INSTITUTO POLITÉCNICO DE BRAGANÇA
ipb Escola Superior de Tecnologia e Gestão

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Speaker Recognition for Door Opening Systems

Enrico Manfron

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Informatics. Work developed during the double degree exchange program between the Instituto Politécnico de Bragança (IPB) and the Universidade Tecnológica Federal do Paraná (UTFPR).

Work oriented by:
Professor PhD João Paulo Teixeira
Professor PhD Rodrigo Minetto

Bragança
November 2023

ipb
INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e Gestão

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Speaker Recognition for Door Opening Systems

Enrico Manfron

Dissertation presented to the School of Technology and Management of Bragança to obtain the Master Degree in Informatics. Work developed during the double degree exchange program between the Instituto Politécnico de Bragança (IPB) and the Universidade Tecnológica Federal do Paraná (UTFPR).

Work oriented by:
Professor PhD João Paulo Teixeira
Professor PhD Rodrigo Minetto

Bragança
November 2023

Dedication

I want to dedicate this work to my beloved Giuliana Martins Silva, who has been the anchor of my life. Her unwavering love and support have been my guiding light throughout this journey. I am also thankful to my family, Thais Manfron, Valdecir Antônio Manfron, and Lucca Manfron, for their endless support and instilled values that have shaped me.

I also want to dedicate this work to my friends, Leonardo, Lorena, and Joyce, who ensured I kept my sanity during the most challenging times in Portugal. Additionally, I am grateful for Lucas Ricardo and Marcelle Pires, who have been an integral part of my academic trajectory in Brazil, and my best friend Dante Diniz, even with the distance, keeps being an essential part of my life.

Acknowledgement

To Instituto Politécnico de Bragança (IPB) and Universidade Tecnológica Federal do Paraná (UTFPR) for the opportunity of being part of the exchange program between them.

To Prof. Dr. João Paulo Teixeira and Prof. Dr. Rodrigo Minetto for all the help, support, and guidance provided throughout this work.

To the professors of IPB and UTFPR, and to each employee of IPB and UTFPR that were part of my academic development.

Abstract

Besides being an important communication tool, the voice can also serve for identification purposes since it has an individual signature for each person. Speaker recognition technologies can use this signature as an authentication method to access environments.

This work explores the development and testing of machine and deep learning models, specifically the GMM, the VGG-M, and ResNet50 models, for speaker recognition access control to build a system to grant access to CeDRI's laboratory. The deep learning models were evaluated based on their performance in recognizing speakers from audio samples, emphasizing the Equal Error Rate metric to determine their effectiveness. The models were trained and tested initially in public datasets with 1251 to 6112 speakers and then fine-tuned on private datasets with 32 speakers of CeDri's laboratory.

In this study, we compared the performance of ResNet50, VGGM, and GMM models for speaker verification. After conducting experiments on our private datasets, we found that the ResNet50 model outperformed the other models. It achieved the lowest Equal Error Rate (EER) of 0.7% on the Framed Silence Removed dataset. On the same dataset, the VGGM model achieved an EER of 5%, and the GMM model achieved an EER of 2.13%.

Our best model's performance was unable to achieve the current state-of-the-art of 2.87% in the VoxCeleb 1 verification dataset. However, our best implementation using ResNet50 achieved an EER of 5.96% while being trained on only a tiny portion of the data than it usually is. So, this result indicates that our model is robust and efficient and provides a significant improvement margin.

This thesis provides insights into the capabilities of these models in a real-world application, aiming to deploy the system on a platform for practical use in laboratory access authorization. The results of this study contribute to the field of biometric security by demonstrating the potential of speaker recognition systems in controlled environments.

Resumo

Além de ser uma importante ferramenta de comunicação, a voz também pode servir para fins de identificação, pois possui uma assinatura individual para cada pessoa. As tecnologias de reconhecimento de voz podem usar essa assinatura como um método de autenticação para acessar ambientes.

Este trabalho explora o desenvolvimento e teste de modelos de aprendizado de máquina e aprendizado profundo, especificamente os modelos GMM, VGG-M e ResNet50, para controle de acesso de reconhecimento de voz com o objetivo de construir um sistema para conceder acesso ao laboratório do CeDRI. Os modelos de aprendizado profundo foram avaliados com base em seu desempenho no reconhecimento de falantes a partir de amostras de áudio, enfatizando a métrica de Taxa de Erro Igual para determinar sua eficácia. Os modelos foram inicialmente treinados e testados em conjuntos de dados públicos com 1251 a 6112 falantes e, em seguida, ajustados em conjuntos de dados privados com 32 falantes do laboratório do CeDri.

Neste estudo, comparamos o desempenho dos modelos ResNet50, VGGM e GMM para verificação de falantes. Após realizar experimentos em nossos conjuntos de dados privados, descobrimos que o modelo ResNet50 superou os outros modelos. Ele alcançou a menor Taxa de Erro Igual (EER) de 0,7% no conjunto de dados Framed Silence Removed. No mesmo conjunto de dados, o modelo VGGM alcançou uma EER de 5% e o modelo GMM alcançou uma EER de 2,13%.

O desempenho do nosso melhor modelo não conseguiu atingir o estado da arte atual de 2,87% no conjunto de dados de verificação VoxCeleb 1. No entanto, nossa melhor implementação usando o ResNet50 alcançou uma EER de 5,96%, mesmo sendo treinado em apenas uma pequena parte dos dados que normalmente são utilizados. Assim, este resultado indica que nosso modelo é robusto e eficiente e oferece uma margem significativa de melhoria.

Esta tese oferece insights sobre as capacidades desses modelos em uma aplicação do mundo real, visando implantar o sistema em uma plataforma para uso prático na autorização de acesso ao laboratório. Os resultados deste estudo contribuem para o campo da segurança biométrica ao demonstrar o potencial dos sistemas de reconhecimento de voz em ambientes controlados.

Contents

List of Tables

List of Figures

Acronyms

ADC Analog-to-Digital Converter.

AUC Area Under Curve.

CeDRI Research Centre in Digitalization and Intelligent Robotics.

CNN Convolutional Neural Network.

DAC Digital-to-Analog Converter.

DFT Discrete Fourier Transform.

DL Deep Learning.

EER Equal Error Rate.

EM Expectation–Maximization.

FFT Fast Fourier Transform.

FrSR Framed Silence Remove.

FrT Framed Trim.

GMM Gaussian Mixture Model.

IPB Instituto Politécnico de Bragança.

MFCCs Mel Frequency Cepstral Coefficients.

MLE Maximum Likelihood Estimation.

MLP Multilayer Perceptron.

ResNet50 Residual Neural Network 50.

RMS Root Mean Square.

SBW Spectral Bandwidth.

SC Spectral Centroid.

SI Speaker Identification.

sr Sampling Rate.

SR Speaker Recognition.

SRO Spectral Rolloff.

STFT Short-Time Fourier Transform.

SV Speaker Verification.

UBM Universal Background Model.

UTFPR Universidade Tecnológica Federal do Paraná.

VGG-M Visual Geometry Group Medium.

ZCR Zero Crossing Rate.

Chapter 1

Introduction

1.1 Contextualization

The human voice is a powerful instrument of communication, conveying emotions and intentions. However, beyond its communicative function, the voice can also serve as an identification tool since it is a unique signature loaded with individual characteristics distinguishing one speaker from another (Mohd Hanifa et al., 2021).

Voices vary among individuals due to multiple factors. Physiologically, differences in the size and shape of our vocal organs, such as vocal folds and the vocal tract, play a role. Gender, age, and even languages and accents further contribute to this diversity (Zhang, 2016).

The Speaker Recognition (SR) technology has potential applications in biometric authentication, which can be used in access control systems for secure environments. This technology can accurately identify individuals by analyzing their unique vocal traits, allowing access to restricted areas (Wang et al., 2020).

SR offers an interesting approach in the context of door-opening systems. Unlike traditional access methods, such as physical keys or numeric codes, an individual's voice is hard to replicate or forge, making it a security option. The voice production process is complex and involves several human body systems, from the respiratory to the articulatory systems. This complexity results in a unique sound identity for each individual.

The field of SR has made significant advancements due to technological progress, including the development of sophisticated algorithms and machine learning models. With the integration of advanced artificial intelligence and large datasets, the accuracy and efficiency of SR systems have improved significantly, allowing voice-based door-opening systems to be viable and reliable (Chung et al., 2018; Nagrani et al., 2017; Zhou et al., 2021).

1.2 Motivation

As technology advances, more reliable and robust authentication systems become possible, particularly in sensitive environments such as university laboratories, where valuable research and confidential information are stored. Biometrics is an interesting solution, offering identification methods based on each individual's unique and non-transferable characteristics.

The SR is one of these biometric techniques that uses an individual's unique voice as an identification key. Unlike passwords or cards, which can be lost, forgotten, or stolen, a person's voice is an inherent characteristic that cannot be easily replicated. This makes it a more secure and reliable method of identification.

It is important to understand that this work is a component of an all-encompassing authentication system for Research Centre in Digitalization and Intelligent Robotics (CeDRI)'s laboratory from IPB, including facial recognition. However, this thesis focuses specifically on speaker recognition. Combining various biometric techniques, such as facial and vocal recognition, can add an extra layer of security, ensuring that only authorized individuals are granted access to the lab.

1.3 Goals

This work discusses the strategies for developing and testing machine learning and deep learning models. The objective is to investigate SR-based access control systems and then evaluate and build a system to grant access to CeDRI's laboratory.

The SR system aims to extract vocal features from an individual, known as voice biometrics. This work intends to identify the individual through a comparative process of the speaker's biometrics with a given sample of speakers. As the goal is to develop a door access authorization system, there is a need for an identity verification process.

This process involves ensuring that the identified individual matches the identity declared by the speaker. Furthermore, there is an intention to refine the system developed on a PC for deployment on a hardware electronic device, which will be developed on a microcontroller platform. This device will receive the speech signal from the microphone and manage the door's opening through direct control of the lock's relay.

1.4 Publications

This work has generated a "Speaker Recognition in Door Access Control System" (Manfron et al., 2023b) paper presented in the *Symposium of Applied Science for Young Researchers*, held in Barcelos, Portugal, 11 July 2023. Another paper entitled "Speaker Identification on Small Datasets" (Manfron et al., 2023a) was presented in the *OL2A: International Conference on Optimization, Learning Algorithms and Applications 2023*, held in Ponta Delgada, Portugal, 27–29 September 2023, to be published in Communications in Computer and Information Science, Springer.

1.5 Document structure

The document begins with a background review of the main concepts and technologies used in Chapter 2. Chapter 3 provides the existing literature on SR. Chapter 4 describes the methodologies employed to develop this work. Chapter 5 presents a comparative performance analysis of the models. Finally, This work concludes in Chapter 6.

Chapter 2

Background

2.1 Audio Fundamentals

This section delves into the fundamentals of audio processing, starting with the basics of audio and acoustics. Speech is one variant of the various audio signals that exist. We can define speech as the articulation of socially significant sounds by humans (“Speech”, 2023). It is a primary mode of human communication and has been instrumental in the evolution of civilizations. Moreover, utterance is an audio clip of speech signals with various attributes, such as the speaker, language, and transcript.

Sound is a longitudinal mechanical wave that travels as an acoustic wave through various mediums, such as gas, liquid, or solid. The peaks and troughs of this wave correspond to the medium’s compression and rarefaction states (Nussenzveig, 2018). Figure 2.1 illustrates the propagation of a compression wave along a spring, which is in equilibrium in Figure 2.1(a) and has its end suddenly compressed in (b). Right after the compressed coils, there are some rarefied regions with greater spacing than in the equilibrium position in (c). Thus, the compression wave is followed by another rarefaction wave. The wave then propagates through the medium to stages (d) and (e).

The complexity of sound waveforms can vary, but we can simplify things by starting with the sine wave. This type of wave is a function of time t, represented by the Equation

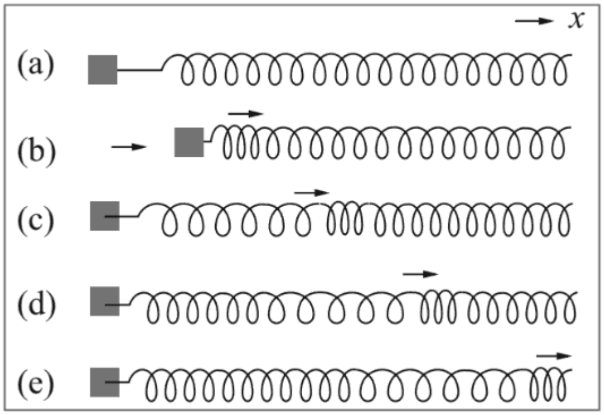

Figure 2.1: Longitudinal Wave Propagation. Extracted from Nussenzveig (2018).

2.1. The equation includes three variables: A for amplitude, f for frequency, and ϕ for phase. In Fourier analysis, we can combine sine and cosine waves with varying frequencies to approximate any periodic function. So, when analyzing a waveform, we can break it down into its individual sine wave components.

$$y(t) = A \sin(2\pi f t + \phi) \tag{2.1}$$

While frequency and amplitude are essential physical quantities, they might not resonate intuitively from a human auditory perspective. From a human hearing perspective, pitch, formants, and loudness are more perceptive features.

For a given signal, after decomposing it into components of varying frequencies, the component with the lowest frequency is the fundamental frequency. While the fundamental frequency (Teixeira, 2013), often denoted as F_0, is a physical measure, its perceptual counterpart is pitch. Beyond F_0, there are F_1 and F_2, referred to as formants. The human vocal tract, a complex system, possesses natural frequencies that amplify specific frequencies through resonance. Each of these natural frequencies is termed a formant.

As the signal $y(t)$ is periodic, we get the period T by taking the inverse of the frequency f. Then, the mean power P is defined by equation 2.2. The auditory threshold, represented as P_0, is the minimum power level at which sounds are audible to humans. The intensity L_{dB} is then defined as the logarithm of the signal's power P divided by

P_0, with the unit being decibels defined by the Equation 2.3. Intensity can also be defined concerning sound pressure. The coefficients differ based on whether the intensity is defined using power or pressure. Loudness is the perceptual counterpart to the physical quantity of intensity.

$$P = \frac{1}{T}\int_0^T y(t)^2 dt \tag{2.2}$$

$$L_{dB} = 10 \log \left(\frac{P}{P_0}\right) \tag{2.3}$$

In order to save or operate audio on a computer, it is necessary to convert the signals into digital form by making use of a sound card. The sound card incorporates an Analog-to-Digital Converter (ADC) and a Digital-to-Analog Converter (DAC). The conversion process comprises two stages: sampling and quantization.

Sampling involves converting a continuous-time signal into a discrete-time signal by taking amplitude values at a fixed frequency, known as the Sampling Rate (sr). The Nyquist frequency concept states that to reconstruct a periodic signal, it should be sampled at least twice in every period.

Quantization is the second step, where sampled amplitude values, which are real numbers, are converted into integers to save space and bandwidth. However, this process can lead to a loss of precision. There are different types of quantization based on the number of levels used to represent amplitude.

2.2 Feature Extraction

A feature is a measurable property or characteristic of an observed phenomenon. The crucial aspect of a feature is its ability to be measured. The effectiveness of a feature depends on the specific problem it addresses. In many machine-learning scenarios, raw data is not directly fed into models. Instead, features are extracted from the data, which machine learning models use to make predictions. For instance, the input is audio in

Speaker Recognition, and the output is a speaker embedding.

While some audio features, like fundamental frequency (F_0) and formants (F_1, F_2), can be computed based on entire audio, they are only sometimes suitable due to their global nature. Since audio signals are often non-stationary, using global features can result in local information loss. Short-time analysis is introduced to address this, where audio is divided into small segments or frames, typically ranging from 10 to 30 ms. These frames are believed to be approximately stationary (Naik, 1990).

Creating frames from audio samples involves considering frame size and frame step. The frame size is a frame's duration, while the frame step is the temporal distance between two consecutive frames. Avoiding introducing artificial discontinuities at the frame edges is essential when analyzing frames. Window functions like the Gaussian, Hanning, and Hamming windows are used to mitigate this. After this process, features from a windowed frame can be extracted (Sahidullah & Saha, 2013).

The Hanning and Hamming windows are closely related, being derivatives of the cosine function. Their difference lies in the constant α in Equation 2.4, where N is the total number of points in the window and n ranges from 0 to $N-1$. For the Hanning window, α is set at 0.5, while for the Hamming window, it is 0.46. Figure 2.2 has a comparison between them.

$$w[n] = (1-\alpha) - \alpha \cos \frac{2\pi n}{N-1} \tag{2.4}$$

When applying a Fourier transform to a signal frame, addressing the discontinuities that can arise is essential. In order to do that, the window function is used, multiplying each sample in the frame by a specific weight. This weight is designed to decrease towards the edges of the frame, almost zero, while remaining constant near the center, close to one. This approach ensures that the core content of the frame remains intact while minimizing edge discontinuities.

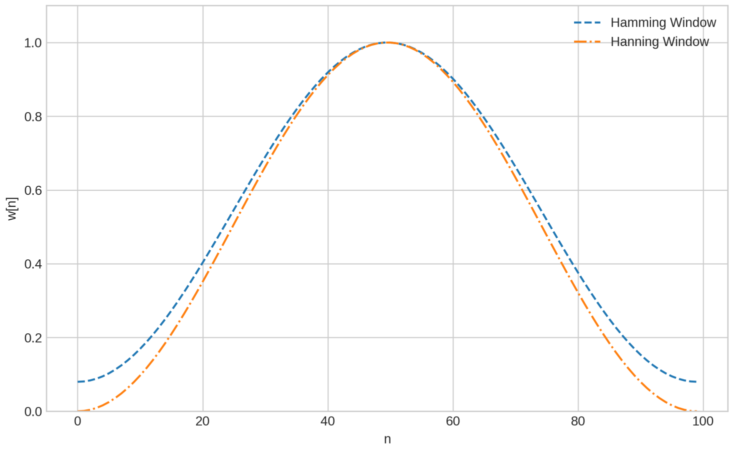

Figure 2.2: Hamming and Hanning window comparison.

Features can be divided into two categories: time-domain features and frequency-domain features. Time domain features are derived directly from the waveform, eliminating the need for a Fourier transform. On the other hand, frequency domain features necessitate a Fourier transform, transitioning from the time domain to the frequency domain.

Given a discrete audio signal, the samples within a windowed frame are denoted by $x[n]$, where N signifies the total sample count in that frame and n ranges from 0 to $N-1$. The Root Mean Square (RMS) measures the power of the audio signal. It does so by squaring its amplitude values, averaging them, and extracting the square root. This is defined by Equation 2.5.

$$RMS = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1} x[n]^2} \tag{2.5}$$

The Zero Crossing Rate (ZCR) is a time-domain feature that measures how often a signal changes its sign (Teixeira, 2013). It is defined by Equation 2.6 and is significant because it correlates with frequency. A higher frequency signal will have more zero crossings, resulting in a higher zero crossing rate.

$$ZCR = \frac{1}{N-1}\sum_{n=1}^{N} \frac{|sign(x[n]) - sign(x[n-1])|}{2} \tag{2.6}$$

Frequency domain analysis methods vary depending on the nature of the signal. Factors such as continuity, periodicity, and finiteness determine the specific method. In speech processing, short-time analyses require the Discrete Fourier Transform (DFT) to transform a sequence of time-domain samples into its frequency-domain counterpart.

A crucial aspect of DFT is its computational complexity of $O(n^2)$, which can be prohibitive for many applications, but this can be mitigated by using Fast Fourier Transform (FFT), a more efficient algorithm that reduces this complexity to $O(n \log n)$. However, FFT requires the input size to be a power of two, which is not always true in real-world scenarios. Zero-padding can be used to make the input length a power of two, making it suitable for FFT without introducing any discontinuities.

Short-Time Fourier Transform (STFT) is a technique that combines framing, windowing, and FFT to analyze signals. The output of STFT, known as a spectrogram, offers a time-frequency representation of signals, capturing how their frequency content evolves. Alongside STFT, other techniques like Chroma STFT categorize the spectrum's magnitude into pitch-specific bins (Alnuaim et al., 2022). Metrics such as the Spectral Centroid (SC) and Spectral Bandwidth (SBW) highlight the spectrum's "center of mass," indicating a sound's brightness, and describe the spread of power across frequencies, respectively (Hosseinzadeh & Krishnan, 2007).

The Cepstrum, often humorously referred to by reversing the letters of the word "spectrum," offers a deeper dive into the periodic patterns within the spectrum itself. The process involves taking the logarithm of the spectral magnitude and then applying the Inverse Fourier Transform, as shown in Equation 2.7.

$$C(y(t)) = F^{-1}[\log(F[y(t)])] \tag{2.7}$$

Mel Frequency Cepstral Coefficients (MFCCs) are cepstral coefficients derived from the Discrete Cosine Transform of a Mel-scaled Log Power Spectrum of an audio signal (J. Fernandes et al., 2019), as shown in Figure 2.3. MFCCs provide a compact representation of the sound's frequency characteristics. By utilizing the Mel scale, MFCCs capture the non-linear human ear perception of frequencies, making them particularly attuned to nuances in human speech. These coefficients encapsulate how the frequency components of a sound signal are constructed, offering a robust and discriminative feature set for various audio analysis tasks.

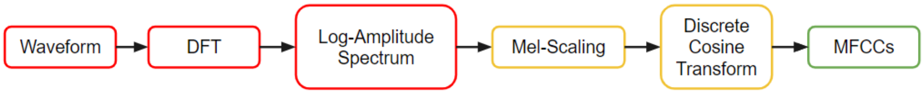

Figure 2.3: MFCCs calculation.

2.3 Speaker Recognition

Speaker Recognition is a subcategory of voice identity techniques that answers the question: "Who is speaking?". It can be seen as a core of voice identity techniques, as many require a SR model. Under Speaker Recognition, there is a range of techniques, including Identification, Verification, Detection, Segmentation, Clustering, and Diarization (Mohd Hanifa et al., 2021), as shown in Figure 2.4.

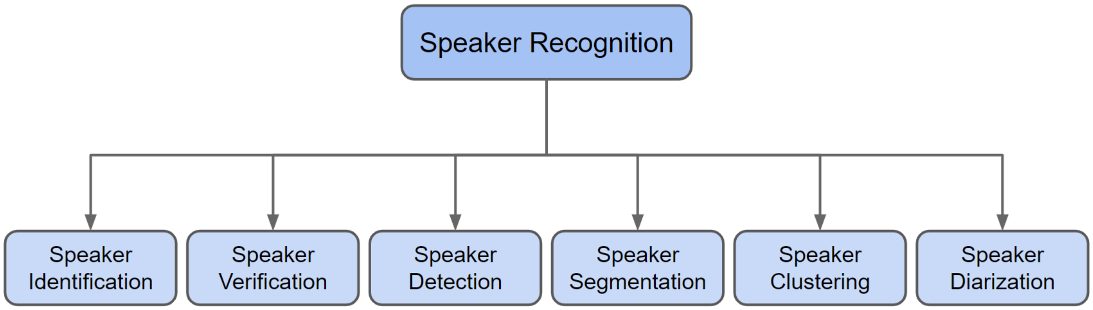

Figure 2.4: Categories of Speaker Recognition.

The Speaker Identification task categorizes an unknown voice as belonging to one of a set of S speakers. Speaker Verification involves determining if the unknown voice matches a specific speaker (Doddington, 1985). Speaker Detection involves correctly identifying the target speaker's speeches when presented together with the testing speeches (“Speaker

Detection", 2009). Speaker Segmentation involves identifying where a speaker changes in an audio stream. Speaker Clustering involves grouping multiple utterances presented to the system. Speaker Diarization involves automatically splitting audio into speaker segments and determining which segments are uttered by the same speaker (Docio-Fernandez & Garcia-Mateo, 2009).

In speech processing, researchers commonly use Speaker Recognition and Speech Recognition to analyze spoken language, but they address distinct challenges. While Speaker Recognition identifies who is the speaker, Speech Recognition determines the content of what was spoken. A Speaker Recognition system should identify the speaker regardless of the words, and a Speech Recognition system should decipher the content irrespective of the speaker (Mohd Hanifa et al., 2021).

Speaker Recognition is only possible because of the uniqueness of each individual's voice. Factors such as the morphology of speech organs such as vocal folds and vocal tract, age, gender, linguistic accent, personal preferences, and myriad other elements contribute to this uniqueness. In practical applications, SR mainly bifurcates into two tasks: Speaker Verification (SV) and Speaker Identification (SI).

Speaker Verification is a binary classification problem that aims to authenticate a single candidate's identity. In a classic workflow, the candidate's voiceprint is first obtained through enrollment. Subsequent audio samples are then compared to this voiceprint to generate a score. A decision is made based on a predefined threshold, determining whether the audio matches the target speaker (Doddington, 1985).

Speaker Identification involves discerning the identity from a pool of N candidates, making it a multiclass classification problem (Sharma & Bansal, 2013). This task can be further subdivided into closed-set and open-set Speaker Identification based on the assumption of the speaker's presence in the candidate set (Jahangir et al., 2021). In an intuitive workflow, each candidate is enrolled separately, and the runtime audio is verified against each voiceprint. Then, the candidate with the highest score is chosen as the speaker (Bai & Zhang, 2021). However, as the number of candidates increases, the complexity of the task escalates due to the potential similarity between voices (Jia et al., 2021).

SR tasks can be divided into three categories based on the number of candidate speakers and the context of the spoken content. These categories are text-dependent, text-independent, and text-prompted speaker recognition (Mohd Hanifa et al., 2021).

Text-dependent speaker recognition assumes that the spoken content remains consistent or has minimal variations. This constraint simplifies the task as the phonetic variations are limited, allowing for modeling only a few phonemes in the fixed text. On the other hand, text-independent speaker recognition does not rely on specific spoken content, which means users can utter any phrase, and the system should still identify the speaker (Büyük, 2011).

Text-prompted speaker recognition was introduced to counter replay attacks. In this method, the system prompts the user with a unique phrase to speak each time, ensuring that recorded voice samples from previous sessions cannot be used maliciously. The system's security relies on Speaker Verification and speech recognition to ensure that the spoken content matches the enrolled user and the prompted phrase.

Speaker Recognition systems typically operate in three stages: training, enrollment, and recognition. During the training stage, a machine learning model is developed using training data, and this model transforms audio features into a unique speaker embedding. In the enrollment stage, speakers provide audio samples, which are processed to create a consolidated speaker profile through aggregation. During the recognition stage, real-time audio is compared against these profiles to identify or verify the speaker, where a similarity score is used (Bai & Zhang, 2021).

The similarity score assumes that an embedding vector represents every speaker's utterance. Let us denote two such embeddings as e_1 and e_2. The objective is to evaluate how similar these embeddings are to each other. A high similarity score suggests that the embeddings are likely from the same speaker, while a lower score indicates different speakers. The most prevalent similarity metric employed is the cosine similarity, represented by Equation 2.8.

Ground truth	Prediction	Case
Same Speaker	Same Speaker	True Accept (TA)
Same Speaker	Different Speaker	False Reject (FR)
Different Speaker	Same Speaker	False Accept (FA)
Different Speaker	Different Speaker	True Reject (TR)

Table 2.1: Pair-Based Evaluation possibilities.

$$\cos\theta = \cos(e_1, e_2) = \frac{e_1 \cdot e_2}{\|e_1\|_2 \cdot \|e_2\|_2} \tag{2.8}$$

The cosine similarity between two vectors equals the cosine value of the angle θ they form. The range of cosine similarity varies between -1 and 1, where a higher value indicates a more significant similarity. Expressly, a cosine similarity of 1 signifies that the vectors are in the same direction, whereas a value of 0 indicates that the vectors are orthogonal. On the other hand, a value of -1 reveals that the vectors are in opposite directions.

Another method to determine similarity is the Euclidean distance, which measures the difference between two vectors, represented by Equation 2.9. Two important considerations with this measure are its sensitivity to vector length and its inherent nature: smaller distances mean a greater similarity.

$$d(e_1, e_2) = \|e_1 - e_2\|_2 \tag{2.9}$$

Beyond these parameter-based measures, model-based similarity scores are also notable. Here, a neural network model called the decision network is fed with two vectors and produces a similarity score. This model requires training data to optimize its parameters.

Several key concepts and metrics are used to measure system performance in SV. The evaluation consists of measure trials. In pair-based evaluation, a trial compares one test utterance against one enrollment utterance. Each trial has four possible cases based on the ground truth and the prediction, as shown in table 2.1.

By knowing the number of FAs and FRs, we can compute the False Accept Rate

(FAR) and False Reject Rate (FRR). FAR is the probability of accepting a different speaker, and FRR is the probability of rejecting the true speaker. However, evaluating SV systems is challenging due to threshold sensitivity, making FAR and FRR unsuitable for direct comparisons across systems (Mingote et al., 2019).

The Receiver Operating Characteristic (ROC) curve is used to overcome this limitation. This curve plots the True Accept Rate against the FAR. A curve in a 2D plane is obtained by sweeping through different thresholds. A system's efficacy is measured by how close its ROC curve is to the top-left corner (Fawcett, 2006). The area under this curve (AUC) quantifies system performance, with values closer to 1 indicating better systems, as shown in Figure 2.5.

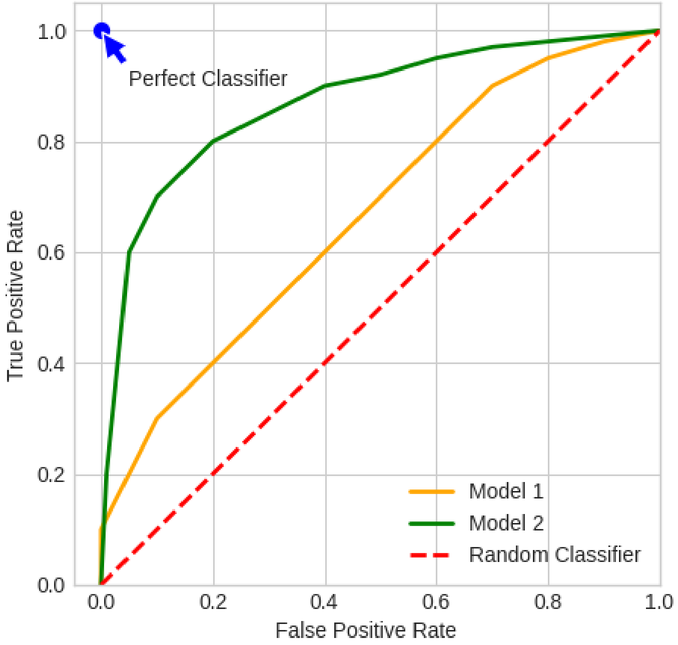

Figure 2.5: ROC Curve.

Equal Error Rate (EER) is a commonly used metric in SV systems. As the system's threshold changes, there is usually a tradeoff between a False Acceptance Rate (FAR) and a False Rejection Rate (FRR). EER is the rate where FAR equals FRR. So, in general, equal error rate is an approximation of total error rates. The smaller the EER, the more proficient the system is.

For Speaker Identification, the metric of choice is often the identification accuracy. Unlike Speaker Verification metrics based on paired trials, Speaker Identification accuracy relies on tuples. Each tuple in this context encompasses the utterance being evaluated and a set of enrolled speakers, which includes the genuine speaker. The core idea is to match the utterance against the roster of speakers and check if it gets correctly identified as a genuine speaker. The identification accuracy is then calculated as the fraction of tuples where the system correctly identifies the speaker. Naturally, the identification challenge amplifies as the dataset grows, potentially decreasing accuracy.

Chapter 3

Literature Review

Speaker Recognition has made significant progress in recent years due to the development of new techniques and the increased availability of large datasets. As a result, numerous research articles have been published, exploring various aspects of SR, ranging from fundamental concepts and methodologies to the latest state-of-the-art models.

Kersta (1962) conducted one of the first academic research on voice identification. It selected ten high-frequency words and enlisted 15 male and 10 female human speakers to pronounce these words in different combinations. A panel of under-18 native female students was chosen to interpret the voiceprints. These students underwent a five-day training session on voiceprint reading, and then they were tasked with identifying the voiceprints of the experiment subjects. The identification results were meticulously recorded and assessed. The study used spectrograms and human interpretation to achieve 97% accuracy.

Speech signals' variations can be captured through statistical methods, as explained by Bricker et al. (1971). Using a mean vector to represent the voiceprint simplifies comparison but may not yield the best results. Enhancements such as standard deviation and segment-wise statistical calculations offer greater precision in analyzing speech signals than human-reading methods.

Among these early approaches, Gaussian Mixture Model (GMM) gained popularity and served as the basis for over a decade before the advent of deep learning (Bai &

Zhang, 2021). The GMM models were widely used in Computer Vision, Speech Recognition, and Speaker Recognition, as they could approximate complex distributions using a combination of simple Gaussian distributions (D. Reynolds & Rose, 1995).

Recent years have shown significant advancements in SR through the use of deep learning (Lei et al., 2014; Snyder et al., 2018). Since the early 2010s, deep learning has become the most widely used approach in machine learning (Mohd Hanifa et al., 2021). This concept has similarities to GMM, using neurons to approximate complex functions.

Recurrent neural networks have been helpful in speech signal processing due to their ability to model sequence data effectively (Hori et al., 2018). Deep learning models are scalable and efficient when working with large datasets, and specialized hardware such as GPUs and TPUs are available for acceleration. Since 2014, deep learning models have been responsible for numerous advancements in SR.

Mohd Hanifa et al. (2021) conducted a comprehensive survey on models for speech recognition, addressing significant issues such as background noise, insufficient data, and model attacks. Their study provides a detailed overview of the development of speech recognition models, highlighting the technological advancements made in the field. The authors discuss various preprocessing techniques, common features extracted in the field, potential model types and classifiers, and application areas.

In a study conducted by Kral (2010), three Discrete Wavelet Transform (DWT) with different coefficients were employed, using both a Multilayer Perceptron (MLP) and a GMM as classifiers. These models achieved 98% and 99% accuracy rates, but the MLP could be trained using audio samples that were half the duration of those used for the GMM. In the subsequent years, the focus of research shifted toward neural networks, such as the Fuzzy Min-Max Neural Network (FMMNN) (Jawarkar et al., 2011). Additionally, variations of the GMM model (Krishnamoorthy et al., 2011) and comparisons with the Hidden Markov Model (HMM) were explored, with all these methods leveraging MFCCs vectors as their input (Tolba, 2011).

Researchers have made various attempts to enhance the quality upon the MFCCs features, including the investigation of alternative approaches such as Normalized Dynamic

Spectral Features (NDSFs) and Linear Prediction Cepstral Coefficients (LPCCs) to determine whether they can provide better feature representation than MFCCs (Chougule & Chavan, 2015).

The field has shifted towards using x-vectors, known for their ability to handle noise effectively (Villalba et al., 2019). According to recent surveys, a combination of features, such as MFCCs and Power Normalized Cepstral Coefficients (PNCC) (P & M, 2020), as well as Linear Discriminant Analysis (LDA) and MFCCs (Zergat et al., 2018), have also been utilized to improve these features.

Nagrani et al. (2017) introduced VoxCeleb, a large-scale Speaker Identification dataset curated from open-source media with hundreds of thousands of 'real-world' utterances from over 1,000 celebrities. Then, it conducts active speaker verification through a Convolutional Neural Network. The researchers also applied and compared various state-of-the-art Speaker Identification techniques on this dataset to establish baseline performance.

In a subsequent study, Chung et al. (2018) presented an enhanced pipeline for curating a Speaker Identification dataset from YouTube videos. The resulting dataset was multilingual and more extensive than their previous work, with over a million utterances from over 6,000 speakers. They compared CNN models and training strategies that can effectively recognize identities from voice under diverse conditions. These models exceeded the performance of previous works.

Then, Nagrani et al. (2020) focused on the challenge of SR in noisy and unconstrained environments. They developed and compared various CNN architectures, aggregation methods, and training loss functions to effectively recognize speaker identities under diverse conditions. The models trained on VoxCeleb significantly outperformed previous works in the domain.

Chapter 4

Methodology

4.1 Dataset

This work's final goal is to create a speaker recognition system that will enable access to CeDRI's laboratory. We employed the CeDRI dataset as our primary data source to achieve this. This data comprises a distinctive collection of speeches delivered by members of the CeDRI community. The CeDRI dataset comprises 169 distinct utterances collected by text-reading from 32 Portuguese-speaking individuals. These utterances are single-channel, 16-bit streams between two sampling rates of 32kHz or 44.1kHz. The durations vary from 2.7 seconds to as long as 21.5 seconds, as shown in Figure 4.1.

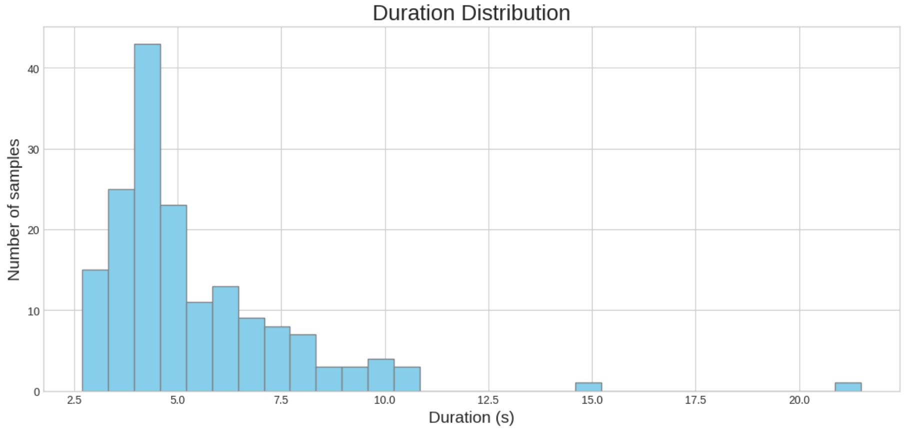

Figure 4.1: Histogram showcasing the duration distribution of audios.

The speakers included in the dataset were from diverse backgrounds and dialects. Approximately 69% of the speakers were male, while 31% were female. The dataset comprised various accents from different Portuguese-speaking regions, such as Portugal, Angola, São Tome, Cabo Verde, Mozambique, Brazil, and Spain. Most speakers were in their early twenties, making the dataset highly representative of young adult speech. For a detailed overview of the dataset's gender, country, and age distribution, see Figure 4.2.

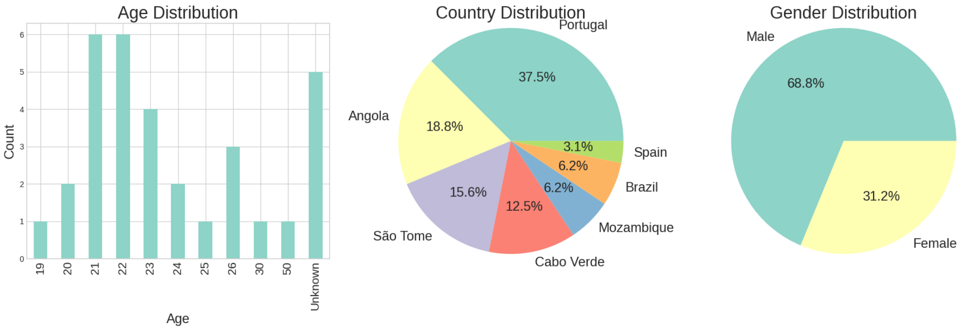

Figure 4.2: Distribution Analysis (age, country, and gender) within the CeDRI dataset.

To standardize our dataset, we performed two strategies. In the first one, a silence removal filter eliminates audio sections below a 30db threshold, resulting in audios with a range duration of 1.824s to 10.358s. The second was a trim filter, which removed silence from the beginning and end of the audio, also at a 30db threshold, resulting in audios ranging from 1.888s to 12.608s. Figure 4.3 shows both strategies, silence removal in the upper flow where each pair of red bars indicates where there is no silence, and trim in the lower flow, where the pair of bars is delimited when the audio utterance begins and ends.

We segmented and resampled the audio files of both strategies to ensure a uniform duration and 16kHz sampling rate. To achieve this, we found the minimal duration of all audio files for each strategy. Next, the silence removal strategy divided the audio durations to a length of 1.824s, while the trim strategy adjusted the audio durations to half the minimum duration, 0.944s. This created two versions of the original dataset: the Framed Silence Remove (FrSR) dataset, derived from the first strategy, and the Framed Trim (FrT) dataset, derived from the second strategy.

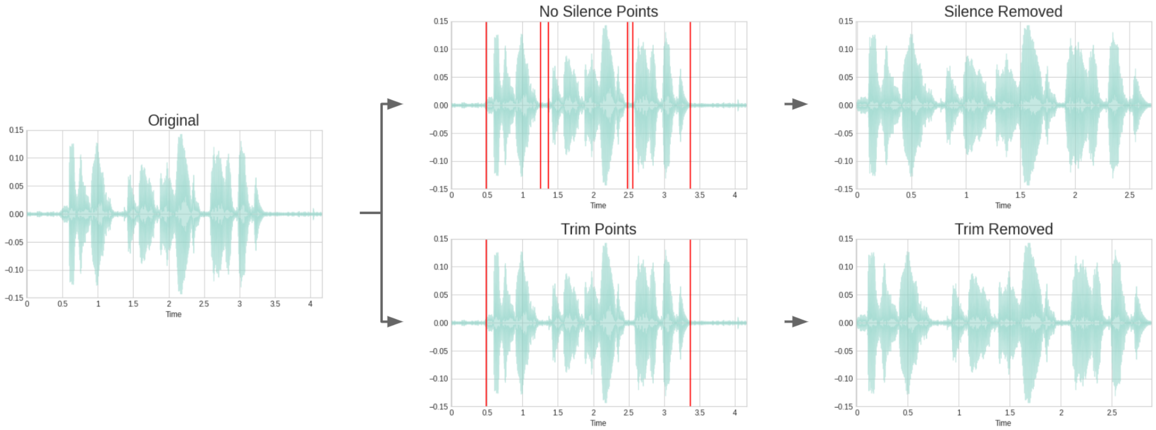

Figure 4.3: CeDRI audio cleaning process.

We needed more data to train our convolutional models than our datasets could provide, so we incorporated the VoxCeleb1 and VoxCeleb2 datasets into our training data. The VoxCeleb1 dataset contains over 100,000 utterances from 1,251 unique speakers, while VoxCeleb2 contains over a million utterances from over 6,000 speakers. Both datasets were collected from open-source media and included speakers from diverse backgrounds, ethnicities, accents, professions, and age groups. The dataset mainly consists of English, but it may include German and French. They contain real-world noise like background chatter, overlapping speech, laughter, and various room acoustics. The audios have a single-channel, 16-bit stream at a 16kHz sampling rate.

For each dataset, 70% of the audio files were allocated for training, 10% for validation, and 20% for testing. This distribution was consistently applied to every speaker, ensuring that each speaker's audio files maintained the same proportion, regardless of the varying total number of audio files they had across datasets. The division is detailed in Table 4.1.

Table 4.1: Separation of data in Framed Trim, Framed Silence Remove, VoxCeleb 1, and VoxCeleb 2 datasets

Dataset	**Total**	**Train**	**Test**	**Validation**
FrT	724	523 (72.24%)	145 (20.03%)	56 (7.73%)
FrSR	289	197 (68.16%)	60 (20.76%)	32 (11.07%)
VoxCeleb 1	148,642	104,590 (70.36%)	29,738 (20.01%)	14,314 (9.63%)
VoxCeleb 2	1,092,009	766,917 (70.23%)	218,373 (20.00%)	106,719 (9.77%)

4.2 Features

This section will discuss the audio features employed in this work. According to J. Fernandes et al. (2019), J. F. T. Fernandes et al. (2023), and Teixeira and Freitas (2003), these features have shown good performance and are commonly used in the field.

- **MFCCs**, Δ **MFCCs**, Δ^2 **MFCCs**: MFCCs are a set of coefficients that capture the short-term power spectrum of a sound. These coefficients are derived by splitting the audio signal into frames, applying the Fourier transform to each frame, taking a logarithmic transformation, and then using Mel-scaling filters. The resulting coefficients are then processed through a Discrete Cosine Transform to yield the MFCCs. The Δ MFCCs are calculated by taking the difference between an MFCC coefficient in a frame and the previous frame, and they provide the rate of change of the MFCCs. Lastly, the Δ^2 MFCCS represent the second-order derivatives of the MFCCs and provide the rate of change of the Δ MFCCs, which are calculated similarly to the Δ MFCCs. Figure 4.4 is an example of these coefficients.

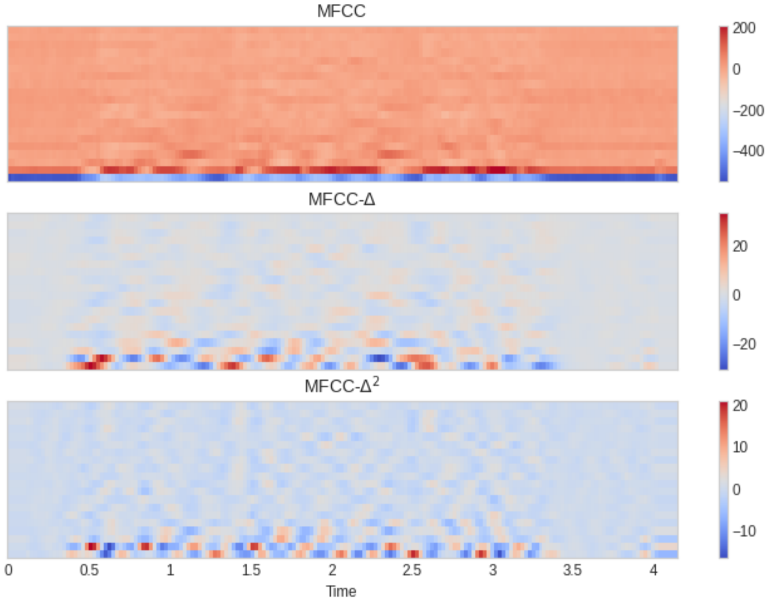

Figure 4.4: MFCCs, Δ MFCCs, Δ^2 MFCCs.

- **Chroma STFT**: The Chroma STFT is a method to extract pitch class information from audio signals. This technique involves using the STFT to analyze the frequency content of a signal over time. The frequencies obtained are then mapped to the 12 pitch classes of Western tonal music. The final output is a matrix where each row represents a pitch class, and each column represents a specific time interval. The values in each cell indicate the energy of a particular pitch class at a specific time. This method helps identify the pitch classes in an audio signal (Alnuaim et al., 2022). Figure 4.5 is an example of this feature.

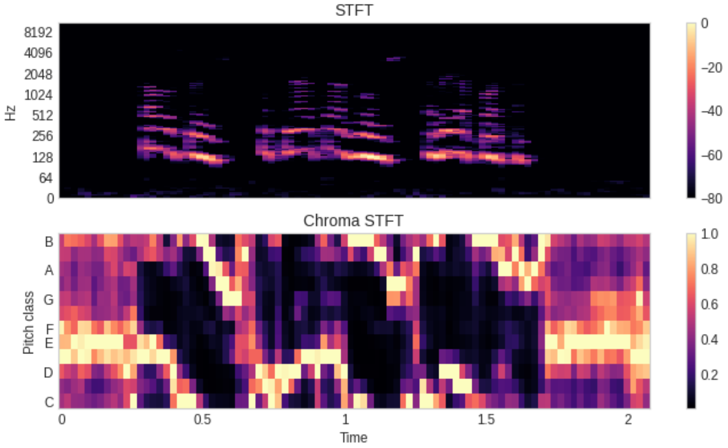

Figure 4.5: Chroma STFT.

- **RMS**: measures the audio signal's power. It is calculated by splitting the signal in frames, squaring the amplitude values, averaging them, and then taking the square root, as indicated in Equation 2.5. Figure 4.6 is an example of this feature.

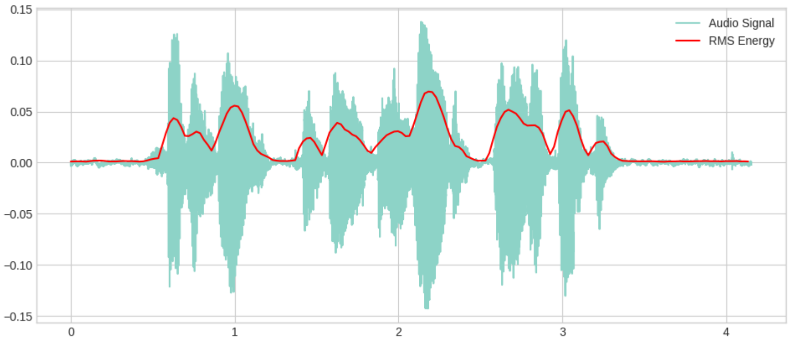

Figure 4.6: RMS.

- **Zero Crossing Rate (ZCR)**: Calculates the number of times a signal shifts from positive to negative and vice versa. It is calculated for each frame in an audio signal by Equation 2.6. Figure 4.7 is an example of this feature.

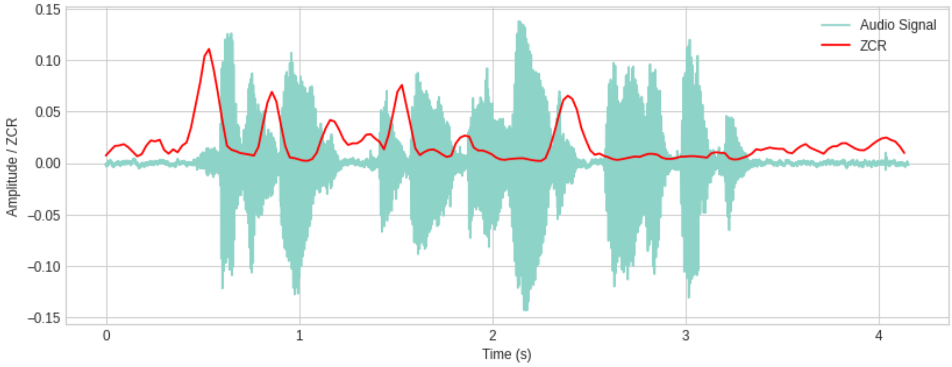

Figure 4.7: Zero Crossing Rate example.

- **Spectral Centroid (SC)**: Measures that indicate the "center of mass" or the "brightness" of a sound. It indicates where the center of the sound's energy lies in the frequency spectrum. Figure 4.8 is an example of this feature.

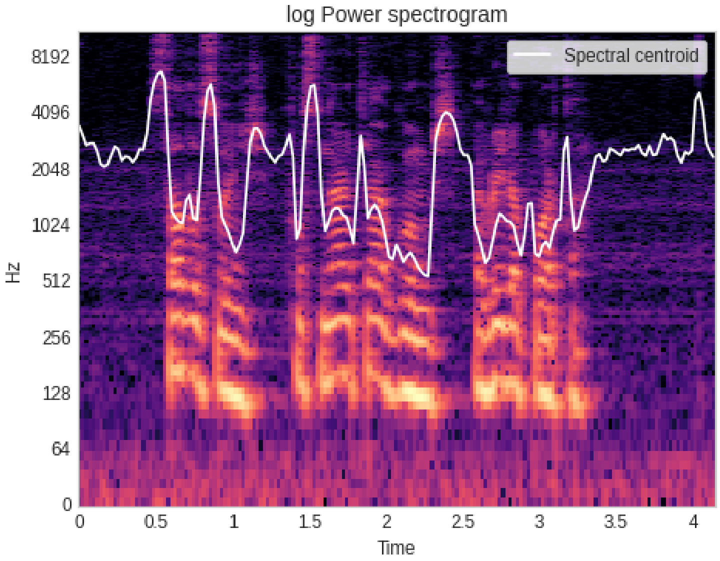

Figure 4.8: Spectral Centroid

- **Spectral Bandwidth (SBW)**: Measurement of spectrum width, determines power spread across frequencies. A sound with multiple harmonics typically has a larger spectral bandwidth than a pure tone. Figure 4.9 is an example of this feature.

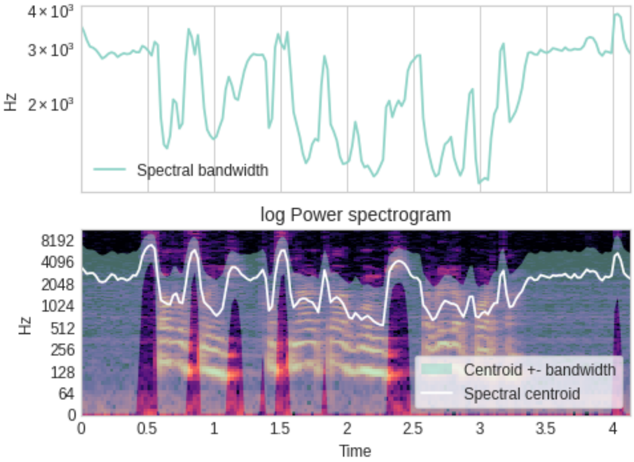

Figure 4.9: Spectral Bandwidth

- **Spectral Rolloff (SRO)**: Measures the frequency below 85% percentage total magnitude distribution of the spectrum is contained. It distinguishes between harmonic content and noise in a signal. Figure 4.10 is an example of this feature.

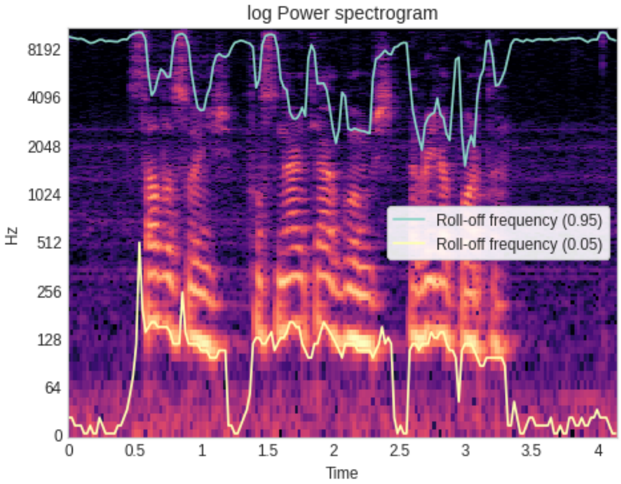

Figure 4.10: Spectral Rolloff.

- **STFT**: It is a Fast Fourier Transform applied to short intervals and is used to analyze a signal's frequency and phase content over time. By examining local sections of the signal, we can determine each segment's sinusoidal frequency and phase content, providing valuable information about the frequency content.

  Specifically, for this feature extraction, we configured the STFT with an FFT size of 1024, which determines the length of the windowed signal after zero-padding. The stride between the two segments was set to correspond to 10 ms, considering a sampling rate of 16kHz. This setting specifies the number of audio samples between adjacent STFT columns. The length of the frame, defining the size of the windowed frame, was set to 25 ms. The window function used for this process was the Hanning. And at last, we also applied magnitude normalization to the results of the STFT. Figure 4.11 is an example of this feature.

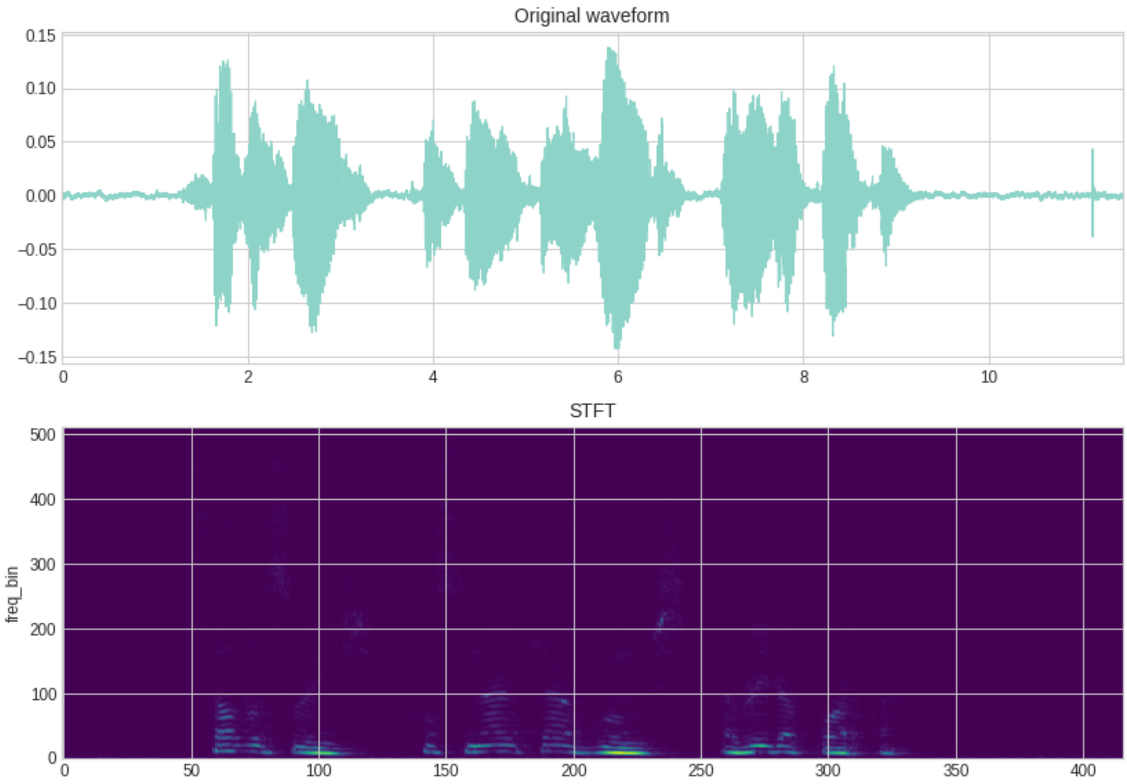

Figure 4.11: Short-Time Fourier Transform Example.

- **Spectrogram**: It is a visual representation of the frequency spectrum in a sound or signal over time. For this work, we used power spectrograms illustrating each frequency component's power over time. It is calculated by computing the square magnitude of the STFT of the signal.

  Specifically, the configurations for this feature are the same as those for the STFT feature, with an FFT size of 1024, the stride between the two segments set to 10 ms, the length of the frame set to 25 ms, using Hanning window function, and magnitude normalization applied to the results. The difference is that the Spectrogram is in the power of 2, and after these calculations, it is transformed from a power scale to a decibel scale. Figure 4.12 is an example of this feature.

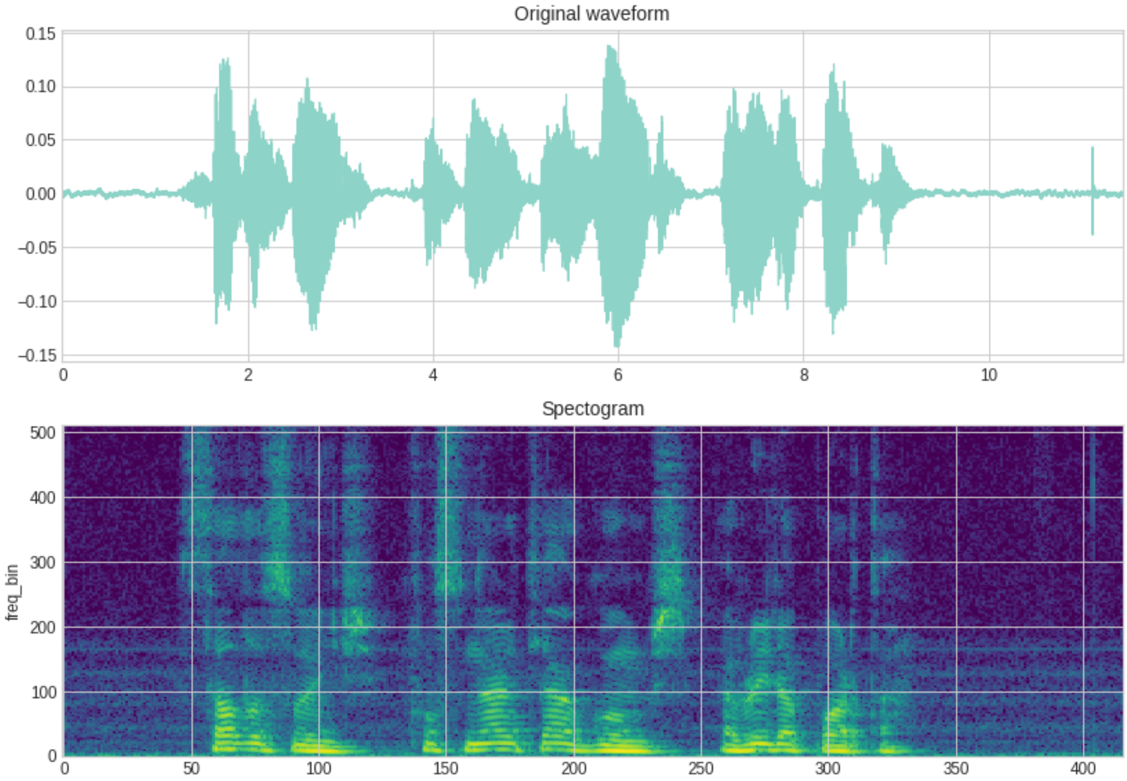

Figure 4.12: Spectogram Example.

4.3 Models

This section explains the models used in this work and details their characteristics and implementations.

4.3.1 GMM

The Gaussian Mixture Model (GMM) (D. Reynolds & Rose, 1995) is a statistical model capable of identifying subpopulations within a dataset without requiring explicit labels. It uses as a core function the Gaussian Distribution, also known as Normal Distribution, denoted by the probability density function shown in Equation 4.1 where the parameter μ represents the mean, and the parameter σ the standard deviation.

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{4.1}$$

When dealing with multiple random variables, we could use multivariate Gaussian distribution, with probability density function denoted by Equation 4.2 where $\boldsymbol{x}$ and $\boldsymbol{\mu}$ are respectively a random and mean K-dimensional vectors and $\boldsymbol{\Sigma}$ is a covariance matrix with dimension $K \times K$.

$$p(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^K |\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})} \tag{4.2}$$

The Central Limit Theorem states that the sum of independent and identically distributed random variables approximates a Gaussian distribution. Speaker recognition may involve variables with Gaussian characteristics, but GMM represents complex distributions as a sum of simpler Gaussian distributions. A GMM is essentially a weighted sum of M Gaussian functions (D. Reynolds, 2009), as given by the Equation 4.3.

$$p(\boldsymbol{x}|\lambda) = \sum_{i=1}^{M} w_i \cdot g(\boldsymbol{x}|\mu_i, \boldsymbol{\Sigma_i}) \tag{4.3}$$

Where $\boldsymbol{x}$ is a K-dimentional continuous-valued data vector, w_i is the mixture weight of the ith Gaussian component, and $g(\boldsymbol{x}|\mu_i, \boldsymbol{\Sigma_i})$ are the component Gaussian densities. Each component density is a K-variate Gaussian function shown in the Equation 4.4. The mixture weights w_i are equal or greater than zero and satisfy the constraint of the Equation 4.5. To represent a GMM, we consider mixture weights, mean vectors, and covariance matrices from all component densities. These parameters are collectively represented by Equation 4.6.

$$g(\boldsymbol{x}|\mu_i, \boldsymbol{\Sigma_i}) = \frac{1}{\sqrt{(2\pi)^K|\boldsymbol{\Sigma_i}|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_i})^T \boldsymbol{\Sigma_i}^{-1}(\boldsymbol{x}-\boldsymbol{\mu_i})} \tag{4.4}$$

$$\sum_{i=1}^{M} w_i = 1 \tag{4.5}$$

$$\lambda = \{w_i, \mu_i, \Sigma_i\} \quad i = 1, ..., M. \tag{4.6}$$

The Maximum Likelihood Estimation (MLE) is commonly employed to determine the parameters λ. For a set of parameters λ and a dataset $\boldsymbol{X}$, where $\boldsymbol{X} = \{\boldsymbol{x_1}, ..., \boldsymbol{x_T}\}$, MLE maximizes the probability $p(\boldsymbol{X}|\lambda)$ assuming independence between the vectors, p is denoted by Equation 4.7. The Expectation–Maximization (EM) algorithm starts with an initial set of λ parameters and estimates a new set $\dot{\lambda}$, such that $p(X|\dot{\lambda}) \geq p(X|\lambda)$. The new model then becomes the initial model for the next iteration, and the process is repeated until some convergence threshold is reached.

$$p(\boldsymbol{X}|\lambda) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|\lambda) \tag{4.7}$$

To perform Speaker Recognition using GMM, a common approach is to model each speaker with a mixture model. If a speaker has multiple utterances, acoustic features can be extracted from each utterance and used to estimate the parameters λ. After parameter estimation, each speaker can be represented using a distinct mixture model, as shown in Figure 4.13.

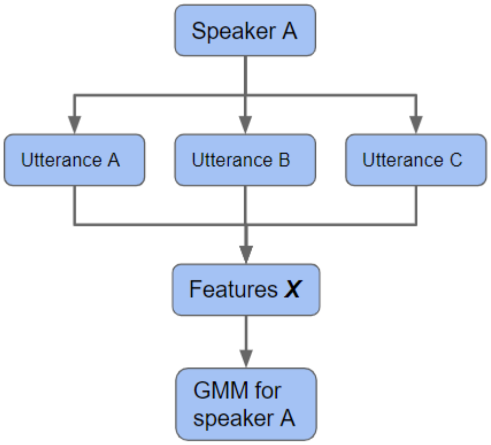

Figure 4.13: GMM modeling flow.

For Speaker Identification, given S candidate speakers and their associated parameters λ_S and acoustic features $\boldsymbol{X}$ extracted from a new audio sample, the objective is to identify the closest speaker between all S candidates. The likelihood of the features $\boldsymbol{X}$ given λ_S is computed for each speaker's GMM. The speaker s whose model provides the highest likelihood for $\boldsymbol{X}$ is identified as the best match using equation 4.8 (D. Reynolds & Rose, 1995).

$$s = \arg\max_{1\leq s\leq S} \ln p(X|\lambda_S) \tag{4.8}$$

For the SI task, we employed our CeDRI dataset, which included 32 speakers, each represented by a GMM model. Each model incorporated 23 Gaussian components and employed a diagonal covariance matrix. The models were fed with audio file features. We experimented with different combinations of features, such as MFCCs, Δ MFCCs, Δ^2 MFCCs, Chroma STFT, RMS, SC, SBW, SRO, and ZCR. The model generates an embedding from the extracted features in the audio input. The embedding is then matched against stored speaker profiles to generate dissimilarity scores. These scores are then thresholded to produce the final identification results.

An issue with this SI approach using GMM is that this process assumes a closed set, which means the identified speaker has to be among the noted speakers. However, to perform Speaker Verification to determine if the speaker is who they claim to be, another approach called the Universal Background Model (UBM) is necessary (D. A. Reynolds et al., 2000). This method utilizes a speaker-independent GMM model to represent alternative speakers or imposters. The problem is then reduced to testing two hypotheses:

- H_0: The utterance is from the hypothetical speaker S.
- H_1: The utterance is not from the hypothetical speaker S.

To calculate the model, we use the log-likelihood ratio as defined in Equation 4.9, where X is the feature vector extracted from the speech utterance. If $L(X) \geq \theta$, we accept Hypothesis H_0; otherwise, we accept Hypothesis H_1. Where θ is a threshold.

$$L(X) = \ln p(X|H_0) - \ln p(X|H_1) \tag{4.9}$$

Creating the UBM model involves combining training data from every speaker using the expectation-maximization algorithm. Once this collective model is established, it is fine-tuned to cater to individual speakers using the Bayesian Adaptation algorithm. This training process is illustrated in Figure 4.14. In this process, all models originate from the same UBM, ensuring a consistent foundation.

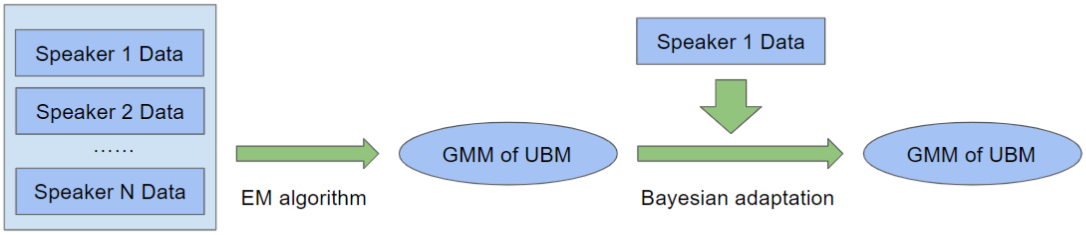

Figure 4.14: GMM-UBM training pipeline using Bayesian Adaptation algorithm.

We followed the approach proposed by Reynolds (D. A. Reynolds et al., 2000) for the Speaker Verification task. We used all the training data from the CeDRI-derived datasets to create a GMM-UBM. Then, we utilized the Bayesian Adaptation algorithm to adapt the UBM model for each speaker's specific data. As a result, we obtained 32 GMMs models, one for each speaker and a single UBM model.

4.3.2 Deep Learning (DL)

Deep learning is an extension of neural networks that relies on vast datasets and advanced computing capabilities and has become essential in speaker recognition and other applications like computer vision and speech recognition. Its dominance in the past decade can be attributed to significant hardware advancements, like GPUs and TPUs, the exponential increase in data availability from digital devices, and the evolution of software with frameworks like TensorFlow and PyTorch.

The concept of neural networks is inspired by the biological workings of neurons, the basic operational units in our biological neural systems. Despite their simplicity, they can perform complex tasks together. This understanding has driven the development of artificial neurons. The artificial neurons are the fundamental units of a neural network. An artificial neuron takes an input vector $\boldsymbol{x}$ of real numbers and applies a function $\boldsymbol{h}$, denoted by Equation 4.10. Here, $\boldsymbol{w}$ represents the weights with the exact dimensions as $\boldsymbol{x}$, while b is a scalar known as the bias. The function σ is called the activation function that emulates a biological neuron's on-off principle.

$$h(\boldsymbol{x}) = \sigma(\boldsymbol{w}^T\boldsymbol{x} + b) \tag{4.10}$$

Neural networks are complex systems that are designed by connecting multiple neurons. It works as a function represented by Equation 4.11, where $\boldsymbol{x}$ is the input, $\boldsymbol{y}$ is the output, and $\boldsymbol{\theta}$ represents the parameters. For instance, in the Equation 4.10 of an artificial neuron, the parameter $\boldsymbol{\theta}$ will encapsulate the weights $\boldsymbol{w}$ and bias b. By using these parameters, the network can infer the output $\boldsymbol{y}$ for a given input $\boldsymbol{x}$.

$$\boldsymbol{y} = f(\boldsymbol{x}|\boldsymbol{\theta}) \tag{4.11}$$

To fully utilize neural networks, we must train them using input $\boldsymbol{x}$ and corresponding output $\boldsymbol{y}$. The network predicts output for each input, and a loss function measures the accuracy of this prediction. The training aims to adjust the parameters $\boldsymbol{\theta}$ to minimize total loss across the dataset using algorithms like gradient descent and iteratively refining the parameters.

A feedforward neural network is a series of artificial neuron layers stacked sequentially, termed MLP. For a given input $\boldsymbol{x}$, the output $h(\boldsymbol{x})$ is derived as $\sigma(\boldsymbol{w}^T\boldsymbol{x} + b)$, where $\boldsymbol{w}$ is the weight vector and b is the scalar termed as bias. When multiple neurons use this input, their collective outputs can be denoted as a vector $\boldsymbol{h}_1$, it can be formulated as $\sigma(\boldsymbol{W}_1\boldsymbol{x}+\boldsymbol{b}_1)$, where $\boldsymbol{W}_1$ is the weight matrix and $\boldsymbol{b}_1$ is the bias vector. Similarly, another layer $\boldsymbol{h}_2$ can use $\boldsymbol{h}_1$ as its input. Denoting the input $\boldsymbol{x}$ as $\boldsymbol{h}_0$, a generic representation of the k^{th} layer in a feedforward neural network can be denoted by Equation 4.12, where $\boldsymbol{W}_k$ and $\boldsymbol{b}_k$ represent the weight matrix and bias vector of the k^{th} layer, respectively.

$$\boldsymbol{h}_k = \sigma(\boldsymbol{W}_k\boldsymbol{h}_{k-1} + \boldsymbol{b}_k) \tag{4.12}$$

When handling high-dimensional data, CNNs are more suited than feedforward networks. While a feedforward network can quickly have an overwhelming number of parameters for this kind of data, such as Spectograms, CNNs efficiently handle this by sharing parameters.

CNNs consist of multiple convolutional layers, each containing many convolutional kernels. These kernels are applied across different input locations, producing an output for each location. The convolution operation involves element-wise multiplication of the input with the kernel and then summing up the products to derive the convolutional operation's output for that location. Various outputs emerge while shifting the kernel across the input, but the kernel remains constant. The output dimension depends on the kernel number, size, and stride. The Pooling Layer is positioned between convolutional layers. Pooling aggregates multiple numbers into a single number by extracting the maximum element (max pooling) or averaging values (average pooling). This process is illustrated in Image 4.15.

Our work employed two neural network models: the Visual Geometry Group Medium (VGG-M) and Residual Neural Network 50 (ResNet50). We chose these models due to

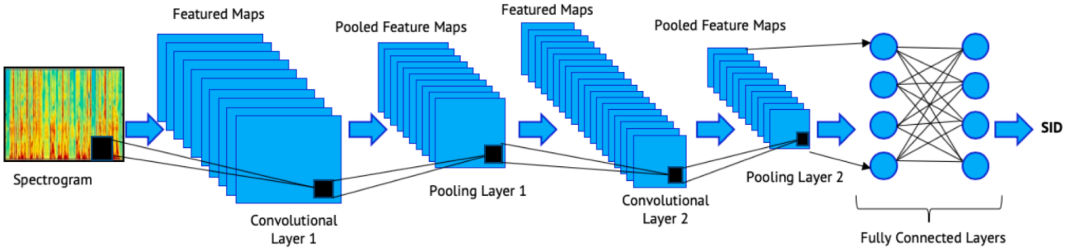

Figure 4.15: Convolutional Neural Network Architecture. Extracted from (Jahangir et al., 2021).

their high performance on the VoxCeleb1 (Nagrani et al., 2017) and VoxCeleb2 (Chung et al., 2018) datasets. Both models are CNN architectures that efficiently classify audio data. The models are insensitive to the temporal position and sensitive to frequency, making it an ideal choice for speech data analysis.

In a research paper by Nagrani et al. (2017), the VGG-M model was further improved by replacing its fully connected layer (fc6) with two layers: a fully connected layer of size 9×1 and an average pool layer with a support of size $1 \times n$, where n is the length of the input speech segment. This modification allowed the network to focus on the temporal aspects of the audio signal rather than the frequency domain while also maintaining the output dimensions of the original fully connected layer and reducing the number of network parameters. The architecture of VGG-M is presented in Table 4.2.

Table 4.2: VGG-M Architecture.

Layer (type)	**Support**	**Input Dim**	**Output Dim**	**Stride**
conv1	7×7	1	96	2×2
mpool1	3×3	-	-	2×2
conv2	5×5	96	256	2×2
mpool2	3×3	-	-	2×2
conv3	3×3	256	384	1×1
conv4	3×3	384	256	1×1
conv5	3×3	256	256	1×1
mpool5	5×3	-	-	3×2
fc6	9×1	256	4096	1×1
apool6	$1 \times n$	-	-	1×1
fc7	1×1	4096	1024	1×1
fc8	1×1	1024	1251	1×1

The ResNet architecture is a variation of a standard CNN that incorporates skip connections, enabling layers to add residuals to an identity mapping on the channel outputs. VoxCeleb2 (Chung et al., 2018) uses ResNet34 and ResNet50 by modifying their layers to accommodate spectrogram inputs. We adopted the ResNet50 architecture for this work and trained it on the VoxCeleb2 dataset. The architecture of the ResNet50 model can be found in Table 4.3.

Table 4.3: Modified ResNet50 architecture. Each row specifies the number of convolutional filters and their sizes as *size* × *size*, # filters.

Layer (type)	**ResNet50**
conv1	7×7, 64, stride 2
pool1	3×3 max pool, stride 2
conv2_x	$\begin{bmatrix} 1 \times 1 & 64 \\ 3 \times 3 & 64 \\ 1 \times 1 & 256 \end{bmatrix} \times 3$
conv3_x	$\begin{bmatrix} 1 \times 1 & 128 \\ 3 \times 3 & 128 \\ 1 \times 1 & 512 \end{bmatrix} \times 4$
conv4_x	$\begin{bmatrix} 1 \times 1 & 256 \\ 3 \times 3 & 256 \\ 1 \times 1 & 1024 \end{bmatrix} \times 6$
conv5_x	$\begin{bmatrix} 1 \times 1 & 512 \\ 3 \times 3 & 512 \\ 1 \times 1 & 2048 \end{bmatrix} \times 3$
fc1	9×1 2048 stride 1
pool_time	$1 \times N$ *avgpool* stride 1
fc2	1×1 5994

When training a neural network, the objective is to identify the ideal set of parameters $\boldsymbol{\theta}$ that minimizes the accumulated loss function over the complete training dataset. However, speaker recognition has two primary categories of loss functions. The first is speaker identification, which involves modeling the identification of multiple speakers, while the second is speaker verification, which focuses on verifying a single speaker's identity.

When working on Speaker Identification tasks, each speaker is considered a separate individual and assigns a unique label to them during training. If two recordings have the same label, they are from the same person. We feed the model a single input, and it predicts an output label. However, during the actual identification process, the assigned label may not be accurate if the speaker is not present in the training data.

We use the following approach to train a neural network for this task. The process involves generating an embedding vector from the input data, which is then projected to match the total number of speakers. This embedding is then passed through a softmax function, transforming it into a probability distribution. To measure the consistency, we can compare the softmax output with the true speaker label using the cross-entropy function.

For the SI task, both models were trained in the Voxceleb datatasets. Then, inspired by Guedes et al. (2019), we employed Framed Trim and Framed Silence Removed datasets in the pre-trained VGG-M and ResNet50 models for transfer learning to identify our 32 speakers. The process involved recreating the classifier after training on the VoxCeleb datasets, specifically modifying the model's output layer for 32, and then training it for our FrT and FrSRdatabase. Figure 4.16 illustrates this process.

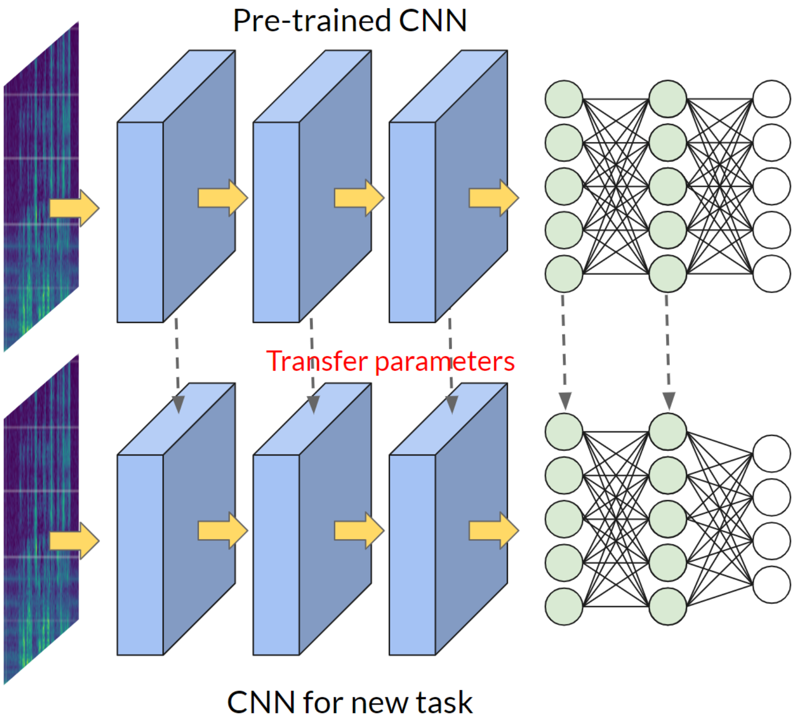

Figure 4.16: Transfer learning process.

When dealing with Speaker Verification tasks, the model is provided with two inputs and is tasked with predicting a binary outcome. Specifically, it returns a 0 if the inputs represent different speakers and a 1 if they are from the same speaker. At least a pair of inputs is needed for training, and we no longer require globally meaningful labels since, in binary decision problems, the label is not usually explicitly calculated.

To illustrate this process, let us look at the flowchart in Image 4.17. Given two inputs x_1 and x_2 to a siamese neural network, we obtain two embeddings e_1 and e_2. These embeddings are then used to calculate a distance score s_{12}. The score and the known speaker labels are then used to define our binary parameters. As a result, our loss becomes a function of s_{12} and y_{12}. These neural networks share the same parameters, which means once we update the parameters based on the Loss Function, both networks are updated simultaneously with the same update. Training with this approach involves selecting balanced input trials as either a positive trial (same speaker) or a negative trial (different speakers).

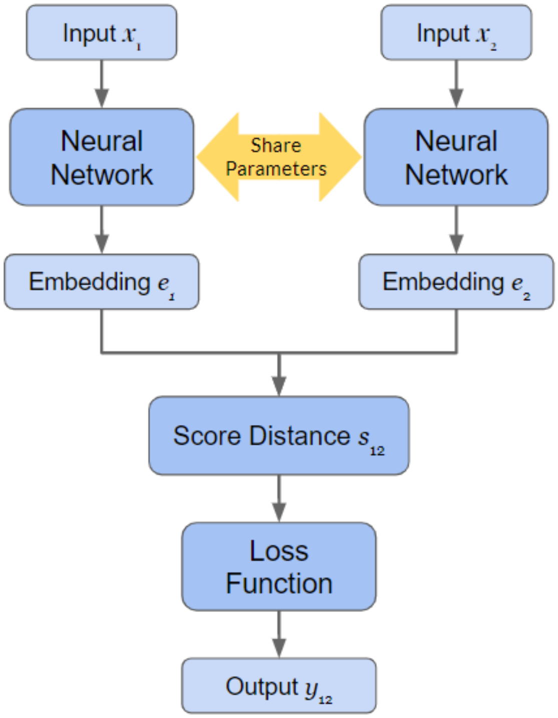

Figure 4.17: Siamese Workflow.

Chapter 5

Experimental Results and Analysis

5.1 Speaker Identification Experiments

This section presents the development and experimentation process of the models used in this work: the GMM, the VGG-M, and the ResNet50. We will discuss the characteristics of each model, the modifications made to adapt them to the SI task, and the results obtained from our experiments.

5.1.1 GMM Experiments

During the first implementation step of the Gaussian Mixture Model, we utilized the FrSR and FrT datasets. To develop the GMM model, we created a unique model for each speaker. All models contain 23 Gaussian components and use a diagonal covariance matrix. The EM algorithm is used to make the GMM converge using the extracted set of features for model training. Once trained, when a new audio sample is introduced, the features are passed through all the models.

The speaker s is identified based on the model that generates the highest score, as shown in Equation 4.8 (D. Reynolds & Rose, 1995). To calculate the score, we use a function that computes the average log-likelihood per sample of the provided data. We evaluated the model's performance by passing each test audio through it and asking it to identify the speaker. We then calculated the accuracy of the model by counting the correct classifications.

We conducted several experiments to determine the optimal input features for the GMM model. These experiments involved variations in the input data for both FrSR and FrT datasets. Initially, we used a small set of features like the 13 MFCCs, and then we added more input features to test for improved accuracy. The experiment results are summarized in Table 5.1.

Table 5.1: Accuracy of the GMM model with different feature combinations

Dataset	**Input Features**	**Accuracy**(%)
Framed Silence Remove (FrSR)	MFCC(13)	95.00
	MFCC(13), ΔMFCC(13), Δ^2MFCC(13)	93.00
	MFCC(13), Chroma STFT, RMS, SC, SBW, SRO, ZCR	91.66
	MFCC(20)	93.00
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20)	90.00
	MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	93.33
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	91.66
Framed Trim (FrT)	MFCC(13)	90.34
	MFCC(13), ΔMFCC(13), Δ^2MFCC(13)	86.89
	MFCC(13), Chroma STFT, RMS, SC, SBW, SRO, ZCR	86.20
	MFCC(20)	93.10
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20)	86.20
	MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	90.34
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	88.27

After analyzing the experiments conducted on the FrSR and FrT datasets, it was found that the FrSR dataset showed higher accuracy when compared to the FrT dataset across various feature combinations. There are two possible reasons for this observation. Firstly, and being an issue to the evaluation, the FrSR dataset is easier to classify than the FrT dataset because it has fewer samples. Secondly, the FrSR dataset is cleaner than the FrT dataset. Furthermore, increasing the input parameters for both datasets resulted in lower model accuracy. This situation indicates a potential trade-off between the complexity of the model and its performance.

To summarize, the efficiency of the GMM model in Speaker Identification highly depends on the data quality and input parameters. Clean data is more effective, as indicated by higher accuracy with FrSR dataset. Adding more parameters can lead to decreased accuracy, indicating sensitivity to complexity.

5.1.2 DL Experiments

In order to develop an environment for testing Neural Networks in the field of Speaker Identification and gain more experience in deep learning development, we implemented a Feed-forward MLP model inspired by a study conducted by Kral (2010). We used only the features extracted from the FrT dataset for this experiment. Unlike the GMM models, this model classifies each row in the set instead of the entire set extracted from an audio sample.

The MLP model used in the study comprised an input layer named **fc1** with 256 neurons that could take variable input, depending on the experiment. It also had two hidden layers, **fc2** and **fc3**, with 128 and 64 neurons, respectively. The output layer, **fc4**, had 32 neurons representing the number of classes in the dataset. The architecture of the MLP model is shown in Table 5.2.

Table 5.2: MPL architecture.

Layer (type)	Input Shape	Output Shape
fc1	N	256
fc2	256	128
fc3	128	64
fc4	64	32

In Experiment 1, we trained a Multilayer Perceptron (MLP) for 50 epochs to classify 32 speakers. The MLP employed CrossEntropy as the loss function and was fed with the first 13 MFCCs, Δ MFCCs, and Δ^2 MFCCs as input features, which resulted in a descriptor size of 39 features. Initially, this setup provided an accuracy of 74%.

In Experiment 2, we adjusted the data allocation for training, validation (early stopping), and testing to 70%, 10%, and 20%, respectively. We set the early stopping criteria based on the loss function but later switched to accuracy as we observed that accuracy could improve even with increased loss. The early stopping implementation led us to increase the number of epochs to 500, and then this model achieved an accuracy of 91.12%

on the test data.

Experiment 3 added more features to the model, including chroma STFT, RMS, SC, SBW, SRO, ZCR, and 20 MFCCs and their derivatives, resulting in 77 features. The model was trained over 500 epochs and achieved a validation accuracy of 93.48% and a test accuracy of 92.19%. With the increase of features, the training process became more stable.

In Experiment 4, we decreased the learning rate from 0.001 to 0.0001, which resulted in more consistent training and a greater enhancement in accuracy. The model accomplished a validation accuracy of 93.40% and a test accuracy of 93.33%. Table 5.3 compares the results of the MLP network.

Table 5.3: Performance of MLP Model Across Attempts

Experiments	**Input Features**	**Val Accuracy**(%)	**Test Accuracy**(%)
Experiment 1	MFCC(13), ΔMFCC(13), Δ^2MFCC(13)	–	74.00
Experiment 2	MFCC(13), ΔMFCC(13), Δ^2MFCC(13)	91.76	91.12
Experiment 3	MFCC(20), ΔMFCC(20), Δ^2MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	93.48	92.19
Experiment 4	MFCC(20), ΔMFCC(20), Δ^2MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	93.40	93.33

To conduct experiments using convolutional neural networks ResNet50 and VGG-M, we downloaded the VoxCeleb1 dataset comprising 1,251 speakers. Next, we implemented the model proposed in the VoxCeleb1 (Nagrani et al., 2017), a modified VGG-M model. The first layer of this model is modified to one channel to accept STFT as input instead of an image. To extract the STFT data, we segmented audio data into 3-second segments, sampled at 16kHz, using a window of 25ms and a step of 10ms. The data was then normalized and passed to the VGG-M model for classification.

We followed the data processing steps described in the VoxCeleb (Chung et al., 2018) and developed a model architecture to classify 40 classes. Therefore, due to the large dataset size, we randomly selected 40 speakers to train the model, which still resulted in over 1GB of data. This reduction in the number of speakers helped us to perform tests quickly. During this training test, we achieved the best accuracy of 92.20% for validation data and 90.25% for test data.

In addition to the VGG-M model, we have implemented the ResNet50 model proposed in VoxCeleb2 (Chung et al., 2018). This model requires as input a Spectrogram derived from the STFT instead of the STFT itself. Therefore, the same modification done in the first layer in VGG-M is also done in this model. The dimensions and sampling rate are the same as that of VGG-M. We have modified the first convolutional layer to accept one-dimensional spectrograms, as the original network accepts three-dimensional RGB images. We have removed the final layers described in VoxCeleb2 (Chung et al., 2018) and added the classifier described in the paper. After training the model, we achieved an accuracy of 88.60% for validation data and 86.40% for test data for this reduced Voxceleb1 dataset with the same 40 speakers.

The lower accuracy of the ResNet50 model in comparison to the VGG-M model in this test was due to the use of an undersized dataset. In the VoxCeleb2 paper (Chung et al., 2018), this model used a much larger dataset of around 6,000 speakers. Then, the ResNet50 model could achieve better generalization results due to using a more significant amount of data for training.

After the previous test, we selected the best model and trained it using the complete VoxCeleb 1 dataset. The VGG-M model was used with a learning rate 0.0001 and a Cross Entropy Loss function. The model successfully achieved an accuracy of 85.66% on validation data and 86.20% on test data. These results are higher than the initial VoxCeleb1 (Nagrani et al., 2017) results of 80.5% for the Speaker Identification task.

In our next experimentation phase, we utilized the VoxCeleb2 dataset, a significantly larger dataset than VoxCeleb1. The dataset comprises $1,092,009$ speeches from $5,994$ speakers. We adopted the method proposed in VoxCeleb2 (Chung et al., 2018), where a 3-second audio segment is randomly selected for each audio and used for training. This approach enables us to use different sections of a recording at different times while training the model.

We trained the VGG-M network using the VoxCeleb2 dataset, which has 5,994 output classes. As we increased the data, the performance of the model also improved. With the

VoxCeleb1 dataset, we achieved an accuracy of 85.66% for validation data and 86.20% for test data. However, with the VoxCeleb2 dataset, the accuracy improved significantly to 89.34% for validation data and 89.19% for test data. To achieve this, we trained the model for 100 epochs with a learning rate $1e-05$.

For the ResNet50 model, we achieved the highest accuracy of all the models we trained using the VoxCeleb datasets. With the VoxCeleb2 dataset, we obtained an accuracy of 95.22% for validation data and 95.09% for test data after 100 training epochs with a learning rate $1e-05$. The results of this experimentation phase are summarized in Table 5.4.

Table 5.4: Summary of the accuracy results for the VGG-M and ResNet50 models on the VoxCeleb1 and VoxCeleb2 datasets.

Model	**Dataset**	**Val Accuracy**(%)	**Test Accuracy**(%)
VGG-M	VoxCeleb1 (40 speakers)	92.20	90.25
ResNet50	VoxCeleb1 (40 speakers)	88.60	86.40
VGG-M	VoxCeleb1	85.66	86.20
VGG-M	VoxCeleb2	89.34	89.19
ResNet50	VoxCeleb2	95.22	95.09

After evaluating the performance of the VGG-M and ResNet50 models with the VoxCeleb1 and VoxCeleb2 datasets, we selected the models with the highest accuracies. Then, we proceed with the transfer learning stage. The best models were the VGG-M model trained with the VoxCeleb2 dataset and the ResNet50 model trained with the same dataset.

To perform transfer learning on the VGG-M model, we first imported the weights from the best-performing VGG-M model and then reset the weights of the **fc7** layer. We adjusted the output of the **fc8** layer to classify 32 classes, corresponding to the number of speakers in our dataset. We followed a similar process for the ResNet50 model. After importing the weights from the best-performing ResNet50 model, we reset the **fc1** layer and adjusted the output of the **fc2** layer to classify 32 classes. Once we made these adjustments, we kept all layers trainable and trained these modified models.

In this stage, we retrained the ResNet50 and VGG-M models for 20 and 60 epochs,

Table 5.5: Summary of Transfer Learning Results for ResNet50 and VGG-M Models

Model	**Dataset**	**Padding**	**Accuracy(%)**
ResNet50	Framed Trim	No	97.93
		Yes	93.10
	Framed Silence Removed	No	100.00
		Yes	100.00
VGG-M	Framed Trim	No	88.97
		Yes	90.34
	Framed Silence Removed	No	88.33
		Yes	98.33

respectively, using our two personal datasets, FrT and FrSR. As the audio samples in these datasets are shorter than 3 seconds, we applied padding to evaluate the models' responses. We added zeros at the end of the audio samples until they reached a length of 3 seconds, and in other testing sessions, we trained the models without padding. We have presented the training sessions' results in Table 5.5.

After analyzing the results, it was observed that adding padding to the FrT dataset caused a decrease in accuracy for the ResNet50 model. However, for the VGG-M model, there was no significant difference in accuracy when padding was added to the FrT dataset. However, a slight improvement was seen in the FrSR dataset when the padding was added for this model. The FrSR dataset has fewer samples and longer audio files and is also cleaner than the FrT dataset. As a result, a higher accuracy was expected when testing on the FrSR dataset. When all speakers in the test set are correctly identified, the system achieves 100% accuracy, as with the ResNet50 model in the FrSR dataset.

5.2 Speaker Verification Experiments

This section will present the development and experimentation process of the models used in this work for speaker verification. The models that will be discussed are the GMM, VGG-M, and ResNet50. We will analyze each model, the adaptations made to fit the speaker verification task, and the results obtained from our tests.

5.2.1 GMM Experiments

The first approach was to develop a GMM model that could identify the speaker of a given phrase, thus serving as a Speaker Identification model. In this design, the model assumes a closed set, where it tries to identify the speaker from a set by assigning a score to the most likely candidate. Therefore, this approach makes it impossible to detect an imposter who is not in the set. When an imposter's speech is processed, each model generates a score, and the imposter is identified with the model with the lowest score, leading to incorrect identification instead of rejection.

However, within a Speaker Verification task, we are now interested in verifying if a speech belongs to a specific speaker. Therefore, another approach called the Universal Background Model is necessary (D. A. Reynolds et al., 2000), which was presented in Chapter 4. This approach utilizes a speaker-independent GMM model to represent alternative speakers or imposters. Essentially, this approach reduces the problem to testing whether the utterance features X are from the hypothetical speaker S (Hypothesis H_0) or not (Hypothesis H_1). The log-likelihood ratio from Equation 4.9 is then used to calculate a score. If this score is greater than or equal to a threshold θ, Hypothesis H_0 is accepted. Otherwise, Hypothesis H_1 is accepted.

To implement the approach proposed by Reynolds (D. A. Reynolds et al., 2000), we followed a two-step process. Firstly, we used all the training data to create a GMM-UBM. Then, using the Bayesian Adaptation algorithm, the GMM-UBM model was adapted for each speaker's specific data to create 32 GMMs, one for each speaker. This training pipeline was illustrated in Figure 4.14. Although it is possible to re-adapt the weights, means, and covariances, the best approach is to adapt only the means. To use the log-likelihood ratio in this situation, we first calculate scores for a given set of utterance features X for each model and then subtract the score from the GMM-UBM model. The speaker is accepted for each model score with a result greater than a threshold θ.

To evaluate the GMM-UBM model, we utilized the same feature sets employed for Speaker Identification. However, for speaker verification, the evaluation process is slightly different. Since this system is used to verify the speaker's identity, we compare the score of the model with the threshold θ to determine whether the speaker is correctly identified, incorrectly identified, or whether an imposter is identified as an imposter or incorrectly identified as the speaker. In total, we have four possible outcomes.

In this context, it is more appropriate to use other metrics like the Equal Error Rate (EER) and Area Under Curve (AUC) of the ROC curve, as explained in Chapter 2. The EER is when the false positive rate equals the false negative rate. To calculate it, we adjust the threshold until we find the most suitable one and then assign a label to the speaker (1 for accepted, 0 for rejected). After we find the threshold, we can also calculate the accuracy by checking how many results were correct - i.e., for Cosine Similarity, accept when they are greater or equal to the threshold and reject when they are below the threshold. Table 5.6 shows the results of these experiments for Speaker Verification.

Table 5.6: Results of the GMM-UBM model with different feature combinations for Speaker Verification

Dataset	**Input Features**	**EER**(%)	**Accuracy**(%)	**AUC**(%)
FrSR	MFCC(13)	2.69	97.29	0.997
	MFCC(13), ΔMFCC(13), Δ^2MFCC(13)	3.12	96.87	0.997
	MFCC(13), Chroma STFT, RMS, SC, SBW, SRO, ZCR	2.42	97.55	0.994
	MFCC(13), ΔMFCC(13), Δ^2MFCC(13), Chroma STFT, RMS, SC, SBW, SRO, ZCR	2.96	97.03	0.994
	MFCC(20)	3.55	96.45	0.996
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20)	7.26	92.76	0.986
	MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	2.13	97.87	0.994
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	7.10	92.91	0.990
FrT	MFCC(13)	5.01	95.00	0.987
	MFCC(13), ΔMFCC(13), Δ^2MFCC(13)	9.12	90.88	0.976
	MFCC(13), Chroma STFT, RMS, SC, SBW, SRO, ZCR	6.38	93.62	0.985
	MFCC(13), ΔMFCC(13), Δ^2MFCC(13), Chroma STFT, RMS, SC, SBW, SRO, ZCR	6.81	93.18	0.986
	MFCC(20)	3.52	96.46	0.995
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20)	10.28	89.74	0.978
	MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	4.85	95.15	0.992
	MFCC(20), ΔMFCC(20), Δ^2MFCC(20), Chroma STFT, RMS, SC, SBW, SRO, ZCR	7.50	92.50	0.986

5.2.2 DL Experiments

Similar to the GMM model, in the speaker identification approach, both CNN models, the ResNet50 and VGG-M, could identify the speaker of a given utterance accurately and performed very well on our private datasets. However, the models also assume a closed set, which means an impostor would be identified as one of our set of speakers. Therefore, this approach makes it impossible to detect an imposter who is not in the set.

In a Speaker Verification task, we need to determine whether a particular speech belongs to a specific speaker. To accomplish this, we require a new approach that utilizes siamese neural networks, as demonstrated in the VoxCeleb papers (Chung et al., 2018; Nagrani et al., 2017, 2020). This approach was visually represented in Figure 4.17 of Chapter 4. By verifying a single speaker's identity, this approach changes the model's task to a binary decision problem.

In this approach, a CNN model is employed to learn a distance metric that can determine whether two utterances are spoken by the same person or not. To train the model, at least a pair of inputs, x_1 and x_2, for each training instance is needed. Then their embeddings are computed, represented by e_1 and e_2. Based on these embeddings, the model calculates a distance score, denoted by $s12$. If this score is equal to or greater than a specific threshold value, the model categorizes the utterances as being spoken by the same speaker and returns the label 1. On the other hand, if the score is less than the threshold, the model assigns the label 0, indicating that the utterances are spoken by different speakers.

In both VoxCeleb papers (Chung et al., 2018; Nagrani et al., 2017), the loss function used in this context was the Contrastive Loss presented by Chopra et al. (2005) and Hadsell et al. (2006). The loss function in most general form is given by Equation 5.1. Where $(y, x_1, x_2)^i$ is the i-th labeled sample pair, L_S is the partial loss function for a pair of similar points, L_D the partial loss function for a pair of dissimilar points, P is the number of training pairs, and D_w represent the distance function between the x_1 and x_2 embeddings. For this work, the specific Loss function used is given by Equations 5.1, 5.2, and 5.3.

$$\mathcal{L}(w) = \sum_{i=1}^{P} L(w, (y, x_1, x_2)^i) \tag{5.1}$$

$$L(w, (y, x_1, c_2)^i) = (1 - y)L_S(D_w^i) + yL_D(D_w^i) \tag{5.2}$$

$$L(w, (y, x_1, c_2)^i) = (1 - Y)\frac{1}{2}(D_w)^2 + (Y)\frac{1}{2}\{max(0, m - D_w)\}^2 \tag{5.3}$$

Given a pair of inputs and a neural network, we obtain two embeddings and compute their distance function, where we have two alternatives: the Euclidean Distance denoted by 2.9 and the Cosine Similarity denoted by 2.8. If the Euclidean distance is chosen, we want embeddings from the same speaker to have a smaller distance than those from different speakers. Otherwise, if the cosine similarity is chosen, we want the score for embedding the same speaker to be bigger than those from different speakers.

Triplet loss is another approach initially proposed by Schroff et al. (2015) for face recognition and has also gained popularity in speaker recognition. In this approach, we use a triplet of inputs in each training step: the anchor utterance x^a, the positive utterance x^p from the same anchor speaker, and a negative utterance x^n from a different speaker than the anchor. The goal is to bring embeddings of the same speaker closer together while distancing embeddings from different speakers, as shown in Figure 5.1.

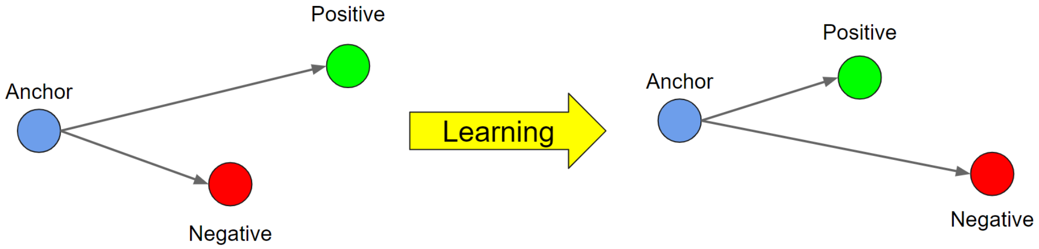

Figure 5.1: Triplet loss learning process.

When using the Euclidean distance, Triplet Loss is defined by Equation 5.4. Additionally, when using cosine similarity, Triplet loss is defined by Equation 5.5. Where $[\cdot]_+$

is the maximum between 0 and the score, α is a predefined margin value between positive and negative. It is important to notice that for each positive trial, the loss function is trying to decrease the score value, and for the negative trial, the loss function is trying to increase the score.

$$L(x^a, x^p, x^n) = [d(f(x^a), f(x^p))^2 - d(f(x^a), f(x^n))^2 + \alpha]_+ \tag{5.4}$$

$$L(x^a, x^p, x^n) = [\cos(f(x^a), f(x^n)) - \cos(f(x^a), f(x^p)) + \alpha]_+ \tag{5.5}$$

However, balancing positive and negative trials during training can be challenging. The efficiency of the training process for contrastive and triplet loss hinges on selecting the most difficult trials. However, this is complicated by the changing nature of the neural network's parameters during training. To address this, we can mine hard samples. This process can be employed in two ways: offline or online. In the offline approach, the trials are computed before the start of the epoch using the entire dataset. For the online approach, the trials are computed during the epochs using the data within the batch.

For our experiments with ResNet50 and VGG-M, we adjusted the size of the last layer to 512 and 1024, respectively, as shown in the Voxceleb 1 (Nagrani et al., 2017) and VoxCeleb 2 (Chung et al., 2018) papers. We selected the best models from the Speaker Identification results and uploaded the parameters for each model. To evaluate the models, we used the trial pairs list from VoxCeleb 1, as this list is commonly used to compare Speaker Verification models using the EER metric.

We use three test time augmentation protocols (TTA) to evaluate model performance and compare it with the Voxceleb papers (Chung et al., 2018; Nagrani et al., 2017). These protocols were utilized in their evaluation of models. The first protocol (TTA-1) uses the entire audio as input for the model. The second protocol (TTA-2) selects ten 3-second temporal crops from each test segment. The embeddings from each crop are then averaged to calculate the distance. The third protocol (TTA-3) selects ten 3-second temporal crops from each test segment. The distances are calculated between every possible pair of crops from the two speech segments, resulting in 100 pairs, and the mean of the 100 distances is used.

Before performing the new training, we evaluated both models using the embeddings from the Speaker Identification step with no retraining. For both models, we took the embeddings from the penultimate layer. Then, we calculated the ERR by measuring the Euclidean distance and cosine similarity between each pair of embeddings obtained from the input utterances in the VoxCeleb1 test set. The results of this evaluation are displayed in table 5.7.

Table 5.7: EER results for testing without any retraining *(lower is better)*.

Model	**Score Function**	**TTA-1(%)**	**TTA-2(%)**	**TTA-3(%)**
VGG-M	Euclidean Distance	40.12	40.83	41.53
	Cosine Similarity	39.94	40.71	41.58
ResNet50	Euclidean Distance	30.40	30.59	30.05
	Cosine Similarity	22.95	23.51	22.41

We tested Contrastive Loss by generating 2^{18} random pairs without mining. Both models utilized Euclidean distance and cosine similarity. In addition, we aimed to determine whether it is more effective to train only the new layer as VoxCeleb, the entire classification module, or even the entire model. The results of these tests are presented in Table 5.8. The True values indicate that the weights of the modules were retrained, whereas False values indicate that the weights were frozen.

We also experimented with testing Triplet Loss by randomly generating 2^{18} pairs without mining. As it was done with Contrastive Loss, both models used Euclidean Distance and Cosine Similarity. We also performed a test to determine whether it is more effective to train only the new layer, the entire classification module, or the entire model. We have included the results of these tests in Table 5.9.

After generating random pairs and triplets, we did offline hard mining for both loss strategies. For the Contrastive Loss strategy, we randomly selected only 1 percent of the generated pairs and mined for the hardest negative samples, similar to what was done in VoxCeleb works. This can be understood as the pairs from different speakers with a

Table 5.8: EER results for retraining with random pairs and Contrastive Loss *(lower is better)*.

Model	**Score Function**	**CNN Module**	**FC Module**	**TTA-1(%)**	**TTA-2(%)**	**TTA-3(%)**
VGG-M	Euclidean Distance	False	False	36.20	36.58	37.09
		False	True	43.36	43.77	43.73
		True	True	48.33	45.94	46.46
	Cosine Similarity	False	False	17.16	17.85	17.93
		False	True	15.96	16.88	16.85
		True	True	13.36	13.84	14.02
ResNet50	Euclidean Distance	False	False	32.87	34.31	31.62
		False	True	34.39	36.48	31.44
		True	True	49.24	43.96	44.77
	Cosine Similarity	False	False	8.51	9.19	9.17
		False	True	8.81	9.36	9.41
		True	True	7.78	8.51	8.70

Table 5.9: EER results for retraining with random triplets and Triplet Loss *(lower is better)*.

Model	**Score Function**	**CNN Module**	**FC Module**	**TTA-1(%)**	**TTA-2(%)**	**TTA-3(%)**
VGG-M	Euclidean Distance	False	False	17.86	18.94	19.01
		False	True	14.18	15.53	15.31
		True	True	14.58	13.63	14.85
	Cosine Similarity	False	False	20.98	21.46	21.85
		False	True	18.62	19.53	19.91
		True	True	18.07	18.62	19.29
ResNet50	Euclidean Distance	False	False	8.17	9.12	9.24
		False	True	7.25	8.13	8.17
		True	True	7.58	8.01	8.22
	Cosine Similarity	False	False	12.69	13.53	13.88
		False	True	12.78	13.35	13.70
		True	True	13.54	14.27	14.65

minimum distance for Euclidean distance. For this strategy, we generated 2^{18} pairs and performed the experiments shown in Table 5.10.

We followed the method proposed by Hermans et al. (2017) to implement the Triplet Loss technique. We randomly selected some anchors, and for each pair with the same label, we chose the positive trial with the maximum distance and the negative trial with the minimum distance. Since we generated around 2^{18} anchors, we had to calculate 2^{36} distances, which are computationally expensive and require a significant amount of memory. Therefore, we divided the anchor into smaller batches to compute the distances. We focused our experiments on the ResNet model, which performed better in the other experiments, and we did not need any retraining of the Convolutional layers. The experiments

Table 5.10: EER results for retraining with hard mining and Contrastive Loss *(lower is better).*

Model	**Score Function**	**CNN Module**	**FC Module**	**TTA-1(%)**	**TTA-2(%)**	**TTA-3(%)**
VGG-M	Euclidean Distance	False	False	32.58	33.78	33.26
		False	True	41.88	40.75	38.2
		True	True	48.7	45.94	44.09
	Cosine Similarity	False	False	25.26	23.65	23.85
		False	True	40.76	40.09	41.07
		True	True	18.38	18.75	19.77
ResNet50	Euclidean Distance	False	False	24.18	25.66	23.36
		False	True	37.35	38.9	34.12
		True	True	49.43	48.81	48.99
	Cosine Similarity	False	False	5.96	6.49	6.44
		False	True	8.12	8.67	8.85
		True	True	9.35	9.92	10.05

are shown in Table 5.11.

Table 5.11: EER results for retraining with hard mining and Triplet Loss *(lower is better).*

Model	**Score Function**	**CNN Module**	**FC Module**	**TTA-1(%)**	**TTA-2(%)**	**TTA-3(%)**
ResNet50	Euclidean Distance	False	False	14.16	14.84	14.51
		False	True	13.48	15.42	14.86
	Cosine Similarity	False	False	16.42	16.99	17.04
		False	True	16.09	16.56	16.83

Finally, for each experiment, Without Retrain, Random Pairs, Random Triplets, Hard Pairs, and Hard Triplets. We selected the models that produced the best results for each Distance function. Then, we performed evaluations on our FrSR and FrT datasets. We used the training set of our private datasets to create an embedding for each speaker, and then we used the data in the test set to compare them in a similar way to the comparison with the GMM model calculating the Equal Error Rate (EER). As the duration of the audio files is less than 3 seconds, we used the entire audio for the evaluation. The results are presented in Table 5.12.

Table 5.12: EER results applied in CeDRI's datasets *(lower is better)*.

			FrSR			FrT		
Model	**Experiment**	**Distance**	**EER**(%)	**Acc**(%)	**AUC**(%)	**EER**(%)	**Acc**(%)	**AUC**(%)
VGG-M	Without Retrain	Euclidean	18.33	88.65	83.49	26.9	78.87	73.32
		Cosine	18.12	88.78	81.88	26.96	78.64	73.04
	Random Pairs	Euclidean	13.33	95.12	87.24	22.76	84.41	76.92
		Cosine	7.69	97.69	92.29	18.09	91.31	81.94
	Random Triplets	Euclidean	5	99.22	93.91	15.17	92.99	84.91
		Cosine	11.83	94.65	88.18	20.69	88.02	79.31
	Hard Pairs	Euclidean	16.67	92.25	82.6	26.21	80.77	73.77
		Cosine	6.08	98.63	93.91	13.84	93.78	86.16
ResNet50	Without Retrain	Euclidean	7.37	97.88	92.66	15.66	92.33	84.33
		Cosine	6.72	97.81	93.28	15.17	91.34	84.83
	Random Pairs	Euclidean	5	99.22	95.73	15.17	92.71	85.67
		Cosine	5.38	98.94	94.64	12.35	94.88	87.65
	Random Triplets	Euclidean	1.67	99.87	97.76	11.03	96.67	89.09
		Cosine	6.34	97.85	93.65	15.19	92.1	84.81
	Hard Pairs	Euclidean	1.67	99.77	97.76	13.1	95.26	86.96
		Cosine	0.7	99.89	99.32	9.39	96.64	90.6
	Hard Triplets	Euclidean	5	99.37	94.95	10.34	96.57	89.85
		Cosine	5.91	97.79	94.06	14.51	93.04	85.5

5.3 Discussion

When working on the Speaker Identification task, we began with the GMM model. This method has been traditionally widely used for speaker recognition. It utilizes a mixture of normal distribution functions to represent the distribution of a speaker's acoustic statistical characteristics. We found that this model performed well in our private datasets, such as FrT and FrSR, achieving high accuracies even for a simple model. Additionally, the saved models were compact, with each file only being about 50kb.

However, during the analysis, we observed that the GMM model's performance was influenced by the complexity of the input features and the quality of the data. As we increased the number of input parameters, the accuracy of the model decreased, indicating a possible trade-off between the model's complexity and its performance. The FrSR dataset yielded the highest accuracy for the GMM model, which was expected as this dataset has less data to classify and is cleaner than the Trim dataset.

It is important to note that the GMM assumes a Gaussian distribution of acoustic characteristics. However, this assumption may not fully capture the complexity of the acoustic characteristics of different speakers. As the number of speakers increases, the SI problem becomes more complex. Furthermore, in many cases, complex data cannot be represented by simple Gaussian distributions.

During our experimentation with Deep Learning, we began by using the MLP model. It was a good way to test the waters and gain more experience in this field. The model was trained on various feature sets and data allocations. Interestingly, unlike the GMM model, this model's accuracy improved as we expanded the feature set and adjusted the data allocation for training, validation, and testing. The highest accuracy we achieved with the MLP model was 93.33% on the Framed Trim dataset.

The VGG-M and ResNet50 models were initially trained on the larger VoxCeleb1 and VoxCeleb2 datasets for transfer learning. For both models, we divided the audio data into 3-second segments and calculated the STFT or spectrogram of the audio for input. During the training of the VGG-M model on the VoxCeleb1 dataset, it achieved an accuracy of 86.20%, which was higher than the best in the first VoxCeleb1 paper for Speaker Identification, which was 80.5%. The VGG-M model achieved the highest accuracy rate of 89.19% by classifying 5994 individuals within the VoxCeleb2 data. On the other hand, the best ResNet50 model achieved an accuracy rate of 95.09% for the test data in the VoxCeleb2 dataset. The results for the Speaker Identification task were impressive, especially for the ResNet50 Model, as it demonstrated robustness by achieving high accuracy for a large number of speakers.

During the transfer learning stage, we fine-tuned VGG-M and ResNet50 models using our datasets, FrT and FrSR. The VGG-M model showed an accuracy of 90.38% on the Framed Trim dataset and 98.33% on the Framed Silence Removed dataset. At the same time, the ResNet50 model performed better, with an accuracy of 97.93% on the FrT dataset and 100% on the FrSR dataset. Both models were able to adapt well to the new data. However, the ResNet50 model slightly outperformed the VGG-M model on all datasets. It was able to train in fewer epochs than VGG-M, which suggests that it might be more robust to variations in the data. As a result, the ResNet50 model was the best-performing model for this task.

It is also important to mention that the FrSR dataset is smaller, has longer audio recordings, and is cleaner than the FrT dataset. Because of these characteristics, all models perform better on the FrSR dataset than on the FrT dataset. However, this does not necessarily mean that the FrSR dataset is superior; it is easier to classify. Table 5.13 summarizes the best models for the FrT and FrSR datasets.

Table 5.13: Summarized results from comparing the best models.

Dataset	**Best Models**	**Acurracy(%)**
Framed Trim Dataset	GMM	93.10
	MLP	93.33
	VGG-M	90.34
	ResNet50	**97.93**
Silence Removed Dataset	GMM	95.00
	VGG-M	98.33
	ResNet50	**100.00**

When dealing with the Speaker Verification task, the first approach we performed was using the GMM-UBM model. With the addition of just one model, we can check the identity of the speakers and classify if there are some impostors on the set. Also, the model generally improves its performance by achieving an accuracy of 97.87% in the FrSR dataset and 96.46% in the FrT dataset, even when dealing with impostors. In these experiments, we can notice that some of the best results are not using just the MFCCs as input, but this happens mainly in the FrSR dataset where several 97% of accuracy happens, suggesting this was insignificant. However, we can see that using just the MFCCs for the FrT dataset has better results.

In the first experiment of the CNN models, we evaluated the models using only the weights from the Speaker Identification task without retraining. These initial results can serve as a ground truth to understand if the model is being improved. Additionally, we observed that the ResNet model has a better EER than VGG-M. Furthermore, the verification performance improved with cosine similarity, at least for the ResNet50 model. However, in this part of the project, the three TTAs did not lead to any significant improvement.

For the second experiment, random trials were sampled. The idea is to observe how the models behave when retraining with their losses and changing some parameters. So, for each model, we retrained for both Euclidean and Cosine Similarity functions. For each function, we performed three tests by retraining only the last layer, the entire fully connected layers, also known as the classifier, or the entire model by retraining the CNN models.

First, for Contrastive Loss, we can observe that for both models, when using the Cosine Similarity, the EER improved when we retrained more layers. However, when using the Euclidean Distance, it has the opposite behavior by increasing the EER. Also, using this loss, we can see that using Cosine Similarity leads to better results for both models. For Triplet Loss, we observed a behavior switch. Now, the Euclidean Distance leads to better results, and now, when retained with more layers, it gets a better EER. Also, the Resnet50 model had better results than the VGG-M for both losses.

In a subsequent experiment, we tried mining hard samples for the models. While there was a slight improvement in some configurations compared to the random samples, the models' ERR rates worsened in other configurations. We observed that using the Contrastive Loss with Euclidean Distance, the VGG-M model slightly improved its rates when we did not retrain the convolutional layers. Similarly, the ResNet50 model showed a similar behavior. However, the improvement was more prominent in the case of ResNet50. When using Cosine Similarity, all the ERRs of the VGG-M model worsened, while the ResNet50 model improved, but only when we did not retrain the convolutional layers.

This experiment suggests that retraining the convolutional layers with hard samples worsens the model, but random samples do not. As the experiments use the same hyperparameters, a possible explanation is the small number of samples compared to the dataset. For the training, 2^{18} trials were sampled a couple of times, but as we have around $1e6$ utterances, we have around $1e12$ pairs. So, this set of selected pairs is tiny, and increasing this number could help us get a better result.

During the use of the Triplet Loss for both distance functions, the EER worsened for the ResNet50 model. In the random experiment, 2^{18} pairs and triplets were chosen for each epoch. With a total of 30 epochs, we can estimate that the models saw almost 8 million triplets during the training. In contrast, the hard mining method sampled these 2^{18} triplets only six times, after ten epochs in 60 epochs. Therefore, the model saw 80% fewer samples in the hardening training than in the random training, which can explain the worse results.

However, the experiments showed that the models achieved good results for both training strategies, even with a few pairs and triples during training. To put this into perspective, there was about $1e12$ total number of pairs, and for the random training with contrastive loss, it sampled around $8e6$ pairs. Then, we can estimate that the model just saw 0.0008% of the total possible number of pairs, and even seeing just this small number, the ResNet50 model achieved an EER of 5.96%. As a comparison, the ResNet50 model from the VoxCeleb2 paper achieved an EER of 3.95% on the same test set using Contrastive Loss.

In our final experiment, we selected the best-performing models for each score function and experiment. We then created an embedding to represent each speaker using the test set from the FrT and FrSD datasets for each model. This allows us to compare the same test set with the GMM model. We have presented all the results for each experiment in Table 5.12.

The table shows that the models performed well. The VGG-M model achieved the best results with an EER of 5% for the FrSD dataset and 15.17% for the FrT dataset using random triples and Euclidean Distance. The ResNet50 model, on the other hand, achieved an EER of 0.7% for the FrSR dataset and 9.39% for the FrT dataset using hard pairs and Cosine Similarity. In comparison, the GMM model achieved an EER of 2.13% as the best result for the FrSR dataset and 3.52% for the FrT dataset.

When comparing these results, we can see that the VGG-M model was not as good as the GMM model. For the FrT, the GMM model also achieved a better metric than ResNet50. Especially for the FrT dataset a possible explanation for the CNN model did not perform well on the FrT dataset because the audio samples were too short. The FrT audios have a length of 0.944s, and as these models were trained with 3s length audios, this could mean that the data from the Trim dataset may not have been sufficient. Therefore, capturing audio at least 3 seconds long in a real-life application is recommended. On the other hand, the FrSR dataset has audio samples that are almost 2 seconds long, which was enough for the ResNet model to achieve an EER of 0.7%, outperforming the other models.

To implement the speaker recognition system, we can utilize the ResNet50 model. First, we must record and store an embedding for each new speaker during the enrolment phase. This embedding will represent the unique characteristics of the speaker's voice. In the authentication phase, new audio will be captured and compared with the stored embeddings to generate a score. The score will be compared to a threshold to determine if the speaker can access the laboratory. We can create different access levels by using multiple thresholds. By building the system this way, we can accurately grant access to the laboratory.

Chapter 6

Conclusions

This study explored different machine learning and deep learning models for speaker recognition technology, which can be used to build a biometric authentication system in the future. We tested models such as GMM, MLP, VGG-M, and ResNet50 on various datasets. These models were trained on different datasets, including Framed Trim and Framed Silence Removed.

The results of the speaker identification experiments showed that the ResNet50 model outperformed the other models by achieving the highest accuracy of 100% on the Framed Silence Removed dataset and 97.93% on the Framed Trim dataset. This suggests that the ResNet50 model may be more robust to variations in the data. The VGG-M model achieved 90.34% accuracy in the Framed Trim dataset and 98.33% accuracy in the Framed Silence Removed dataset after being trained on larger datasets. The Multilayer Perceptron model had an accuracy of 93.33% on the Framed Trim dataset. The GMM model had a speaker identification accuracy of 93.10% in the Framed Trim dataset and 95% in the Framed Silence Removed dataset. However, it is essential to note that the Framed Silence Removed dataset, which has fewer and cleaner samples than the Framed Trim dataset, contributed to all models achieving higher accuracy.

The results of the speaker verification experiments also showed that the ResNet50 model outperformed the other models by achieving the lowest EER of 0.7% on the Framed Silence Removed dataset. The VGG-M model achieved an EER of 5% accuracy in the

Framed Silence Removed dataset. Regarding the Framed Trim dataset, the CNN models did not perform so well, probably because of the duration of the audio. However, as the CNN models were retrained in a tiny subset of data for the speaker verification task, there is an excellent margin to explore by training this model in a large set of pairs and triplets and consequently improving the models. The GMM model had the best EER of 3.15% in the Framed Trim dataset and 2.13% in the Framed Silence Removed dataset. However, ResNet achieving the best EER rate even with the best training shows its potential.

It is now possible to create a system for the CeDRI laboratory by using the ResNet50 model. To achieve this, we can follow the same process we used when we used our private datasets to compare with the GMM model. First, we can collect an audio sample of each speaker in the laboratory, which will be used to create an "embedding" to represent them. This process is called the enrollment phase. Once someone requests access, we use the model to generate an embedding of their voice and compare it to the stored embedding to verify their identity. Furthermore, we can improve the same model by exploring hard mining techniques and replacing just the model weights when necessary. Also, it is important to notice that the model does not need to be retrained if a new speaker is introduced. In this case, just the enrollment phase will be enough.

To summarize, this work establishes a basis for more extensive research and development in speaker Recognition through neural networks. Several factors, such as the quality and complexity of the input data, the choice of features, and the model architecture, influence speaker identification models' performance. Our research provides valuable insights for developing speaker recognition-based access control systems.

Future investigations may study advanced feature extraction techniques and innovative model architectures to enhance the capabilities of our system and improve accuracy. Additionally, the application of more complex mining sample methods can be explored.

Bibliography

Alnuaim, A. A., Zakariah, M., Shashidhar, C., Hatamleh, W. A., Tarazi, H., Shukla, P. K., & Ratna, R. (2022). Speaker gender recognition based on deep neural networks and resnet50. *Wireless Communications and Mobile Computing*, *2022*, 1–13.

Bai, Z., & Zhang, X.-L. (2021). Speaker recognition based on deep learning: An overview. *Neural Networks*, *140*, 65–99.

Bricker, P., Gnanadesikan, R., Mathews, M., Pruzansky, S., Tukey, P., Wachter, K., & Warner, J. (1971). Statistical techniques for talker identification. *Bell System Technical Journal*, *50*(4), 1427–1454.

Büyük, O. (2011). *Telephone-based text-dependent speaker verification* [Doctoral dissertation, Bogazici University].

Chopra, S., Hadsell, R., & LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, *1*, 539–546.

Chougule, S. V., & Chavan, M. S. (2015). Robust spectral features for automatic speaker recognition in mismatch condition. *Procedia Computer Science*, *58*, 272–279.

Chung, J. S., Nagrani, A., & Zisserman, A. (2018). VoxCeleb2: Deep speaker recognition. *Interspeech 2018*.

Docio-Fernandez, L., & Garcia-Mateo, C. (2009). Speaker segmentation. In S. Z. Li & A. Jain (Eds.), *Encyclopedia of biometrics* (pp. 1277–1284). Springer US.

Doddington, G. (1985). Speaker recognition—identifying people by their voices. *Proceedings of the IEEE*, *73*(11), 1651–1664.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern recognition letters*, *27*(8), 861–874.

Fernandes, J., Silva, L., Teixeira, F., Guedes, V., Santos, J., & Teixeira, J. P. (2019). Parameters for vocal acoustic analysis - cured database. *Procedia Computer Science*, *164*, 654–661.

Fernandes, J. F. T., Freitas, D., Junior, A. C., & Teixeira, J. P. (2023). Determination of harmonic parameters in pathological voices—efficient algorithm. *Applied Sciences*, *13*(4), 2333.

Guedes, V., Teixeira, F., Oliveira, A., Fernandes, J., Silva, L., Junior, A., & Teixeira, J. P. (2019). Transfer learning with audioset to voice pathologies identification in continuous speech. *Procedia Computer Science*, *164*, 662–669.

Hadsell, R., Chopra, S., & LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, *2*, 1735–1742.

Hermans, A., Beyer, L., & Leibe, B. (2017). In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*.

Hori, T., Cho, J., & Watanabe, S. (2018). End-to-end speech recognition with word-based rnn language models. *2018 IEEE Spoken Language Technology Workshop (SLT)*.

Hosseinzadeh, D., & Krishnan, S. (2007). On the use of complementary spectral features for speaker recognition. *EURASIP Journal on Advances in Signal Processing*, *2008*, 1–10.

Jahangir, R., Teh, Y. W., Nweke, H. F., Mujtaba, G., Al-Garadi, M. A., & Ali, I. (2021). Speaker identification through artificial intelligence techniques: A comprehensive review and research challenges. *Expert Systems with Applications*, *171*, 114591.

Jawarkar, N. P., Holambe, R. S., & Basu, T. K. (2011). Use of fuzzy min-max neural network for speaker identification. *2011 International Conference on Recent Trends in Information Technology (ICRTIT)*.

Jia, Y., Chen, X., Yu, J., Wang, L., Xu, Y., Liu, S., & Wang, Y. (2021). Speaker recognition based on characteristic spectrograms and an improved self-organizing feature map neural network. *Complex & Intelligent Systems*, *7*, 1749–1757.

Kersta, L. G. (1962). Voiceprint identification. *The Journal of the Acoustical Society of America*, *34*(5_Supplement), 725–725.

Kral, P. (2010). Discrete wavelet transform for automatic speaker recognition. *2010 3rd International Congress on Image and Signal Processing*.

Krishnamoorthy, P., Jayanna, H., & Prasanna, S. (2011). Speaker recognition under limited data condition by noise addition. *Expert Systems with Applications*, *38*(10), 13487–13490.

Lei, Y., Scheffer, N., Ferrer, L., & McLaren, M. (2014). A novel scheme for speaker recognition using a phonetically-aware deep neural network. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1695–1699.

Manfron, E., Teixeira, J. P., & Minetto, R. (2023a). Deep learning and machine learning techniques applied to speaker identification on small datasets. *III International Conference on Optimization, Learning Algorithms and Applications (OL2A 2023), Springer*.

Manfron, E., Teixeira, J. P., & Minetto, R. (2023b). Speaker recognition in door access control system. *In the third Symposium of Applied Science for Young Researchers - SASYR 2023*.

Mingote, V., Miguel, A., Ribas, D., Giménez, A. O., & Lleida, E. (2019). Optimization of false acceptance/rejection rates and decision threshold for end-to-end text-dependent speaker verification systems. *INTERSPEECH*, 2903–2907.

Mohd Hanifa, R., Isa, K., & Mohamad, S. (2021). A review on speaker recognition: Technology and challenges. *Computers & Electrical Engineering*, *90*, 107005.

Nagrani, A., Chung, J. S., Xie, W., & Zisserman, A. (2020). Voxceleb: Large-scale speaker verification in the wild. *Computer Speech & Language*, *60*, 101027.

Nagrani, A., Chung, J. S., & Zisserman, A. (2017). VoxCeleb: A large-scale speaker identification dataset. *Interspeech 2017*.

Naik, J. (1990). Speaker verification: A tutorial. *IEEE Communications Magazine*, *28*(1), 42–48.

Nussenzveig, H. M. (2018). *Curso de física básica: Fluidos, oscilações e ondas, calor* (Vol. 2). Editora Blucher.

P, B. K., & M, R. K. (2020). ELM speaker identification for limited dataset using multitaper based MFCC and PNCC features with fusion score. *Multimedia Tools and Applications*, *79*(39-40), 28859–28883.

Reynolds, D., & Rose, R. (1995). Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing*, *3*(1), 72–83.

Reynolds, D. (2009). Gaussian mixture models. In S. Z. Li & A. Jain (Eds.), *Encyclopedia of biometrics* (pp. 659–663). Springer US.

Reynolds, D. A., Quatieri, T. F., & Dunn, R. B. (2000). Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, *10*(1-3), 19–41.

Sahidullah, M., & Saha, G. (2013). A novel windowing technique for efficient computation of MFCC for speaker recognition. *IEEE Signal Processing Letters*, *20*(2), 149–152.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 815–823.

Sharma, V., & Bansal, P. (2013). A review on speaker recognition approaches and challenges. *International Journal of Engineering Research and Technology (IJERT)*, *2*(5), 1581–1588.

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

Speaker detection. (2009). In S. Z. Li & A. Jain (Eds.), *Encyclopedia of biometrics* (pp. 1253–1253). Springer US.

Speech. (2023). In *Oxford English Dictionary*. Oxford University Press.

Teixeira, J. P. (2013). *Análise e síntese de fala-modelização paramétrica de sinais para sistemas tts*. Editorial Académica Española.

Teixeira, J. P., & Freitas, D. (2003). Segmental durations predicted with a neural network. *Proceedings of Eurospeech'03 – International Conference on Spoken Language Processing*, 169–172.

Tolba, H. (2011). A high-performance text-independent speaker identification of arabic speakers using a CHMM-based approach. *Alexandria Engineering Journal*, *50*(1), 43–47.

Villalba, J., Chen, N., Snyder, D., Garcia-Romero, D., McCree, A., Sell, G., Borgstrom, J., Richardson, F., Shon, S., Grondin, F., Dehak, R., García-Perera, L. P., Povey, D., Torres-Carrasquillo, P. A., Khudanpur, S., & Dehak, N. (2019). State-of-the-art speaker recognition for telephone and video speech: The JHU-MIT submission for NIST SRE18. *Interspeech 2019*.

Wang, C., Wang, Y., Chen, Y., Liu, H., & Liu, J. (2020). User authentication on mobile devices: Approaches, threats and trends. *Computer Networks*, *170*, 107118.

Zergat, K., Selouani, S., & Amrouche, A. (2018). Feature selection applied to g.729 synthesized speech for automatic speaker recognition. *2018 IEEE 5th International Congress on Information Science and Technology (CiSt)*.

Zhang, Z. (2016). Mechanics of human voice production and control. *The Journal of the Acoustical Society of America*, *140*(4), 2614–2635.

Zhou, T., Zhao, Y., & Wu, J. (2021). Resnext and res2net structures for speaker verification. *2021 IEEE Spoken Language Technology Workshop (SLT)*, 301–307.