11].

The Diarization Error Rate [1, 12] sums the time of False Alarm Rate (FA) (when the system determines someone is speaking and there is no one speaking), the missed time (time where the system concludes no one is speaking and someone is doing so) and the speaker confusion (when the SD system labels a period as spoken by one speaker and in reality a different speaker is the one speaking). Later, it divides this sum between the total time. The formula can be seen in Equation 2.10 [1]:

$$DER = \frac{FA + Missed + Speaker\ confusion}{Total\ time} * 100 \tag{2.10}$$

2.5.2 Conventional Speaker Diarization systems

There have been many approaches to Speaker Diarization systems. However, the vast majority of them followed a similar structure and the same essence before the recent advances in Deep Learning that revolutionized every field of Natural Language Processing.

Usually, Speaker Diarization systems have been formed by many independent sub-modules. Each contributed to the overall process of distinguishing and labeling different speakers within an audio recording and could be tuned.

The first step to diarize any track conventionally was to preprocess the raw audio signal. This preprocessing aimed to enhance the audio quality by mitigating various issues such as background noise, speech enhancement, and the removal of echoes or reverberations. The subsequent modules could more accurately analyze and differentiate speech segments by optimizing the audio quality upfront.

Following the preprocessing step, the next module was the Speech Activity Detection (SAD), whose task, as its name suggests, was to determine if someone was talking at a given moment or not. This module played a significant role in influencing both the False Alarm Rate (FA) and "Missed" errors, which are integral components of the Diarization Error Rate formula (Equation 2.10). However, the preceding preprocessing module naturally impacted these errors as well.

The subsequent module in the traditional Speaker Diarization system was responsible for segmentation. The term "segmentation" has been vastly used in the speech recognition community to describe many different tasks, and even in Speaker Diarization bibliography, it has slight differences in goals and implementation. Despite this fact, generally speaking, the segmentation module of a traditional SD system does separate the parts in which there is speech into different segments according to who is talking at each time. Another way to see it is that they detect speaker change and split the speech part of the audio according to it.

The next vital step is to create embeddings of each segment. In Machine Learning, embedding refers to a representation of data that transforms high-dimensional input into a lower-dimensional space while retaining the original data's most relevant and discriminative features. This transformation is typically achieved through a mathematical function that captures patterns, relationships, and other information present in the data. Embeddings are very useful tools in various Machine Learning tasks, as they can help alleviate computational complexity, enhance model performance, and reveal hidden structures within the data. However, these embeddings usually relied on handcrafted features in traditional SD systems.

Subsequently, a clustering algorithm is applied to the previously generated embeddings, thereby forming clusters or groups of speaker interventions. Concurrently, a classification task is undertaken to assign these clusters to distinct speaker identities. It is important to note that these speaker identities do not inherently carry specific information about the speakers themselves; they merely serve as labels for differentiation and can be freely interchanged without any adverse effects on the process.

Finally, the clustering and classification can be refined in a post-processing stage or module, obtaining a RTTM file with the diarization results as output.

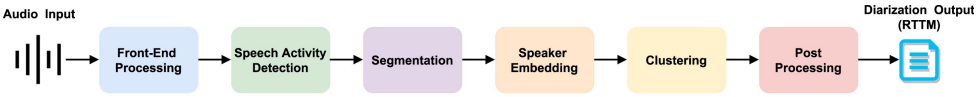


Figure 2.5: Conventional speaker diarization system [1].

2.5.3 A review of speaker recognition technology used in SD

2.5.3.1 Deep Neural Network speaker recognition: x-vectors

As discussed in Section 2.5.2, one crucial step in the Speaker Diarization process is the creation of speaker embeddings, and in conventional SD systems, these relied mainly on handcrafted features.

In 2018, [13] introduced the concept of x-vectors, which are fixed-dimensional embeddings extracted from a Deep Neural Network.

The concept was created for standalone speaker identification. Nevertheless, its ability to create embeddings that represented speaker information better than conventional ones made them very useful in the Speaker Diarization context.

The original X-Vectors were extracted from a Neural Network, whose features were 24-dimensional filter banks.

To understand this NN, assume that the input is an audio segment with T frames.

The network has five initial layers using a Time Delay Neural Network architecture [14, 13, 15], which works at frame-level, that is, takes a frame t and some context of k frames before and after t. The first layer starts with a $[t-2, t+2]$ context, and when the information arrives at the fifth layer, it has a total context of 15. After the fifth layer, a "statistics pooling" was applied, aggregating all the T frame-level outputs from the fifth layer and allowing the network to start working at the segment level. The X-Vector is extracted from the first segment level layer before applying the ReLU nonlinearity. The complete architecture is shown in Table 2.1.

Table 2.1: The original embedding DNN architecture. x-vectors are extracted at layer segment6, before the nonlinearity. The N in the softmax layer corresponds to the number of training speakers [13].

Layer	Layer context	Total context	Input x output
frame1	[t-2, t+2]	5	120x512
frame2	t-2, t, t+2	9	1536x512
frame3	t-3, t, t+3	15	1536x512
frame4	t	15	512x512
frame5	t	15	512x1500
stats pooling	[0,T)	T	1500Tx3000
segment6	0	T	3000x512
segment7	0	T	512x512
softmax	0	T	512xN

2.5.3.2 Improving speaker embeddings: ECAPA-TDNN

The success of x-vectors [13] made the creation of speaker embeddings and speaker verification using Time Delay Neural Network vigorous areas of investigation. This resulted in many x-vectors improvements [16, 17, 18, 19], one of the most relevant improvements for the TDNN was the inclusion of elements from the Residual Neural Network (ResNet) architecture [16, 20].

On top of all those advances, a paper [16] was published to improve the creation of speaker embeddings and the recognition of speakers. This was achieved by modifying the TDNN architecture and the statistical layer of the architecture proposed in the original x-vectors paper [13].

The ECAPA-TDNN proposes the extension of Self-Attention used in previous x-vectors improvements [21, 22], which proved to obtain better results, further to the channel dimension [16].

Another key feature of the proposed architecture was the use of 1-Dimensional Convolutional Neural Networks based on ResNet together with the introduction of 1-Dimensional Squeeze-Excitation (SE) Res2Blocks. These blocks proved in the Computer Vision (CV) field to successfully model global channel interdependencies, and, because of this, it seemed like an excellent way to rescale frame-level features given global properties of the audio, similar to the global context of the previously explained SA [16].

The final important feature of ECAPA-TDNN is that the output of each SE-Res2Block is concatenated and later processed by the Multi-layer Feature Aggregation (MFA), a dense layer which generates features for the attentive statistics pooling [16].

More interesting for this work is the extraction of speaker embeddings, which is done from the final fully connected layer. The entire structure of the ECAPA-TDNN, together with the structure of its particular SE-Res2Block can be seen in Figure 2.6.

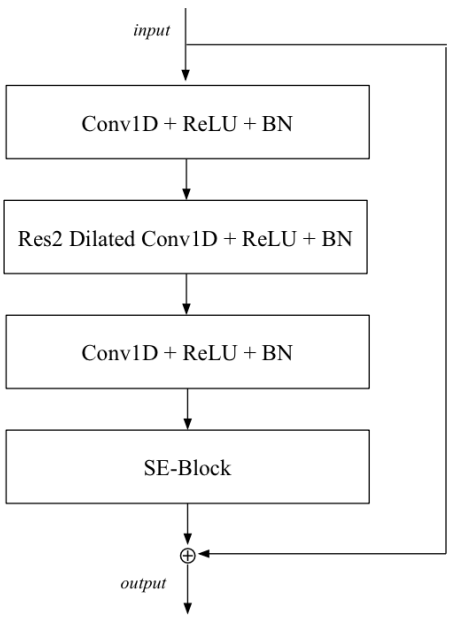
2.5.3.3 Deep Neural Network speaker recognition: SincNet

When talking about speaker recognition in Section 2.5.3.1, it has been seen that many systems in this area developed before the rise of deep learning were based on handcrafted features, which was not the optimal way to proceed.

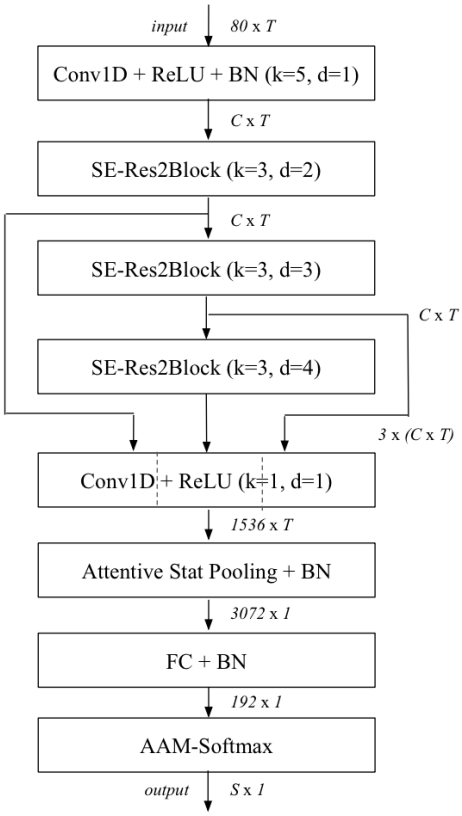
In opposition, when Neural Networks and especially Convolutional Neural Networks became increasingly popular, researchers started focusing their efforts on developing systems based on them. This led to a lot of promising results [23, 24, 25, 26, 27].

Despite this, [23] describes some problems with feeding raw waveforms to convolutional layers. First, this approach made the convolutional layers handle very high dimensional data. Secondly, the first convolutional layers of previous approaches were quite sensitive to the vanishing gradient problem. Finally, when assuming that the convolutional layers were modeling filters, many were taking noisy and chaotic shapes, which did not seem to help the network achieve a good performance.

For this reason, this paper presented SincNet, a modification of Convolutional Neural Networks in which instead of learning a full convolutional filter of length L from scratch, a predefined filter was provided, leaving only some parameters of it (θ) to be learned by the Neural Network.



(a) The SE-Res2Block of the ECAPA-TDNN architecture. The standard Conv1D layers have a kernel size of 1. The central Res2Net [20] Conv1D with scale dimension $s = 8$ expands the temporal context through kernel size k and dilation spacing d [16].



(b) Network topology of the ECAPA-TDNN. We denote k for kernel size and d for dilation spacing of the Conv1D layers or SE-Res2Blocks. C and T correspond to the channel and temporal dimension of the intermediate feature-maps, respectively. S is the number of training speakers [16].

Figure 2.6: Architecture of ECAPA-TDNN and its SE-Res2Block [16].

Those predefined filters could have taken many forms, but in the SincNet original paper, they took the form of rectangular bandpass filters inspired by classical digital signal processing. This choice made the learned parameters θ only two: The low and high cutoff frequencies (f_1 and f_2 respectively).

The SincNet filters not only solved the initial problems stated in the paper but also brought with them more advantages, mainly reflected in the computational complexity of the model. In the first place, the number of learnable parameters reduced drastically, from LF, being F the number of filters and L its length, to just $2F$. Moreover, the proposed SincNet filters are also symmetrical, so the convolution can

be performed using only one side of the filter and reducing by 50% the number of computations it needs. All this leads to a quicker convergence of the network.

2.5.4 End-to-End Neural Speaker Diarization (EEND)

In recent years, with the increase in computing power and the popularization of new Machine Learning models and architectures, end-to-end models have been on the rise in every field, from ASR to SD.

In contrast to the conventional SD systems, composed of many blocks, as explained in Section 2.5.2, in 2019, a group of Hitachi researchers published a paper [28], proposing an end-to-end structure for SD and quickly becoming the state-of-the-art.

This paper proposed a solution in which the input is a T-length observation sequence $X = (x_t \in \mathbb{R}^F | t = 1, ..., T)$ of audio. Given this input, the goal is to predict a speaker label sequence $Y = (y_t | t = 1, ..., T)$ where $y_t = [y_{t,c} \in \{0, 1\} | c = 1, ..., C]$ denoting C speakers. Having $y_{t,c} = 1$ and $y_{t,c'} = 1$ while $c \neq c'$ means that both c and c' are talking at the same time, generating an overlap.

For this reason, the most probable speaker label sequence $\hat{Y}$ should be selected as shown in Equation 2.11.

$$\hat{Y} = \arg\max_{Y \in \mathcal{Y}}(P(Y|X)) \tag{2.11}$$

Then, using the conditional independence assumption, $P(Y|X)$ can be factorized as in Equation 2.12.

$$P(Y|X) = \prod_t P(y_t|y_1, ..., y_{t-1}, X) \approx \prod_t P(y_t|X) \approx \prod_t \prod_c P(Y_{t,c}|X) \tag{2.12}$$

Finally, by assuming that $P(y_{t,c}|X)$ is conditioned on all inputs and that each speaker presence is independent, $P(y_{t,c}|X)$ is estimated using a Neural Network either based on Bidirectional Long Short-Term Memory networks BLSTM or Self-Attention [1, 28, 29].

2.5.5 The state-of-the-art toolkit: PyAnnote

PyAnnote, specifically *pyannote.audio*, is an open-source toolkit that provides state-of-the-art speech processing tools. It specializes in Speaker Diarization, for which it provides many modular blocks to create SD pipelines and even preconfigured pipelines.

It incorporates many versions of state-of-the-art technologies as its default blocks, such as the ones explained in Subsection 2.5.4.

The different blocks used in the development of the experiments of this work will be further explained in Chapter 4

Chapter 3
Datasets

3.1 *PyAnnote Database* standard

PyAnnote Database is a versatile Python library designed to help manage and organize multimedia datasets for various machine learning tasks, particularly those related to Speaker Diarization, even though it can be used for other Speech Processing tasks. It provides a structured framework for handling metadata, annotations, and associated audio or video files effectively. This structure improves dataset organization and helps maintain clarity and control over data.

Of course, one of the main strengths of PyAnnote Database is that it is part of the PyAnnote ecosystem, which is really useful in this work since PyAnnote is the main library that will be used for conducting different experiments and trying to achieve the goals established in Section 1.4.

PyAnnote Database promotes consistency and reproducibility in research endeavors. Establishing a standardized dataset management format ensures that the research community can faithfully reproduce, validate, and share experiments. This standardization looks for more transparent research practices and facilitates the reliable advancement of knowledge.

The subsequent sections (3.1.1, 3.1.2, 3.1.3 and 3.1.4) will examine the parts of this standard that carry the highest significance for the development of this work.

3.1.1 Annotation RTTM files

Rich Transcription Time Marked (RTTM) files are among the most critical files for Speaker Diarization in the PyAnnote Database standard. In this specific context, they are used to annotate the different speaker interventions in the multiple dataset files.

Rich Transcription Time Marked files are space-delimited files containing one turn per line, where ten fields form each line:

- Field 1 (Type): This field contains the type of segment stored in this line. In this work, it will always be set to "SPEAKER".

- Field 2 (File ID): The second field contains the audio file name where the speaker is intervening, without path names or extensions.
- Field 3 (Channel ID): The third field stores the audio channel. In the case of this work, this value will always be 1.
- Field 4 (Beginning time): As its name suggests, the fourth field of the lines of RTTM files contains the beginning time of the "object", which, in the context of this work, is a speaker intervention. This time will be indicated in seconds elapsed from the start of the audio file.
- Field 5 (Duration): The fifth field of each line contains the duration of the "object", that is, in our case, the duration of the speaker intervention, also in seconds.
- Field 6 (Ortography field): This field is used for storing orthographic representations of some "objects". However, for SPEAKER, and hence, in this work, this field will always be <NA>.
- Field 7 (Subtype): The seventh file should contain the subtype of the "object". For this work, this field will always be <NA>.
- Field 8 (Name): In this field, a name will be given to the "object". It will be used to identify which speaker has intervened in each line and will be a unique speaker ID.
- Field 9 (Confidence): The last field contains the confidence that the object information is correct. Again, this field will not be used in this work, so it will always stay fixed to <NA>.

A RTTM file example can be seen in Table 3.1.

Table 3.1: Exemple of RTTM file extracted from the "Debates" dataset, showing five speaker interventions.

1	2	3	4	5	6	7	8	9
SPEAKER	autonomic	1	7509.57	4.61	<NA>	<NA>	3	<NA>
SPEAKER	autonomic	1	7755.85	1.29	<NA>	<NA>	3	<NA>
SPEAKER	autonomic	1	7757.98	57.93	<NA>	<NA>	3	<NA>
SPEAKER	autonomic	1	886.47	14.16	<NA>	<NA>	4	<NA>
SPEAKER	autonomic	1	901.11	15.94	<NA>	<NA>	4	<NA>

3.1.2 Annotated region (*.uem*) files

Unpartitioned Evaluation Map (UEM) files will be the ones that will be used to indicate which parts of the different audio files on the dataset are annotated.

This is very useful to prevent errors in the evaluation due to the system detecting speaker activity where there really is but has not been annotated and also errors in the annotation RTTM files, removing any part of the annotation outside of the annotated region set in these UEM files.

The UEM files can also be employed to use different parts of the same audio file in different dataset partitions. For instance, a third of an extensive audio file can be used for train, another for dev, and another for test.

UEM files contain the information of the annotated region of one audio file in each line, are space-delimited files, and are formed by four fields [30]:

- Field 1 (File ID): The first field contains the audio file name where the speakers are intervening, without path names or extensions.
- Field 2 (Channel ID): The second field stores the audio channel. In the case of this work, this value will always be NA.
- Field 3 (Start time): The third field contains the start time of the annotated region. This time will be indicated in seconds elapsed from the start of the audio file.
- Field 4 (End time): The fourth and last field contains the end time of the annotated region. This time will be indicated in seconds elapsed from the start of the audio file.

A UEM file example can be seen in Table 3.2.

Table 3.2: Example of UEM file extracted from the "Debates" dataset.

File ID	Channel ID	Start time	End time
autonomic	NA	4343.000000	7994.780000

3.1.3 List files

Another simple but critical type of file in the PyAnnote Database library are list files. These are simple text files containing a "resource" name in each line.

In this work, they will be used to give PyAnnote a list of the files that contain each subset of the dataset (train, dev, and test).

They contain the filename of the "resource", without any extension. By doing it this way, PyAnnote can use the list files to locate both RTTM, UEM, and audio files using only a list file. It is important to note that every file of the "resource" should have the name appearing in the list file plus its corresponding extension (.rttm, .uem, .flac, .wav, etc.).

3.1.4 Audio files

Another vital type of file in the PyAnnote Database library is, of course, audio files. These will be the files later used to train the Speaker Diarization model, perform inference and adjust hyperparameters, and test and evaluate the final system.

During the development of this work, both .flac, .aac and .wav audio files have been used.

3.2 RTVE2018 Database - Albayzin Challenge

As mentioned in the Abstract of this master's thesis, with the goal to adapt Speaker Diarization (SD) systems to audiovisual production, it could be interesting to experiment over the Radio y Televisión Española (RTVE) datasets that the Machine Learning and Language Processing Group (MLLP) has available.

Specifically, the "RTVE2018 Database" used for the "Albayzin Challenge" was available since it was used in recent MLLP projects. This database is a collection of TV programmes from RTVE, which were broadcasted between 2015 and 2018. The database is divided into four partitions and has two of them where Speaker Diarization (SD) labeled data is available.

The first partition with Rich Transcription Time Marked (RTTM) files, containing speaker diarization information, is the "dev2" partition. The second one was the "test" partition. Their duration and composition can be seen in Tables 3.3 and 3.4, respectively.

Table 3.3: Duration of each program of the RTVE2018 Database "dev2" partition [31].

Program	Duration
Millennium	7h 42m 44s
La noche en 24H	7h 26m 41s
Total	15h 09m 25s

On the one hand, the shows of the "dev2" dataset, as shown in Table 3.3, are "Millenium", a debate show which discusses everyday events, and "La Noche en 24H", a talk show analyzing the events that happened the day it is emitted.

Table 3.4: Duration of each program of the RTVE2018 Database "test" partition [32].

Program	Duration
España en Comunidad	8h 09m 32s
La Mañana	1h 36m 31s
La Tarde en 24H (Tertulia)	8h 52m 20s
Latinoamérica en 24H	4h 06m 57s
Total	22h 45m 20s

On the other hand, the programmes composing the "test" dataset are four: "España en comunidad", which offers reports and information about the Spanish autonomous communities, "La Mañana", which is a live magazine, "La Tarde en 24H", a talk show about political and economic news, and "Latinoamérica en 24H", which provides analysis and information focused on Ibero-America.

3.3 À Punt dataset

As it has been explained in Sections 1.2 and 1.3, this master's thesis aligns with the MLLP-CVMC agreement, and both parties emphasized the need to develop a SD system, driving this work.

Therefore, the primary dataset used in this work will be one containing "À Punt" content.

3.3.1 Manually annotating interventions

One of the most important steps when developing any Machine Learning system is always to gather good data. In fact, this, combined with massive computing power, is one of the principal motors of the swift evolution AI has been experimenting this last year. It is also what differentiates big players from more modest researchers.

This work was no exception to this rule, and one of the biggest and most time-consuming challenges this work has faced was to get annotated data.

While the Corporació Valenciana de Mitjans de Comunicació (CVMC) had a lot of previously annotated data for developing Automatic Speech Recognition systems in the context of the MLLP-Corporació Valenciana de Mitjans de Comunicació (CVMC) agreement, it was not the case for Speaker Diarization data.

This reality was caused by the fact that many TV programs already had subtitles that a team of people had manually written for many years. It was not the case for subtitles separating the speaker interventions. Some available subtitles had colors and looked like they could have been used for this task. However, after exploring them at the beginning of this work, some crucial problems showed up, making their use impossible.

In the first place, the available subtitles with colors only represented the four most relevant speakers in each program. To be specific, the following scheme of colors:

- Yellow, for the most relevant speaker.
- Green, for the second most relevant speaker.
- Cyan, for the third most relevant speaker.
- Magenta, for the fourth most relevant speaker.

This was a problem since, as it has already been explained multiple times, Speaker Diarization systems do not identify the speakers. For this reason, it is impossible to

ask a SD system to recognize these speakers only, causing a very high error rate in inference.

Moreover, all the programs with colored subtitles had way more speakers than available Speaker Diarization systems can handle with an acceptable error rate for state-of-the-art systems.

For all these reasons, it was decided to invest a major part of the development of this project into manually annotating "À Punt" programs. This was done using an editor developed by MLLP, which allows the creation and edition of subtitles by using a web interface.

Two different TV programs were annotated, forming two independent datasets and one mixed dataset. These three datasets will be further explained in subsections 3.3.2, 3.3.3 and 3.3.4.

3.3.2 *La Cuina de Morera*

"La Cuina de Morera" is a cuisine TV program where there are two main protagonists: The cook and a nutritionist. On top of that, usually, there are three more speakers: A narrator, some guest who explains or advertises something for a few minutes, and a viewer who sends an audio with some sort of doubt to the program.

In developing a cutting-edge Speaker Diarization system, "La Cuina de Morera" presents a unique opportunity. With a diverse cast of five distinct voices, this show provides an ideal testing ground for contemporary state-of-the-art diarization systems.

It is worth highlighting that the key speakers to identify for generating accurate colored subtitles are primarily the two main speakers. This aspect somewhat simplifies the task. Nonetheless, their conversational style is informal and unscripted, resulting in a lot of overlaps. These overlapping segments pose a significant challenge for state-of-the-art SD systems.

The dataset used for developing the Speaker Diarization is formed by three episodes of the shows, one for train, one for dev, and another one for test, which duration can be seen in Table 3.5.

Table 3.5: Duration of each part of the "La Cuina de Morera" dataset.

Subset	Duration
train	0h 42m 10s
dev	0h 40m 18s
test	0h 43m 18s

3.3.3 *Electoral debates*

The second dataset intended for utilization in this work is focused on political debates.

Whenever a significant election is imminent, "À Punt" typically organizes political debates involving the candidates. Due to the distinct and well-defined turns, these

debates seemed promising as a Speaker Diarization application field. These made the debates task look very suitable for achieving notable performance.

This dataset is basically the entire electoral debate of the candidates for the presidency of the "Generalitat Valenciana" for the autonomical elections celebrated on the 28th of May of 2023.

This debate showcased six candidates, each representing one of the six most influential political parties participating in the electoral process. The proceedings were further guided by the presence of two moderators of the debate, summing up a total of eight speakers.

This number is inside the boundaries of speakers state-of-the-art Speaker Diarization systems can handle but is close to the maximum number the literature analyzes. It is widely acknowledged that accommodating a larger number of speakers tends to strain the accuracy of diarization, potentially diminishing its quality.

Despite the structured conversational turns that typified this electoral debate, the occurrence of multiple interruptions introduced instances of overlapping speech segments also in this dataset. This, in turn, introduced a certain degree of intricacy to the diarization task, augmenting the complexity of accurate speaker attribution.

The autonomical debate was separated into different parts to create a train, a dev, and a test subset. The duration of these parts is shown in Table 3.6

Table 3.6: Duration of each part of the "debates" dataset.

Subset	Duration
train	1h 00m 51s
dev	0h 27m 17s
test	0h 27m 17s

3.3.4 *La Cuina de Morera* and electoral debates mixed task

After the creation of both "La Cuina de Morera" and "debates" datasets, a specific Speaker Diarization system could have been created using them.

This was one of the main goals. Nonetheless, it was also interesting to try creating a more general SD system by combining both datasets and comparing it to the specific task-adapted systems.

For these reasons, both datasets were combined, achieving a dataset with the length shown in Table 3.7.

Table 3.7: Duration of each part of the "La Cuina de Morera" + "debates" mixed dataset.

Subset	Duration
train	1h 43m 01s
dev	1h 07m 35s
test	1h 10m 35s

Chapter 4

Baseline System: The PyAnnote pretrained pipeline

As explained in Section 2.5.5, PyAnnote is an open-source toolkit that provides researchers and developers with state-of-the-art speech processing modules, specializing in Speaker Diarization.

Among other features, PyAnnote provides a pre-trained and preconfigured SD pipeline, and this pipeline is the Speaker Diarization system that will be used as the baseline in this master's thesis.

On top of that, several modified and optimized versions of this initial pre-trained *pyannote.audio* pipeline will be developed in the experiments showcased in this section to obtain a state-of-the-art Speaker Diarization system specifically adapted to the CVMC tasks, as stated in Section 1.4.

For both these reasons, it is a requisite to understand how the *pyannote.audio* pre-trained pipeline works and its different components and parameters.

4.1 The segmentation model

The first module of the *pyannote.audio* Speaker Diarization pretrained pipeline is the segmentation module presented in [33].

As explained in Section 2.5.2, the word "segmentation" has been used for describing a wide variety of tasks across speech processing. However, when it is used inside the field of Speaker Diarization, segmentation refers to the task of splitting a conversation in an audio file into speaker turns. In conventional SD, the segmentation task was usually divided into three subtasks:

- Speech Activity Detection (SAD): To detect whether there is anyone speaking at time t or not.

- Speaker Change Detection: As its name suggests, to identify when one speaker c stops speaking and another one c' being $c \neq c'$ starts doing so. Sometimes, this subtask alone was also called "segmentation" at the time.

- Overlapped Speech Detection: The task to detect whether two speakers c and c' being $c \neq c'$ were talking at the same time. Not every conventional Speaker Diarization system considered this possibility; hence, not all of them performed this subtask.

In contrast, the aforementioned paper proposed to train an end-to-end model for performing the entire segmentation task. This segmentation model was inspired in the original End-to-End Neural Speaker Diarization (EEND) [28, 29] approach explained in Section 2.5.4.

The main differences with the original EEND implementation are that this model works on 5s audio chunks at a high temporal resolution (every 16ms) and that these chunks are passed to SincNet [23] convolutional layers, which have been explained in Subsection 2.5.3.3. These layers maintained the original SincNet configuration except for the first one, where the stride was set to 10.

Then four layers of Bidirectional Long Short-Term Memory (BLSTM) and two FF-DNN layers of 128 units each work at frame-level, and the output of the second fully connected layer is passed to the final classification layer [33, 34].

4.2 The local embedding model

After the segmentation has been performed, the next step of the *pyannote.audio* pipeline is the creation of local, that is, intra-window, speaker embedding.

The embedding model can be changed, but the one provided by default by the pipeline is the Speechbrain [35] implementation of ECAPA-TDNN [16], which architecture has been explained on Subsubsection 2.5.3.2.

This Speechbrain model has been trained on Voxceleb 1 [36] and Voxceleb 2 [37].

4.3 Global clustering

After creating the local speaker embeddings using ECAPA-TDNN, the next step performed by the pipeline is a clustering step that aims to create global clusters from the local speaker embeddings (see Section 2.3).

This is done using classical agglomerative hierarchical clustering with centroid linkage. Centroid linkage is a method used to measure the distance between clusters by computing the Euclidian distance between its centroids. The centroid of a cluster is the average of the individual feature values of each data point in a given cluster.

4.4 Final Aggregation

The final step of the *pyannote.audio* pipeline is to join the results of the three previous ones to get the final diarization output from which a RTTM file can be extracted.

This final aggregation consists of three, and optionally four, steps [34]:

- Frame number of speakers estimation: For each frame f the number of active speakers K_f is estimated based on the output of the segmentation model.
- Instantaneous cluster score estimation: Each cluster is evaluated, and a score is assigned by summing the local speaker segmentation.
- Top cluster selection: The K_f clusters with higher scores are selected, and the frame index is converted to the temporal domain.
- Gap filling: This step is optional and consists of filling those intra-speaker gaps that are shorter than what is considered normal.

4.5 Hyper-Parameters

4.5.1 Segmentation threshold (θ)

The segmentation threshold(θ) is the most important parameter of the *pyannote.audio* pipeline.

The output of the segmentation model explained in 4.1 before applying the segmentation threshold (θ) is the probability of each speaker being active at each time t (every 16ms) of the window, being this probability a real value between 0 and 1.

The segmentation threshold θ is in the same range, $\theta \in [0, 1]$. It is used to binarize the output, keeping only those speakers whose probability goes above it at each step of the window, assigning to those speakers the value one and to those who do not go above θ the value zero.

4.5.2 Clustering threshold (δ)

The clustering threshold (δ) is the stopping criterion for the agglomeration process of the applied global hierarchical agglomerative clustering explained in Subsection 4.3.

Agglomerative clustering is based on starting with each data point being its own cluster and then iterating, finding the two closest clusters at each iteration, and merging them. Before this merge step, it is checked if the distance between the two clusters is greater than the clustering threshold (δ). If it is, the agglomeration process stops.

The clustering threshold (δ) takes values in the range $[0, 2]$.

4.5.3 Minimum cluster size

This parameter sets a number of data points below which a smaller cluster is assigned to the most similar large cluster.

The $min\_cluster\_size$ can be any integer greater than one.

4.5.4 Minimum duration off (Δ)

This parameter is used on the last optional step of the pipeline, explained in Subsection 4.4. Its goal is to fill intra-speaker gaps that are smaller than its value, that is, that are considered not normal.

This parameter can have a relatively relevant effect on the diarization output if the pre-trained pipeline is used. However, if the segmentation model is finetuned, these gaps tend to disappear just because of the training with an in-domain dataset, so this parameter becomes not very relevant. For this reason, and in order to adapt to the time restrictions of this master's thesis, this parameter will not be explored.

The $min\_duration\_off$ can take any positive value.

Chapter 5

Experiments and results

5.1 First baseline experiment: RTVE2018 - Albayzin

To test the chosen baseline, the first approach was to use the dataset that was available since the start of this work, without any need for manual annotation, the RTVE2018 - Albayzin dataset presented in Section 3.2.

To do this, the "test" partition of the dataset was evaluated using the *pyannote.audio* pretrained pipeline.

The result of this experiment was discouraging. The Diarization Error Rate obtained by the system was 6.2%. This result was too good.

The results of the Albayzin Challenge evaluation over this dataset were way higher [32]. Even considering that the technology could have improved in the last years, the result was breaking those of the state-of-the-art for the number of speakers that the audio files in this dataset had.

This anomaly led to the decision of abandoning the experiments with this dataset since it seemed to yield misleading results.

5.2 Baseline over *À Punt* datasets

To set a baseline for the "À Punt" datasets, the test partition of each task was processed with *pyannote.audio* 2.1, using the pre-trained pipeline with the default hyperparameters ($segmentation\_threshold = 0.4442333667381752$, $cluster\_threshold = 0.7153814381597874$ and $cluster\_size = 15$)

As a result, this experiment set the test Diarization Error Rate (DER) to beat for each dataset, as seen in Table 5.1.

These results have been obtained executing the aforementioned pipeline in a system with 62GB of RAM and an NVIDIA GeForce GTX 1080 Ti with 11264MiB of vRAM.

The results presented in Table 5.1 are sensible compared with those appearing in the literature of the state-of-the-art.

Table 5.1: DER obtained by the *pyannote.audio* pretrained speaker diarization pipeline over the *dev* and *test* partitions of the three "À Punt" datasets.

Dataset	DER(dev)	DER(test)
La Cuina de Morera	10.3%	19.2%
Debates	6.3%	3.4%
La Cuina de Morera + Debates	8.6%	13.1%

It can be clearly seen that the best result is initially obtained in the "Debates" dataset. This can be explained because, even though the number of speakers is higher than in "La Cuina de Morera", speaker interventions are longer, more organized, and less spontaneous, leading to fewer overlaps and speaker changes.

The dataset from "La Cuina de Morera" shows a worse performance. However, this performance is still within the boundaries of what state-of-the-art considers appropriate for a dataset with its challenging features.

To conclude this section, the mixed dataset: "La Cuina de Morera" + "Debates", shows a result that is in between the other two, as expected.

5.3 Initial pipeline hyperparameter optimization

The *pyannote.audio* pretrained pipelines offer several hyper-parameters that can be adjusted to optimize the results in the inference stage.

From these parameters, three of them seem to have a remarkable importance on the final diarization result. For this reason, this section will explore these hyper-parameters, analyzing their implications over the *dev* partition of the tree "À Punt" datasets by setting each other parameter to the default values and changing them in an isolated way.

5.3.1 *La Cuina de Morera*

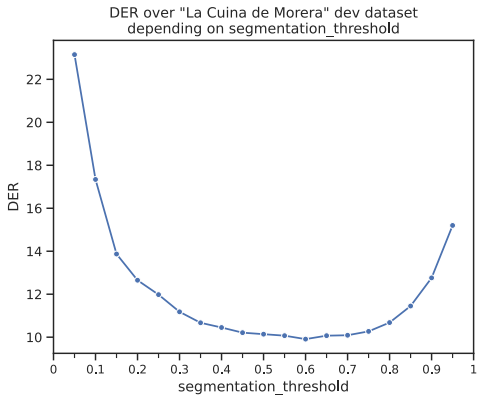
Our first experiments were carried out on the "La Cuina de Morera" dataset. The result of this exploration is shown in Figure 5.1.

5.3.1.1 Segmentation Threshold

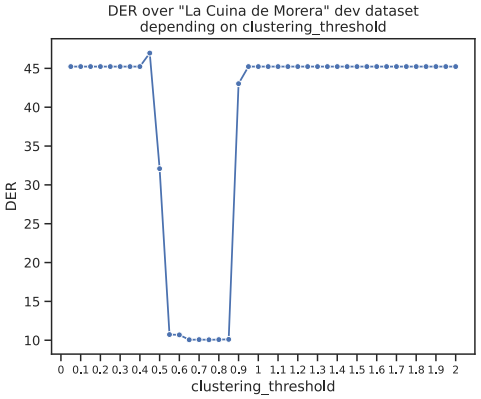
The first hyperparameter that will be explored will be the *segmentation_threshold*.

The Diarization Error Rate (DER) has been represented on the Y-axis depending on the *segmentation_threshold* (θ) on the X-axis and computed over the "dev" partition of "La Cuina de Morera" dataset, creating a line plot that can be seen in Subfigure 5.1a.

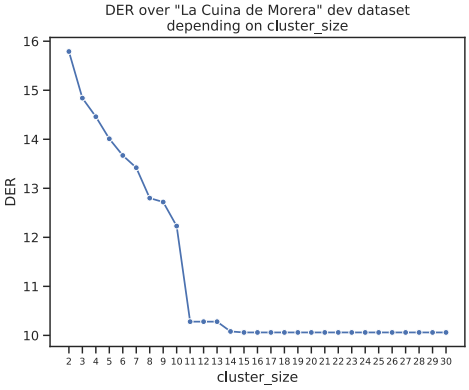
Taking a look at this subfigure, it can be seen that the Diarization Error Rate is very high in lower values, starting at 23.2% when the *segmentation_threshold* takes 0.05 as the value. This is the expected result since setting a very low threshold will



(a) Line plot of the DER over "La Cuina de Morera" dev(—) dataset depending on *segmentation_threshold*.



(b) Line plot of the DER over "La Cuina de Morera" dev(—) dataset depending on *clustering_threshold*.



(c) Line plot of the DER over "La Cuina de Morera" dev(—) dataset depending on *cluster_size*.

Figure 5.1: Line plot of the DER over "La Cuina de Morera" dev(—) dataset depending on inference hyperparameters.

lead to the system considering as active way more speakers than those who are really talking.

From there, the DER rapidly decreases. When arriving at the values that approach, from the lower side to the range's ($[0, 1]$) center, this decrease becomes lighter but continues until arriving at $segmentation\_threshold = 0.6$, where we find the lowest Diarization Error Rate, being it 9.9%.

Then, the DER starts increasing again. First, in a slow way and from the $\theta =$

0.8 more rapidly. This trend is also expected since being too strict when selecting the active speakers will increase the misses, that is, the errors in which the system considers a speaker is not intervening and he is.

5.3.1.2 Clustering Threshold

To see the effect of the *clustering_threshold* on the Diarization Error Rate (DER), those two values have been represented on a line plot.

The DER, computed over the "dev" partition of "La Cuina de Morera" dataset, is represented on the vertical axis, and the *clustering_threshold*, of which the first one depends, on the horizontal axis. This line plot can be seen in Subfigure 5.1b.

The Diarization Error Rate over the dev part of the dataset stars being very high (DER = 45.2%), and continues being very high until the cluster size takes a 0.45 value (included). Selecting a very low distance where the agglomeration process stops can leave the process with too many clusters.

From there, the DER decreases extremely quickly, lowering to 10.7% in only two steps. This is probably caused by the fact that most points that should belong in the same clusters are separated at a distance near this value.

Then the DER becomes very stable until $clustering\_threshold = 0.85$, with the lowest value within the *clustering_threshold* range being 10.06%.

At $clustering\_threshold = 0.9$ the Diarization Error Rate takes a considerable step up, reaching the value of 43%. This is believed to be caused by the fact that points that should belong to different clusters are not further than this distance between them.

5.3.1.3 Cluster Size

The last parameter that will be analyzed in a line plot for "La Cuina de Morera" "dev" dataset is the *clustering_size*. This line plot can be seen in Figure 5.1c.

The analysis will start at $clustering\_size = 2$ and end at $clustering\_size = 30$, two times the default value $clustering\_size = 15$.

The Diarization Error Rate (DER), is represented on the Y-axis, opposed to the *cluster_size*, on the X-Axis.

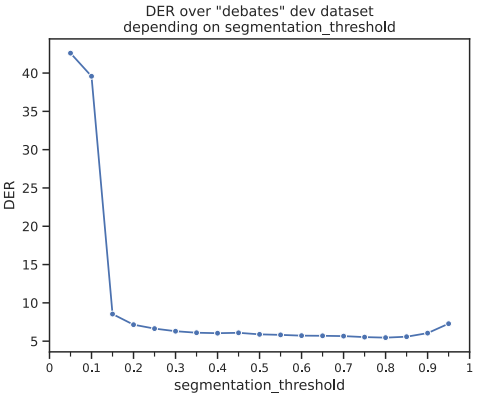
From the starting point, the DER decreases quickly and constantly until arriving at a value of 10.3% when $clustering\_threshold = 11$. This value keeps constant until $clustering\_threshold = 13$.

After that value, the Diarization Error Rate value takes another significant step down, arriving at 10.1%, and after that, it keeps nearly constant.

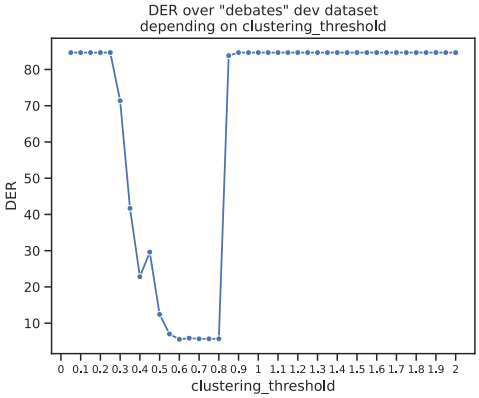
The constant decrease in the line indicates that the vast majority of clusters of 10 points or less do not represent speakers correctly. The fact that the value keeps constant from 14 means that the incorrect clusters are smaller than this number, and the correct ones are bigger than 30.

5.3.2 Electoral debate

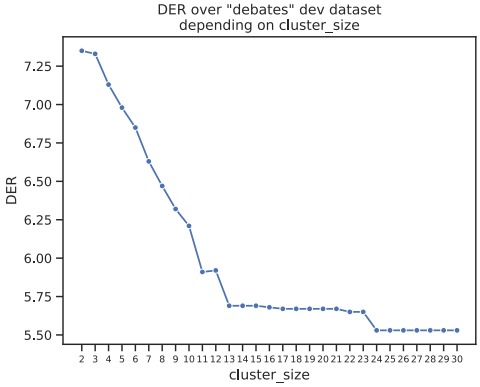
The second dataset in which hyperparameter exploration will be performed will be the "debates" dataset. The result of this exploration is shown in Figure 5.2.



(a) Line plot of the DER over "debates" dev(—) dataset depending on *segmentation_threshold*.



(b) Line plot of the DER over "debates" dev(—) dataset depending on *clustering_threshold*.



(c) Line plot of the DER over "debates" dev(—) dataset depending on *cluster_size*.

Figure 5.2: Line plot of the DER over "debates" dev(—) dataset depending on inference hyperparameters.

5.3.2.1 Segmentation Threshold

First, Subfigure 5.2a shows the evolution of DER (vertical axis) with respect to the Segmentation Threshold parameter (horizontal axis), computed over the "dev" set of "Debats".

Examining it, a trend in the DER values as the segmentation threshold changes can be rapidly noticed. Starting at an initial segmentation threshold of 0.05, the DER is notably high, reaching a value of 42.6%. Again, this is the expected behavior since a very low segmentation threshold is way too permissive, and any hint of intervention is taken as an intervention, when the vast majority of the time, such minor signs are misleading.

However, a significant reduction in Diarization Error Rate is witnessed as the segmentation threshold increases, indicating improved diarization performance. This reduction continues as we incrementally raise the segmentation threshold to 0.4, where the Diarization Error Rate stabilizes at around 6%.

Notably, the Diarization Error Rate remains relatively consistent in this range, with a very slight downtrend, reaching the minimum DER at $segmentation_threshold = 0.8$, where the DER is 5.5%, indicating a robust performance plateau. However, a slight upward trend in Diarization Error Rate can be observed as the segmentation threshold is pushed further to the higher end. Again, this behavior shows the fact that a very high segmentation threshold causes the rise of missed interventions.

5.3.2.2 Clustering Threshold

The line plot confronting the clustering threshold (horizontal axis) versus the Diarization Error Rate (vertical axis) over the "Debates" dev dataset, as we have also seen with the "La Cuina de Morera" dataset in Subfigure 5.1b, starts in a very high position, at 84.7% in this case, and keeps constant until $clustering\_threshold = 0.25$. Again, this shows that points separated less than this value should belong to the same cluster.

When the clustering threshold reaches the 0.3 value, the Diarization Error Rate (DER) starts a very steep fall until reaching $clustering\_threshold = 0.4$. After that, a slight increase is seen at $clustering\_threshold = 0.45$ breaking the downtrend, but this is only an exception since this trend continues after this data point until reaching the clustering threshold of 0.6 where the DER is the lowest one in the plot, 5.6%. The slight increase in DER during the fall can be attributed to the fact that stopping there still stops agglomerating clusters that are together in the optimal configuration but joins some clusters that, in fact, are different and were not joined with the previously tried distance.

To end, at $clustering\_threshold = 0.85$, the DER suddenly reaches again over 80%, and that plateau continues until the end of the range. Again, this can be attributed to the fact that clusters that should not be together have a smaller separation than this value.

5.3.2.3 Cluster Size

Taking a look at the line plot of the Diarization Error Rate (Y-axis) versus the *cluster_size* (X-axis) over the "Debates" "dev" dataset, plotted at Subfigure 5.2c, it can easily be seen that, again, the DER starts at a pretty high value compared to the one on the rest of the parameter range. However, in this case, this high value is only 7.35%, which is low compared to the results over the "La Cuina de Morera" dataset, which line plots can be seen in Figure 5.1. Still, the reason behind this is expected to be the same: The clusters with smaller sizes are, in general, not correct as standalone speaker clusters.

From that point, the Diarization Error Rate decreases every time the cluster size increases until reaching 13, where it seems to stabilize, getting a very similar value around 5.7%. Then, at *clustering_size* = 24 it takes a little step down again and stabilizes until the end of the studied range around DER = 5.5%. This shows that processing this dataset, bigger incorrect clusters appear, some of them even with 23 points, that should be joined with the bigger ones.

5.3.3 *La Cuina de Morera* + Electoral debate

5.3.3.1 Segmentation Threshold

In this last *segmentation_threshold* exploration, the Diarization Error Rate (DER) is presented through another line plot on the vertical axis, depicting its relationship with various segmentation thresholds across the "La Cuina de Morera" + "Debates" "dev" dataset in the vertical axis. The plot is displayed in Subfigure 5.3a.

Upon examination of the subfigure, a discernible trend is observed in the Diarization Error Rate values as the segmentation threshold is adjusted. At the lowest threshold of 0.05, a substantial elevation in DER is noted, peaking at 31.3%. Again, we see that, like in the two previously analyzed datasets, setting a very permissive threshold causes a lot of false alarm errors.

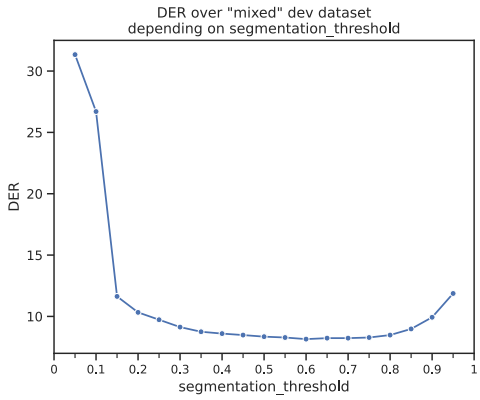
However, an appreciable reduction in DER becomes evident as the segmentation threshold is incrementally raised. A significant decrease occurs when the threshold reaches 0.15, with the DER plummeting to a mere 11.6%

Nevertheless, an upward trend in DER is observed as the segmentation threshold is pushed beyond this optimal range. At a threshold of 0.9, the DER breaks back the 9% barrier and further increases to 11.87% at 0.95, probably due to increased missed interventions.

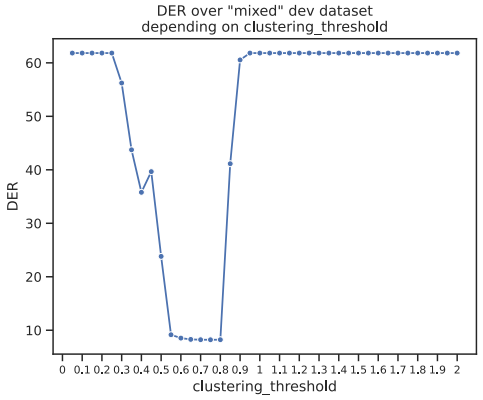
5.3.3.2 Clustering Threshold

Regarding the clustering threshold for this combined "dev" dataset, the line plot exploring its range of possible values is shown on the horizontal axis of Subfigure 5.3b.

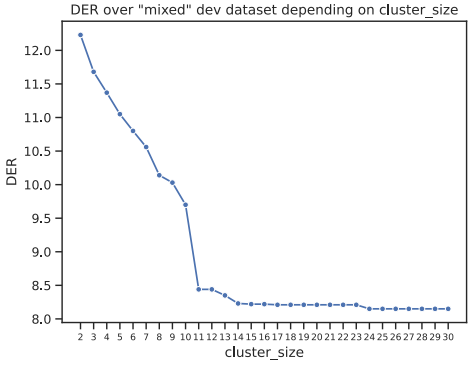
It can be seen again that the Diarization Error Rate, represented on the vertical axis, for the first few points starts at a very high value. If a closer look is taken, it can be figured out that the value that appears in this case, 61.8%, is a value between



(a) Line plot of the DER over "La Cuina de Morera" + "debates" dev(—) dataset depending on *segmentation_threshold*.



(b) Line plot of the DER over "La Cuina de Morera" + "debates" dev(—) dataset depending on *clustering_threshold*.



(c) Line plot of the DER over "La Cuina de Morera" + "debates"' dev(—) dataset depending on *cluster_size*.

Figure 5.3: Line plot of the DER over "La Cuina de Morera" + "debates" dev(—) dataset depending on inference hyperparameters.

the ones seen in the two separate datasets, as it can be seen in subfigures 5.1b and 5.2b.

At *clustering_threshold* = 0.3 the characteristic quick fall starts, stopping at 0.45 to show a peak that was also seen in 5.2b, and then continuing until reaching the clustering threshold of 0.55 when the Diarization Error Rate goes under 10% for the first time. After that, the lowest value at 0.75 with a 8.2% DER. Finally, the clustering threshold goes up again, reaching the initial values, and keeps at that level

for all the remaining range values.

It can be clearly seen that the behavior of the line is a combination between those shown in Figures 5.1b and 5.2b, and, hence, can be explained with the same reasons.

5.3.3.3 Cluster Size

This last line plot of the first hyperparameter exploration, which can be seen in Subfigure 5.3c, shows the cluster size on the horizontal axis and the DER on the vertical axis. It has been computed over the combined "dev" dataset of "La Cuina de Morera" + "Debates", and it is the one among those representing the cluster size with more plateaus appearing.

As always, when analyzing this parameter, the Diarization Error Rate starts being high and quickly and progressively decreases until reaching a plateau.

In this case, the first reached plateau is tiny, consisting of only two data points, the ones in which the cluster size is 11 and 12, giving a DER of 8.4%. Then, it takes two points to reach the next plateau, which is the bigger one, starting at $cluster\_size = 15$ and ending at $cluster\_size = 23$. At 24, the value of the Diarization Error Rate decreases slightly again, reaching the plateau of the lowest DER, 8.2%.

The plateaus shown are a combination of the ones shown in the two previous sections and can be explained by the same reasons.

5.4 Finetuning the segmentation model

The segmentation model can be finetuned in an isolated way, and this can be done even with a small quantity of training data. By doing this, the segmentation explained in Subsection 4.1 is expected to improve, reducing the Diarization Error Rate (DER).

Moreover, finetuning the segmentation model of the *pyannote.audio* pipeline makes the false gaps between intra-speaker turns nearly disappear, making exploring the minimum duration off hyperparameter (Δ) unnecessary, as explained in Subsubsection 4.5.4.

For these reasons, the segmentation model will be finetuned using a variety of batch sizes and learning rate values. These values will be a few ones that are close to the ones recommended by the *pyannote.database*, since they are supposed to be the best for this architecture.

The finetuning has been performed for 30 iterations with early stopping set to a patience of 10.

The used partitions of the datasets will be the "train" partition to finetune the model and the "dev" partition to determine whether to stop and the best model generated among those created for each iteration.

5.4.1 *La Cuina de Morera*

The finetuning of the segmentation model for the "La Cuina de Morera" dataset has been done using three different values of learning rate and another three values of

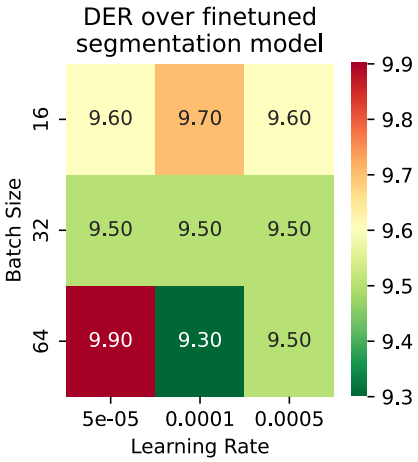


Figure 5.4: Heatmap of DER over "La Cuina de Morera" dev dataset depending on Learning Rate and Batch Size

batch size. By performing a grid search, the total number of finetuned models has been nine, which have been separately evaluated over the dev partition of the dataset.

The result of these evaluations is shown on the heat map that can be visualized in Figure 5.4, with the learning rate on the horizontal axis and the batch size on the vertical one.

By looking at this figure, it can be seen that no clear trend can be appreciated. The models which have been finetuned with a batch size of 32 show a stable Diarization Error Rate of 9.50%. Those trained with a batch size of 16, the smaller one tried, delivered a relatively lousy performance with values of 9.60% and 9.70%.

The best result appears when training the model with a batch size of 64. This, combined with a 0.0001 learning rate, achieves a DER of only 9.30%. However, the batch size of 64 seems to be less stable, having also the worst result when combined with the lowest learning rate and, when combined with the highest learning rate, a result equal to those of batch size 32.

5.4.2 Electoral debate

The second finetuning of the segmentation model has been done using the "Debates" dataset, and the nine finetuning processes performed, with the same parameters applied to "La Cuina de Morera" dataset, have produced the results shown in 5.5, for the "dev" partition. The learning rate is represented on the X-axis, and the batch size is on the Y-axis.

The majority of the values of this heatmap are around 6% Diarization Error Rate, appearing this value three times, 5.9% three more times, and 6.1% two times, which are the worst values of DER over this dataset, and appear for the 32 a 64 batch sizes

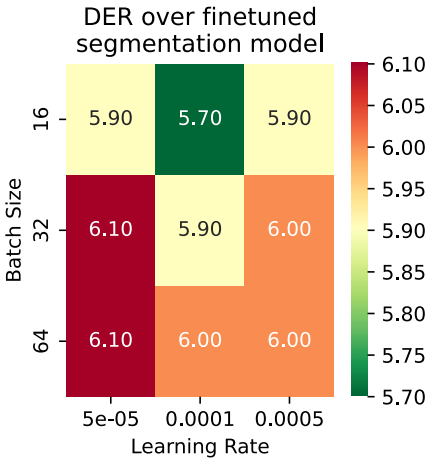


Figure 5.5: Heatmap of DER over "Debates" dev dataset depending on Learning Rate and Batch Size

combined with the 5e-05 learning rate.

The best value is the one obtained when finetuning the segmentation model with a batch size of 16 and a learning rate of 0.0001, and it is a Diarization Error Rate of 5.7%.

5.4.3 *La Cuina de Morera* + Electoral debate

The last finetuning was performed with both "La Cuina de Morera" and "Debates" datasets combined. Again, it has been evaluated over the "dev" partition and represented on a heat map, with the learning rate on the horizontal axis and the batch size on the vertical one.

Figure 5.6 shows that finetuning the model with a batch size of 16 does not bring the best results. Every value obtained using this batch size exceeds 8% Diarization Error Rate.

Once the batch size is increased to 32, the first DER not greater than 8% appears, showing precisely that 8% as value when the learning rate is set to 5e-05.

To end, the greater batch size tried, 64, yields the best results. Those improve when the learning rate increases, starting at 8.10% DER when it is 5e-05, bringing the first value under 8% when the learning rate is 0.0001 and finally giving the best result (of 7.8% Diarization Error Rate) with the biggest batch size, 0.0005.

5.5 Hyper-parameter tuning in inference

The last step to find the best Speaker Diarization system for each of the three analyzed datasets is to adapt the hyper-parameters of the finetuned pipeline to the specific

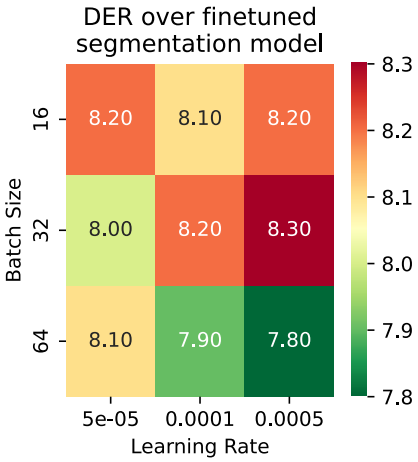


Figure 5.6: Heatmap of DER over "La Cuina de Morera" + "Debates" dev dataset depending on Learning Rate and Batch Size

features of each dataset in order to reduce the Diarization Error Rate (DER) as much as possible, taking the development subset as reference.

As it has already been explained in Subsection 4.5, there are two hyperparameters that, when correctly tuned, can reduce the DER of the diarization system in a very relevant way.

These are the segmentation threshold and the clustering size, in this order. For this reason, they will be analyzed using a grid search process, and the different combinations of values will be shown in one heatmap per dataset. After that, the data plotted in them will be commented.

To end, the clustering size has been explored over the finetuned models, and the best values for the segmentation and clustering threshold.

5.5.1 *La Cuina de Morera*

The hyper-parameter tuning has been performed to the pipeline with the segmentation model finetuned for "La Cuina de Morera" dataset. It can be seen in Figure 5.7.

The grid search of the segmentation threshold and clustering threshold has been done within the best-performing ranges discovered for this dataset in Subsection 5.3.1. Specifically, for both parameters, the range has been set by choosing the values that yielded less than 11% Diarization Error Rate (DER). These ranges are:

- $segmentation\_threshold \in [0.35, 0.8]$
- $clustering\_trheshold \in [0.55, 0.85]$

Looking at the heatmap, where the segmentation threshold and clustering threshold have been represented on the horizonal and vertical axes respectively, it can be

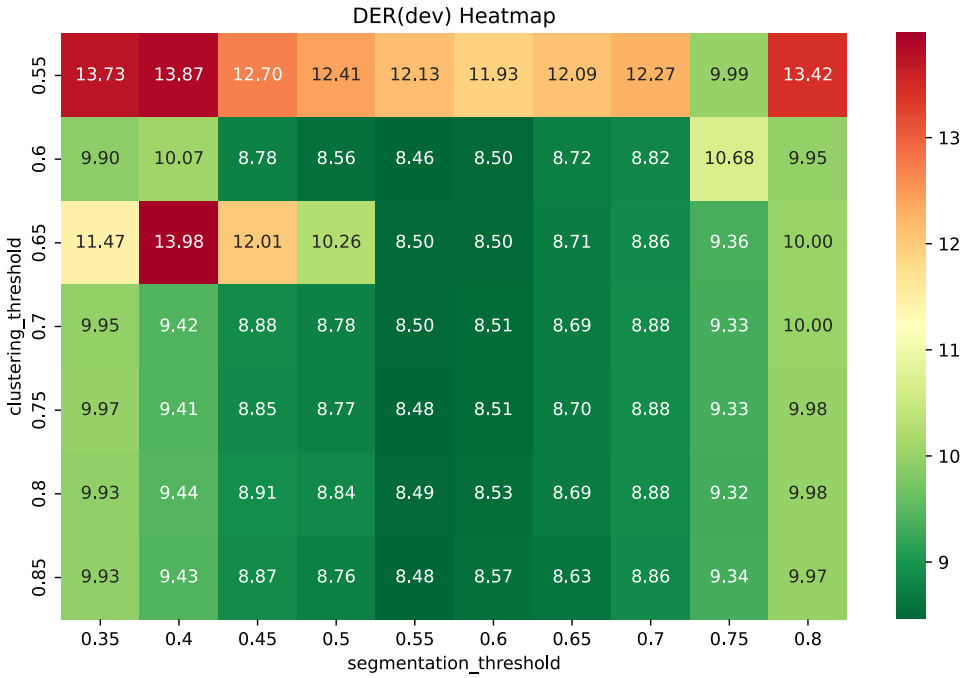


Figure 5.7: Heatmap of DER over "La Cuina de Morera" dev dataset depending on segmentation and clustering threshold used in inference with its finetuned segmentation model

easily noted that the worst results generally appear for the lowest clustering threshold, 0.55.

It is also essential to notice a clear trend when changing the segmentation threshold. The best values for this hyperparameter are those close to the center of the exploration. One column excels over the other ones, the one which represents the segmentation threshold equal to 0.55.

In that column, every value is around 8.5%, and the best value of that column is also the lowest value obtained in the experiment and is the product of choosing a clustering threshold of 0.6. However, the difference with other values on the column is so slight that nearly every system on it could perform well.

5.5.2 Electoral debate

The second heatmap exploring hyper-parameters in a grid search has been performed over the pipeline with its segmentation model trained with the "Debates" dataset. It can be seen in Figure 5.8. The segmentation threshold has been represented on the X-axis, and the clustering threshold is on the Y-axis.

The selected range for the grid search exploration gathers the values that produced

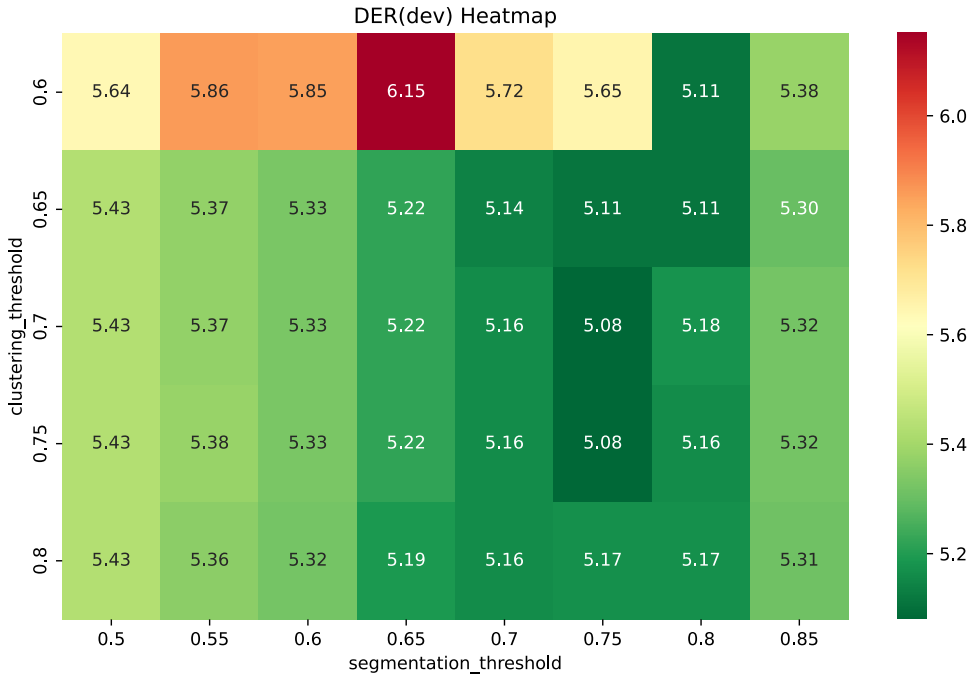


Figure 5.8: Heatmap of DER over "Debates" dev dataset depending on segmentation and clustering threshold used in inference with its finetuned segmentation model

a Diarization Error Rate over the development subset under 6%. Those values are:

- $segmentation\_threshold \in [0.5, 0.85]$
- $clustering\_trheshold \in [0.6, 0.8]$

This heatmap shares some features with the one obtained in the previous subsection for "La Cuina de Morera" system. In the first place, the lower value for the segmentation threshold, in this case 0.6, shows the worst results. Secondly, there is a column that concentrates on the best results.

However, in this case, this column is not in the center of the exploration range but is skewed to the higher part of it. This column is the one where the segmentation threshold is 0.75.

It is also worth noting that there is more than one single best value. Two segmentation and clustering threshold combinations give the same lowest DER: the segmentation threshold in the column with the best results ($segmentation\_threshold = 0.75$) and the clustering threshold set to either 0.7 or 0.75. Again, there are other values around 5.1%, which could have been splendid systems since the difference is very small.

5.5.3 *La Cuina de Morera* + Electoral debate

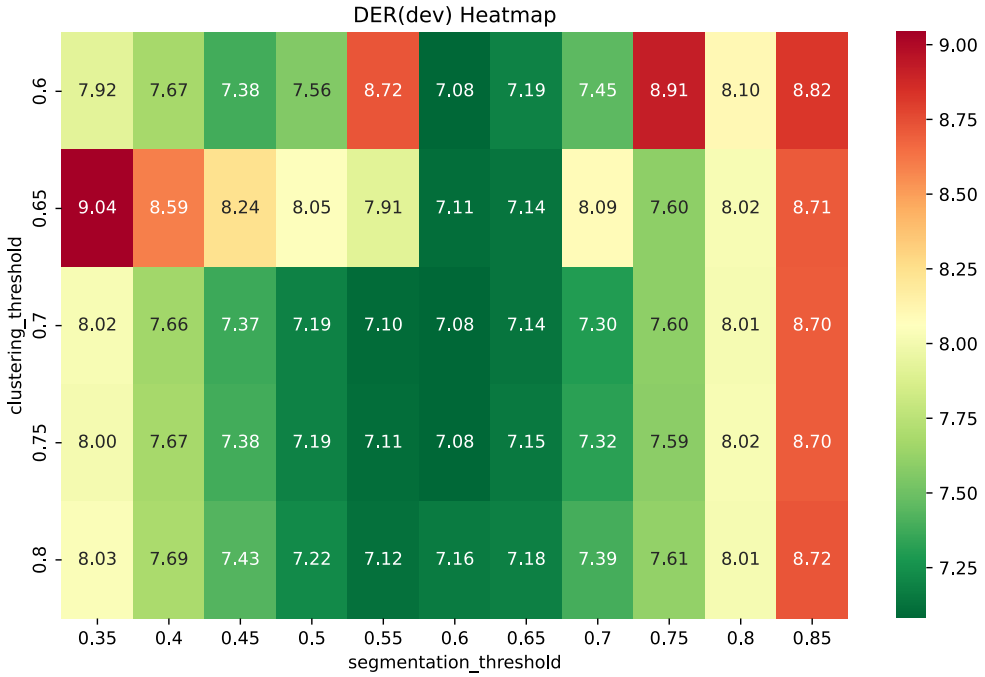


Figure 5.9: Heatmap of DER over "La Cuina de Morera" + "Debates" dev dataset depending on segmentation and clustering threshold used in inference with its finetuned segmentation model

The last grid search over the inference hyperparameters has been performed over "La Cuina de Morera" + "Debates" combined dataset. The heatmap with its result is shown in Figure 5.9. As always, the segmentation threshold appears on the horizontal axis and the clustering threshold on the vertical one, and the DER has been computed over the "dev" part of the dataset.

The range for the grid search, in this case, is formed by the values that produced a Diarization Error Rate over the development subset under 9%. Those values are:

- $segmentation\_threshold \in [0.35, 0.85]$
- $clustering\_trheshold \in [0.6, 0.8]$

In this heatmap, the worst values are not concentrated in one clustering threshold row like in the previous systems but in a segmentation threshold column. Specifically, in the one with the biggest value for this hyperparameter, this value 0.85.

Like in the two previous datasets, there is a column that sets a segmentation threshold as the best one, showing a relatively good performance for nearly each value

of the clustering threshold. In this case, the segmentation threshold for achieving this is 0.6.

In this column, the three best values, which are a tie, appear for the clustering thresholds of 0.6, 0.7, and 0.75. As well as in the two previous sections where a hyperparameter grid search is performed, there are systems around this value which were also valid candidates to be a final system, with DERs near 7.1%.

5.6 Final systems

To conclude the experimentation phase, it is essential to try the developed systems over the test partitions of each dataset to see whether the improvements observed in the dev partition while developing the systems reflect, as expected, in better results also for this test partition.

To do that, three Python scripts were developed to use each dataset's respective finetuned segmentation model and the segmentation and clustering threshold values found in Section 5.5. The cluster size was set to 30 as this value proved to be consistently excellent in Section 5.3.

Table 5.2: DER obtained by the *pyannote.audio* speaker diarization pipeline over the *test* partition of the three "À Punt" datasets, before (baseline), and after (final) adapting it to each one by finetuning the segmentation model and tuning its inference hyper-parameters.

Dataset	Baseline DER(test)	Final DER(test)
La Cuina de Morera	19.2%	10.1%
Debates	3.4%	2.4%
La Cuina de Morera + Debates	13.1%	7.2%

In Table 5.2, the Diarization Error Rates of the three baseline systems are compared to the DERs of the final improved systems.

In the "La Cuina de Morera" dataset test split, the DER has reduced from 19.2% to 10.09%, which implies a relative improvement of 47% of the error rate.

Following, in the "Debates" test subset, the Diarization Error Rate has seen a relative improvement of 29% by going from a DER of 3.4% to one of only 2.4%.

To finish, the "La Cuina de Morera" - "Debates" mixed dataset had a 13.1% Diarization Error Rate when processed by the baseline system. After finetuning its segmentation model and its inference hyperparameters, it now yields a DER of only 7.2%. This means a relative improvement of 45%.

Chapter 6

Conclusions and future work

In this chapter, different parts of this master's thesis will be reviewed and summarized to obtain conclusions. The goals set in Section 1.4 will be taken into account, and the results obtained in Chapter 5 will be analyzed in order to determine if the aforementioned goals have been accomplished. To close this chapter, some interesting future work will be proposed, taking the systems developed in this work and their result as the foundation.

If one thing is vital in this chapter, it is to remark that the two goals set in Section 1.4 have been both achieved.

The first goal ("Implementing a baseline system from the state of the art and applying it to the MLLP-CVMC context") was accomplished on Chapter 4. In this section, the *pyannote.audio* pre-trained pipeline was selected as the baseline system. This pipeline uses state-of-the-art technologies, as established on the goal, such as the EEND diarization models [28, 29, 33] and ECAPA-TDNN embeddings [16] (Explained in Subsection 2.5.4 and Subsubsection 2.5.3.2 respectively). The baseline was used to process the different datasets that were relevant to the MLLP-CVMC agreement, and from that, the Diarization Error Rate to beat for each dataset was set in Table 5.1.

Regarding the second goal: "Adapting the baseline system to different tasks of the MLLP-CVMC context in order to get improved systems with better results", many adaptations have been performed to the original system with the three datasets of the MLLP-CVMC context.

In Section 5.3, a general overview and exploration of the hyperparameters have been performed to later adapt the *pipeline.audio* to the different "À Punt" datasets.

After that, a first adaptation of the baseline to the desired context has been done by finetuning the segmentation model of the *pyannote.audio* pretrained pipeline.

To conclude the system's adaptation, a more specific tuning of the most relevant hyperparameters has been executed for every previously adapted system, employing a grid search involving the segmentation and clustering threshold.

With these three steps, it can be claimed that the baseline has been adapted to the different tasks of the MLLP-CVMC context as the goal told.

Whether these adaptations led to improved systems with better Diarization Error Rates, this has been clarified in Section 5.6. In that section, it has been seen that the adaptations for each dataset have brought relevant reductions of their DERs. The systems adapted to "La Cuina de Morera", "Debates", and "La Cuina de Morera" + "Debates" datasets caused relative improvements of, respectively 47%, 29%, 45% as it can be seen in Table 5.2.

Once the goals have been reviewed and marked as achieved, some proposals for future work can be made.

It will be really interesting to try to improve the effect of finetuning the segmentation model, which seems to still have some potential to extract. As it has been mentioned multiple times in this document, one of the most time-consuming tasks performed in this work was data annotation to obtain the datasets. However, it is likely that labeling more videos of each program to use as training and to make the development and test sets even more representative of any other episode of the program will unlock the full potential of finetuning the segmentation model.

Going a step further, if the quantity of labeled data gets to be big enough, the finetuning of the embedding ECAPA-TDNN [16] can also be tried. To show improvements, this process needs way more data than was available for the development of this master's thesis.

Another thrilling task to propose as future work is to explore the potential of transformers [38] in the Speaker Diarization field.

As it is widely known in the Machine Learning research community, transformers [38] have brought a huge revolution to many application fields such as Machine Translation, Automatic Speech Recognition or Computer Vision.

However, in the Speaker Diarization field, despite the fact that a light use of attention and Self-Attention has been done, and has proved to improve some specific models used in SD such as End-to-End Neural Speaker Diarization [29] or ECAPA-TDNN [16]; transformers [38] potential has not been extensively explored.

For this reason, it will be worth investing time in researching the application of this technology to the Speaker Diarization field. Moreover, its results on Automatic Speech Recognition, which, as seen in Chapter 2, is a field that has many features similar to those found in SD, have been excellent and a significant improvement over previous systems. This can lead to thinking that a similar improvement can be seen in Speaker Diarization if enough research is done.

The use of transformers [38] seems to have a vast potential to create embeddings that encapsulate information in an outstanding way. This could be a very attractive alternative to the ECAPA-TDNN embeddings if developed.

Also, in the segmentation model, to extract the features from raw waveforms, instead of using SincNet [23], as the *pyannote.audio* segmentation model does at the moment, transformers [38] can be tried.

In conclusion, exploring transformers [38] is an exciting task to set as future work.

It must be noted that this work is only focused on offline Speaker Diarization; that is, the developed systems need an audio file as input and can not work with live

audio streaming. Online SD systems are a challenging but very interesting field of research for the audiovisual industry. Taking this into account, another interesting future work can be trying to develop online SD systems.

Bibliography

[1] Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J. Han, Shinji Watanabe, and Shrikanth Narayanan. A Review of Speaker Diarization: Recent Advances with Deep Learning, November 2021.

[2] Shelly Chadha, Kaloyan Kamenov, and Alarcos Cieza. The world report on hearing, 2021. *Bulletin of the World Health Organization*, 99(4):242–242A, April 2021.

[3] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[4] T.M. Mitchell. *Machine Learning*. McGraw-Hill International Editions. McGraw-Hill, 1997.

[5] Kevin P. Murphy. *Probabilistic Machine Learning: An Introduction*. The MIT Press, 2022.

[6] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, November 1958.

[7] Johannes Lederer. Activation Functions in Artificial Neural Networks: A Systematic Overview, January 2021.

[8] Alex Sherstinsky. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. *Physica D: Nonlinear Phenomena*, 404:132306, March 2020.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997.

[10] Anupama Ray, Sai Rajeswar, and Santanu Chaudhury. Text recognition using deep BLSTM networks. In *2015 Eighth International Conference on Advances in Pattern Recognition (ICAPR)*, pages 1–6, January 2015.

[11] Владимир Иосифович Левенштейн. Двоичные коды с исправлением выпадений, вставок и замещений символов. In *Доклады Академии наук*, volume 163, pages 845–848. Российская академия наук, 1965.

[12] Jonathan G. Fiscus, Jerome Ajot, Martial Michel, and John S. Garofolo. The Rich Transcription 2006 Spring Meeting Recognition Evaluation. In Steve Renals, Samy Bengio, and Jonathan G. Fiscus, editors, *Machine Learning for Multimodal Interaction*, Lecture Notes in Computer Science, pages 309–322, Berlin, Heidelberg, 2006. Springer.

[13] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. X-Vectors: Robust DNN Embeddings for Speaker Recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333, April 2018.

[14] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur. Deep Neural Network Embeddings for Text-Independent Speaker Verification. pages 999–1003, August 2017.

[15] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. pages 3214–3218, September 2015.

[16] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In *Interspeech 2020*. ISCA, October 2020.

[17] David Snyder, Daniel Garcia-Romero, Gregory Sell, Alan McCree, Daniel Povey, and Sanjeev Khudanpur. Speaker recognition for multi-speaker conversations using X-vectors. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5796–5800, 2019.

[18] Daniel Garcia-Romero, Alan McCree, David Snyder, and Gregory Sell. Jhu-HLTCOE system for the voxsrc speaker recognition challenge. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7559–7563, 2020.

[19] Hossein Zeinali, Shuai Wang, Anna Silnova, Pavel Matějka, and Oldřich Plchot. BUT system description to VoxCeleb speaker recognition challenge 2019, 2019.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[21] Koji Okabe, Takafumi Koshinaka, and Koichi Shinoda. Attentive statistics pooling for deep speaker embedding. In *Interspeech 2018*. ISCA, September 2018.

[22] Yingke Zhu, Tom Ko, David Snyder, Brian Mak, and Daniel Povey. Self-attentive speaker embeddings for text-independent speaker verification. In *Proc. Interspeech 2018*, pages 3573–3577, 2018.

[23] Mirco Ravanelli and Yoshua Bengio. Speaker Recognition from Raw Waveform with SincNet, August 2019.

[24] Dimitri Palaz, Mathew Magimai-Doss, and Ronan Collobert. Analysis of CNN-Based Speech Recognition System Using Raw Speech as Input.

[25] George Trigeorgis, Fabien Ringeval, Raymond Brueckner, Erik Marchi, Mihalis A. Nicolaou, Björn Schuller, and Stefanos Zafeiriou. Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network.

In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5200–5204, March 2016.

[26] Pegah Ghahremani, Vimal Manohar, Daniel Povey, and Sanjeev Khudanpur. Acoustic Modelling from the Signal Domain Using CNNs. In *Interspeech 2016*, pages 3434–3438. ISCA, September 2016.

[27] Jee-Weon Jung, Hee-Soo Heo, Il-Ho Yang, Hye-Jin Shim, and Ha-Jin Yu. A Complete End-to-End Speaker Verification System Using Deep Neural Networks: From Raw Signals to Verification Result. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5349–5353, April 2018.

[28] Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Kenji Nagamatsu, and Shinji Watanabe. End-to-End Neural Speaker Diarization with Permutation-Free Objectives, September 2019.

[29] Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Yawen Xue, Kenji Nagamatsu, and Shinji Watanabe. End-to-end neural speaker diarization with self-attention, 2019.

[30] David Van Leeuwen and Jos Bouten. The NFI/TNO Forensic Speaker Recognition Evaluation Plan.

[31] Eduardo Lleida, Alfonso Ortega, Antonio Miguel, Virginia Bazan, Manuel Gomez, and Alberto de Prada. RTVE2018 Database Description. 2018.

[32] Eduardo Lleida, Alfonso Ortega, Antonio Miguel, Virginia Bazán-Gil, Carmen Pérez, Manuel Gómez, and Alberto de Prada. Albayzin 2018 Evaluation: The IberSpeech-RTVE Challenge on Speech Technologies for Spanish Broadcast Media. *Applied Sciences*, 9(24):5412, January 2019. Number: 24 Publisher: Multidisciplinary Digital Publishing Institute.

[33] Hervé Bredin and Antoine Laurent. End-to-end speaker segmentation for overlap-aware resegmentation, 2021.

[34] Herve Bredin. PYANNOTE.AUDIO 2.1 SPEAKER DIARIZATION PIPELINE: PRINCIPLE, BENCHMARK, AND RECIPE.

[35] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, et al. SpeechBrain: A general-purpose speech toolkit, 2021.

[36] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. VoxCeleb: A large-scale speaker identification dataset. In *Interspeech 2017*. ISCA, August 2017.

[37] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. VoxCeleb2: Deep speaker recognition. In *Interspeech 2018*. ISCA, September 2018.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, December 2017.

List of Figures

List of Tables

Agraïments

Com la gran majoria de vegades que he d'agrair coses a gent, en primer lloc, vull agrair als meus pares. El seu suport i l'educació que m'han donat tota la meua vida son fonament indispensable de tot allò que aconseguisc. Arribar a aquest punt de la meua vida acadèmica no només no és excepció, és un dels màxims exponents.

També vull agrair a tota la gent del MLLP. Ells han creat un ambient de treball idoni on desenvolupar aquest treball de fi de màster.

Agraïsc als meus tutors Alfons i Joan Albert haver-me guiat en aquest treball.

També vull enviar un agraïment formal a ValgrAI per la beca que em concediren per estudiar aquest màster.

Finalment (encara que és un dels agraïments més especials), vull agrair a tots els amics que m'han acompanyat i m'han donat suport. Als que porten molts anys amb mi, als que he fet fa poc en aquest màster o en el treball, i inclús a aquells que m'han acompanyat en altres etapes i ara recorren camins separats del meu. Moltes gràcies a tots!

APPENDIX

SUSTAINABLE DEVELOPMENT GOALS

Degree to which the work relates to the Sustainable Development Goals (SDGs).

Sustainable Development Goals (SDGs)	High	Medium	Low	Not applicable
SDG 1. **No Poverty.**				✓
SDG 2. **Zero Hunger.**				✓
SDG 3. **Good Health and Well-Being.**	✓			
SDG 4. **Quality Education.**		✓		
SDG 5. **Gender Equality.**				✓
SDG 6. **Clean Water and Sanitation.**				✓
SDG 7. **Affordable and Clean Energy.**				✓
SDG 8. **Decent Work and Economic Growth.**			✓	
SDG 9. **Industry, Innovation, and Infrastructure.**		✓		✓
SDG 10. **Reduced Inequalities.**	✓			
SDG 11. **Sustainable Cities and Communities.**				✓
SDG 12. **Responsible Consumption and Production.**				✓
SDG 13. **Climate Action.**				✓
SDG 14. **Life Below Water.**				✓
SDG 15. **Life on Land.**				✓
SDG 16. **Peace, Justice and Strong Institutions.**				✓
SDG 17. **Partnerships for the Goals.**	✓			

As explained in the introduction of this master's thesis, according to the World Health Organization (WHO), in 2021, more than 1.5 billion people were affected by hearing loss. The data of the WHO is not optimistic since these statistics are expected to worsen with demographic changes, with predictions suggesting that hearing loss could be affecting around 2.5 billion people in 2050. This is a massive health problem, and trying to mitigate its effects definitely contributes to the health and well-being of the population. For this reason, this work aligns highly with SDG 3: "Good Health and Well-Being".

The inclusion of people who have hearing loss is not only a great way but an essential need to reduce inequalities, as the SDG 10: "Reduced Inequalities" requires.

Escola Tècnica Superior d'Enginyeria Informàtica

ETS Enginyeria Informàtica
Camí de Vera, s/n, 46022, València
T +34 963 877 210
F +34 963 877 219
etsinf@upvnet.upv.es - www.inf.upv.es

For that reason, developing new techniques to adapt audiovisual content for them is contributing to the SDG 10.

One of the main ways of adapting audiovisual content for people suffering from hearing loss is the creation of quality subtitles, and the goal of this work is to use cutting-edge technology based on Machine Learning to make this adaptation viable and easy for any entity or person producing media content.

It has also been explained that subtitles are helpful not just for people who can not hear well but, when used with translations, they become a valuable tool for making content easy to understand at a reasonable cost. This benefits individuals who don't speak the original language of the media content without the necessity of adding voiceovers, which can be more complex and costly. Taking this into account, they can also be considered a very helpful asset for education, contributing to SDG 4: "Quality Education".

As it has also been already explained in the introduction, this master's thesis has been developed while working at the Machine Learning and Language Processing Group (MLLP), a research group of the Universitat Politècnica de València (UPV). Specifically, it was conducted as part of the MLLP-Corporació Valenciana de Mitjans de Comunicació (CVMC) agreement: "Subtitulacio Assistida Per Ordinador en Temps Real i Basada en la Intel·ligencia Artificial, de Continguts Audiovisuals". For this reason, it aligns with SDG 17: "Partnerships for the Goals".

Since the developed Speaker Diarization systems are innovative and can be used by the audiovisual industry, this master's thesis also aligns with the SDG: "Industry, Innovation, and Infrastructure". Moreover, other industries can also use diarization systems to improve processes, be more efficient and be more sustainable.

This master's thesis is also lightly aligned with the SDG 8. "Decent Work and Economic Growth", since, as it has already been explained, Speaker Diarization systems will make the creation of quality subtitles much more easy and more efficient. This will lead to an increased capacity in accessible audiovisual production, witch, at the same time, will promote economic growth. Moreover, these systems serve as tools for people working in the subtitling industry. If the parts of these jobs which are more monotonous and time-consuming are reduced by these Speaker Diarization and Automatic Speech Recognition systems, these jobs will become better.

Escola Tècnica Superior d'Enginyeria Informàtica

ETS Enginyeria Informàtica
Camí de Vera, s/n, 46022, València
T +34 963 877 210
F +34 963 877 219
etsinf@upvnet.upv.es - www.inf.upv.es