Universidad Autónoma de Yucatán

Facultad de Matemáticas

Online Speaker Diarization Using Single Speaker Model Update with Embeddings

T H E S I S

Submitted for the degree of

Bachelor in Computer Engineering

by

Br. Carlos Rodrigo Castillo-Sanchez

Thesis advisors

Dr. Leibny Paola Garcia-Perera

Dr. Anabel Martin-Gonzalez

Mérida, Yucatán, México

January, 2020

This work is wholeheartedly dedicated to my parents
Gabriela and Carlos, who inspired me to be a researcher.
Furthermore, to my girlfriend Karen, who gave me
the strength when I thought of giving up.

Acknowledgments

Throughout the development of this work, I have received an incredible amount of support and assistance beyond my highest expectations; without it, this work could not have been possible.

First, I would like to thank Ramón Peniche Mena, Ph.D., and Jorge Armando Argáez Sosa, Ph.D., for providing the resources required for my research stay at the Johns Hopkins Center for Language and Speech Processing (CLSP) for the development of this study. Also, Martha Imelda Jarero Kumul, M.Sc., for its resolution of uncertainties throughout the process.

I would also like to thank my supervisor, Anabel Martín González, Ph.D., who saw my potential and pushed me to go further, inspiring me to begin this research. With your expertise and insightful feedback, this work was brought to a higher level.

Furthermore, I would like to thank my supervisor, Leibny Paola García Perera, Ph.D., who taught me everything I know about speech technologies and enhanced my development as a researcher. Thank you for your patient support and all the opportunities you gave me in this research area. I am forever in your debt for having taught me so much, specifically, not to give up.

I want to acknowledge Diego Nígel Joaquín Campos Sobrino, M.Sc., Mario Alejandro Campos Soberanis, M.Sc., and Juan José Negrón Granados, M.Sc, for their helpful discussions and guidance.

Besides, I would like to thank my parents for their support and sympathetic ear. You are always there for me. Finally, I could not have completed this work without the support of my girlfriend, Karen. You were in my worst moments and gave me the strength to go ahead. Without you, this could not be possible.

Abstract

Speech has become one of the most essential means for human-machine communication; with technologies such as assistant devices, wearable devices, and in-vehicle information systems performing each day more sensitive tasks, improving human-machine communication is an essential task to assure safe execution.

The speech technologies field can be divided into many subfields that tackle different speech-related tasks; from all of them, speaker diarization is an enabling technology that solves the question "*who spoke when?*" by finding the number of speakers in an audio recording and labeling speech regions of the recording accordingly to the identity of whom uttered them.

In recent years offline speaker diarization techniques have reached state-of-the-art performance. Such techniques cannot be used in real-time applications since they require complete speech data upfront. Suppose the application is latency-sensitive, such as a multi-person voice interactive system. In that case, it must have speaker labels generated as soon as speech segments are available to the system; this online variant of speaker diarization has hardly been reviewed as extensively compared to the offline variant.

This work aims to research online speaker diarization and develop a computer system capable of performing it using speech features embeddings as models of the speakers found in an audio recording, with a clustering algorithm capable of updating the speaker representations during execution.

The experimental results showed that the developed system could perform online speaker diarization successfully using x-vectors as speech representations. Furthermore, the test results using oracle initialization of speaker models show that the proposed methodology haves competitive performance against an offline diarization baseline.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

There is a multitude of data in human speech; by means of speech, we transmit not only the ideas we want to share with the world but also the emotions we feel, the ethnic group we belong to, and our own unique footprint that differentiates us.

Speaker diarization is the process that answers the question *"who spoke when"* by partitioning the speech signal into groups of speech segments that correspond to the same speaker. Many applications benefit from the diarization process, such as Automatic Speech Recognition (ASR), speaker indexing, and document content structuring. Most speaker diarization systems consist of four independent components [1, 2, 12]: 1) Speech segmentation, where the audio signal is split into short segments; 2) Audio embedding extraction, where specific speech features such as MFCC [3] or d-vectors [12] are extracted from the splitted audio segments; 3) Clustering, where the speech features embeddings are clustered into speakers; and optionally 4) Resegmentation, where the clustering output is refined to produce better diarization results [1].

In recent years offline speaker diarization techniques have reached state-of-the-art performance, especially those based on neural network audio embeddings [4, 5, 6]. Such techniques cannot be used in real-time applications since they require complete speech data upfront. If the application is latency-sensitive, such as a multi-person voice interactive system, it must have speaker labels generated as soon as speech segments are available to the system.

In this work, we aim to develop an online speaker diarization system that uses speech features embeddings as speaker models along with a clustering algorithm capable of updating the speaker representations during execution.

1.1 Objectives

1.1.1 General Objective

The general objective of this project is to develop a computer system based on x-vectors embeddings capable of performing online speaker diarization.

1.1.2 Scientific Objectives

- Define an online speaker diarization clustering process based on x-vector embeddings.
- Find the minimum number of audio segments required to generate an accurate speaker model that can be used to find that same speaker appearances along the audio stream.
- Estimate performance of the proposed speech feature vector (x-vector) for online speaker diarization.
- Compare our system performance with state-of-the-art methods.

1.1.3 Academic Objectives

- Strengthen scientific collaboration between the Universidad Autónoma de Yucatán (UADY), Mexico and Johns Hopkins University (JHU), USA.
- Generate scientific knowledge through experimentation with new computational techniques and approaches to improve speaker diarization in an online scheme.
- Encourage research in new areas, such as speech processing, among the UADY students.
- Publish an article in a scientific journal.

1.2 Problem statement

As technology goes forward, speech becomes one of the most essential means for human-machine communication; technologies such as assistant devices, wearable devices, and in-vehicle information systems execute more sensitive tasks, therefore fostering a more precise human-machine communication bridge is necessary.

Speaker diarization can improve real-world applications such as automatic speech recognition [7], allowing machines to have a more precise understanding of human instructions, and, in

its *online* variant, it can be used in latency-sensitive applications, improving communication in real-time multi-user applications.

This work's proposed methodology consists of developing an online speaker diarization system using speech features embeddings and a clustering algorithm capable of performing its task in an online fashion. In this work, the clustering process will be treated as a dynamic speaker tracking problem with a real-time updating process for speaker models.

1.3 Thesis structure

In this chapter, this thesis's objectives have been presented, and the problem statement has been described. Chapter 2 presents the literature review that has been carried out to know the state-of-the-art of the research topic. In Chapter 3 the theoretical background is presented to put the reader in context on all the specific topics that have some relationship with this work.

Chapter 4 presents the methodology used for the development of the system, explaining the operation of each of its modules, the experimentation process is described, the data set, and the procedures used to obtain the performance of the system. In the Chapter 5 the system results are presented, and followed by their analysis in Chapter 6.

In Chapter 7 the conclusions obtained through the development of the system are presented, in addition to the following steps for this work.

Chapter 2

Literature review

Historically, the development of state-of-the-art systems for speaker diarization has been focused on its offline variant, as offline clustering algorithms typically outperform their online counterparts. This performance advantage is because offline clustering algorithms have available complete contextual information of the entire audio during inference to generate speech regions hypothesis, determine the number of speakers, and assign labels to such regions to resemble the real speaker turns. Moreover, most of the re-segmentation techniques that further refine the speech regions' hypotheses and clusters can only be applied in the offline setting.

For some years now, the technology industry has been the most interested entity pushing for online speaker diarization systems development. As the increasing popularity of consumer-demanded smart devices calls for real-time applications capable of retrieving information and analytics with low latency in domains such as broadcast news, call-centers, and meetings. Examples of such interest are Intel, IBM, and Google's works in recent years, where they implement various techniques, ranging from the usage of multiple independent, unsupervised modules; to fully-supervised systems.

This does not mean that the academic sector has wholly left behind this research field, as some online speaker diarization papers are available from universities of France, Switzerland, Japan, and Germany. Nevertheless, compared to the development of offline speaker diarization systems, it is a topic with plenty of room for development, specifically a good performing online clustering method.

Everything seems to indicate that as offline systems are perfected, its real-time variant will be more significant over time as it is the next step in difficulty for this already challenging task.

2.1 The standard diarization pipeline

A speaker diarization pipeline is usually built on several independent submodules. Out of them, four are required in some shape or form in order to perform diarization successfully. Depending on the source material, one may find them named differently, but the core functionality they provide is the same. The first of these required submodules is the *speech segmentation* module; its task is to segment the input audio into short sections; most of the time this module tries to remove non-speech regions from the posterior analysis. The second of the required submodules is the *embedding extraction* module; it is in charge of converting the audio from the segments extracted by the previous module into mathematical representations that preserve the speakers' relevant information. The third one is the *scoring* module, which computes similarity measures between the embeddings; usually, this comparison is made for all pairs of embeddings. The last of the mandatory submodules is the *clustering* module, where the similarity scores are used to group audio segments that belong to the same speaker, and the number of speakers in the recording is computed.

2.1.1 Speech segmentation methods

The task of the *speech segmentation* module is to extract speaker-homogeneous fragments of audio from the input recording; this process simplifies the next steps in the diarization pipeline as it converts the continuous stream of audio into a stepped stream with a known pre-defined time resolution. Most of the time, this module also has the task of removing non-speech regions from the input recording to reduce the complexity of posterior steps, as it can be assumed the resulting segments do not include non-speech instances, such as silence, music, or noise.

For this purpose, a Voice or Speech Activity Detector (VAD or SAD, respectively) is used; the main difference is that the first one detects if there is a voice in the analyzed region, whereas the SAD detects if there is speech. Most recent literature uses the term SAD, as VAD may refer to energy-based (power of the signal) detection of voice, but sometimes both terms are used interchangeably, so it should be taken into account that in this work, we will be using SAD for both.

Dimitriadis and Fousek (2017) showed in their work that SAD is a critical part of diarization, as downstream tasks suffer from non-speech data being fed into algorithms that assume there is a speaker in each audio segment. In their experiments, i-vectors extracted from non-speech audio degraded the system results; specifically, the segmentation had too many false positives due to the pass of non-speech to a speaker turn detector used before embedding extraction.

Given the weight that this process carries in later steps, many researchers decide to work with an *oracle* SAD, which uses the ground-truth labels to provide a selection of speech regions

with no errors. The thinking behind this is to test later modules (embedding extraction and clustering) with the best conditions to assess their effectiveness as independent modules for speaker diarization. Examples of this procedure are both Pal et al. (2019) and Qingjian et al. (2019) works, where they use an oracle VAD with uniform segmentation (with a pre-defined duration and overlap) before embedding extraction.

The oracle SAD can assess the effectiveness of subsequent diarization modules, but in real-life conditions, the speech labels are not available to be used by the system, requiring an automatic method capable of classifying audio segments into speech or non-speech. A standard method to perform this classification is used by Zhang et al. (2019) and Wang et al. (2018), where a Gaussian Mixture Model (GMM) model with only to full covariance Gaussians is used for speech and non-speech classification.

Given the availability of labeled data, many modern systems take advantage of supervised learning with Deep Neural Networks (DNN), specifically recurrent architectures that can process data sequences (such as audio). Sell et al. (2018) uses a 5-layer Time-Delay Neural Network (TDNN)-based SAD to make a frame-wise speech and non-speech decision.

We previously mentioned that speaker diarization results are helpful for downstream tasks such as automatic speech recognition, but sometimes the order can be inverted; ASR timestamps available before diarization; this is the case for Diez et al. (2018), where they use an aggressive SAD based on BUT's phoneme recognizer, so the speech classification matches the words being uttered in the recording.

2.1.2 Speaker representation module

As previously mentioned, the speaker representation module task is to map the audio segments from the speech segmentation module into a fixed-dimensional feature space while keeping as much information about the speaker available for downstream tasks. So it is evident that better representations (embeddings) provide better diarization results. Most of these representations are borrowed from the speaker verification field; since, in the same way, they will be used to compare audio segments to decide if they come from the same speaker; examples of such embeddings are Mel Frequency Cepstral Coefficients (MFCC), speaker factors, i-vectors, x-vectors, and d-vectors.

For some years now, i-vectors have been one of the most widely used speaker embeddings in the speaker diarization field as they provide a compact way to represent speech. Moreover, it keeps being used as a baseline or fusion with other modern speech representations up to this date. Now, most speaker diarization pipelines take advantage of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM)-based speaker-discriminative embeddings. As

they have proven to outperform i-vectors in setups where they can capitalize large amounts of training data, because their performance keeps improving, more data is fed to their training, whereas i-vectors stop improving with added data. Both Sell et al. (2018) and Qingjian et al. (2019) compared the performance of i- and x-vectors for speaker diarization and confirmed that x-vectors outperform i-vectors. A common strategy that can be seen in both works is the usage of a large dataset with augmentations for x-vector training. In this case, VoxCeleb1+2 with additional augmentations, such as added noise, reverberation, and music, which play a crucial role in deep speaker embedding extractor training, as the increased variability forces the DNN extractor to better generalize what the relevant information of a speaker is. Both also comment on the improvement provided by a fusion of i- and x-vectors systems to gain additional performance than the single embedding systems. The extraction of i- and x-vectors is performed in two main steps; in the first one, statistics are computed with a background GMM or a TDNN for i- and x-vectors, then, a dimensionality reduction is performed with Joint Factor Analysis (JFA) for i-vectors and with a feedforward neural network for x-vectors.

Some researchers decide to develop their speaker representations instead of using i- or x-vectors to test newer DNN architectures or increase the network's embeddings' generalization on unseen data by increasing the difficulty during the training phase. In their paper, Ghahabi and Fischer (2019) represent speech segments with vectors referred to as speaker-corrupted embeddings. Their method uses Baum-Welch statistics to create super-vectors and make dimension reduction through a speaker discriminative neural network. They apply the speaker *corruption* by adding low-quality super-vectors from other speakers with low energy after UBM normalization. To further corrupt the input data, they apply dropout on the input super-vector, finding that developing their embeddings results in better accuracy than the baseline i-vectors.

Another example of neural network embeddings outperforming i-vectors is provided by Wang et al. (2018) and Zhang et al. (2019); the first of the two works proposed using a new neural network embedding known as d-vector to perform speaker diarization. They use an LSTM-based neural network to transform the audio signal into the speaker representation, and they found that the neural embedding performed well on a multi-lingual dataset even with the extractor model being trained on an only-English dataset. The second work improved the performance of d-vectors by making modifications to the neural network's training procedure. First, they trained a new embedding extractor model with additional non-English speakers, data from far-field devices, and public datasets; Second, they made the new model capable of working with variable-size speech windows by averaging the frame-level d-vectors into a segment-level d-vector.

2.1.3 Clustering methods

The next step in the diarization pipeline is to group the extracted representations (embeddings) according to the speaker's identity that uttered them. In the case of speaker diarization, when we talk about identity, we do not refer to specific speaker identification, as: "This recording's timestamps belongs to John.", which is the task of *speaker tracking*; but to a label that defines each unique speaker in the analyzed recording, as "This recording's timestamps belong to the first speaker that appeared in the recording, and those timestamps belong to the second speaker.". This difference between speaker diarization and tracking is because speaker tracking has available information about the participants in the recording; an example of this is a broadcast show, where there are available utterances from the host from previous emissions, so we could use them to search for his timestamps more efficiently in new recordings.

In speaker diarization, a representation of the speaker's speech identity must be found during the computation of clusters, using only the information available in the recording, which could lead to the labeling of two similar-sounding speakers as one. For this reason, some clustering algorithms have stopping thresholds to select how similar two clusters can be before being merged into a single cluster.

Some of the speaker diarization systems require a parameter optimization step in which a database with similar recording conditions to the expected test data is used to find the best clustering parameters for the given conditions. An interesting phenomenon is that within the same database, the best threshold for clustering is not constant, as it depends on the specific scenario in which a recording of the dataset was produced. If an utterance were recorded in a busy scenario, such as a restaurant or a party, with a lot of background noise and babble, its clustering parameters would not be the same as the utterance recorded in a meeting room where people wait for their turn to speak, even when both utterances were recorded with the same equipment and belong to the same database. This is the case of Novoselov et al. (2019) entry for the DIHARD II Challenge [14]; their speaker diarization system used an environment classifier, as the challenge's database consisted of English utterances from 11 different conversational domains, they found that it was required to have a conversational-domain-dependent AHC threshold to get the best diarization results. Still, this strategy is not the best; as mentioned by the authors, the training and development subsets for the challenge missed two of the 11 conversational domains, making the domain classification model perform poorly on the never-seen domains. For this reason, domain classification is primarily used for research challenges to squeeze most of the performance, but for production environments, diarization requires a broadly-applicable clustering model to meet good performance in unknown environments.

As previously mentioned, the speaker clustering module is in charge of grouping the speech segments according to the speaker's identity; its performance depends on its clustering algo-

rithm's capability to find the precise number of speakers in the recording and to compute labels that mimic the ground-truth speaker turns. To this date, there are a plethora of clustering algorithms that can be used for speaker diarization, ranging from Gaussian Mixture Models (GMM) to Deep Neural Network (DNN)-based techniques. These clustering algorithms can be classified following two distinct criteria. The first one is the usage of labeled data; in this case, the algorithms can be classified into supervised and unsupervised clustering algorithms. When we talk about unsupervised clustering algorithms, we refer to the learning algorithms that do not require labeled data to perform clustering, as it groups the speech segments accordingly to their similarity or dissimilarity using a pre-defined measure.

Over the last years, the speaker diarization clustering module has kept mostly on unsupervised algorithms such as Gaussian Mixture Models (GMM), Agglomerative Hierarchical Clustering (AHC) based on similarity measures like Bayesian Information Criterion (BIC), generalized log-likelihood ratio, and Information Bottleneck (IB); mean shift, k-means, spectral clustering, integrated linear programming, variational Bayesian clustering, and links. During this time, a few supervised clustering algorithms have been proposed for speaker diarization, such as unbounded interleaved-state recurrent neural networks (UIS-RNN) and affinity propagation.

The second criteria to classify clustering algorithms is according to their run-time latency, this being the time it takes the diarization system to produce speaker labels relative to the recording's processing. According to this, the algorithms can be classified into offline clustering or online clustering. Where offline clustering algorithms produce speaker labels after the recording has been completely processed, all speech segment embeddings are available to produce clusters. In comparison, online clustering algorithms have to produce a speaker label as soon as a speech segment is available, so the task of selecting the label is performed using only information from previous speech segments. For this reason, offline clustering algorithms tend to outperform online clustering algorithms as they can use all contextual information to produce a label. Nonetheless, selecting one of these kinds of clustering algorithms not only depends on performance, as for some applications offline clustering is not an option, precisely, real-time applications, or in some cases, for long-duration recordings, offline clustering would require large amounts of computational resources (memory) to be performed.

One of the most widely used clustering algorithms for speaker diarization is Agglomerative Hierarchical Clustering (AHC), which requires a similarity measure such as Probabilistic Linear Discriminant Analysis (PLDA) or Bayesian Information Criterion (BIC) to calculate the dissimilarity between speech segments and use it in an iterative process that clusters the most similar segments until a stopping criterion is reached. In their work, Sell et al. (2018) follow an i-vector PLDA AHC with an average linking clustering pipeline using MFCC as acoustic features and found that supervised threshold selection was consistently effective for the challenge; as previ-

ously mentioned, after clustering, a post-processing process was applied with Variational Bayes refinement. Similarly, Novoselov et al. (2019) used AHC for their diarization system for the DIHARD II challenge; in their case, they experimented with two different similarity measures, PLDA and discriminatively trained Cosine Similarity Metric Learning (CSML), and also used an optimal AHC threshold selector. In both works, the fusion of two of their tested systems provided better results than any single system, a common strategy to take advantage of different system configurations.

Another widely used offline unsupervised clustering algorithm is used in both Diez et al. (2018) and Landini et al. (2019) works, their systems are based on Bayesian Hidden Markov Model (HMM) with eigenvoice priors. This method assumes that the sequence of speech embeddings is generated from an HMM, where states represent the speakers and the transitions the speaker turns. The specific speaker distribution (model) is modeled by Gaussian Mixture Models (GMM) with eigenvoice priors imposed on the GMM parameters.

For each recording, a HMM is built, assigning more states than the assumed number of speakers in the recording; the HMM is started with an initial assignment that can be random and another external clustering algorithm. The surviving HMM states provide the final diarization assignments, as zero transition probabilities are learned and remove unnecessary speakers. They use AHC as initialization with PLDA similarity scores in both works, as they found that it provides better performance than random initialization.

Up to this moment, we can notice that Probabilistic Linear Discriminant Analysis (PLDA) is one of the most used similarity measures for speaker diarization. Its task is to score the similarity between two speech segments; in the traditional offline diarization pipeline, this task is performed for all pairs of speech segments producing a similarity matrix, also known as an affinity matrix, which is the input to the clustering algorithm. Qinqjang et al. (2019) propose a supervised method to produce the similarity matrix between all recording segments with a Bi-directional Long Short-Term Memory (Bi-LSTM)-based architecture, as the PLDA ignores the temporal sequence of speech segments, ignoring the highly structured, turn-taking behavior of a conversation. Their results show that temporal sequence is essential information that most traditional unsupervised methods tend to leave aside.

As mentioned above, the temporal sequence is essential information for speech analysis, so Recurrent Neural Network (RNN) architectures, specifically Long Short-Term Memory (LSTM), are being researched for speaker diarization, as they better capture the sequential nature of audio signals. A magnificent example of this is the two papers of Google Inc; in the first one, Wang et al. (2018) use an LSTM-based text-independent speaker verification speaker model with non-parametric spectral clustering to perform speaker diarization. In their work, they propose a novel refinement algorithm for the segments affinity matrix and test it with four different clustering

algorithms, two of them being online. The first clustering algorithm is referred to as "naive" online clustering, where a similarity measure (in their case, cosine distance) is used as a metric with a threshold to select if a given segment embedding belongs to one of the existing speaker clusters. Their second online method is Links; it estimates cluster probability distributions and models their substructure based on the embeddings vectors. Their third method is K-Means++, an offline clustering algorithm, and finally, their best performing clustering method was spectral clustering. This paper focused on offline clustering as its performance was twice as good compared to the tested online clustering algorithms; they mention that most of the online error is generated at the beginning of the recording, so they talk about a "burn-in" stage before entering the online diarization mode.

In the second Google's paper, Zhang et al. (2019) build over the previous experiments by replacing the unsupervised clustering module with an online generative process, named Unbounded Interleaved-State Recurrent Neural Network (UIS-RNN), where a parameter-sharing RNN models each speaker, naturally integrated with a Distance-Dependent Chinese Restaurant Process (DDRP) able to accommodate an unbounded number of speakers. They compared their new method against two of their previous systems, K-Means, and spectral clustering, obtaining better performance against the two offline clustering methods using the same verification speaker model. We can see that online clustering algorithms can have competitive performance against their offline counterparts.

Many online speaker diarization systems share the same clustering strategy, sometimes referred to as "naive" online clustering; it consists of developing representation for each cluster that can be directly compared against a segment embedding with some of the previously mentioned similarity measures. Such representation is known as the speaker model, as its task is to represent a single speaker from the recording. Having the speaker models, each incoming segment embedding is compared against all of them, and then, using a threshold, the decision to mark the segment as one of the known speakers or generate a new speaker model is made.

Soldi et al. (2015) developed a speaker diarization system using Gaussian Mixture Models (GMM) for speaker models and a Universal Background Model (UBM) trained from external data to perform online diarization. They use log-likelihood to compare against the similitude of the different speaker models and the UBM. If the similitude does not meet a threshold, a new speaker model is computed from the UBM using Maximum A Posteriori (MAP) adaptation; if the threshold was met, the selected speaker model is updated with MAP adaptation. They found that performance deteriorates if the speaker model is too big due to insufficient data for MAP adaptation. A similar approach is used by Patino et al. (2018) with the replacement of GMM with i-vectors and log-likelihood with PLDA. Ghahabi and Fischer (2019) propose a Deep Neural Network (DNN) to produce what they define as "speaker corrupted" embeddings, then

using cosine similarity measure, they compare the current speaker's models, and as previously mentioned, if the model gives a score higher than a speaker-dependent threshold, the segment is labeled as the selected speaker, if not, a new speaker model is created but only if two halves of the segment are similar enough; the speaker models in their method are the average overtime of speaker vectors assigned to each detected speaker. From these three works, we can view that the "naive" clustering method is still a good option given the correct embeddings and similarity measure.

2.2 The speaker overlap problem

First, we have to introduce how most speaker diarization systems measure their performance to talk about the speaker's overlap problem. As previously mentioned, a diarization system hypothesis does not need to identify the intervening speakers by a specific identity or definite ID, so the system's output labels do not need to be the same as the ground-truth labels. Simultaneously, non-speech tags must be marked as non labeled gaps between two speaker segments, being implicitly identified.

The primary metric used for speaker diarization is the Diarization Error Rate (DER) described by NIST [19]. It comprises three types of errors, speaker error, false alarm speech, and missed speech. When the reference file states that multiple speakers are overlapping with each other at the same time, it is expected from the diarization system to provide multiple labels indicating the speaker overlap; failing this, the speakers in overlap will count as missed speaker errors.

A common practice in most speaker diarization research papers is that the DER computation is made with two simplifications; the first one is that a collar of $\pm$250ms around every reference speaker turn is defined to account for inexactitudes in the ground-truth labeling. The second simplification is the problematic one, where no speaker overlap is taken into account for DER scoring. This caused that most of the speaker diarization systems assume that a speech segment only contains one speaker; therefore, the definition of the diarization systems carry this assumption in the clustering stage, making the search of overlapped speech an additional module in the diarization pipeline instead of a joint solution in the clustering algorithm. The DIHARD challenge [14] disrupted this common practice by reviewing all contestant systems with no forgiveness collar of $\pm$250ms and taking into account overlapped speech; with the expectation that state-of-the-art systems fare poorly. This was important as real-life scenarios such as professional meetings have high percentages of overlapped speech (5 - 10%), while informal get-togethers can easily exceed 20%[20]. As previously mentioned, speaker diarization systems had to add a pipeline module to handle overlapped speech. This was the case of Landini et al.

(2019) where a logistic regression model was trained to classify overlapped and non-overlapped speech segments, and because of the speech representations assumed one speaker per segment, it was needed to assign each frame in an overlapped speech segment to the closest speaker according to the diarization labels given by the clustering algorithm. Diez et al. (2018) work implemented a model similar to a Speech Activity Detector (SAD), but in this case, it had three outputs "speech", "non-speech", and "overlapped speech". Those labels were used in two ways to remove overlapped speech from the clustering algorithm, as it performed worse when handling overlapped speech, and the second way was to label overlapped speech with the output from the clustering algorithm. From these examples, we can see how researching a clustering algorithm capable of handling overlapped speech by default is needed.

2.3 Online inference

As presented in Section 2.1.3, online speaker diarization has some specific issues that its offline counterpart does not experience. The first one is that inference should be made as soon as possible to produce a speaker label in real-time, only using information from the current segment embedding and contextual information from previous segments. Soldi et al. (2015) found that the number of speakers in the recording was inferred within the first two to three minutes, so if there were other speakers down the line, it would generate an error as the system would infer more speaker than there are. They also concluded that good DER requires a balance with the duration of the speech segments, as shorter duration produced worse results, but after a given segment length, longer segments damage the DER due to missed speakers (a longer segment can contain two speakers, but it is assumed to contain only one). Their most significant finding was that the speaker model update is effective up to a specific time; further, it damages the results.

Patino et al. (2018) conclude that when performed online, speaker diarization systems have the potential to introduce errors into the diarization hypothesis, which the system can never recover from, as impure clusters (containing more than one speaker) are likely to remain impure as long as the online diarization proceeds.

Nevertheless, even with these problems, online speaker diarization seems to be the next step for researchers in the topic. It provides challenging conditions to test state-of-the-art techniques and also allowing the analysis of arbitrarily long recordings with low computational resources.

Chapter 3

Theoretical background

3.1 Speech technologies

Speech technologies is a subfield of signal processing that investigates how to produce, perceive and understand the human spoken language. It is an interdisciplinary research area, joining efforts from psychology, linguistics, engineering, and artificial intelligence [21].

Knowledge on the nature of spoken language is essential for efficient coding and transmission of speech, as well as satisfactory human-machine speech interaction. In recent years many methods have been designed with the objective of solving problems such as: dialogue systems based on speech recognition and synthesis, detecting emotions in speech, identifying speakers, as well as speech coding and transmission, denoising, detection of speech in the presence of noise, quality enhancement, and medical diagnostics based on the analysis of human voice.

Speech technologies has many subfields, such as, speech encoding, speech synthesis, speech recognition, natural language processing, language classification, speaker verification, et cetera [22, 23]. In this chapter the most relevant for our work are presented.

3.2 Speech recognition

Speech recognition, also known as *automatic speech recognition* is the process of extracting from the human speech signal the corresponding sequence of words, or other linguistic entities, by mean of computer algorithms [24]. There has been some confusion in the public with *speech* and *speaker* recognition. In a speech recognition application, it is not the voice of the speaker which is being recognized, but the contents of his/her speech [22].

Speech recognition is an important medium for better human-human and human-machine communication. With technology such as, mobile devices, wearable devices, assistant devices,

and in-vehicle infotainment systems, interaction mediums such as keyboard and mouse became less convenient, with speech becoming a more suitable medium. As it is the natural way of human-human communication and a skill that most of people already have [25].

3.3 Speaker recognition

Speaker recognition, sometimes referred as *speaker biometrics*, is a generic term used for any process which extracts, characterizes and recognizes information about the identity of a person based on his/her voice [22] answering the question *"who is speaking"*.

The underlying methodology of a speaker recognition application is to model the voice characteristics of a person by a mathematical model of the physiological system that produces the human speech [26, 27, 28] or by a statistical model with an output similar to the one of the human vocal tract [29, 30]. Once the person model is created, new speech instances may be assessed to determine the likelihood of them being generated by the individual model in contrast with other known models.

Speaker recognition branches into multiple areas of study which are either directly or indirectly related to each other [22, 31], the most relevant for this work are presented below.

3.3.1 Speaker verification

Speaker verification, also known as *speaker authentication* aims to verify whether the test speaker (the person being verified) input speech corresponds to it's claimed identity [32, 31]. In a typical speaker verification application the person provides his/her identity by a non-speech method (e.g., an identification number, a user-name, et cetera) [22], then the provided ID is used to retrieve the *target speaker model* for that identity from a database. Then the input speech of the test speaker is compared against the target speaker model to verify the identity of the test speaker.

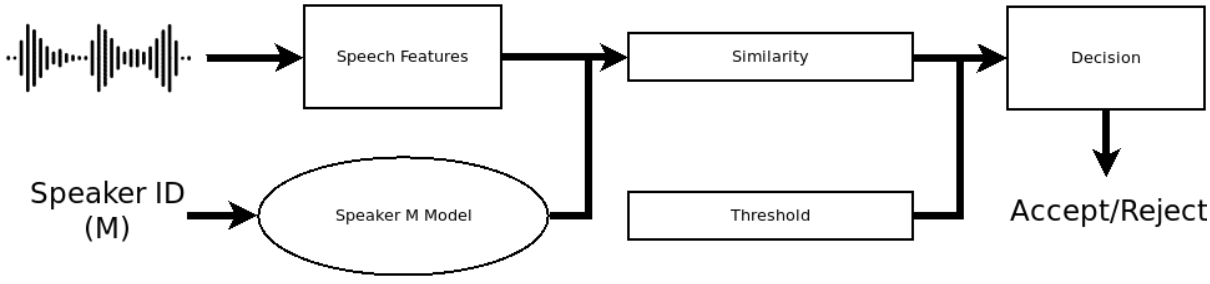


Figure 3.1: Speaker verification overview.

To increase reliability, speaker verification systems require to add a medium for contrast

when making a comparison, as comparison against the target speaker's model is not enough [22, 33]. In order to get a quantitative assessment of the likeness, we not only need to know how similar is the test speaker to the target speaker model, but how similar is the test speaker to other speakers.

Two of the major approaches deal with the closeness of the target speaker, with the addition of competing models [34, 35]. The first method uses a model based on data from a large population, with the idea that if the test speaker is closer to the average population than the target speaker, it is not the target speaker, this method is known as *Universal Background Model*.

The second method uses a model selected from speakers who sound similar to the target speaker, this method is known as *cohort model* [36]. The idea behind this method is that if the test speaker is closer to the target speaker model, rather than the cohort model, it is the target speaker.

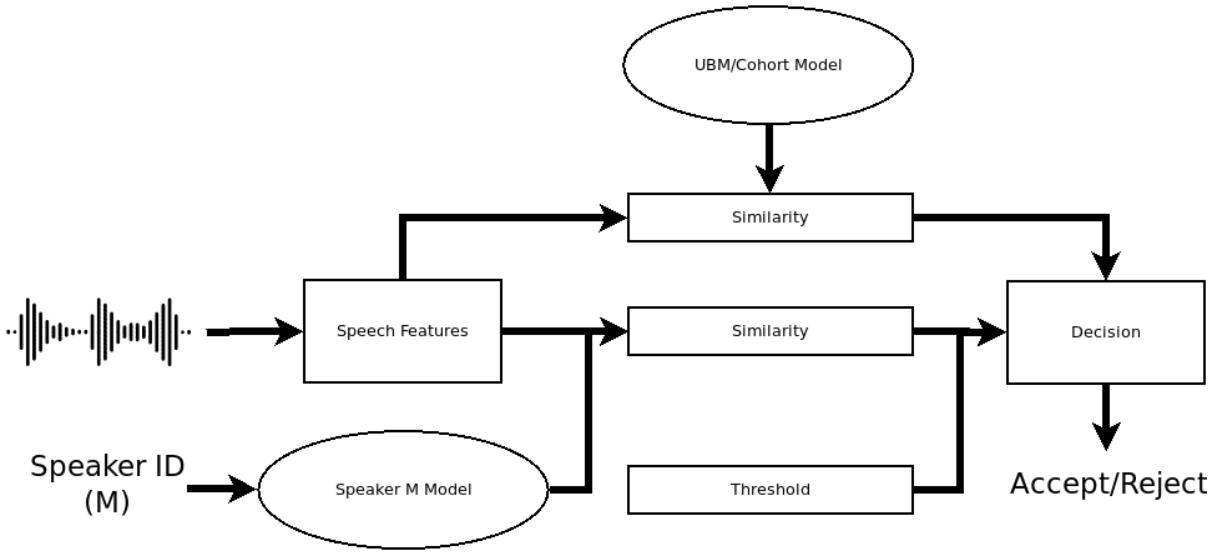


Figure 3.2: Speaker verification with UBM/Cohort overview.

3.3.2 Speaker identification

Speaker identification is the task of determining from which of the known speakers a given utterance comes. It solves the question *"whose voice is this"*. There are two different types of speaker identification, *closed-set* and *open-set* [22]. In closed-set identification, the utterance of the test speaker is compared against all known speaker models, then the speaker ID is selected from the model with the closest match.

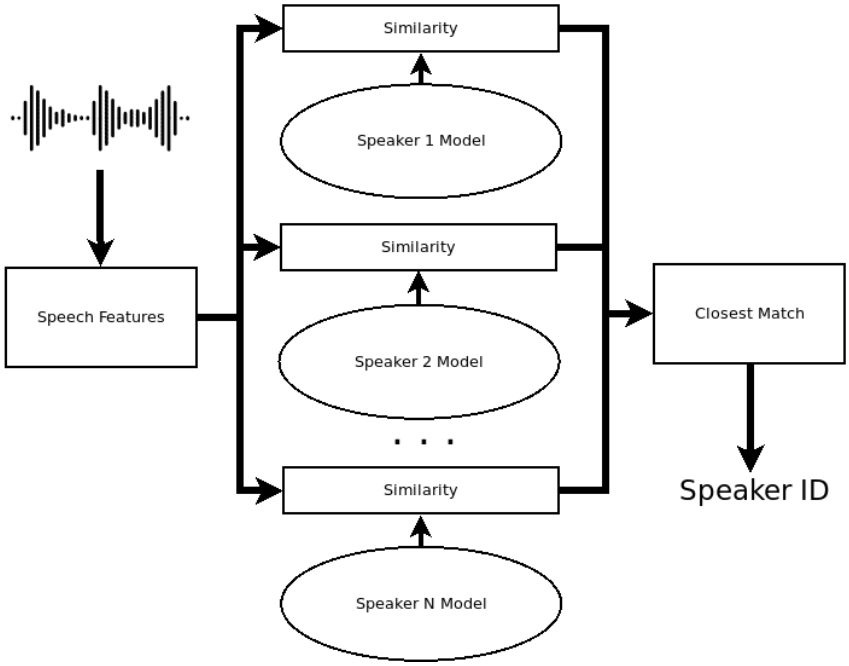


Figure 3.3: Closed-set speaker identification overview.

Open-set speaker identification can be seen as the addition of a verification step to the output of a closed-set identification. For example, in figure 3.4 a closed-set identification returns *Speaker M* as the closest speaker ID, so the *Speaker M* model is used to run a speaker verification session. If the test utterance matches the target model, the ID is accepted as the open-set identification result. On the other hand if the verification fails the utterance is rejected with no identification result.

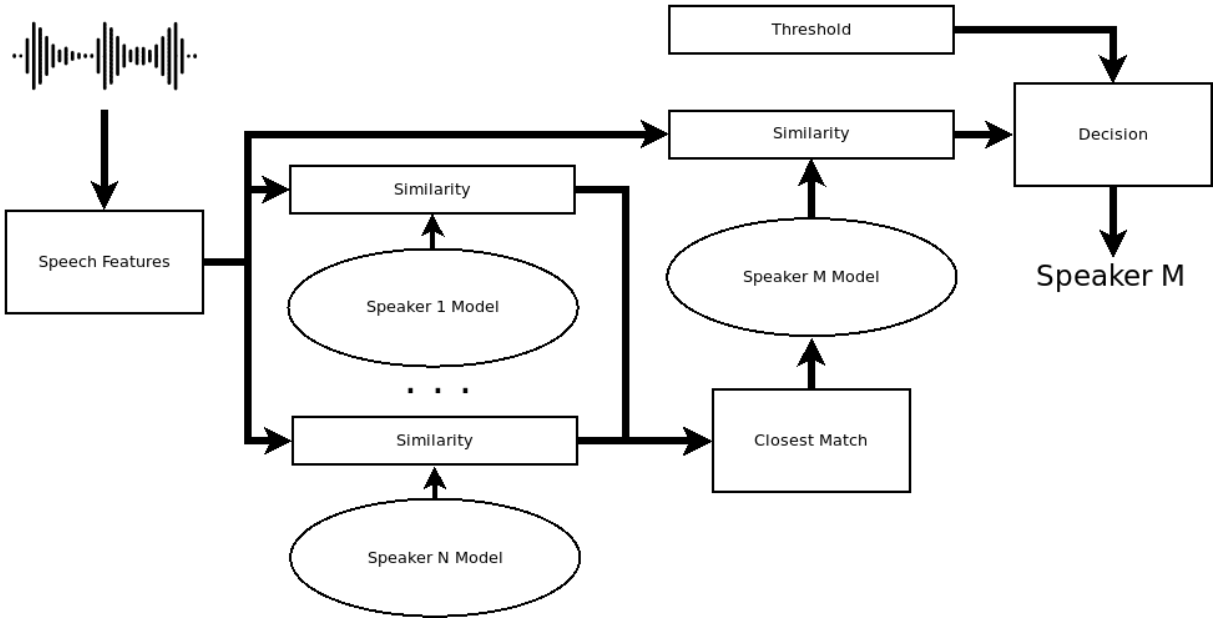


Figure 3.4: Open-set speaker identification overview.

3.3.3 Speaker diarization

Speaker diarization is the process that solves the question *"who spoke when"* by organizing and tagging an input audio stream with multiple speakers into groups of homogeneous segments according to each speaker identity [37].

There are many applications that benefit from the diarization process, such as Automatic Speech Recognition (ASR) [7] and speaker indexing [38], making the transcriptions of such systems searchable by the ID of each speaker who was recognized.

For diarization, many speaker recognition branches are employed [22]. The typical speaker diarization system consist of four independent components [1, 39, 12]: 1) Speech segmentation, where the audio signal is splitted into short segments; 2) Audio embedding extraction, where specific speech features such as Mel Frequency Cepstral Coefficients (MFCC) [3], i-vectors [5], x-vectors [39, 5] or d-vectors [12] are extracted from the splitted audio segments; 3) Clustering, where the speech features embeddings are clustered into speakers; and optionally 4) Resegmentation, where the clustering output is refined to produce better diarization results [1].

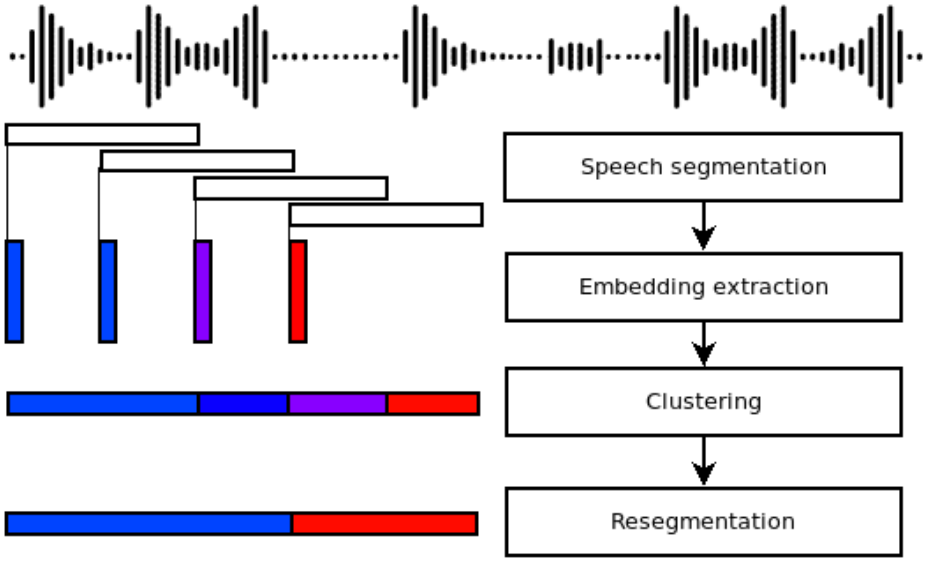


Figure 3.5: Speaker diarization overview.

Depending on the type of audio input, speaker diarization can be classified as *offline speaker diarization* or *online speaker diarization*. In offline speaker diarization all the speech data is available before the system makes any decision on how many speakers are present in the recording and when each of them is speaking [17]. Latency-sensitive applications such as a multi-person voice interactive system require to have speaker labels generated as soon as speech segments are available to the system without any knowledge of future segments, therefore a real-time option is needed.

Online speaker diarization generates labels as soon a speech segment is available to the system, it is more difficult than the offline variant because the decisions are based on much less data, so typically the performance is far from what can be achieved with offline approaches.

3.4 Speaker diarization pipeline

3.5 Feature extraction

In this section we present the topics needed to understand how the feature extraction for speaker recognition tasks is performed.

3.5.1 Discrete Fourier transform

The Discrete Fourier transform (DFT) allows us to transform a finite sequence of audio samples into a finite sequence of frequency components [22]. It is a discretization of the complex Fourier

transform given by:

$$H(\omega) = \int_{-\infty}^{\infty} h(t)\, e^{-i(\omega t)} dt \tag{3.1}$$

and,

$$h(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} H(\omega)\, e^{i(\omega t)} d\omega \tag{3.2}$$

where $h(t)$ is a periodic time-domain function that meets the Dirichlet conditions necessary for the convergence of the Fourier Series, such conditions state that $h(t)$ should be periodic with period $2T$, it must be absolutely integrable in the period $[-T,T]$, and it must have a finite number of finite discontinuities in the interval $[-T,T]$; and $H(\omega)$ is a frequency-domain representation of $h(t)$.

To apply discretization to 3.1 first we need to assume that we have a finite set of N samples, from time $t = 0$ to $t = N-1$, that we are going to map to a finite set of frequencies. We can write the discrete time instances as:

$$t_n = nT \quad n = \{0, 1, \ldots, N-1\}$$

Assuming our signal has been sampled based on the *Nyquist* sampling theorem, there are only spectral components present with frequencies less than f_c, therefore our frequency range will be from $-f_c$ to f_c instead of $-\infty$ to ∞. This means, for a frequency resolution of N, the discrete frequency-domain step is $\frac{2f_c}{N}$ frequency levels. Therefore the discrete frequency is:

$$f_k = \frac{k}{NT} \; k = \{0, 1, \cdots, N-1\}$$

$$\omega_k = \frac{2\pi k}{NT} \; k = \{0, 1, \cdots, N-1\}$$

To perform the discretization of 3.1 we will use the following definitions:

$$h_n = h(t_n) = h(nT)$$

$$\hat{H}_k = H(\omega = \omega_k) = H\left(\omega = \frac{2\pi k}{NT}\right)$$

In the discrete form, the integral of 3.1 changes to a sum of the N values with $dt \to T$:

$$\hat{H}_k = \sum_{n=0}^{N-1} h_n e^{-i\frac{2\pi kn}{N}} T$$

We can get rid of sampling frequency dependence, by defining the discrete Fourier transform

H_k such that $\hat{H}_k = H_k T$, therefore we get the definition of the DTF as:

$$H_k = \sum_{n=0}^{N-1} h_n e^{-i\frac{2\pi kn}{N}}$$

3.5.2 Discrete cosine transform

The DFT can be splitted into its real (discrete cosine transform) and imaginary parts (discrete sine transform), in the speech processing task, the Discrete Cosine Transform (DCT) is commonly used, because of its relation to the real axis. It is defined as follows:

$$H_k = \sum_{n=0}^{N-1} h_n \cos\left(\frac{\pi\,(2n+1)\,k}{2N}\right)$$

where $k = \{0, 1, \cdots, N-1\}$ is the frequency index.

3.5.3 Mel scale filter banks

The human auditory system has a series of peculiarities that may be unwanted for some applications. One of them is the way we perceive pitch, as it is not perceived in a linear manner, i.e., two sine waves of 100 and 200 Hz might sound *farther* apart than two sine waves of 10,100 and 10,200 Hz, even though in both cases the difference is 100 Hz. For that reason the *Mel scale* was developed, as a way to compensate for this unwanted feature.

The *Mel* is a unit of pitch, by definition it is equal to one thousand of the pitch of a simple tone with frequency of 1000 Hz with an amplitude of 40 dB above the auditory threshold [22].

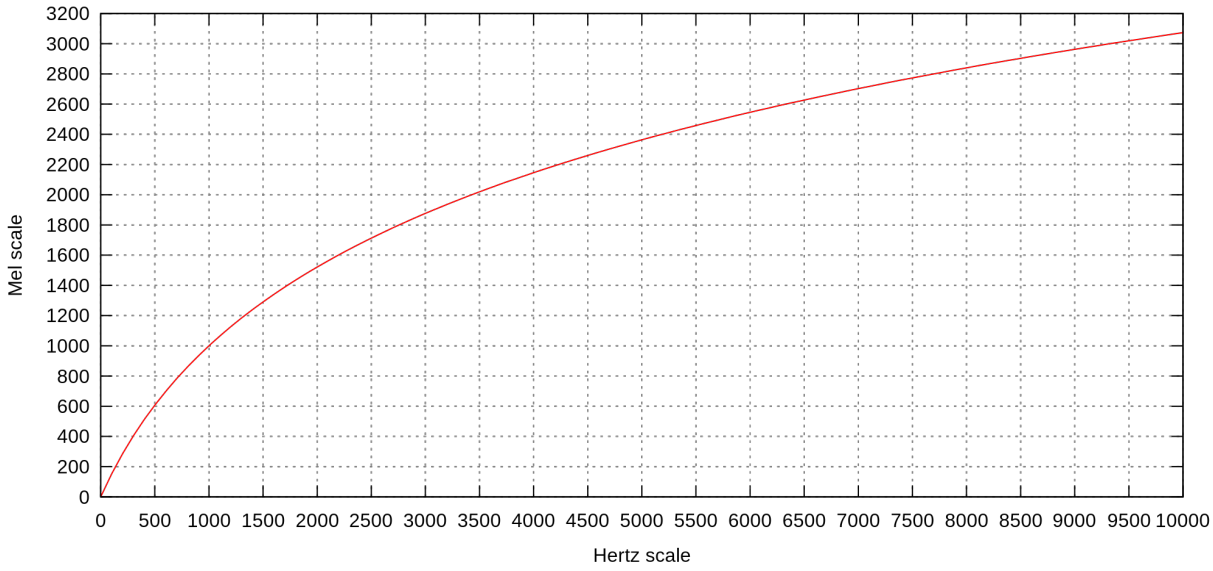


Figure 3.6: Relationship between the frequency scale and the Mel scale.

There are many Mel scale formulas [40], as they are made by fitting an equation to the data points reported by the experiments done by Stevens, Volkman and Newman in late 1930's. The above definition equation is:

$$mel = \frac{1000}{ln\left(1+\frac{1000}{700}\right)} ln\left(1+\frac{Hz}{700}\right)$$

Another popular formula is:

$$mel = 2595 \log_{10}\left(1+\frac{hz}{700}\right)$$

The Mel scale is widely used in speaker and speech recognition applications [22, 23] as it describes more precisely the audio signal, seen from the perception of the human auditory system.

A *filter bank* is defined as an array of band-pass filters, used to separate an input signal into its multiple components, each one carrying a specific frequency sub-band of the original signal, there are many applications that use this process as a way to analyze an input signal. For our purposes, a filter bank provides a way to adapt the speech signal to pertain to the critical bands in which the human perception works [22]. So far the DFT provides us the frequency-domain spectra of the speech signal in a linear scale, so, it has to be mapped to a smaller set of values that correspond to the critical bands in Mel scale.

If we consider a 8000 Hz signal sampling frequency, as it is one of the most frequently used in the telephony industry, the *Mel scale filter bank* is built by dividing a signal sampling

frequency of 2840 Mels (roughly 8000 Hz), into a set of band-pass filters. To do that, we use the difference between the first and second critical bands to model the rest of them, such difference is known as *Bark*, it was defined by Zwicker as the width of one critical band over the whole frequency range, corresponding to nearly 100 Mel.

For a sampling rate of 8000 Hz we get 24 filters in the *filter bank* with their center frequencies 110 Mels apart. In Fig. 3.7 a triangular shaped Mel scale filter bank is shown, one of the most popular and simplest approaches is to use triangular shaped filters[22].

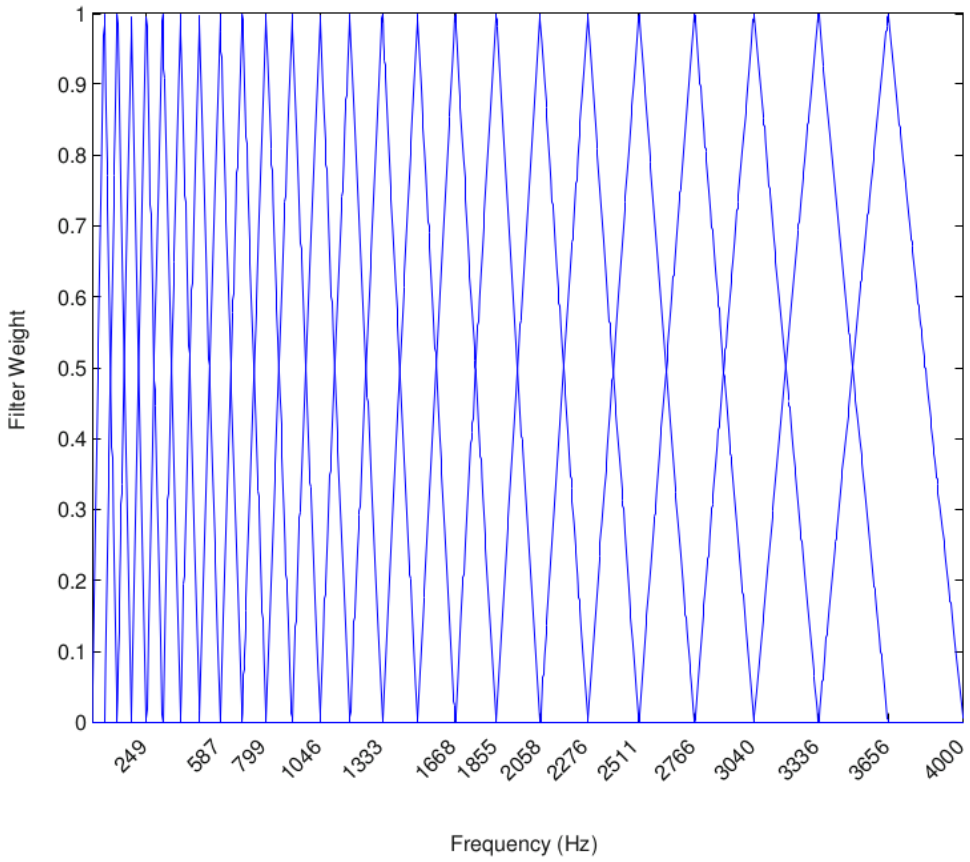


Figure 3.7: Shape of the Mel scale filter bank.

3.5.4 Mel Frequency Cepstral Coefficients

The Mel Frequency Cepstral Coefficients (MFCC) are audio features commonly used in speech and speaker recognition tasks [25, 22]. MFCC represent the power spectrum of a sound in a human-like perceptual way, this coefficients are always real and convey significant information about the structure of the speech signal [22]. In Fig. 3.8 the MFCC extraction process from the raw audio signal is shown.

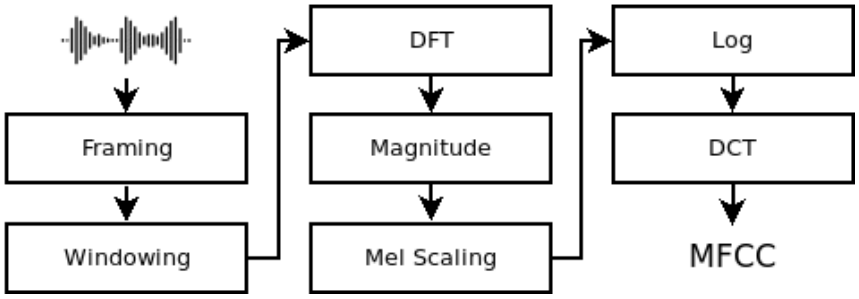


Figure 3.8: MFCC extraction overview.

The first step in the MFCC extraction process is to divide the input speech signal into *frames*

3.6 Voice activity detection

A Voice Activity Detector (VAD) is a speech processing system that identifies human speech segments from background noise in audio streams. [41]. Voice activity detectors are widely used to improve performance of speech recognition systems [42, 43, 44]. Roughly the typical VAD system has three steps; noise reduction; feature extraction; and a classification rule to select if the features are from speech or non-speech [45].

3.7 Speech embeddings

An embedding is a representation of a topological object, field, graph, etc. in a certain space, in such a way that its connectivity or algebraic properties are preserved. We can say that a space X is embedded into a Y space if the the properties of Y restricted to X are the same as the properties of X [46, 47].

Let $\mathbb{A} = (A, (c^{\mathbb{A}}_{c\in C}, (P^{\mathbb{A}})_{p\in P}, (f^{\mathbb{A}})_{f\in F}))$ and $\mathbb{B} = (B, (c^{\mathbb{B}}_{c\in C}, (P^{\mathbb{B}})_{p\in P}, (f^{\mathbb{B}})_{f\in F}))$ be structures for the same first-order language L, and let $h : A \to B$ be a homomorphism from $\mathbb{A}$ to $\mathbb{B}$. Then h is an embedding provided that it is injective [47].

Within the neural network context, embeddings are a useful way to reduce dimensionality of categorical variables, keeping a meaningful, low-dimensional representation of discrete variables in the transformed space.

In the speech recognition field, there are many speaker embeddings that have proven to have a good performance, such as i-vector, x-vector, s-vector, d-vector, etc [48]. For our purposes an embedding can be seen as the mapping of a discrete variable (speaker characteristics) to a vector of continuous numbers.

3.7.1 I-vectors

Before presenting the definition, it is necessary to touch on some topics. The i-vector [49] is an embedding that provides a way to reduce large-dimensional input speech data to a small-dimensional feature vector that retains most of the relevant speaker information. In the last years it has proven to provide state-of-the-art performance for speech recognition tasks, such as, speaker verification [50, 51, 49] and speaker diarization [52, 53]. This approach is based on the idea of a new low-dimensional speaker- and channel-dependent space representation. Before presenting the i-vector definition, it is necessary to touch on some required topics.

Gaussian mixture models

The Gaussian Mixture Model (GMM) is a probability distribution of a random variable expressed in terms of a weighted sum of its components, each one described by a Gaussian density function:

$$p(x|\phi) = \sum_{\gamma=1}^{\Gamma} p(x|\theta_\gamma)P(\theta_\gamma) \tag{3.3}$$

Where ϕ is the super-vector of parameters, defined as an augmented set of Γ vectors constituting the free parameters associated with the Γ mixture components, θ_y, $\gamma \in \{1,2,\cdots,\Gamma\}$ and the $\Gamma-1$, and the mixture weights, $P(\theta=\theta_\gamma)$, $\gamma=\{1,2,\cdots\Gamma-1\}$, being the prior probabilities of each of the mixture models known as the *mixing distribution* [22].

Each mixture component parameter vector is the parameters of the normal density function:

$$\theta_\gamma = [\mu_\gamma^T \; u^T(\Sigma_\gamma)]^T$$

Where the *unique parameters* vector (u) is an invertible transformation that stacks all the free parameters of a matrix into a vector form. For example if Σ_γ is a diagonal matrix, then:

$$\big(u(\Sigma_\gamma)\big)_{[d]} \triangleq (\Sigma_\gamma)_{[d][d]} \forall d \in \{1,2,\cdots,D\}$$

We may always reconstruct Σ_γ from u_γ with the inverse transformation:

$$\Sigma_\gamma = u_\gamma^{-1}$$

With the parameter vector for the mixture model being constructed as follows:

$$\phi \triangleq [\mu_1^T \cdots \mu_\Gamma^T \; u_1^T \cdots u_\Gamma^T \; p(\theta_1)\cdots p(\theta_{\gamma-1})]^T$$

Where only $(\Gamma - 1)$ prior probabilities, $p(\theta_\gamma)$ are included in ϕ due to:

$$\sum_{\gamma=1}^{\Gamma} p(\phi_\gamma) = 1$$

Assuming a sequence of *independent* and *identically distributed* (i.i.d.) observations, $\{x\}_i^N$, the log of likelihood of the sequence is written as:

$$l(\phi|\{x\}_1^N) = ln\left(\prod_{n=1}^{N} p(x_n|\phi)\right) = \sum_{n=1}^{N} ln\ p(x_n|\phi)$$

With 3.3, the log-likelihood may be written in terms of the mixture components:

$$l(\phi|\{x\}_1^N) = \sum_{n=1}^{N} ln\left(\sum_{\gamma=1}^{\Gamma} p(x_n|\theta_\gamma)P(\theta_\gamma)\right) \tag{3.4}$$

The GMM parameters are estimated by maximizing the log-likelihood (3.4) given a set of training examples $\{x\}_i^N$:

$$\phi^* = \arg\max_{\phi} \sum_{n=1}^{N} ln\left(\sum_{\gamma=1}^{\Gamma} p(x_n|\theta_\gamma)P(\theta_\gamma)\right)$$

This task may be performed with the use of *expectation-maximization* algorithm. The *expectation* step is formulated by the *a-posteriori* probability of each mixture component, given the observed data $\{x\}_i^N$:

$$p(\theta_\gamma|x_n) = \frac{p(\theta_\gamma, x_n)}{p(x_n)} = \frac{p(x_n|\theta_\gamma)p(\theta_\gamma)}{\sum_{y'=1}^{\Gamma} p(x_n|\theta_{\gamma'})p(\theta_{\gamma'})}$$

In the *maximization* step, we would like to maximize the expected log-likelihood for all the observations $\{x\}_i^N$ and the parameter vector ϕ. The maximization function is defined as the expectation:

$$\begin{aligned} Q(\phi) &= \mathscr{E}\left\{ ln\left(\prod_{n=1}^{N} p\left(x_n, \phi\right)\right) |x \right\} \\ &= \mathscr{E}\left\{ \sum_{n=1}^{N} ln\left(p\left(x_n, \phi\right)\right) |x \right\} \\ &= \sum_{n=1}^{N} \mathscr{E}\left\{ ln\left(p\left(x_n, \phi\right)\right) |x \right\} \end{aligned}$$

$$= \sum_{n=1}^{N} \sum_{\gamma=1}^{\Gamma} p\left(\theta_{\gamma}^{(k)} | x_n\right) ln\left(p\left(\theta_{\gamma}, x_n\right)\right)$$

$$= \sum_{n=1}^{N} \sum_{\gamma=1}^{\Gamma} p\left(\theta_{\gamma}^{(k)} | x_n\right) ln\left(p\left(x_n | \theta_{\gamma}\right) p\left(\theta_{\gamma}\right)\right)$$

We redefine ϕ to include all Γ prior probabilities, in order to solve the constrained optimization problem with the constraint on the summation of the prior probabilities:

$$\widetilde{\phi} \triangleq \left[\mu_1^T \cdots \mu_{\Gamma}^T \; u_1^T \cdots u_{\Gamma}^T \; p(\theta_1) \cdots p(\theta_{\gamma})\right]$$

So we have the following constraint:

$$\sum_{\gamma=1}^{\Gamma} p\left(\gamma_k\right) = 1$$

In the following iterative maximization problem:

$$\widetilde{\phi}^{(k+1)} = \arg\max_{\widetilde{\phi}} Q\left(\widetilde{\phi} | \widetilde{\phi}^{(k)}\right)$$

When solving, we assume a multi-dimensional normal density for $p\left(x|\phi\right)$, and we use the method of *Lagrange multipliers*:

$$\mathscr{L} = \sum_{n=1}^{N} \sum_{\gamma=1}^{\Gamma} p\left(\theta_{\gamma}^{(k)} | x_k\right)$$

$$\left(-\frac{D}{2} ln\left(2\pi\right) - \frac{1}{2} ln\left|\Sigma_{\gamma}\right| - \frac{1}{2}\left(x_n - \mu_{\gamma}\right)^T \Sigma_{\gamma}^{-1}\left(x_n - \mu_{\gamma}\right) + ln\left(p\left(\phi_{\gamma}\right)\right)\right)$$

$$-\lambda\left(\sum_{\gamma} p\left(\theta_{\gamma}\right) - 1\right)$$

To solve $\widetilde{\phi}^{(k+1)}$, we set the gradient of the *Lagrangian* to zero and solve for $\widetilde{\phi}$, $\nabla_{\widetilde{\phi}} \mathscr{L}\left(\widetilde{\phi}, \lambda\right) = 0$, we may break up the problem since $\widetilde{\phi}$ is made up from different partitions:

$$\nabla_{\mu_{\gamma} \mathscr{L}\left(\widetilde{\phi}, \lambda\right) = 0}$$

$$\frac{\partial \mathscr{L}\left(\widetilde{\phi}, \lambda\right)}{\partial \Sigma_{\gamma}} = 0$$

$$\frac{\partial \mathscr{L}(\widetilde{\phi}, \lambda)}{p\left(\theta_\gamma\right)} = 0$$

By solving such equations for $\mu_\gamma^{(k+1)}$, $\Sigma_\gamma^{(k+1)}$, and $p(\theta_\gamma^{(k+1)})$ respectively, we get the steps needed to form an iterative *expectation maximization* process to compute the elements of the parameter vector, ϕ:

Expectation step:

$$p(\theta_\gamma|x_n) = \frac{p(x_n|\theta_\gamma)p(\theta_\gamma)}{\sum_{y'=1}^{\Gamma} p(x_n|\theta_{\gamma'})p(\theta_{\gamma'})}$$

Maximization step:

$$\mu_\gamma^{(k+1)} = \frac{\sum_{n=1}^{N} p\left(\theta_\gamma^{(k)}|x_n\right) x_n}{\sum_{n=1}^{N} p\left(\theta_\gamma^{(k)}\right) x_n}$$

$$\Sigma_\gamma^{k+1} = \frac{\sum_{n=1}^{N} p\left(\theta_\gamma^{(k)}|x_n\right)\left(x_n - \mu_\gamma^{(k+1)}\right)\left(x_n - \mu_\gamma^{(k+1)}\right)^T}{\sum_{n=1}^{N} p\left(\theta_\gamma^{(k)}|x_n\right)}$$

$$p\left(\theta_\gamma^{(k+1)}\right) = \frac{1}{N}\sum_{n=1}^{N} p\left(\theta_\gamma^{(k)}|x_n\right)$$

In speech recognition tasks GMM are widely used as a way to model the data and for statistical classification, due to their well known ability to represent complex distributions[25].

Universal Background model

The Universal Background Model (UBM) is a speaker-independent speech model. It offers a robust and rich model of speech, statically generated with utterances of thousands of individuals pooled to create a model that averages features of everyone[22, 25]. It is used as a reference for adapting an individual speaker model, and, in a speaker verification scenario, as an *anti-model* that helps us to decide whether to accept or not a test speaker as shown in Fig. 3.2 [54].

A target model is usually generated by adapting from the UBM, using *Maximum A-Posteriori* (MAP) adaptation. This strategy has multiple benefits, such as, keeping the correspondence between the target and the UBM mixtures, and allowing the generation of a reliable target model with limited target data. In Fig 3.9 an overview of the target model GMM adaptation from the UBM is shown.

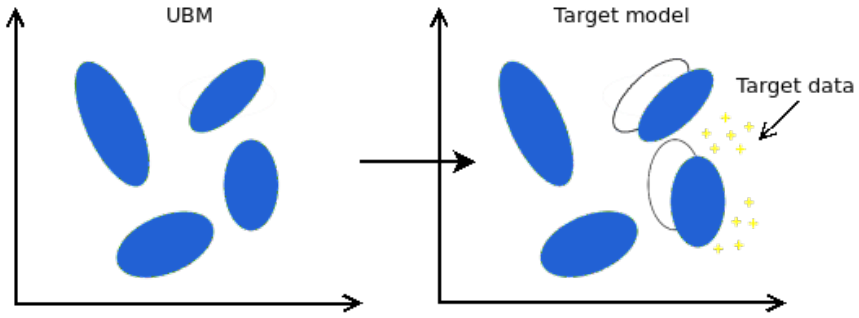


Figure 3.9: GMM-UBM target model MAP adaptation overview.

Total variability

In Joint Factor Analysis (JFA) a speaker utterance is represented by a GMM super-vector M that consists of the speaker's auditive characteristics and a channel/session subspace [22, 49], with M being defined as:

$$M = m + V_y + U_x + D_z \tag{3.5}$$

Where: m is a speaker- and session-independent GMM super-vector (generally UBM), V_y is a speaker-dependent component, U_x is a channel-dependent component, and D_z is a speaker-dependent residual component.

The *total variability* space [49] contains both, the speaker, and the channel variabilities simultaneously, so there is no distinction between the speaker and channel effects in the GMM super-vector space. In the total variability space the GMM super-vector defined in 3.5 is rewritten as follows:

$$M = m + Tw \tag{3.6}$$

Where m is the speaker- and session-independent GMM super-vector (UBM), T is a rectangular matrix of low rank total variability matrix, and w is a random vector that has a standard normal distribution $N(0, I)$, with its components being the total factors.

The vector w is known as *i-vector*.

3.7.2 X-vectors

The recent progress in speaker recognition has been given by the adoption of Deep Neural Networks (DNN) techniques [55, 56, 57, 58]. The x-vector is an embedding computed from a variable-length speech segment, such that, it contains the speaker characteristics. In the speaker recognition and verification fields, x-vectors are the state of the art method [56, 57, 58]. This approach provides a small-dimensional embedding that can be used with superior performance than i-vectors [55]. These representations are extracted from a DNN, where its input is the

MFCC that represents the raw audio signal.

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	$512xN$

Table 3.1: X-vector DNN architecture [55].

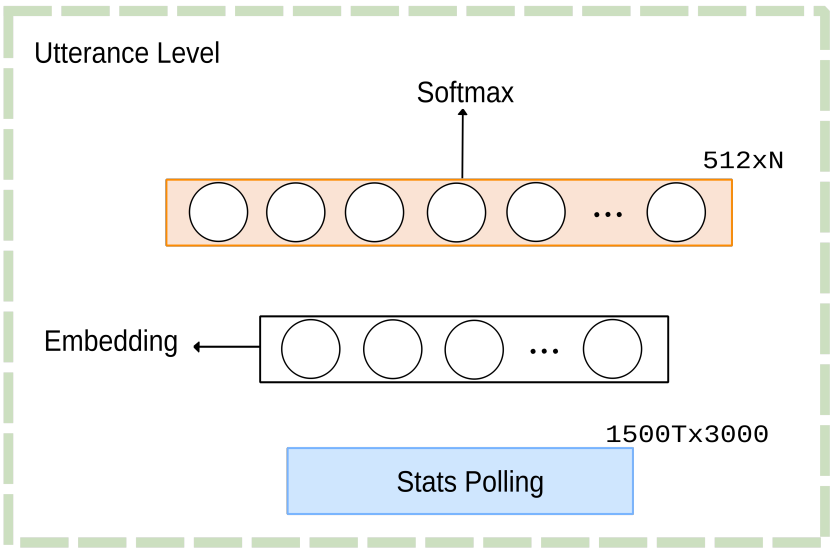


Figure 3.10: X-vector extraction layer.

In Table 3.1, N represents the number of training speakers. As shown in Fig. 3.10 the extraction of the x-vector embedding is performed at the layer *segment6* before the softmax layer [55].

3.8 Backend

3.8.1 Cosine distance

Cosine distance is a similarity measure between two vectors, it measures the cosine of the angle between the vectors, so it is a measurement of the orientation and not the magnitude. It can

be applied for vectors with any number of dimensions, with the advantage of being a low-complexity operation. Cosine distance is defined as:

$$cosine\ distance = 1 - cosine\ similarity$$

Where:

$$cosine\ similarity = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

In previous years, cosine similarity has been successfully applied in the i-vector space, comparing utterances for making speaker detection decisions [59, 60].

3.8.2 Probabilistic linear discriminant analysis

Linear Discriminant Analysis (LDA) is a statistical pattern classification technique commonly used to model both intra-class and inter-class variance, identifying linear features that maximize the inter-class separation of the input data, while, at the same time minimizing the intra-class scatter [22, 61]. This technique makes the following assumptions:

- Vectors considered to be within the same class, are identically distributed about their own class mean according to a covariance matrix, Σ_W, known as the *within class covariance matrix*.

  $$\Sigma_W = \frac{\sum_k \sum_{i \in C_k} (x^i - m_k)(x^i - m_k)^T}{N}$$

  Where x^i is a vector of length d from the training data set $\{x^1 \ldots x^N\}$. Each x^i belongs to one of the K classes. Let C_k be the set of all examples of class k, with $m_k = \frac{1}{n_k} \sum_{i \in C_k} x^i$ being the mean of the kth class, where $n_k = |C_k|$ is the number of elements in class $k = 1 \ldots K$.

- Mean values of all classes are themselves distributed according to a probability density function of the same family as the within class distributions, with a central mean and their own covariance matrix, known as the *between class covariance matrix*, Σ_B.

  $$\Sigma_B = \frac{\sum_k n_k (m_k - m)(m_k - m)^T}{N}$$

  Where $m = \frac{1}{N} \sum_i x^i$ is the mean of the data set.

We seek the linear transformation $x \rightarrow W^T x$ that maximizes the inter-class variance relative to the intra-class variance, where W is a dxd' matrix, with d' being the desired number of dimensions.

The LDA projection can be derived by fitting a GMM to the training data, such model can be used to classify examples of the classes in the training data, but no new classes. For that purpose Probabilistic LDA is required[61].

Probabilistic Linear Discriminant Analysis (PLDA) is an extension of LDA, it has delivered state-of-the-art performance in speaker recognition tasks, as it provides a powerful distortion-resistant mechanism to distinguish between-speakers variability [62, 63]. By introducing a GMM prior on the mean vector of classes, such GMM can be seen as a latent variable model.

A standard Gaussian PLDA assumes that, the jth utterance vector x_{ij} of the ith speaker can be formulated as follows:

$$x_{ij} = u + Vy_i + z_{ij}$$

Where u is the speaker-independent global factor, y_i represent the speaker-level factor, and $z_i j$ the utterance-level factor. Note that both y_i and z_{ij} are assumed to follow a full-rank GMM prior. The model is trained via expectation-maximization, and the similarity of two vectors can be computed as the ratio of the likelihood of two hypothesises: whether or not the two vectors belong to the same speaker [63].

3.9 Clustering

3.10 Metrics

The Diarization Error Rate (DER) is the main metric used for speaker diarization, it measures the fraction of time that is not attributed correctly to a speaker or to non-speech. It was described and used by NIST [19].

As per the definition of the speaker diarization task, the system output does not need to identify the speakers by the reference ID, therefore the ID tags assigned to the speakers in both output and reference do not need to be the same, as the evaluation script does an optimum one-to-one mapping of all speaker ID labels between the output and reference. The DER score is defined as:

$$DER = E_{spkr} + E_{miss} + E_{fa} + E_{ovl}$$

Where the *speaker error* (E_{spkr}) is the percentage of scored time that a speaker ID was assigned to the wrong speaker. *Missed speech* (E_{miss}) is the percentage of scored time of a reference speaker hypothesized as non-speech. *False alarm speech* (E_{fa}) is the percentage of scored time selected as a speaker where the reference labels non-speech. And finally *overlap speaker* (E_{ovl}) is the percentage of scored time of multiple speakers in a segment that do not get assigned any speaker ID [19].

3.11 Kaldi ASR Toolkit

Kaldi is an open source toolkit written in C++ for voice-related applications, such as speech and speaker recognition. It provides flexible and extensible tools targeted for researchers, available under the Apache License v2.0.

The toolkit was initially developed as part of the 2009 *"Low Development Cost, High Quality Speech Recognition for New Languages and Domains"* workshop in Johns Hopkins University. The development continued since 2010 with the aim to create a recipe of the previous year work that was clean and releasable, and as a byproduct of such work, create a general-purpose speech toolkit. Kaldi provides features as: extensive linear algebra support, extensible design with generic algorithms, and complete recipes [64].

Chapter 4

Methodology

It was decided to use Kaldi ASR Toolkit to develop the online speaker diarization system, as it is one one the most widely used automatic speech recognition toolkits in research and industry scenarios. A vital advantage of the toolkit is that it includes out-of-the-box working examples for distinct speech-related tasks, commonly known as "recipes". There are several examples of offline speaker diarization systems within these recipes, so the time to have an initial working system is almost none. One of the most friendly recipes for offline speaker diarization is the one specifically made for the CALLHOME dataset [65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77], which includes two variants, the first one uses i-vectors as speech representations, and the second one uses x-vectors.

The first problem encountered in using one of the recipes was the database, as it is not publicly available and requires purchasing a license. Nevertheless, thanks to the Center for Language and Speech Processing of Johns Hopkins University, access to the database was granted and computational power provided by their computer cluster; the cluster consisted of several nodes, each one having around 64 GB of system memory, around 20 CPU cores, and four last-generation GPUs; so computational power would not be a problem during the development of this speaker diarization system.

Both variations of the {callhome_diarization} recipe were used during the development of the system. To begin, both recipes were executed step by step as provided in the toolkit; this was to be familiar with their internal structure and to obtain baseline results to compare both embeddings. After this, it was evident that the x-vector provided better performance for speaker diarization, as they significantly improved the Diarization Error Rate (DER) in the same dataset.

4.1 Datasets

For the development of the system, the 2000 NIST Speaker Recognition Evaluation (LDC2001S97) corpus was used, which is referred in the Kaldi toolkit *callhome_diarization* recipe as *CALLHOME* [65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77]. It contains 500 utterances distributed across six languages: Spanish, English, Arabic, German, Japanese, and Mandarin. Each utterance contains up to 7 speakers.

A consideration that must be noted is that the CALLHOME dataset contains only recordings from telephone conversations, so the produced diarization model most probably not perform well in other types of environments, so it would be necessary to re-train the embedding extraction and PLDA model to work in other scenarios.

The development of the speaker diarization system followed the same 2-fold strategy as used by the *callhome_diarization* recipe, where half of the dataset (250 utterances) is used to train the embedding extraction model, the PLDA model, and to optimize the AHC threshold and tested with the other half of the dataset and vice-versa, with the final DER being the mean of both CALLHOME folds.

4.2 System overview

The system proposed in this work carries out speaker diarization in an online fashion. Fig. 4.1 shows our system overview, which consists of two main stages that perform the online diarization task. The first stage gets an audio segment, defined with a specific length, and extracts its speech features (x-vector). The second stage clusters all the incoming feature sets based on a similarity measure so that segments belonging to the same speaker stay in the same cluster.

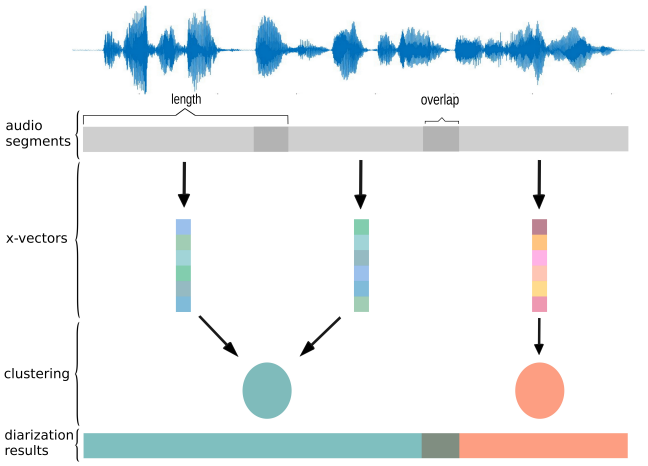


Figure 4.1: System overview.

Each stage has its own set of modules performing specific tasks. All modules are described below in order of usage in the online speaker diarization pipeline.

4.2.1 Speech activity detection

The audio pre-processing module is in charge of getting the audio segments out of the audio stream from a recording. This module stores a pre-established time length segment of the incoming audio signal into a buffer, whose content is delivered to the feature extraction module. This module has a Speech Activity Detector integrated, so every audio segment contains mostly speech rather than noise or silence.

It was decided to use an oracle SAD for the development of the system, which fulfills the SAD task using the ground-truth labels; in this way, the system's performance is not affected by this module as it produces perfect speech and non-speech annotations. This decision aims to test the proposed clustering module in the best possible conditions to assess its effectiveness.

For this reason, the system cannot be used as-is because it cannot produce speech segments automatically.

4.2.2 Segmentation

The segmentation module is commissioned to produce length-restricted speech segments from the SAD output; in this case, the speaker diarization system uses 1.5s long segments with an overlap of 0.5s. During development, it was not tested different configurations of these two parameters, so it would be necessary to research further how they affect the online clustering algorithm's performance.

4.2.3 Embedding extraction

To develop the x-vector extractor, the *callhome_diarization* recipe was followed as is; this ensured that our experiments could be easily replicated. There was no further experimentation with i-vectors as it was clear that they provided worse performance in speaker diarization than x-vectors.

Once the segmentation module produces a segment, a 400-dimensional x-vector is extracted from it and is used throughout the rest of our online speaker diarization process, with no length normalization or whitening.

4.3 Speaker model update

After a segment embedding was extracted, the speaker model update module produces a speaker label. This module is the heart of the online speaker diarization system, as it gives it the ability to produce speaker labels with each incoming segment in an online manner. It uses previously seen embeddings to generate speaker models for all intervening speakers in the recording.

The clustering is performed by computing the similarity between the incoming segment embedding and the available speaker models generated from previous iterations; thus, the speaker diarization system only performs as many comparisons as speakers stored on the system memory.

At the beginning of a new recording, the system does not have any speaker model stored on its memory, so the first incoming x-vector is used as the first speaker model. Then, as the second x-vector arrives, the system computes its similitude to the current speaker model, and, if a similarity threshold is exceeded, the segment is labeled with the same label as the first x-vector, and the speaker model is updated by a weighted sum of its current value and the now labeled embedding. If the similarity threshold is not exceeded, a new speaker model is initialized, with its initial value being the current segment's x-vector. Following this strategy, all the recording segments are clustered according to their similitude to the newly-generated speaker models.

During development, two different similarity measures were tested, cosine similarity and Probabilistic Linear Discriminant Analysis (PLDA). At the beginning of the experimentation, cosine similarity was selected because it does not require training. The only supervised module was the x-vector extractor, but the performance was not good, as it had significant problems differentiating x-vectors from two different speakers. A script provided in the Kaldi toolkit was used to measure Diarization Error Rate (DER) with no forgiveness collar. Given the poor performance of the cosine similarity measure, the system was updated to use PLDA as a similarity measure; this second criterion produced less DER; as expected, the PLDA model was trained following a 2-fold strategy CALLHOME dataset.

Once the similarity measure problem was solved, the system's performance was still lacking compared to the baseline offline systems; and after some analysis, it was found that the speaker model update was helpful until a certain point, after what it hindered the performance if updates still were performed on it. To solve this problem, updating threshold was defined, so the system only updated models that had fewer updates than the threshold.

The system performance was further improved by the definition of a mechanism to update the similarity measure threshold as segments were labeled; in this way, at the beginning of the recording, it is easy to cluster a segment into one of the existing speaker clusters, but as the system proceeds with the recording's segments, the similarity measure required to update an

existing speaker model is increased. As segments came by, it is expected that speaker models converge into a better representation so that the system can ask for more significant similarity to update an existing speaker model. It was found that this strategy allowed the system to find speakers that appeared after a considerable time in the recording.

Chapter 5

Results

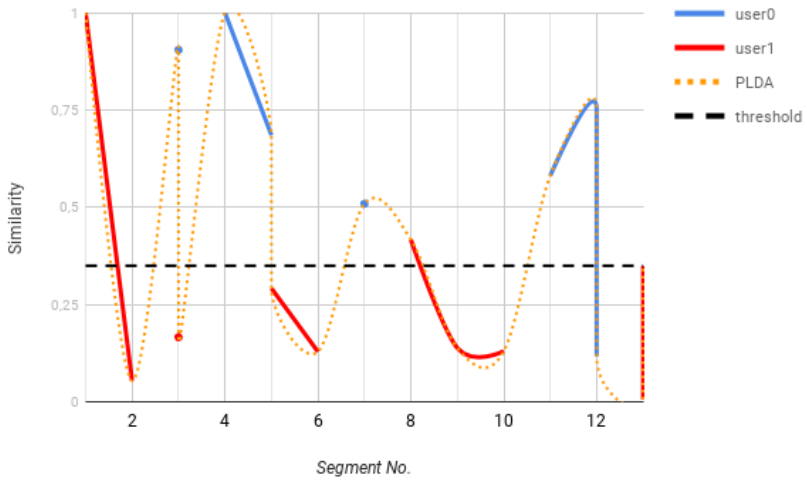
Figure 5.1 presents the results of the first approach used for the online speaker diarization system, which used i-vectors as speech features. The figure shows the PLDA score over time for a single recording following two different strategies; the first one computes the similarity of the current segment embedding to the selected speaker models and outputs a label without updating the speaker models. The second strategy is the same as the first one, but as the system selects one of the speakers to use its label, it updates the selected speaker model.

Table 5.1 shows the results comparison between the i-vector and x-vector-based systems for the CALLHOME dataset. For both, a dynamic threshold was used; as segment embeddings were labeled as a speaker, a speaker-specific threshold was increased, so further down in the recording, a greater similitude would be necessary to update the speaker model with the segment embedding data.

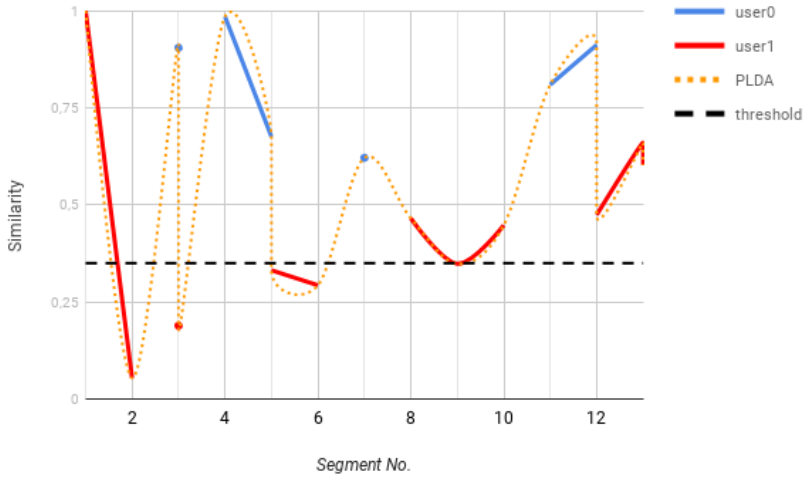
Table 5.2 shows the comparison of the results between the i-vector and x-vector-based systems; the difference between the experiments shown in this table and the ones shown in the last table is that here a hard limit is used, limiting how many segment embeddings are allowed to be used to update a speaker model, so even if down the line a segment embedding has a great similitude to the speaker model, the speaker model would not be updated.

Threshold	**Max segments**	**i-vector DER**	**x-vector DER**
0.88	-	49.47	38.66
0.9	-	49.47	38.16
0.6	2	47.96	**37.71**

Table 5.1: DER (%) comparison between i-vector and x-vector for the CALLHOME dataset with dynamic threshold adjustment.



(a) Online clustering without speaker model update.



(b) Online clustering with speaker model update.

Figure 5.1: PLDA score (%) comparison with speaker model update of the i-vector baseline system.

Threshold	**Max segments**	**i-vector DER**	**x-vector DER**
0.6	1	48.35	38.45
0.6	2	47.96	**37.71**
0.6	3	47.9	39.42
0.6	4	47.64	40.29

Table 5.2: DER (%) comparison between i-vector and x-vector for the CALLHOME dataset with speaker model update limit.

Figure 5.2 shows how a good speaker model initialization could improve the speaker diariza-

tion results. These experiments use the ground-truth labels during inference to signal the system if a new speaker model has to be created; as soon as a not-seen speaker appears in the recording, the system generates a new speaker model ignoring the previously mentioned similitude strategy. This can be seen as a warm-up step for the online speaker diarization system; after that, the system follows its standard procedure to incoming label segments as one of the generated speaker models.

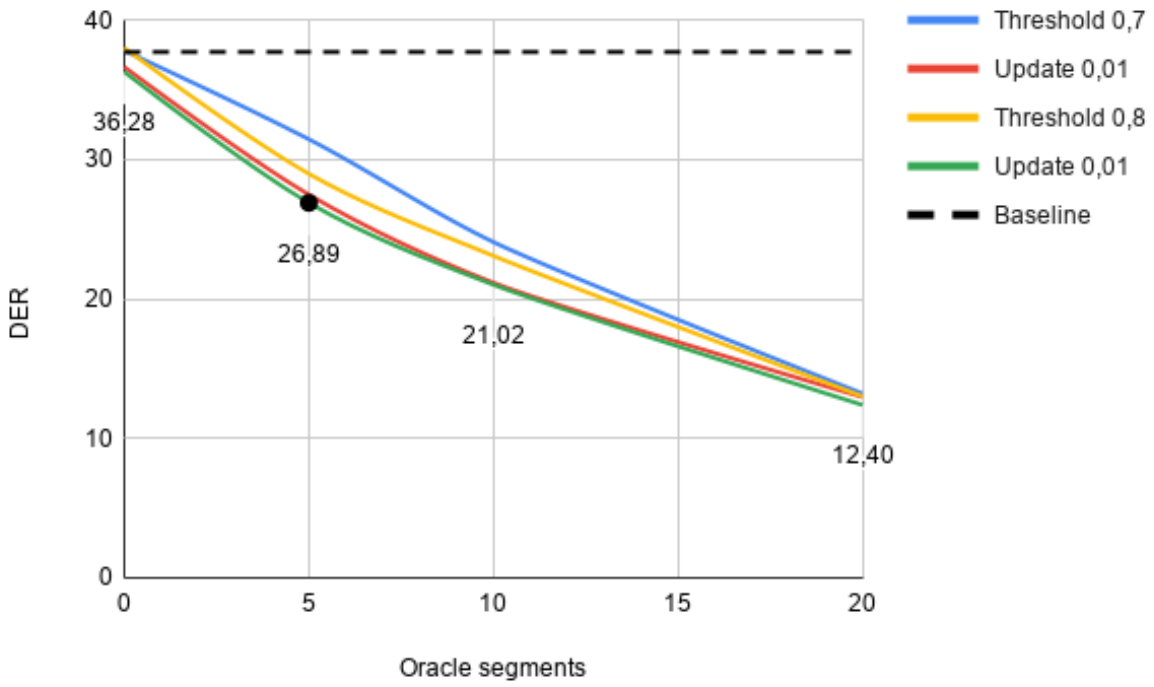


Figure 5.2: DER (%) graph of the x-vector system using oracle model update in CALLHOME.

Chapter 6

Discussion

As shown in Table 5.1, the developed online speaker diarization system can perform its task successfully. Its performance falls behind the Kaldi toolkit's baseline system for both i-vectors and x-vectors; this is because of the latency difference between both systems; the developed system works in an online fashion, while the baseline system makes offline inference. Having all recording data available to perform speaker diarization allows the baseline system to cluster the speech segments more accurately as the contextual information for inference comprises the full-length of the recording. In contrast, the developed system has to label each speech segment one at a time, using as contextual information the speaker models it generated in previous iterations; So the clustering module in the online speaker diarization system cannot use the similitude between all speech segments to perform clustering, as the offline variant can.

Another phenomenon shown in Table 5.1 is that x-vectors provide a 21.37% improvement over the i-vectors for the system, which is expected, as they better capture speaker-relevant information due to its extractor being trained with more data that was augmented with noise. So it can better handle disturbances in the input audio to return a speaker-representative embedding.

It should be noted that the first experiments presented in the table were executed allowing the diarization system to update the speaker models unlimited times, while the last experiment limited the system to update the speaker model up to 2 times. In both cases, the threshold required to update the speaker model was dynamic, so with each speech segment labeled as some of the speaker models, a better similitude score would be needed in a subsequent iteration to update the speaker model. The idea behind this is that as a speaker model is updated, it should converge into a more representative model, so the similitude of speech segments produced by the same speaker to the model should increase with each update. However, an unexpected behavior was found during experimentation; updating the speaker model further down the recording decreased the system performance; this phenomenon is because the weighted sum tends to make all speaker models more similar if more updates than needed are made. This decrease in per-

formance should be further investigated because, at this moment, a pre-defined hand-optimized threshold is used to limit the number of updates that can be made to a speaker model; and most probably, it would not be the same for all speaker models, and also it would differ depending on the recording conditions.

The results presented in Table 5.2 further support this claim, because as with each experiment, the number of allowed updates to the speaker model is increased, and the performance of the system decreased. It can be seen that the update of the speaker model helps the system to perform its task up to 2 updates, so the idea of updating a model that represents each speaker found in the recording by the system is correct for both i-vector and x-vector. However, as shown in the table, the improvement given by the speaker model update is minor in i-vectors, which indicates that the speaker model does not suffer heavy changes with the updates, this is primarily due to the gaussian nature of the i-vectors, but further research in the topic should be done.

As discussed so far, the speaker model update seems to be a suitable method, but only up to a certain number of updates, from which the performance of the system falls behind. Given the experiments performed during this system's development, the speaker model's equally weighted sum and the selected segment embedding was the only update strategy tested. Hence, it is feasible that another update strategy could improve the system's performance and allow it to use a speaker model update strategy along all recording duration instead of stopping it with a pre-defined limit of updates.

Furthermore, as shown in Figure 5.2, the selection of a good set of speech segments to initialize a speaker model is crucial for good diarization performance. In the figure, it could be seen that with five oracle segments for speaker model initialization, the performance of the online speaker diarization system reaches the same DER as the offline system. This selection would require a good Speech Activity Detector (SAD), equivalent to the oracle one used during experimentation, which is one of the areas of opportunity of this work. Knowing that only with 5 seconds of oracle segments the system can reach offline-like performance presents the idea of a warm-up step in the pipeline, where an offline speaker diarization system selects the best speech segments at the beginning of the input audio stream and computes the speaker models used further down in the online process.

Another way to test the speaker model update idea would be to build upon this research into a speaker tracking system, where speaker models are available to the system, and it has to search for the occurrences of those specific speakers within a recording in an online fashion. As shown in the results from Figure 5.2, the tracking system should have good performance.

Chapter 7

Conclusion

Speech has become one of the most essential means for human-machine communication; with technologies such as assistant devices, wearable devices, and in-vehicle information systems performing each day more sensitive tasks, improving human-machine communication is an essential task to assure safe execution. From all speech technologies fields, speaker diarization provides a way to solve the question, "who spoke when?" by exploiting each person's speech's unique characteristics like a fingerprint, allowing an automatic system to know which utterances come from the same speaker, so commands from different speakers are not mixed up. So, speaker diarization can be seen as an enabling technology for downstream tasks such as automatic speech recognition, improving upon them by providing contextual information about how many speakers are intervening in a conversation and when each of them talks.

In recent years speaker diarization has been thoroughly researched by industry and universities, but research has mainly focused on its offline variant, as it produces the best results due to its use of all recording data as contextual information to perform its task. Currently, most state-of-the-art systems use deep neural network-based speech representations for this task, specifically x-vectors, which has become the most widely used embedding for speaker diarization. So at this moment, there has been extensive research in offline speaker diarizaton with x-vectors. In contrast, the online variant of speaker diarization has hardly been reviewed as extensively in comparison.

This work's objective was to research online speaker diarization and develop a computer system capable of performing it using x-vectors with a clustering algorithm that used embeddings to represent the speakers found in the recording, with an update mechanism that improves such speaker models to represent its speakers better. During the literature review, it was found that online speaker diarization using x-vectors had not been extensively reviewed, as they were some works from industry, but they used their own-developed speech representations. So this work would contribute to a research area that seems to be the next step on speaker diarization.

The experimental results showed that the developed system could perform online speaker diarization; Two different speech representations were tested, i-vectors and x-vectors; the first one has been one of the most widely used embeddings for speaker diarization, and even to this day, is still used along with new neural speech representations. The second embedding tested was x-vectors, a state-of-the-art DNN-based speech representation that exploits large amounts of data during its extractor training. As expected, our online speaker diarization system had better performance with x-vectors, as it was better suited to define the different speakers within the recordings. Future research should test how the mixture of both embeddings impacts system performance and other embedding architectures such as d-vectors.

The developed online clustering algorithm provided an efficient way to label the speech segments with low computational cost, as it only makes as many comparisons as they are speech segments in the recording times the number of speakers. Also, the system can find an unlimited number of speakers as it produces a new speaker model when the similitude score does not reach a pre-defined threshold. This kind of clustering strategy was used in other works, but to the author's knowledge, it is the first time it has been used for online speaker diarization in the CALLHOME dataset.

Using an x-vector as speaker model updated by a weighted sum of its current value and the x-vector from the speech segment selected to be labeled as the speaker has been proven to work, as shown in the experimental results. Further research has to be made to find if the weighted sum is the best method to update the speaker model; due to it best working with a limited amount of updates, controlled by a pre-defined threshold, because, during development, the system results deteriorated as the speaker model was further updated. Another interesting finding within the online clustering strategy was that using a limited amount of oracle segments for speaker model initialization provided competitive results against the offline diarization baseline. Knowing that a better speaker model initialization significantly improves the diarization results indicates that a warm-up period at the beginning of a recording using an offline speaker diarization system is the next step to improving system performance.

Considering these findings, the developed online speaker diarization system was a success, as it provided the baseline knowledge of developing a diarization system in its mostly unexplored variation: online speaker diarization.

Bibliography

[1] G. Sell and D. Garcia-Romero, "Diarization resegmentation in the factor analysis subspace," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 4794–4798.

[2] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree, "Speaker diarization using deep neural network embeddings," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 4930–4934.

[3] Tae Jin Park and Panayiotis Georgiou, "Multimodal Speaker Segmentation and Diarization using Lexical and Acoustic Cues via Sequence to Sequence Neural Networks," 2018.

[4] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker Recognition for Multi-speaker Conversations Using X-vectors," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 5796–5800.

[5] Gregory Sell, David Snyder, Alan McCree, Daniel Garcia-Romero, Jesús Villalba, Matthew Maciejewski, Vimal Manohar, Najim Dehak, Daniel Povey, Shinji Watanabe, and Sanjeev Khudanpur, "Diarization is Hard: Some Experiences and Lessons Learned for the JHU Team in the Inaugural DIHARD Challenge," 09 2018, pp. 2808–2812.

[6] Mireia Diez, Federico Landini, Lukáš Burget, Johan Rohdin, Anna Silnova, Kateřina Žmolíková, Ondřej Novotný, Karel Veselý, Ondřej Glembek, Oldřich Plchot, Ladislav Mošner, and Pavel Matějka, "BUT System for DIHARD Speech Diarization Challenge 2018," in *Proc. Interspeech 2018*, 2018, pp. 2798–2802.

[7] Laurent El Shafey, Hagen Soltau, and Izhak Shafran, "Joint Speech Recognition and Speaker Diarization via Sequence Transduction," *arXiv e-prints*, p. arXiv:1907.05337, Jul 2019.

[8] Dimitrios Dimitriadis and Petr Fousek, "Developing On-Line Speaker Diarization System," 08 2017, pp. 2739–2743.

[9] Monisankha Pal, Manoj Kumar, Raghuveer Peri, and Shrikanth Narayanan, "A STUDY OF SEMI-SUPERVISED SPEAKER DIARIZATION SYSTEM USING GAN MIXTURE MODEL," 2019.

[10] Qingjian Lin, Ruiqing Yin, Ming Li, Hervé Bredin, and Claude Barras, "LSTM Based Similarity Measurement with Spectral Clustering for Speaker Diarization," *Interspeech 2019*, Sep 2019.

[11] Aonan Zhang, Quan Wang, Zhenyao Zhu, John Paisley, and Chong Wang, "FULLY SUPERVISED SPEAKER DIARIZATION," 2019.

[12] Quan Wang, Carlton Downey, Li Wan, Philip Andrew Mansfield, and Ignacio Lopez Moreno, "SPEAKER DIARIZATION WITH LSTM," 2018.

[13] Omid Ghahabi and Volker Fischer, "Speaker-Corrupted Embeddings for Online Speaker Diarization," 09 2019, pp. 386–390.

[14] Neville Ryant, Kenneth Church, Christopher Cieri, Alejandrina Cristia, Jun Du, Sriram Ganapathy, and Mark Liberman, "The Second DIHARD Diarization Challenge: Dataset, task, and baselines," 2019.

[15] Sergey Novoselov, Aleksei Gusev, Artem Ivanov, Timur Pekhovsky, Andrey Shulipa, Anastasia Avdeeva, Artem Gorlanov, and Alexander Kozlov, "Speaker Diarization with Deep Speaker Embeddings for DIHARD Challenge II," 09 2019, pp. 1003–1007.

[16] Federico Landini, Shuai Wang, Mireia Diez, Lukáš Burget, Pavel Matějka, Kateřina Žmolíková, Ladislav Mošner, Oldřich Plchot Ondřej Novotný, Hossein Zeinali, and Johan Rohdin, "BUT System Description for DIHARD Speech Diarization Challenge 2019," 2019.

[17] Giovanni Soldi, Christophe Beaugeant, and Nicholas Evans, "ADAPTIVE AND ONLINE SPEAKER DIARIZATION FOR MEETING DATA," 01 2015.

[18] Jose Patino, Ruiqing Yin, H'ector Delgado, Hervé Bredin, Alain Komaty, Guillaume Wisniewski, Claude Barras, Nicholas Evans, and S'ebastien Marcel, "Low-latency speaker spotting with online diarization and detection," in *The Speaker and Language Recognition Workshop*, ISCA, Ed., Les Sables d'Olonne, France, June 2018, ISCA.

[19] Jonathan G. Fiscus, Jerome Ajot, Martial Michel, and John S. Garofolo, "The Rich Transcription 2006 Spring Meeting Recognition Evaluation," in *Proceedings of the Third International Conference on Machine Learning for Multimodal Interaction*, Berlin, Heidelberg, 2006, MLMI'06, pp. 309–322, Springer-Verlag.

[20] Thilo von Neumann, Keisuke Kinoshita, Marc Delcroix, Shoko Araki, Tomohiro Nakatani, and Reinhold Haeb-Umbach, "ALL-NEURAL ONLINE SOURCE SEPARATION, COUNTING, AND DIARIZATION FOR MEETING ANALYSIS," 05 2019, pp. 91–95.

[21] Vlado Delić, Zoran Perić, Milan Sečujski, Nikša Jakovljević, Jelena Nikolić, Dragiša Mišković, Nikola Simić, Siniša Suzić, and Tijana Delić, "Speech Technology Progress Based on New Machine Learning Paradigm," *Computational Intelligence and Neuroscience*, vol. 2019, pp. 19, 2019.

[22] Homayoon Beigi, *Fundamentals of Speaker Recognition*, 12 2011.

[23] Ian Vince McLoughlin, "Applied Speech and Audio Processing: With Matlab Examples," 2009.

[24] Jinyu Li, Li Deng, Reinhold Haeb-Umbach, and Yifan Gong, "Chapter 1 - Introduction," in *Robust Automatic Speech Recognition*, Jinyu Li, Li Deng, Reinhold Haeb-Umbach, and Yifan Gong, Eds., pp. 1 – 7. Academic Press, Oxford, 2016.

[25] Dong Yu and Li Deng, *Automatic Speech Recognition: A Deep Learning Approach*, Springer Publishing Company, Incorporated, 2014.

[26] Brett R. Schofield and Diana L. Coomes, "Pathways from auditory cortex to the cochlear nucleus in guinea pigs," *Hearing Research*, vol. 216-217, pp. 81 – 89, 2006, The Cochlear Nucleus.

[27] Patrick J. Loughlin and Berkant Tacer, "On the amplitude- and frequency-modulation decomposition of signals," *The Journal of the Acoustical Society of America*, vol. 100, no. 3, pp. 1594–1601, 1996.

[28] M.M. Straka, R. Hughes, P. Lee, and H.H. Lim, "Descending and tonotopic projection patterns from the auditory cortex to the inferior colliculus," *Neuroscience*, vol. 300, pp. 325 – 337, 2015.

[29] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Moreno, and Javier Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," 05 2014, pp. 4052–4056.

[30] Johan Rohdin, Anna Silnova, Mireia Diez, Oldřich Plchot, Pavel Matějka, Lukáš Burget, and Ondřej Glembek, "End-to-end DNN based text-independent speaker recognition for long and short utterances," *Computer Speech & Language*, vol. 59, pp. 22 – 35, 2020.

[31] Daniel Jurafsky and James Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, vol. 2, 02 2008.

[32] Minho Jin and Chang Yoo, "Speaker Verification and Identification," *Behavioral Biometrics for Human Identification: Intelligent Applications*, pp. 264–289, 01 2009.

[33] M. T. S. Al-Kaltakchi, W. L. Woo, S. S. Dlay, and J. A. Chambers, "Comparison of I-vector and GMM-UBM approaches to speaker identification with TIMIT and NIST 2008 databases in challenging environments," in *2017 25th European Signal Processing Conference (EUSIPCO)*, Aug 2017, pp. 533–537.

[34] Douglas A. Reynolds, Thomas F. Quatieri, and Robert B. Dunn, "Speaker Verification Using Adapted Gaussian Mixture Models," *Digital Signal Processing*, vol. 10, no. 1, pp. 19 – 41, 2000.

[35] Jayanth and B. Roja Reddy, "Speaker Identification based on GFCC using GMM-UBM," 2016.

[36] D. E. Sturim and D. A. Reynolds, "Speaker adaptive cohort selection for Tnorm in text-independent speaker verification," in *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, March 2005, vol. 1, pp. I/741–I/744 Vol. 1.

[37] "IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)," .

[38] S. Araki, M. Fujimoto, K. Ishizuka, H. Sawada, and S. Makino, "Speaker indexing and speech enhancement in real meetings / conversations," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, March 2008, pp. 93–96.

[39] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree, "Speaker diarization using deep neural network embeddings," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 4930–4934.

[40] W. Dixon Ward, "Chapter Eleven - Musical Perception," in *Foundations of Modern Auditory Theory*, Jerry V. Tobias, Ed., pp. 405 – 447. Academic Press, 1970.

[41] T. Drugman, Y. Stylianou, Y. Kida, and M. Akamine, "Voice Activity Detection: Merging Source and Filter-based Information," *IEEE Signal Processing Letters*, vol. 23, no. 2, pp. 252–256, Feb 2016.

[42] A. Benyassine, E. Shlomot, H. . Su, D. Massaloux, C. Lamblin, and J. . Petit, "ITU-T Recommendation G.729 Annex B: a silence compression scheme for use with G.729 optimized for V.70 digital simultaneous voice and data applications," *IEEE Communications Magazine*, vol. 35, no. 9, pp. 64–73, Sep. 1997.

[43] Damjan Vlaj, Bojan Kotnik, Bogomir Horvat, and Zdravko Kačič, "A Computationally Efficient Mel-Filter Bank VAD Algorithm for Distributed Speech Recognition Systems," *EURASIP Journal on Advances in Signal Processing*, vol. 2005, no. 4, pp. 561951, Mar 2005.

[44] I. McCowan, D. Dean, M. McLaren, R. Vogt, and S. Sridharan, "The Delta-Phase Spectrum With Application to Voice Activity Detection and Speaker Recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2026–2038, Sep. 2011.

[45] J. Ramirez, J. M. Gorriz, and J. C. Segura, "Voice Activity Detection. Fundamentals and Speech Recognition System Robustness," in *Robust Speech*, Michael Grimm and Kristian Kroschel, Eds., chapter 1. IntechOpen, Rijeka, 2007.

[46] Stanley Burris and Hanamantagouda Sankappanavar, *A Course in Universal Algebra*, vol. 91, 01 1981.

[47] Herbert Enderton, *A Mathematical Introduction to Logic*, 01 1972.

[48] Shuai Wang, Yanmin Qian, and Kai Yu, "What Does the Speaker Embedding Encode?," 08 2017, pp. 1497–1501.

[49] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-End Factor Analysis for Speaker Verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[50] Ondrej Novotny, Oldrich Plchot, Ondrej Glembek, Lukas Burget, and Pavel Matejka, "Discriminatively Re-trained i-vector Extractor for Speaker Recognition," 2018.

[51] Jinxi Guo, Ning Xu, Kailun Qian, Yang Shi, Kaiyuan Xu, Yingnian Wu, and Abeer Alwan, "Deep neural network based i-vector mapping for speaker verification using short utterances," *Speech Communication*, vol. 105, 10 2018.

[52] Jiamin Xie, L. Paola García-Perera, Daniel Povey, and Sanjeev Khudanpur, "Multi-PLDA Diarization on Children's Speech," in *INTERSPEECH 2019*, 2019.

[53] Prasanna V. Kothalkar, Dwight Irvin, Ying Luo, Joanne Rojas, John Nash, Beth Rous, and John H. L. Hansen, "Tagging child-adult interactions in naturalistic, noisy, daylong school environments using i-vector based diarization system," in *Proc. SLaTE 2019: 8th ISCA Workshop on Speech and Language Technology in Education*, 2019, pp. 89–93.

[54] Angga Dwi Firmanto, Miranti Indar Mandasari, Suprijanto, and Fadjar Fathurrahman, "Applying GMM-UBM framework for Indonesian forensic speaker verification," *AIP Conference Proceedings*, vol. 2088, no. 1, pp. 050013, 2019.

[55] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-Vectors: Robust DNN Embeddings for Speaker Recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2018, pp. 5329–5333.

[56] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker Recognition for Multi-speaker Conversations Using X-vectors," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 5796–5800.

[57] David Snyder, Daniel Garcia-Romero, Alan McCree, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "Spoken Language Recognition using X-vectors," 06 2018, pp. 105–111.

[58] Longting Xu, Rohan Kumar Das, Emre Yılmaz, Jichen Yang, and Haizhou Li, "Generative x-vectors for text-independent speaker verification," 2018.

[59] Najim Dehak, R. Dehak, James Glass, Douglas Reynolds, and Patrick Kenny, "Cosine similarity scoring without score normalization techniques," 01 2010.

[60] Mohammed Senoussaoui, Patrick Kenny, Pierre Dumouchel, and Najim Dehak, "New cosine similarity scorings to implement gender-independent speaker verification.," in *Interspeech*, 2013, pp. 2773–2777.

[61] Sergey Ioffe, "Probabilistic Linear Discriminant Analysis," in *Computer Vision – ECCV 2006*, Aleš Leonardis, Horst Bischof, and Axel Pinz, Eds., Berlin, Heidelberg, 2006, pp. 531–542, Springer Berlin Heidelberg.

[62] Abbas Khosravani and Mohammad M. Homayounpour, "A PLDA approach for language and text independent speaker recognition," *Computer Speech & Language*, vol. 45, pp. 457 – 474, 2017.

[63] C. Zhao, L. Li, D. Wang, and A. Pu, "Local training for PLDA in speaker verification," in *2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA)*, Oct 2016, pp. 156–160.

[64] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely, "The Kaldi Speech Recognition Toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Dec. 2011, IEEE Signal Processing Society, IEEE Catalog No.: CFP11SRW-USB.

[65] Alvin Martin and Mark Przybocki, "2004 NIST Speaker Recognition Evaluation LDC2006S44," in *Web Download*. 2006, Philadelphia: Linguistic Data Consortium.

[66] NIST Multimodal Information Group, "2005 NIST Speaker Recognition Evaluation Training Data DC2011S01," in *Web Download*. 2011, Philadelphia: Linguistic Data Consortium.

[67] NIST Multimodal Information Group, "2005 NIST Speaker Recognition Evaluation Test Data LDC2011S04," in *Web Download*. 2011, Philadelphia: Linguistic Data Consortium.

[68] NIST Multimodal Information Group, "2006 NIST Speaker Recognition Evaluation Training Set LDC2011S09," in *Web Download*. 2011, Philadelphia: Linguistic Data Consortium.

[69] NIST Multimodal Information Group, "2006 NIST Speaker Recognition Evaluation Test Set Part 1 LDC2011S10," in *Web Download*. 2011, Philadelphia: Linguistic Data Consortium.

[70] NIST Multimodal Information Group, "2006 NIST Speaker Recognition Evaluation Test Set Part 2 DC2012S01," in *Web Download*. 2012, Philadelphia: Linguistic Data Consortium.

[71] NIST Multimodal Information Group, "2008 NIST Speaker Recognition Evaluation Training Set Part 1 LDC2011S05," in *Web Download*. 2011, Philadelphia: Linguistic Data Consortium.

[72] NIST Multimodal Information Group, "2008 NIST Speaker Recognition Evaluation Test Set LDC2011S08," in *Web Download*. 2011, Philadelphia: Linguistic Data Consortium.

[73] David Graff, Kevin Walker, and Alexandra Canavan, "Switchboard-2 Phase II LDC99S79," in *DVD*. 1999, Philadelphia: Linguistic Data Consortium.

[74] David Graff, David Miller, and Kevin Walker, "Switchboard-2 Phase III Audio LDC2002S06," in *Web Download*. 2002, Philadelphia: Linguistic Data Consortium.

[75] David Graff, Kevin Walker, and David Miller, "Switchboard Cellular Part 1 Audio LDC2001S13," in *DVD*. 2001, Philadelphia: Linguistic Data Consortium.

[76] David Graff, Kevin Walker, and David Miller, "Switchboard Cellular Part 2 Audio LDC2004S07," in *DVD*. 2004, Philadelphia: Linguistic Data Consortium.

[77] David Snyder, Guoguo Chen, and Daniel Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015, arXiv:1510.08484v1.