UCLouvain

FACULTE DES SCIENCES

Faculté des sciences

Sentiments analysis in speech data

Auteur·es : Geeraerd Arthur
Promoteur·rices : Dr. Robin Van Oirbeek, Dr. Andrea Pennisi
Lecteur·rices : Dr. Johan Segers
Année académique 2022-2023

UNIVERSITÉ CATHOLIQUE DE LOUVAIN

Abstract

LSBA
Ecole de Statistiques, Bio-Statistiques et Sciences Actuarielles

Master Data Sciences Student

Sentiment Analysis

by Arthur GEERAERD

This master thesis explores sentiment analysis in speech data, focusing on its intricacies and the methodologies that can optimize its accuracy. Initial sections delve into the area of signal processing, studying sound properties and the application of transforms like the Fourier transform and its derivatives. A comprehensive literature review provides insights into the current state of the art and the advancements made in this domain.

In the methodology section, emphasis is placed on the techniques of feature extraction pivotal for sentiment analysis. This involves an in-depth examination of feature selection and dimensionality reduction methods. The study evaluates different models, notably the Support Vector Machine (SVM), Multilayer Perceptron (MLP), and MLP regression, in their effectiveness for sentiment classification.

The application of this research, utilizing features such as the first 20 Mel-frequency cepstral coefficients (MFCC), zero crossing rate, and root mean squared, revealed SVM to be particularly potent. When classifying sentiments into categories like "sad", "happy", "angry", and "calm", SVM achieved an accuracy of 85%, significantly outperforming the MLP which registered at 74%. This was tested on a mixed dataset combining RAVDESS and SAVEE.

The findings of this study underscore the potential of SVM in sentiment analysis of speech data and pave the way for further research to enhance accuracy and utility in real-world applications.

Acknowledgements

To my esteemed promoters, Dr. Robin Van Oirbeek and Dr. Andrea Pennisi,

My journey through this thesis was marked by moments of doubt and discouragement, and in those moments, both of you stood by me with unwavering patience and endless encouragement. Your combined guidance, expertise, and wisdom not only shaped this work but also molded my academic and personal growth. I am profoundly grateful for the knowledge you shared and the faith you placed in me. Your belief in my potential, even when I faltered in my own belief, was the catalyst that propelled me forward. Thank you, from the depths of my heart, for being more than just promoters — for being mentors, guides, and stalwart supporters.

A heartfelt thank you to Simon and Adrien, for our collaborative efforts, camaraderie, and the invaluable experiences we have shared during the completion of this master degree.

To my friends and family, thank you for your unwavering belief in me, for offering emotional support during challenging times, and for reminding me of the importance of balance.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

In today's world, we are witnessing an exponential increase in interactions between man and machine. This is largely due to the rapid technological progress and expansion of artificial intelligence in our daily lives. Whether in the home with voice assistants, at work with voice recognition software, or even when travelling with navigation systems, we are increasingly required to communicate vocally with machines.

In this context, the ability of machines to understand and respond to our emotions could offer considerable advantages. Emotions play a central role in human communication and greatly influence the way we perceive and react to our environment. In other words, emotions are an integral part of our lived experience. Therefore, if a machine is able to recognise our emotions, it may be able to provide more tailored and personalised responses to our needs.

More specifically, identifying emotions by voice could prove particularly beneficial in a number of areas. For example, in the emergency services sector, a system capable of detecting stress or panic in a caller's voice could help prioritise calls and deploy help more quickly. Similarly, in the context of helplines for people in emotional distress, such as SOS Suicide, recognition of vocal emotions could help identify the most urgent calls and provide appropriate support. In the commercial sector, the detection of emotions in telephone calls could help to improve customer service, enabling companies to understand and respond more effectively to their customers' feelings. Furthermore, the analysis of vocal emotions could also offer valuable insights for the evaluation of commercial calls.

In short, as voice communication with machines becomes more widespread, recognising emotions in the voice becomes a priority. This field of study promises many practical applications, from improving customer service to helping people make decisions in emergency situations.

1.2 Research Objectives and Questions

In an era of accelerating digitalisation, human-machine interaction has continued to develop, becoming an inexorable part of our daily lives. Verbal communication, intrinsically linked to our humanity, is the focus of increased attention, particularly with the growing development of voice interfaces. As we look to the near future, when we'll be interacting more and more with machines by voice, the prospect of incorporating a degree of sensitivity to our emotions into them could constitute a veritable revolution.

The recognition of emotions by spoken language has the potential to considerably improve the efficiency and humanity of many services, such as emergency call classification and prioritisation systems), after-sales services or even the analysis of cold calls to customers.

The voice, this extremely sophisticated communication tool, very often serves as a vector for our emotions. It therefore seems particularly interesting to formalise the criteria for categorising these emotions. In this way, voice analysis could offer a new dimension to artificial intelligence, improving its ability to understand and respond to users' needs.

Speech Emotion Recognition (SER) is a fast-growing area of research, and the number of studies in this field is constantly growing. In this context, we propose to carry out our own study, aimed at understanding and exploiting this phenomenon, with the ambition of contributing to the evolution of this promising technology.

Chapter 2

Background

The main goal of this section is to present the tools needed for signal definition and processing in general. These methods are widely used in other sectors, such as the analysis of electroencephalograms, electrical signal analysis and time series.

2.1 Sound Signal and Waveform

The data used in this research is a sound sample. Understanding how this signal is composed is essential for its processing and, it will also allow to understand the limitations and the performances of the proposed classification models with regard to the specificities of each of them.

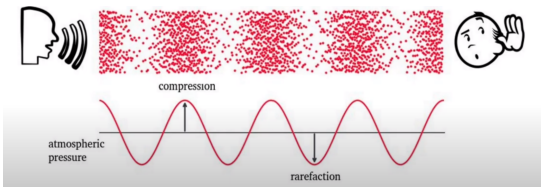


FIGURE 2.1: raw sound signal

Therefore, we define the notion of sound as the product of the vibration of an object. In the case of the human voice, it is the vibrations emanating from the air sent in the vocal cords. Furthermore, the modulation of the oral cavity is also important. These emitted vibrations cause the oscillation of the medium in which they are propagated, in our case: the air molecules. The oscillation is characterized by periodical pressure changes in the air that creates the waves. In Figure 2.1, we see the speaker that emits a sound signal, the air molecules that compress and release periodically. Corroborating this, the graph shows the air pressure as a function of time. We can therefore formalize that sound is made of a mechanical wave that has the following properties:

- Oscillations that travels trough space

- Energy levels going from one point to everywhere in space linearly
- The medium (air) is deformed

In order to understand the structure of a waveplot, Figure 2.2 shows the structure of a 2.5 seconds voice sample taken from the RAVDESS database.

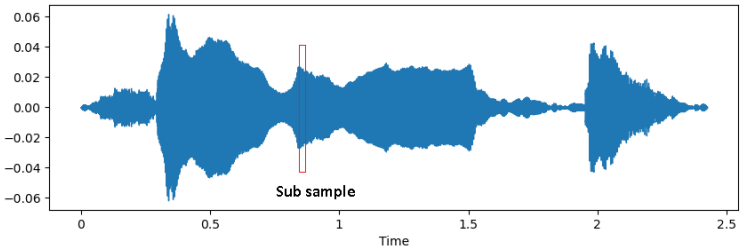


FIGURE 2.2: waveplot of a sound sample

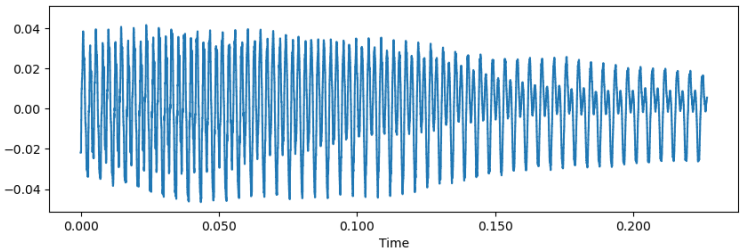


FIGURE 2.3: zoomed in waveplot

We can therefore take a closer look at what makes up our signal in Figure 2.3. In order to understand its structure, we introduce the notion of frequency, amplitude and phase. These three parameters allow us to formalize the structure of this signal.

The sine function serves as the most fundamental and simplified representation for a synthetic periodic signal:

$$S(t) = sin(2\pi \cdot (f_t - \rho)) \tag{2.1}$$

When visualizing this function, the resulting graph exhibits the following characteristics:

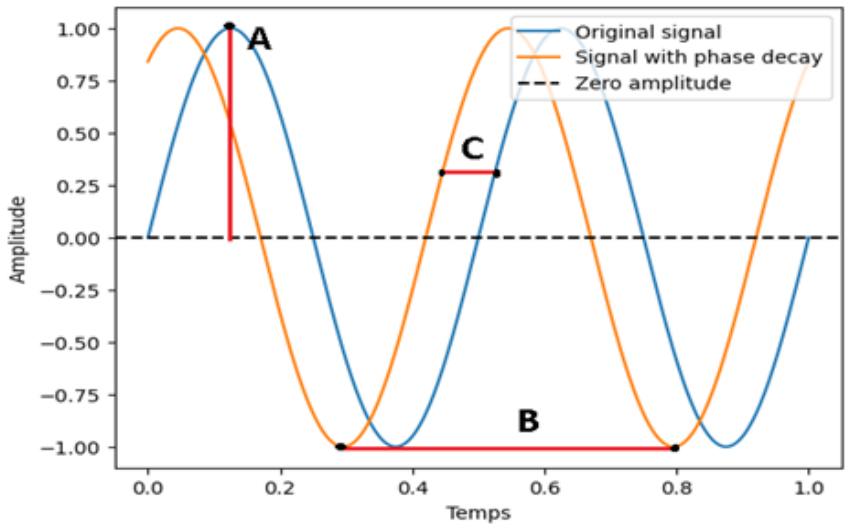


FIGURE 2.4: raw sound signal

Where :

- **A** : is the amplitude argument. The greater the amplitude (resp. the smaller), the louder the sound (resp. the weaker).
- **B** : is the period parameter. It represents the time that the wave takes to make a complete oscillation, to return to its initial state. We can obtain the value of the frequency on the basis of the period since the frequency is the number of complete oscillations executed on during one second: $f(t) = 1/T$ where T is the period.
- **C** : is the phase parameter: ρ. It gives the offset from the signal that passes through the y-intercept.

A real signal of a human voice however, is a combination of several sine functions. Indeed if we aggregate several sine functions of different frequencies, amplitude and phase we can then find something visually more similar, such as:

$$S(t)_1 = 0.01 \cdot sin(2\pi \cdot (20 - 0))$$

$$S(t)_2 = 0.005 \cdot sin(2\pi \cdot (60 - 0))$$

$$S(t)_3 = 0.005 \cdot sin(2\pi \cdot (22 - 0.2))$$

$$S(t)_{aggregated} = S(t)_1 + S(t)_2 + S(t)_3$$

That gives us :

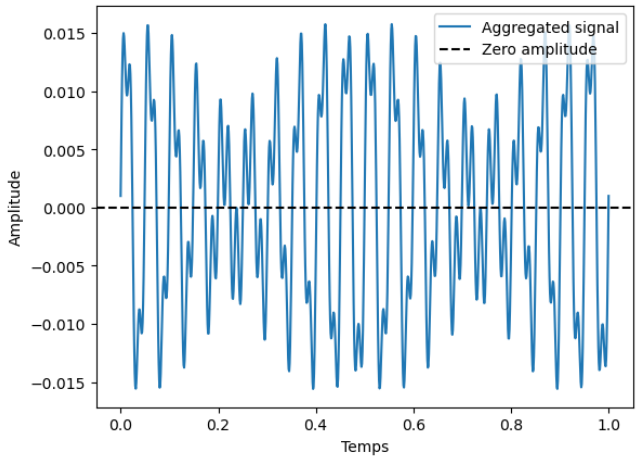


FIGURE 2.5: aggregated synthetic signal

This seems to correspond more to the sound signal present in the human voice with regard to Figure 2.3. In order to relate these tree concepts (amplitude, frequency and phase) to common notions, a high amplitude (resp. low)

sound is one characterized by a loud (resp. soft) sound. Also, a sound with a high (resp. low) frequency will be perceived as high (resp. low). The phase constant has no direct sensory purpose except to synthesize the signal.

The intensity of the sound signal is a fundamental measure in acoustics, as it translates the amount of energy transmitted per unit area. To describe it, we must use the concept of sound power which is defined as the amount of energy transferred by the sound per unit of time emitted by the sound source in all directions. This energy is measured in Watt (W). The sound intensity is simply the sound power per unit area (W/m^2). The intensity is proportional to the amplitude of the signal. The more the signal is intense means that it is made of waves with a larger amplitude. This is our measure of the energy that reaches our ear. This measure is very important in the study of the sound signal because we will see that the human ear does not treat the intensity variations in a linear way. In fact, we treat the intensity variations in a logarithmic way. For such a reason, the unit decibel was proposed.
The decibel is a relative measure of sound intensity. This means that it is based on a completely arbitrary measure, i.e. the threshold of hearing or (TOH) which characterizes the minimum threshold of hearing of the human ear. It is estimated at TOH = 10-12W/m^2. Beside that, it was defined, just as arbitrarily, that the threshold of pain (TOP) is equivalent to 10 W/m^2. In order to calculate the decibel value of a sound intensity and taking into account the ability of the human ear to capture the intensity variations, the decibel scale is defined as follows:

$$dB(I) = 10.log_{10}\left(\frac{I}{I_{TOH}}\right) \tag{2.2}$$

Where :

- I is the intensity of the signal
- I_{TOH} is the threshold of hearing $\sim 10^{-12} W/m^2$.

In this way, our initially linear scale is transformed such that :

From Figure 2.6 it is obvious that this relationship is absolutely not linear. This aspect is duly noted for consideration in the forthcoming preprocessing step.

2.2 Digital Audio Signal

In this section, we are interested in how sound is recorded and stored. Then, we will look at the three general categories of processing that we can use to apply

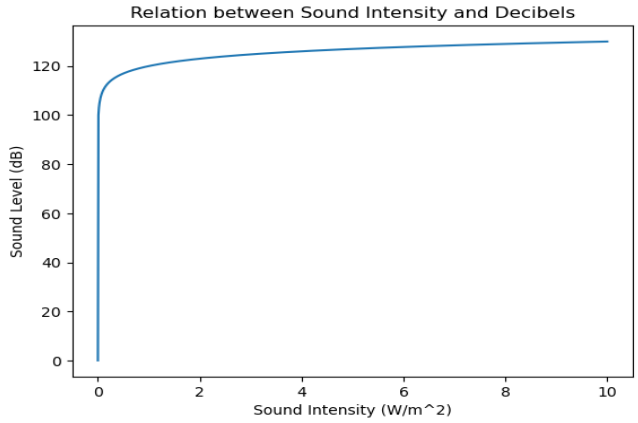


FIGURE 2.6: linear vs decibel scale for amplitudes

machine learning technique. The sound is a machanical wave that propagates in the air. These oscillations are sinusoidal and therefore continuous in time. However, if we want to digitize this signal, we cannot keep it in its continuous form. In fact, when we record voice samples to transform them into a digital signal, we are actually capturing sequences of discrete values sampled at equal time intervals, see Figure 2.7 as an example:

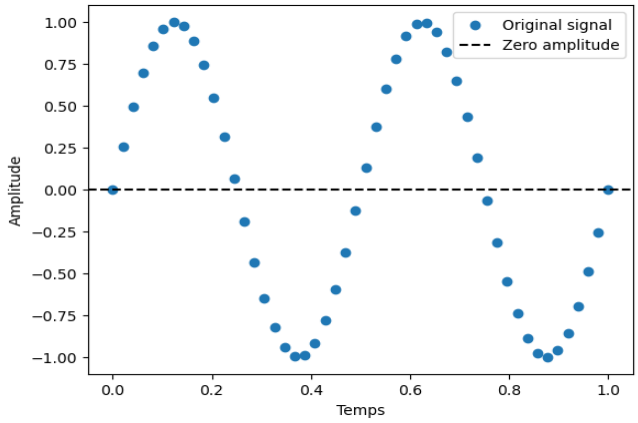


FIGURE 2.7: digital signal

The above example represents a signal that lasts 1 second and has 50 sampling points. This means that :

- T : the sampling period, lasts 1/50 *seconds*
- t_n : the snippet taken at the nth time step = $n.T$
- S_r : the sampling rate = $1/T$.

These notions are essential for a good understanding of the following signal processing. Also, we can see that the sampling rate is very important in the

digital retranscription of the continuous signal. Indeed, if our original signal is sampled at a rate that is too low, we might lose information.

The notion of an "ideal" sampling rate is worthy of exploration. In order to answer in the best way, we introduce the notion of Nyquist frequency noted f_N. The Nyquist frequency is expressed as

$$f_N = S_r/2 \tag{2.3}$$

Since we are now dealing with points and no longer with a continuous curve, we need to be as sure as possible of what is happening between the points. Otherwise we could be confronted with the phenomenon of alliasing. This occurs when a signal is sampled at a rate that is too slow to correctly represent the original signal. To prevent this, the Nyquist Frequency indicates that you should sample twice as fast as the maximum frequency value you can expect to have. This explains why CD (Compact Disc) have a sampling rate of 40100Hz. Indeed, the human ear covers a frequency spectrum from 20HZ to 20kHz (Sharma et al., 2020). Therefore, the highest frequency that can be represented by a CD is $40100/2 = 20500$Hz, which corresponds to the maximum frequency of the human ear.

2.3 Fourier Transform

The purpose of this transformation is to convert our waveplot (Fig. 2.8), which represents the intensity of the signal captured over time, into a graphical representation that elucidates the magnitude of all frequencies within the audio sample's spectrum.

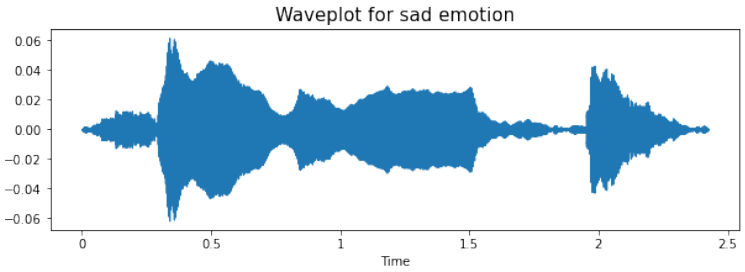


FIGURE 2.8: Representation of the vocal spectrum.

We will illustrate these freshly introduced concepts by analyzing in detail a sample from the database representing a sad emotion and spoken by an actress.

"Kids are talking by the door". We see in 2.8 that the sample lasts about 2.5 seconds. This graph represents the amplitude of the sound as a function of time. The stronger the signal, the greater the amplitude and the more the frequency is acute, the more the oscillations are short.

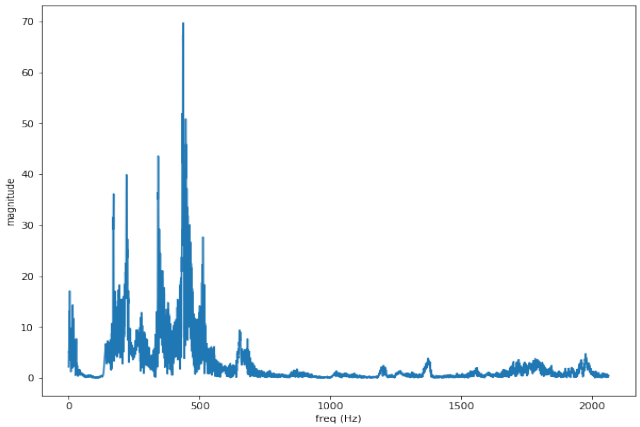


FIGURE 2.9: Sample of voice in frequency domain.

Figure 2.9 represents the above voice sample in the frequency domain. We can see that the vocal spectrum mainly exploited is between 100 and 750Hz, with some frequencies very present in this spectrum. Also note the presence of frequencies between 0 and 30 Hz. These are either the result of an artifact or noise during the recording

The concept of the Fourier transformation will now first be introduced using an intuitive explanation, by manipulating the signal sample in order to obtain the amplitude value for a well-represented frequency. This will be followed by a more conceptional demonstration, as well as a discussion of the implications for further pre-processing.

In order to obtain these magnitude values as a function of frequency we will need to decompose the initial signal into several steps. First, we will zoom in on the sample in order to perceive the oscillations of the sound signal with the naked eye. We will then estimate the phase and frequency parameters in order to find a synthetic signal that matches well with our sample sequence. This is necessary to obtain the magnitude for a frequency f.

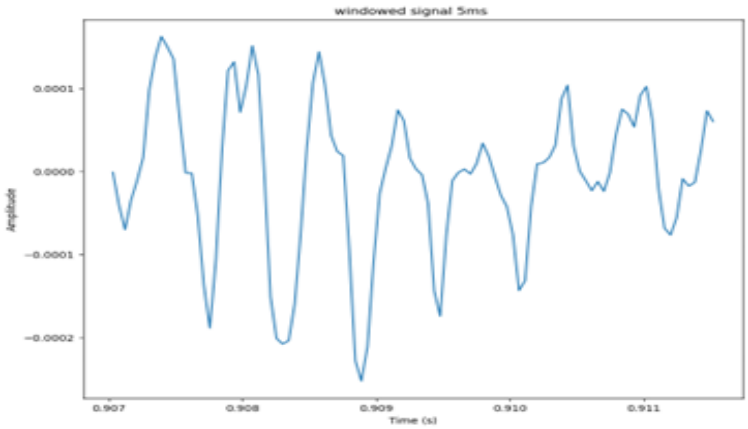


FIGURE 2.10: Sample of voice in frequency-domain.

Figure 2.10 : is our voice sample from 0.907 seconds to 0.912 seconds. It is composed of 100 points of different amplitudes. We see in this zoomed view, the sinusoidal characteristic of the contained waveform. We will define what the phase and frequency components are. Then, we will estimate their value by hand.

Considering the presented by Ma et al., 2010, we can assume that the frequency is stationary as a function of time. Indeed, this hypothesis can be supported given the duration of the windowing applied to the signal. Thus, it allows us to ignore frequency variations over time.

A sinusoidal signal can be modeled as follows :

$$S(t) = sin(2\pi \cdot (f_t - \rho)) \tag{2.4}$$

Where :

- f_t is the frequency given the time.
- ρ is the phase component of the signal.

Once the values of phase and frequency are found, the magnitude of frequency f in the signal sample is obtained as:

$$\rho_f = argmax_{\rho \in [0,1)} \left(\int s(t) \cdot sin(2\pi \cdot (ft - \rho)) \cdot dt \right) \tag{2.5}$$

As such, we can find the phase that maximizes the cumulative positive area between the signal $s(t)$ and the pure tone obtained with equation (2.1). This process is repeated for a continuous number of frequencies. The magnitude of a frequency f then corresponds to:

$$d_f = \max_{\rho \in [0,1)} \left(\int s(t) \cdot sin(2\pi \cdot (ft - \rho)) \cdot dt \right) \tag{2.6}$$

It can be shown that the complex coefficient of Fourier for a defined frequency can be obtained via :

$$\hat{g}(f) = g(t) \cdot e^{-i2ft} dt \tag{2.7}$$

Where :

- $\hat{g}(f)$ is the complex Fourier coefficient for the frequency f
- $g(t)$ the signal
- e^{-i2ft} the imaginary component

This formulation uses the notion of imaginary numbers and is therefore by nature more intricate but has the advantage of not requiring any adjustment of the phase parameter to find the magnitude of f.

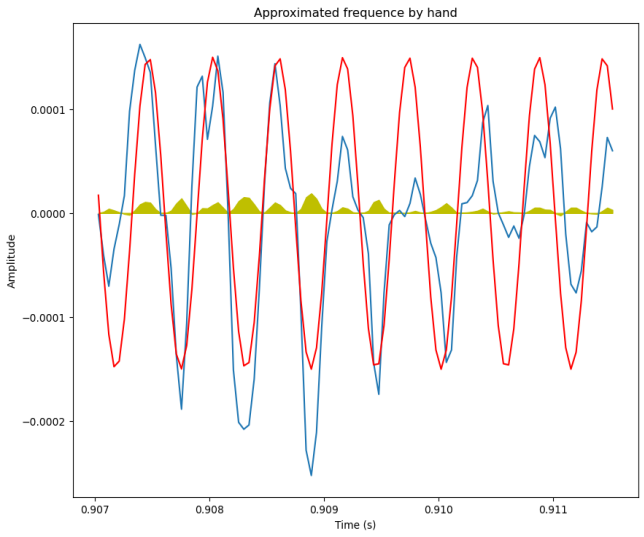


FIGURE 2.11: synthetic signal matching the basic one

We must find a value of frequency and phase that allows us to match the two signals: sample and synthetic. We find the component of phase = 0.85 and frequency = 1761. Result shown in Figure 2.11, where the synthetic signal is shown in red and the basic signal in blue. Both signals overlap relatively well.

Moreover, the green area (cumulative area) is always positive, which indicates a good alignment between the two signals. The reader will find in Appendix 1 the graph of a bad alignment.

Using Equation (2.7), we find for the frequency f = 1761 the magnitude of 0.0046. We compare the results obtained via the numerical method of the Fourier transform below:

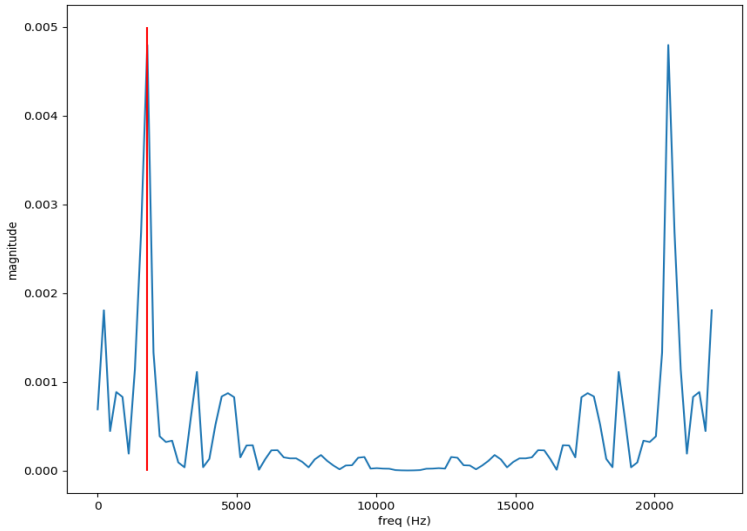


FIGURE 2.12: magnitudes of f = 1761

The vertical line in red indicates the frequency value for f = 1761. This one is indeed equal to 0.005. Of course, this is only the magnitude analysis for a single frequency value. This is replicable at all frequency values present in the spectrum.

Discrete Fourier Transform

In the computation process to determine the magnitude corresponding to each frequency, we make two mathematical assumptions which do not hold true in the context of dealing with discrete values:

- f: The frequency is treated as a continuous variable which implies that we are handling an analog signal. However, this assumption is incorrect since we are in fact dealing with a digital signal, which is inherently discrete in nature. (see Section 2.2).

- The signal is assumed to be stationary, which means that its properties do not change over time. In reality, speech and other forms of audio data are non-stationary signals, meaning their properties do vary over time. This discrepancy may impact the accuracy of our frequency analysis.

To do this, we will formalize two things:

1. We consider the frequency as non-negative in a finite time interval $[0, N]$'. Thus, during the whole duration of the voice sample.

2. We look for the magnitude of a finite number of frequencies f and we assume that the number of frequencies is equal to the number of points in the sample. So, $T \times S_r$ or S_r is the sampling speed. In speech processing, we expect at least a sampling around 14000 samples per second.

We therefore transform the previous expression (2.7) but as we are dealing with a digital (or discrete) signal, we need to use the Discrete Fourier Transform (DFT), which can be written as:

$$\hat{x}(n) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi fn} \tag{2.8}$$

In the DFT, the integral in the continuous Fourier Transform is replaced by a summation since we now have a set of discrete points in the time domain. The limits of the summation are from 0 to $N-1$, where N is the total number of samples in the discrete time signal $x(n)$.

And in view of the clarifications made above, we transform the expression as follows:

$$\hat{x}(k/N) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi fk/N} \tag{2.9}$$

Now, we substitute f with $\frac{k}{N}$ where k is the frequency bin (from 0 to $N-1$) and N is the total number of samples. This effectively scales the frequency term to the range [0, 1].

$$\hat{x}\left(\frac{k}{N}\right) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi \frac{k}{N} n} \tag{2.10}$$

Where :

- $f(k)$ = k/NT = $k/N \times S_r$ and T equals sampling period
- K = [0,N] = [0,M] where N equals # of sample serving as an approximation of the frequency

This allows us to process the digital signal of our samples in a mathematically rigorous way and always based on what has been stated in Equation 2.7. However, the Discrete Fourier Transform method is subject to a problem. When we calculate the frequency magnitudes, we find a mirrored distribution of magnitudes around the axis k = N/2 (see Figure 2.12).

Also, the discrete Fourier transform is a very computationally intensive method. In the rest of the preprocessing, we will use the Fast Fourier transform algorithm which is a numerically optimized version but based on the same principles.

We now have a digital signal, processed in a digital way. We must now work on the signal processing for the whole voice sample. Indeed, with the Fourier transform method, we assume that the signal is stationary in time. This implies that the signal properties (frequency and amplitude) do not change in time. However, this is not the case.

Therefore, maintaining the quasi-stationarity assumption is essential for successful signal analysis. It is a crucial aspect of our method that must be addressed carefully. To do so, we have dedicated the subsequent section specifically to the techniques and strategies used to ensure that this condition is met for each sub-sample of our voice signal data.

Discrete Cosine Transofm

The Discrete Cosine Transform (DCT) is a mathematical tool that's primarily used for converting a signal from the spatial or temporal domain into the frequency domain. In essence, it's a Fourier-like transform but it only uses cosine functions, avoiding complex numbers and better approximating the way human perception of audio and images works.

The general formula for the DCT for a discrete signal of length N is as follows:

$$DCT[k] = \sum_{n=0}^{N-1} x[n] \cos\left[\frac{\pi}{N}(n+0.5)k\right] \tag{2.11}$$

In this formula:

- $x[n]$ represents the n^{th} sample in the input signal
- N is the total number of samples in the signal
- k is the index of the DCT coefficient being computed
- The sum is taken over all n from 0 to $N-1$.

Each term in the sum is a cosine wave at frequency k multiplied by the signal value $x[n]$ at time n. The summation essentially "projects" the signal onto the cosine wave, calculating how much of the signal's energy lies at the cosine wave's frequency.

There are several types of DCTs (I, II, III, and IV) that differ slightly in their mathematical definitions. The DCT-II, often simply referred to as the DCT, is the most commonly used form, particularly in signal processing applications. The formula given above is for the DCT-II.

Short Time Fourier Transform

Short-Time Fourier Transform (STFT) is a signal processing technique widely used in voice processing. More generally, it is an essential step in obtaining key metrics to classify emotions. As mentioned above, we do not want to process the whole signal at once. We seek to process the signal in small subsets of time in order to find stationarity in the voice (Akçay et al., 2020). The STFT allows us to convert our temporal signal into a frequency-time signal by calculating the Fourier transform on small windows of our sample. Therefore, thanks to the STFT we obtain frequency magnitudes of small sections of our initial signal. The spectrogram of our sample is represented in Figure 2.13.

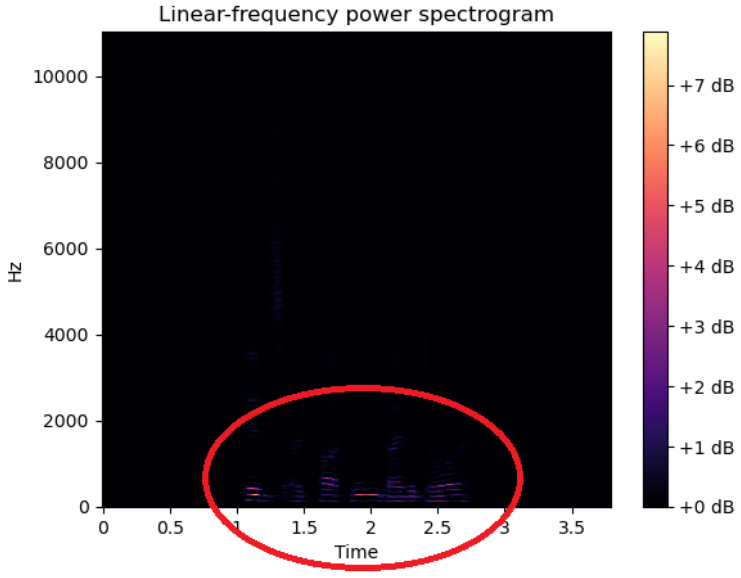


FIGURE 2.13: Spectrogram, linear Scale

Thanks to the color scale in Fig. 2.13, we can see that depending on the time window we are in, the frequency magnitudes vary. We oscillate between 0db and 7db, this representing the intensity of the magnitudes present for each frequency in each sub time interval.

Thus, we begin by considering the expression of the Discrete Fourier Transform (DFT) as shown in Equation (2.10). Our objective is to derive an expression that incorporates information about the specific frames of the signal, enabling us to calculate the complex Fourier coefficients accordingly.

As such, the window function corresponds to :

$X_w(k) = x(k) \cdot w(k)$ where $X_w(k)$ is the signal present in the window w.

The windowing function allows us to select the sub interval of time in which we study the signal and to make the original signal equal to 0 everywhere else. Thanks to this windowing function, we obtain the following expression:

$$S(m,k) = \sum_{n=0}^{N-1} x(n+mH) \cdot w(n) \cdot e^{-i2\pi fk/N} \tag{2.12}$$

Where :

- m : time operator, the frame we are currently in.
- H : hop parameter, the offset of the window of interest.
- mH : starting sample of the current frame.
- $w(n)$: the windowing function.

By now, we passed from the discrete Fourier Transform where we had a spectral vector for all the signal (depending on the number of frequency bins) of complex coefficient to the Short-Time Fourier Transform that is a spectral matrix (depending on the number of frequency bins, depending on the number of frames) of complex Fourier Coefficients

We obtain the Figure 2.13. However, when we look at it more closely, we can see that the magnitudes of high frequencies are completly erased and even low frequencies are not very well represented. Indeed the magnitudes themselves and their variations are not visible on the spectrogram with a linear scale. We will see in the next section that a correction can be made to make our spectrogram much more nuanced. To do this we will base ourselves on the way the human ear perceives sounds.

Linear scales vs Log scales

As explained in Section 2.1 , the human ear perceives the intensity present in the sound signal logarithmically. Conversely, this influences the way we vary the intensity of the signal when we speak. Therefore, if we want our spectrogram to represent both the way we emit sounds and the way we perceive them, we must use the decibel scale. Indeed, since this scale is non-logarithmic, it will tend to emphasize high frequencies of the signal which are, in the basic spectrogram, completely erased by the linear scale. Applying the logarithmic transformation

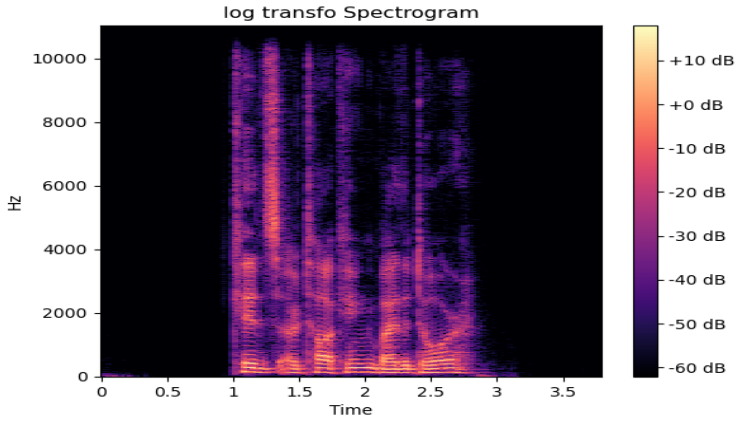


FIGURE 2.14: Spectrogram, dB scale

we obtain the spectrogram depicted in Figure 2.14

In studying speech emotion recognition, it's important to understand sound signals. These signals can be viewed as waveforms, which show sound changes over time. When we use computers, we change these signals into digital form. Tools like the Discrete Fourier Transform and the Discrete Cosine Transform help us see these signals in a different way, by showing their frequency. The Short Time Fourier Transform is especially useful for looking at speech. Also, when we look at these signals, we can use either linear or logarithmic scales. Both have their uses, but they show the data differently. Overall, these tools and ideas are key to understanding and working with speech and emotions.

Chapter 3

Litterature Review

3.1 Introduction

This section presents the various scientific studies that revolve around the research topic of emotion classification in the human voice. We approach each element sequentially in order to draw the logical red thread that leads to a good understanding of the knowledge needed for this purpose. First, we will look at Emotion Theory. In this part, we will present the different ways of conceiving emotions based on the literature. We will then present the two main models that allow us to differentiate between emotions. We will then address the subject of data collection. Due to the highly subjective nature of emotions, we need to be careful when recording voice samples. We also need to be careful about the method of labelling emotions in speech. And then we'll discuss the different types of dataset that exist, and what their advantages and disadvantages are. The next section covers all the theory involved in analysing emotions in speech. Feature extraction and selection being a key aspect of the field, we review the main categories of features in order to provide the most complete information possible. Finally, we present an outline of previous work to show the techniques already in place. From the most basic to the most complex, always giving as much information as possible on the types of features used, the models implemented and the results obtained.

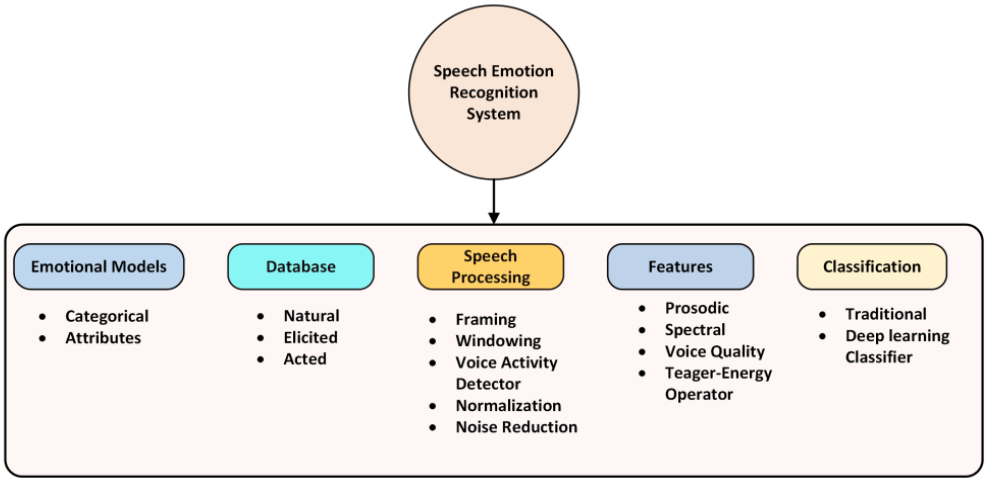


FIGURE 3.1: General SER system (Wani et al., 2021)

3.2 Emotion Theory

In the upcoming section, we delve into emotion theory, examining traditional models and their evolution. We focus on the categorical and dimensional models, commonly utilized for emotion classification. The categorical model underscores discrete emotion categories, including the basic groups – anger, disgust, fear, joy, sadness, and surprise – and additional expressive categories. The dimensional model, characterized by valence, arousal, and dominance, offers a contrast by positioning emotions along these dimensions. We critically assess both models, discussing their benefits and limitations. Our goal is to elucidate their functionality and constraints, contributing to the ongoing task of understanding and classifying emotions.

3.2.1 An Historical Overview

Charles Darwin's evolutionary theory of emotions posits that emotions are adaptive responses to the environment, developed by humans and animals to survive and procreate (Hess et al., 2009). This theory suggests that emotions like love motivate mate-seeking, and fear triggers responses to threats, enhancing survival chances.

The James-Lange theory, one of the earliest physiological theories of emotion, asserts that emotions arise from bodily reactions to events (James, 1884). For example, encountering a bear would trigger an increased heart rate, which

is then recognized as fear.

Contrarily, the Cannon-Bard theory proposed by Walter Cannon and Philip Bard, asserts that emotions and physiological reactions occur simultaneously (Cannon, 1927). Upon reacting to a stimulus, the brain's thalamus triggers a physiological response and simultaneously processes signals to generate the emotional sensation.

Lastly, the Schachter-Singer theory, also known as the two-factor theory of emotion, combines elements from the James-Lange and the Cannon-Bard theories (Bąk, 2016). It suggests that similar physiological responses can produce different emotions, depending on cognitive interpretation and context.

3.2.2 Categorical Model of Emotion

The categorical model identifies emotions via descriptive words, tags or audio features, typically using six primary categories: anger, disgust, fear, joy, sadness, and surprise (Hess, 2017). Each emotion in this model carries unique traits that signify specific situations or reactions. Most affective computing studies have focused at least on these six basic emotions. However, in areas like education, a broader emotional spectrum may be needed, as students frequently experience emotions like boredom or joy, underlining the necessity for domain-specific classes.

The categorical model offers clear emotion labels, making it easier to understand and categorize human emotions. However, it also comes with limitations that we will delve bellow. As stated by PS et al., 2017, not all emotions are included, as they are grouped under one category. Due to cultural, environmental, and personal differences, the same emotional states can fall under different categories, complicating the categorization process. This might lead to inefficient emotion detection and forced choices among the presented categories, rather than allowing subjects to identify their own emotional state. Despite these challenges, the categorical model remains prevalent due to its simplicity, with variations based on the number of classes included.

Drawing upon the six-emotion categorical model, Table 3.1, as described in (Khalil et al., 2019) illustrates some discernible characteristics or traits across

Emotion	Pitch	Intensity	Speaking rate	Voice quality
Anger	Abrupt on stress	Much higher	Marginally faster	Breathy, chest
Disgust	Wide , downward inflections	Lower	Very much faster	Grumble, chest tone
Fear	Wide, normal	Lower	Much faster	Irregular voicing
Happiness	Much wider, upward inflections	Higher	Faster/slower	Breathy, blaring tone
Joy	High mean, wide range	Higher	Faster	Breathy, blaring timbre
Sadness	slightly narrower	Downward inflections	Lower	Resonant

TABLE 3.1: 6 basic emotions and their attributes (Khalil et al., 2019)

these fundamental emotions, thereby aiding in the understanding of unique elements inherent to each emotion category.

In evaluating the categorical emotion model, as delineated in PS et al., 2017, it is crucial to consider both its merits and drawbacks. These attributes play a substantial role in the model's applicability and overall effectiveness.

- **Pros:**
  1. *Intuitive and Accessible:* The categorical model simplifies emotions into understandable labels.
  2. *Universality:* The basic emotion categories are recognized across cultures, enhancing the model's global relevance.
  3. *Domain-specific Utility:* In domains like affective computing, this model offers practicality due to its well-defined classes.
- **Cons:**
  1. *Limited Scope:* Not all emotions are captured due to broad categorization, risking oversimplification.
  2. *Bias:* Cultural, environmental, and personal differences can lead to varied categorization of the same emotional state.

3. *Forced Choice:* The model's predefined categories may not align with subjects' true emotional states.
4. *Inflexibility:* The model's rigidity might not accommodate the fluidity of emotions.

Despite its limitations, the categorical model's simplicity and straightforward approach make it a prevalent choice in many research fields. However, the need for more nuanced and flexible models is evident to address to the diverse and complex landscape of human emotions.

3.2.3 Dimensional Model of Emotion

The Circumplex Model of Emotions, formulated by James A. Russell (Posner et al., 2005) in the 1980s, presents an innovative approach to map emotions. It employs a two-dimensional circular layout, portraying the relationship between various emotions and situating them within the dimensions of valence and arousal as depicted in Figure 3.2.

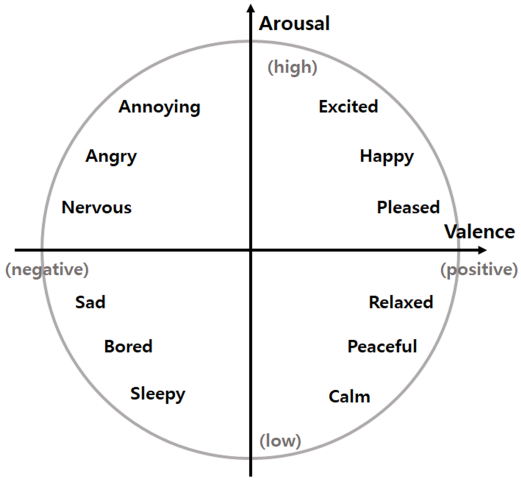


FIGURE 3.2: Circumplex model

The valence dimension indicates the positivity or negativity of an emotion. Emotions like joy and happiness are positioned on the positive end of the valence scale, representing positive emotions. Conversely, fear, anger, and sadness reside on the negative side, denoting negative emotions.

The arousal dimension, on the other hand, captures the intensity or calmness of an emotion. Emotions characterized by high energy like anger, excitement, or fear are situated on the high arousal end, whereas emotions linked with low

energy like sadness, relaxation, or boredom lie on the low arousal end.

The Circumplex Model, due to its intuitive and continuous structure, provides a more comprehensive illustration of emotional experiences, suggesting that emotions are not isolated but interconnected. This model, due to its visual representation, enables easier understanding of the complex landscape of human emotions.

Russell et al., 1977 have substantially enriched the dimensional model of emotion, thereby augmenting its representation of emotional states. Introducing the 'dominance' dimension, they've extended the model to capture the degree of control or influence an individual perceives over a situation or state. This advanced model is also known as the Pleasure-Arousal-Dominance (PAD) model, signifies a notable improvement to the original circumplex model, while maintaining its distinctive circular layout .

In conclusion, while the categorical model provides an accessible, universal framework for understanding emotions, it may oversimplify the complexity of emotional experiences. On the other hand, the circumplex model and its evolution into the PAD model offer a more nuanced, dynamic representation of emotions. However, the PAD model can be more challenging to interpret due to its increased complexity and the need to understand interactions across multiple dimensions. Despite these potential challenges, it enriches our comprehension of the intricate interplay of different emotional states.

3.3 dataset and data collection

In the field of speech emotion recognition, the foundation of any classification task is the dataset. This encompasses labelled data, and the quality of these labels, as well as the raw audio files, is essential for an effective classification process. The collection of accurate and useful data often presents a significant challenge due to the inherent subjectivity of emotions, making it a potentially demanding aspect of the research process.

According to Akçay et al., 2019, datasets used in this field can be broadly categorized into three types. Firstly, there are acted datasets, in which professional or semi-professional actors record utterances while portraying specific emotions. Still, it's crucial to note that these recordings are typically made in a

studio setting, resulting in clean audio samples that may not totally represent real-world conditions. An example of this kind is the RAVDESS dataset.

Secondly, we have elicited datasets. These are created by prompting individuals to express themselves freely, leading to a range of emotional displays based on the context and the individual's reactions. IEMOCAP (Busso et al., 2008) is an example of this dataset type.

Lastly, there are natural datasets, which comprise real-life recorded data, providing an authentic view of emotional expression. The MSP Podcast Corpus (*MSP-Podcast* n.d.) dataset serves as an example of this type.

Moreover, the demographic details of the participants, particularly their age and gender, are also crucial factors to consider. For instance, the RAVDESS dataset, which falls under the category of acted datasets, consists of 24 professional actors (12 male, 12 female) enacting various emotions in their utterances. Each sample is labelled through a consensus-based method involving over 200 North American individuals, representing a wide cross-section of the community. In this labeling process, an emotion label is assigned to a sample only when at least 75% of the participants agree on that label (Livingstone et al., 2018).

In the field of research, we are witnessing the emergence of larger datasets. For instance, the Large-Scale Emotional Speech Dataset (LSSED) includes over 206 hours of labeled audio content, produced by more than 820 speakers (Fan et al., 2021).

In sum, an in-depth understanding of the nature and quality of the dataset used is a prerequisite for effective research in speech emotion recognition, given the vital role it plays in the analysis and interpretation of results.

3.4 Preprocessing

Preprocessing represents a critical stage to competently address the classification problem, particularly considering the inherently raw nature of data derived from vocal samples. Preprocessing functions to transform the audio signal into a manageable format. Additionally, this stage plays a pivotal role in feature extraction, serving to identify and isolate valuable characteristics for further analysis (Zhou et al., 2016).

3.4.1 Framing

Framing is a technique employed in signal processing to convert a continuous time-domain signal into a sequence of fixed-length segments, an essential practice particularly when dealing with long segments of real-world data.

Achieving quasi-stationarity is one of the primary motivations for this practice, as it is generally agreed that speech signals exhibit stationary properties over short periods of time (typically around 20-50 ms) (Akçay et al., 2019).

In order to preserve signal continuity and avoid information loss at the boundaries of each segment, an overlap is usually applied between successive frames. This overlapping, commonly referred to as the "hop" parameter, typically ranges between 20% to 50% of the frame size.

Lastly, the segmentation of the signal into frames facilitates the use of spectral analysis techniques, such as the Discrete Cosine Transform (DCT), which are crucial for feature extraction in tasks such as speech emotion recognition.

3.4.2 Windowing

According to Wani et al., 2021, the windowing phase is an essential preprocessing step in speech emotion recognition that follows the framing of the speech signal. Its primary goal is to mitigate a phenomenon known as spectral leakage.

Spectral leakage is the distortion in the frequency domain that results from discontinuities at the edges of a framed signal when a Fast Fourier Transform (FFT, see Section 2.3) is applied. These discontinuities can blur the accurate representation of the signal's frequency content, which is critical when extracting features for speech emotion recognition.

A window function, applied to each frame, serves to attenuate or dampen the signal at its edges. This reduction in amplitude decreases the discontinuities, effectively curbing spectral leakage. There are various window functions that can be employed, each with unique characteristics and impacts on the frequency spectrum produced.

The Hamming window, expressed by the equation 3.1 :

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{M-1}\right), \quad 0 \leq n \leq M-1 \tag{3.1}$$

In equation 3.1:

- $w(n)$ denotes the window function
- M denotes the size of the window or frame
- n denotes the sample index within the frame

The Hamming window function tapers the signal to zero at the frame's edges, reducing boundary discontinuities and thus the spectral leakage. This process provides a more accurate frequency representation of the signal, enabling more effective feature extraction.

Apart from the Hamming window, there are other options such as Hann, Triangular, Gauss, Welch, Blackman, and Bartlett windows. The choice is dependent on specific analysis requirements (Sonmez et al., 2019).

3.4.3 Normalization

The process of feature normalization is a crucial stage in reducing the variability caused by individual speakers and varying recording conditions, while simultaneously maintaining the distinctiveness of the extracted features. By employing feature normalization techniques, the adaptability and generalization capabilities of these features are enhanced. The scope of normalization encompasses different scales, ranging from specific functions to the entire corpus. Z-normalization (standard score) is one of the most commonly applied methods in this context. It utilizes the mean μ and standard deviation of the dataset, allowing the computation of the normalized score (z) as z = (x - μ) σ (Akçay et al., 2019).

3.5 Feature Extraction

3.5.1 Prosodic Feature

The study of prosodic features, such as the Zero-Crossing Rate (ZCR), energy, and the pitch offers a nuanced understanding of the emotional undertones within human speech.

The ZCR measures the rate at which a signal transitions from positive to negative values and vice versa within a defined time period and provides valuable insights into the frequency content of the signal. Emotions such as anger or excitement can cause more abrupt changes in pitch and intensity, resulting

in a higher ZCR, while calm or sad emotional states may be associated with a lower ZCR (Akçay et al., 2019).

Energy and pitch are additional key prosodic features. Energy, measured as the sum of the squares of the amplitude values in a signal frame, reflects the intensity of speech and provides cues about emotional states. As seen in Table 3.1, high energy could be indicative of intense emotions like anger, while low energy might suggest a state of fear or disgust.

Pitch, the perceived frequency of a sound, varies within a single utterance depending on the emotional state of the speaker. High pitch might be associated with surprise or fear, while lower pitch could suggest sadness or boredom.

These prosodic features, while individually informative, offer the most insight when integrated and considered in tandem. They each contribute to the complex characterization of emotional states within human speech, reflecting the variety of emotions.

3.5.2 Spectral Feature

Human speech is produced by the vocal tract, and the shape of the vocal tract filters this sound. The resultant sound is determined by this shape, meaning an accurate simulation of the shape can result in an accurate representation of the vocal tract and the sound produced. Spectral features, which represent characteristics of the vocal tract, are obtained by transforming the time domain signal into the frequency domain signal using the Fourier transform (see Section 2.3). They are extracted from speech segments of 20 to 30 milliseconds, partitioned by a windowing method thanks to a method called Short time Fourrier Transform.

As Zheng et al., 2001 presents, Mel Frequency Cepstral Coefficients (MFCC) represent the short-term power spectrum of the speech signal and are obtained by dividing utterances into segments (i.e. framing), converting each segment into the frequency domain using short time discrete Fourier transform (see section, calculating a number of sub-band energies using a Mel filter bank, calculating the logarithm of those sub-bands, and finally applying inverse Fourier transform.

Linear Prediction Cepstral Coefficients (LPCC) embody vocal tract characteristics that vary with different emotions. They are obtained directly from Linear Prediction Coefficient (LPC) using a recursive method.

Log-Frequency Power Coefficients (LFPC) measure spectral band energies using Fast Fourier Transform to mimic the logarithmic filtering characteristics of the human auditory system. Ancilin et al., 2021 achieved a better accuracy with LFPC than with MFCC.

Gammatone Frequency Cepstral Coefficients (GFCC) are similar to MFCC, but apply a Gammatone filter bank to the power spectrum instead of Mel filter bank. Gammatone filter banks and their role in deriving acoustic features for speech recognition are extensively explored in the work of Yin et al., 2011.

In the work of Sugan et al., 2018, the authors derived features from speech samples, which included the first 16 MFCC and LPCC as baseline features. They further generated wavelet-based MFCC and LPCC features, referred to as WMFCC and WLPCC, respectively. These features were employed to create a Radial Basis Function Network (RBFNN) classifier. For the Emo-DB database, the reported average recognition accuracies were 59.57% for MFCC, 60.43% for LPCC, 75.37% for WMFCC, and 73.62% for WLPCC. When the same feature sets were applied to the SAVEE database, the recognition accuracies were slightly higher: 60.04% for MFCC, 60.64% for LPCC, 77.23% for WMFCC, and 74.70% for WLPCC.

3.5.3 Voice Quality

Voice quality features, notably the Harmonics-to-Noise Ratio (HNR), are vital in SER. According to Aouani et al., 2020, HNR measures the speech signal's harmonic to noise components. High HNR indicates clear speech, often linked to calmness or happiness. Conversely, a lower HNR, denoting more noise, is frequently associated with emotions like anger or fear. These features, though context-dependent, enhance the emotional decoding in SER systems.

3.6 Dimensionality reduction and Feature selection

Feature selection is a process that involves identifying and selecting a subset of valuable and pertinent features from a given dataset. This process seeks to eliminate unnecessary, redundant, or irrelevant attributes, thereby optimizing the accuracy of the predictive model by focusing on the most informative aspects of the data.

According to Özseven, 2019, it's been observed that Principal Component Analysis (PCA), Sequential Forward Selection (SFS), and Fast Correlation-Based Filter (FCBF) are often at the leading edge of feature selection techniques being applied.

In the field of SER, a variety of feature selection methodologies are applied. These include, but are not limited to, techniques such as Forward Feature Selection (FFS) and Backward Feature Selection (BFS), as referenced in numerous studies. Additional methods such as Sequential Floating Forward Selection (SFFS), and a wrapper approach incorporating forward selection, are also commonly used (Akçay et al., 2019).

As per the findings of Özseven, 2019, it is not definitively established that a specific feature selection technique consistently improves or diminishes the performance of Speech Emotion Recognition (SER). The impact of such methods on SER success is contingent upon multiple factors such as the classifier used, the nature of the data, and the rate of dimensionality reduction. Moreover, existing feature selection techniques have wide applications across numerous domains, including SER.

Another fresh and innovative approach in the domain of feature extraction involves the utilization of a pre-trained convolutional neural network (CNN), specifically AlexNet. The strength of this approach lies in the CNN's ability to autonomously extract pertinent local features from the augmented input spectrogram of a speech audio signal. When employed in SER system, the spectrogram (see Figure 2.14) is often the preferred input for the CNN, enabling the extraction of high-level features. This approach has gained substantial traction in recent years, as illustrated by various studies, including the work of Amjad,

Khan, and Chang Amjad et al., 2021.

Given this, conducting in-depth investigations into the efficiency of various methods based on different types of classifiers and datasets, as well as developing a feature selection method specifically designed for SER, could provide valuable guidance for future research in this field. These efforts are expected to contribute to advancing the knowledge and techniques in the SER domain.

3.7 Previous work

The methodology, as described by Dahake et al., 2016, revolves around the extraction of features, which is a crucial component in speech recognition. They proposed the fusion of cepstral, Mel-frequency cepstral coefficients (MFCC), and formants into a feature vector, which, according to their findings, significantly enhances the accuracy of emotion recognition. For the classification of speech, a variety of traditional classifiers have been employed, including SVM (Support Vector Machine), HMM (Hidden Markov Models), and kNN (k-Nearest Neighbors). However, the SVM, due to its simplicity, lesser complexity, and high accuracy, was evaluated to be the most effective. Comparing different SVM kernel functions can yield valuable insights, with the choice depending on the specific application. The study reported the highest overall emotion recognition rate (84%) achieved by the RBF (Radial Basis Function) kernel. However, the linear and quadratic functions demonstrated superior recognition rates for specific emotions, namely fear, joy, and sadness. Conversely, the polynomial function was found to be the least effective for speech emotion recognition.

Ke et al., 2018 performed a comparison study between Support Vector Machines (SVM) using a polynomial kernel of degree 3 and Artificial Neural Networks (ANN) using the Chinese Emotional Corpus (CASIA dataset). Without employing dimensionality reduction techniques, the SVM model obtained an accuracy of 46.67%, slightly outperforming the ANN model which recorded an accuracy of 45.83%. Nevertheless, when dimensionality reduction was introduced via PCA, both models observed significant performance enhancements. Specifically, the SVM model, combined with PCA and using the polynomial kernel of degree 3, achieved an impressive accuracy of 76.67%. Concurrently, the ANN model combined with PCA, structured with four layers (32, 50, 100,

and 200 nodes per layer), obtained a close accuracy of 75%.

Zhao et al., 2019 introduced 1D and 2D Convolutional Neural Network (CNN) Long Short-Term Memory (LSTM) architectures for speech emotion recognition, focusing on the extraction of local correlations and global contextual data from raw audio clips and log-mel spectrograms. The proposed design, termed Local Feature Learning Block, comprised a convolutional layer, a Batch Normalization (BN) layer, an Exponential Linear Unit layer, and a max-pooling layer, all aimed at local feature extraction. These features, once reshaped, were incorporated into an LSTM layer for contextual learning.
Performance evaluations of the networks were conducted on two benchmark databases, revealing that the CNN LSTM designs could effectively discern and model emotional nuances. The 2D CNN LSTM network outperformed the 1D variant in overall performance and demonstrated superior average accuracy compared to other conventional feature representations and methodologies.

Amjad et al., 2021 proposed SER system using standard databases and ten-fold cross-validation. Results indicated that the SVM classifier generally outperformed other models, achieving the highest accuracy across multiple databases, including 92.11% for the Emo-DB database. However, the Multi-layer Perceptron (MLP) classifier showed the highest accuracy of 91.51% on the Emo-DB database and 86.75% for the IEMOCAP database. The Single-Speaker-Out (SSO) method was employed for speaker-independent experiments, in which the MLP model exhibited the best performance, especially on the Emo-DB dataset. Preliminary results suggest that implementing feature selection and data resampling approaches can improve the accuracy of the models.

We close this literature review with a summary of the information collected. We note that there are several types of emotion annotation. The most widely studied model for recognising emotions in speech is the categorical model. Another, quite different, model proposes to position emotions on two axes, 'arousa' and 'valence'.

Next, we looked at the different types of datasets containing labelled data. We then looked at the scientific literature on pre-processing, setting out the various stages involved in obtaining data with high discriminant values for classifying emotions. To do this, we looked at the concepts of framing, windowing and signal normalization in order to obtain a quasi-stationarity of the signal and be able to extract relevant information from it. This precedes the most

important part of the study of emotions in speech: feature extraction.

Indeed, most of the methods proposed in the literature are based on a fairly extensive feature extraction stage. The types of features and their related characteristics have been developed in order to understand what each feature covets and finally to understand the different associations proposed. It is clear that spectral features (MFCC and others) are essential in most case studies.

Next comes an important part of the data science field: reducing the dimensionality of data. There is no consensus in the literature on this subject. It mentions the classic methods: PCA, SFS, FFS, BFS. But there are also filtering and encoding methods. We also learn that transfer learning is an emerging field of study.

We conclude this literature review with case studies that are as detailed as possible of what was provided in the scientific literature. We note that the emergence of emotion recognition in speech has been driven by classical classification models such as SVM, HMM and KNN. It seems that SVMs have a preponderance to work well within the framework of our study. As a result, the reputation of deep learning and recurrent neural networks has grown. This has led to some great improvements in the systems put in place, such as the pairing of convolutional neural networks with recurrent neural networks like LSTM.

After reflection and considering the existing scientific literature which shows the superiority of Support Vector Machines (SVM) in the majority of cases where linear methods are used, we propose to develop two approaches for the classification of audio signals based on the MFCC, ZCR and RMS characteristics. The first approach will use a classical classification model (SVM), while the second approach will explore the use of a Multilayer Perceptron (MLP). What is the comparative effectiveness of these two approaches for classifying audio signals in terms of precision, recall and F1 score?

Chapter 4

Methodology

As expressed in the conclusion of the literature review, we propose to compare the effectiveness of support vector machines with Multilayer perceptrons by comparing them on the basis of their accuracy, recall and F1 score with MFCC, zero-crossing rate (ZCR) and root mean squared (RMS) as extracted features.

To this end, we propose a methodology as follows:

- **Data collection**: where we discuss the points of attention to be brought in the way of selecting its dataset and to understand the implications of the method of collection of the audio samples.
- **Preprocessing**: We go through the data preparation steps required for functional feature extraction. Then we detail how to obtain the desired features. As ZCR and RMS are fairly simple to understand, we move on to the more substantial derivation of MFCC via Mel Spectrogram. This, using the concepts detailed in Section 2.3. We then explain what the RMS of a signal is and how it is extracted. We close this section by explaining the input space obtained from feature extraction.
- **Dimensionality reduction** Although there is no consensus on dimensionality reduction in the literature review, we describe two methods in detail: PCA and SFS.
- **Model Development**: We present the two models we are studying in our research question, namely SVM and MLP. For each of the two models we explain the underlying principle of their operation in order to clearly identify the parameters to be taken into account when using them. At the end of the section, we detail the metaparameters tested for the training phase.

4.1 Data Collection

The databases employed for the current research include the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) and the Surrey Audio-Visual Expressed Emotion (SAVEE) dataset. Table 4.1 and Figure 4.1 show the details of the two datasets.

The SAVEE dataset, contains recordings from 4 male actors expressing 7 different emotions neutral, calm, happy, sad, angry, fear, disgust, surprised. This corpus of 480 British English utterances was carefully selected from the standard TIMIT corpus, ensuring a phonetic balance for each emotional representation. The recordings were conducted in a high-quality visual media lab and were thoroughly evaluated by multiple subjects under audio, visual, and audio-visual conditions. As a result, this database ensures a high degree of reliability for emotion recognition research.

The RAVDESS is a comprehensive database, featuring a balanced gender distribution, with a total of 24 professional actors voicing matched statements in a neutral North American accent. The emotions represented include calm, happy, sad, angry, fearful, surprised, and disgusted expressions in speech, with song expressions of calm, happy, sad, angry, and fearful emotions. Each emotion is expressed at two intensity levels, with an additional neutral expression, culminating in a set of 1440 unique recordings. It is to be noted that, unlike the other databases, RAVDESS includes a 'calm' label. These audio file have been subjected to rigorous validation, ensuring their emotional accuracy and authenticity.

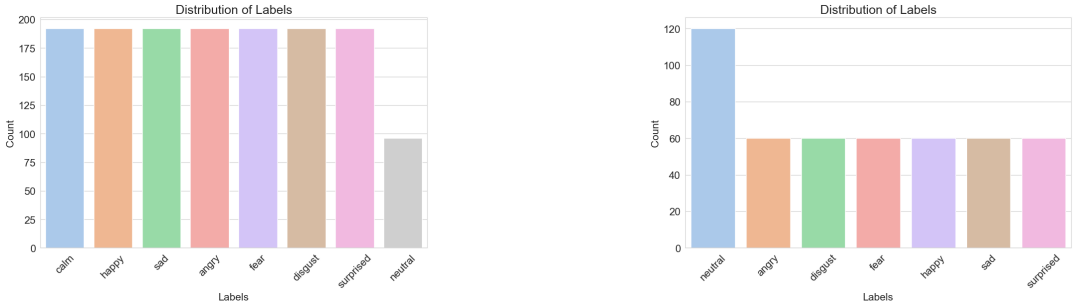
Apart from the 'calm' label unique to the RAVDESS database, all databases share the same labels, providing a consistent framework for the emotional analysis. All the databases used in the study are free to access and are available in English.

Table 4.1 summarizes the specificities of the three datasets.

The three datasets presented contain audio data in .wav format. This format is extremely raw since it contains the signal in the shape presented in Section 2.2. Now that we have validated our data sources, we can move on to the processing of the data that is essential for our classification models to work properly.

Attribute	RAVDESS	SAVEE
Number of Speakers	24	4
Gender of Speakers	Male and Female	Male
Total Recordings	1440	480
Number of Emotions	8	7
Type of Speech	Speech	Speech
Audio Format	WAV	WAV
type of dataset	Acted	Acted

TABLE 4.1: Comparison of RAVDESS and SAVEE datasets



(A) Ravdess (B) SAVEE

FIGURE 4.1: Emotion label distribution

4.2 Preprocessing

This section is of utmost importance in this study. When aiming to process the audio signal for emotion classification in voice, it is necessary to modulate the signal to match how it is perceived by the receiver. Our goal is to obtain measurements with high discriminating power from the raw signal. Failure to do so will hinder proper learning of our models.

This section is divided into five parts. Firstly, we focus on signal processing to enhance its quality. This step allows us to avoid deteriorating the signal when applying feature extraction techniques (2.2: Aliasing, Nyquist frequency). It is essential to ensure quasi-stationary properties in the signal.

Next, we present and demonstrate the extraction of the features chosen for analysis. We also provide an explanation of the characteristics of these extracted variables.

Following that, we introduce the classification models and discuss the rationale behind their selection.

Finally, we explain the evaluation of model performance in general terms. Additionally, we delve into specific notions relevant to the SER domain.

4.3 Data Prepartion

1. **Framing:** We performed a comparative analysis to determine the optimal frame size, given that existing literature provides a range (20-50ms) rather than a specific value. Given the sampling rate of at least 41000Hz of our audio data, we adjusted our framing sample rate to 22050Hz. Indeed, it allows to avoid aliasing (See section 2.3). For 2.5-second samples with a frame size of 25ms and a hop parameter of 10ms (40% of frame size), we obtained a frame size of 551 samples per frame.

2. **Windowing:** A window function was applied to the frames - specifically, a Hamming window function, as per the recommendation from our literature review. We used a Hop length of 20% (Section 3.4.1).

3. **Normalization:** Normalization was not applied a priori, as it should only be applied after dividing the data into training, validation, and testing sets. This process aims to reduce subtle differences in audio samples due to gender, physiological conditions, and recording quality. It also help in the generalization of our model by averaging our parameters during the training phase. The test set was normalized using the mean and variance computed from the training set.

4.4 Feature Extraction

Feature extraction is a key step in human speech signal processing. It plays a central role in a variety of tasks such as speech recognition, speech synthesis and speaker identification. By extracting informative and discriminative features from speech, we can transform raw signals into a form that can be easily and efficiently processed by machine learning systems.

Our aim is to try to cover the most popular range of features that could be extracted from the human voice with a focus on prosodic and spectral features. However, it is important to note that the field of feature extraction is vast and complex, especially in the field of SER. We do not pretend to cover the whole subject, but rather to provide a sufficiently deep overview of what has been achieved in this area. In this way, we hope to propose a good basis for future work.

Our study explores a range of spectral and prosodic features, some of which have been widely used in speech recognition, while others are more innovative. Our aim is to understand the relative importance of these different features and

how they can be combined with other features to improve the performance of voice-based systems.

4.4.1 Zero-crossing rate

The zero-crossing rate metric is a commonly used characteristic in sound signal analysis. It represents the frequency at which a sound signal passes through zero in one second. Specifically, it measures the number of times the value of the audio signal changes sign in one second.

The zero-crossing rate can provide information about the frequency and nature of sound signal variations. For example, a tone with a high zero-crossing rate may indicate the presence of impulsive noises, while a tone with a low zero-crossing rate may be more stable and regular.

The calculation of the zero crossing rate can be done using the following formula:

$$ZCR = \frac{1}{T} \cdot \sum \left(|\text{sign}(x[n]) - \text{sign}(x[n-1])| \right) \tag{4.1}$$

where T is the duration of the sound signal and sign is the function that returns the sign of the number. The sum is performed on all samples of the sound signal from 0 to $N - 1$.

For each pair of consecutive samples, we calculate the difference between their signs (i.e. whether they are positive or negative) and we take the absolute value of this difference. Then, we sum all these differences and divide by the total duration of the sound signal to obtain the zero crossing rate.

Although the zero-crossing rate is a useful feature for sound signal analysis, it also has some drawbacks. For example, it does not take into account the amplitude of the sound signal, which can be problematic for certain types of sound signals such as those with high amplitude. In addition, it can be sensitive to noise and fluctuations in the sound signal, which can make the results less reliable.

The zero-crossing rate is therefore a commonly used characteristic in sound signal analysis. It represents the frequency at which a sound signal passes through zero in one second and can provide information about the frequency and nature of sound signal variations. The formula for calculating the zero-crossing rate is based on the difference between the signs of pairs of consecutive samples and the sum of these differences. However, this metric also has drawbacks and should be used with caution.

4.4.2 Mel Spectrogram

As we can see in Figure 2.13 showing the power spectrum obtained via the Short Time Fourier Transform, the energy levels and their variations are very low. Furthermore, the human ear does not pick up frequency magnitudes and their variation in the spectrum in a linear fashion. For example, a frequency with twice the intensity will not be perceived systematically as twice as "loud" by the human ear. This is why we usually use decibel-scaled (Figure 2.14) signal to transform the basic signal. But there are ways to better translate the energy given to the signal. Thus, as suggested by Toyoshima et al., 2023, CITATION : *"the Mel scale is a scale reflecting human speech perception with a frequency axis that plots lower frequencies in narrower intervals and higher frequencies in wider intervals"*. .

So, if we want the signal to be the way we perceive sounds, we must obtain an "ideal" signal that would include :

- **Time-frequency representation**: such as STFT, it allows the capture of frequency magnitudes as a function of time.
- **Perceptually-relevant amplitude representation**: The perception of the amplitude or energy of a signal is nonlinear and is often addressed using logarithmic scales, such as decibel scales.
- **Perceptually-relevant frequency representation**: typically what the MEL-Spectrogram gives as a representation.

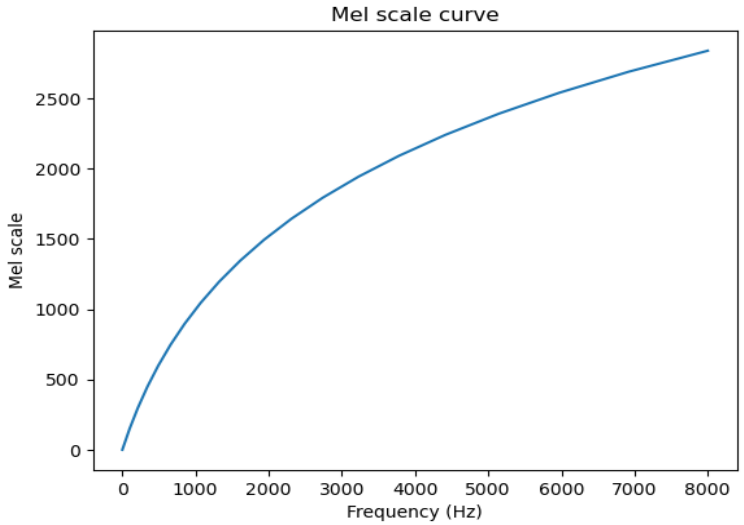


FIGURE 4.2: mapping function between Hz scale and Mel scale

We see in Figure 4.2 the transformation function of frequencies (Hz) in Mel scale (Mel). It is via this transformation function that we can obtain the Mel Spectrogram.

The transformation from Fourier transform to the Mel spectrogram involves a few mathematical steps. We start by computing the power spectrum of the audio signal using the Fourier transform:

$$P(f) = |\mathcal{F}x(t)|^2 \tag{4.2}$$

Where $x(t)$ is the audio signal, $\mathcal{F}$ denotes the Fourier transform, and $P(f)$ is the power spectrum. Given our interest in the signal's power, we square the Fourier Transform, which results in the power spectrum, moving our focus from magnitude to power.

Then we apply a filter bank that is spaced according to the Mel scale to the power spectrum. The Mel scale is a perceptual scale of pitches judged by listeners to be equal in distance from one another. The filter bank is typically a set of triangular filters that are designed to approximate the shape of the human auditory system's frequency response. Figure 4.3 depict 10 triangular filters and the mapping of each filter in power of the signal. The output of each filter is computed by convolving the filter with the power spectrum:

$$H_m(f) = \begin{cases} 0, & f < f_{m-1} \\ \dfrac{f - f_{m-1}}{f_m - f_{m-1}}, & f_{m-1} \leq f \leq f_m \\ \dfrac{f_{m+1} - f}{f_{m+1} - f_m}, & f_m \leq f \leq f_{m+1} \\ 0, & f > f_{m+1} \end{cases} \tag{4.3}$$

$$S_m = \sum_{f=0}^{N/2} P(f) H_m(f) \tag{4.4}$$

Where $H_m(f)$ is the frequency response of the m-th filter, S_m is the output of the m-th filter, N is the length of the power spectrum, and the filter bank has M filters. We take the logarithm of the filter bank output to obtain the Mel spectrogram: $M_m = \log(S_m)$ where M_m is the Mel spectrogram value for the m-th filter. The resulting Mel spectrogram is a representation of the power spectrum of the audio signal that has been transformed into a space that is more perceptually uniform, making it easier to analyze and model certain aspects of the signal contained in the sound sample.

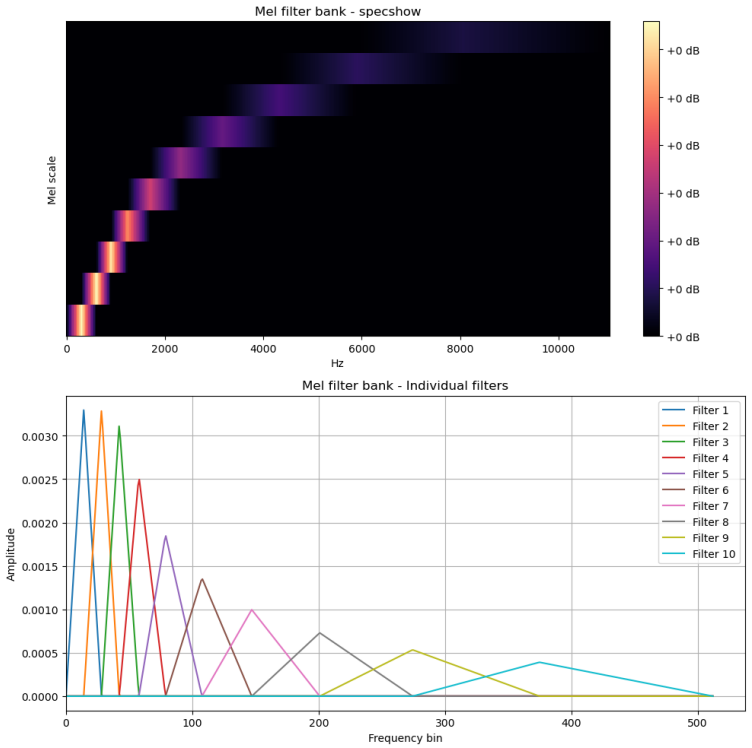


FIGURE 4.3: Mel filterbank and mapping function

We can see on Figure 4.3 the effect that each filter of the filter bank has on the starting signal. By applying 4.4 to all the frame of the signal we obtain the following mel spectrogram. We overlay the plots and assign equal widths to each filter in order to provide an intuitive understanding of the transformation performed by the Mel filter bank.

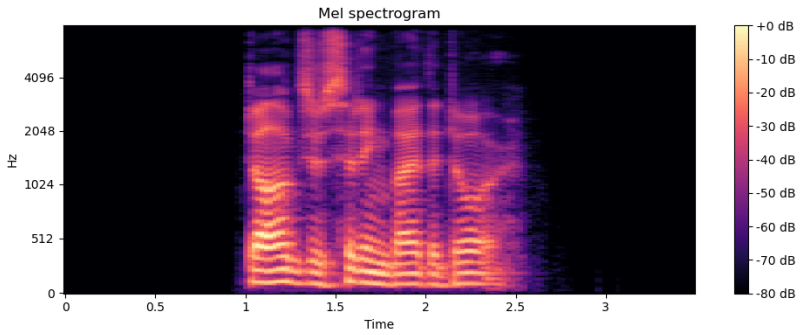


FIGURE 4.4: Mel spectrogram

Note that the number of Mel filters in Figure 4.3 is not the same as in Figure 4.4. This is because it would not have been easy to distinguish between the filters if we had taken too many. Figure 4.3 is useful for the reader's understanding because it allows to visualize the enlargement of the bands of mel-frequency the more we go up in the Hertz frequencies. However, in order to keep the

signal as smooth as possible we work with 128 Mel filters to get our final Mel spectrogram.

4.4.3 Mel-Frequency Cepstral Coefficients

Mel-Frequency Cepstral Coefficients (MFCC) are a standard tool in speech recognition and emotion detection. They represent the characteristics of the human voice by modeling how the human auditory system perceives different frequencies.

The process of calculating MFCCs involves several steps:

1. The speech signal is divided into short frames.
2. Each frame is then transformed into a frequency spectrum using the Short Term Fourier Transform (STFT). (Section 2.12)
3. As the raw frequency spectrum doesn't account for the human ear's perception of sound, it is passed through a Mel filter which models the ear's frequency perception. (Section 4.4)
4. The MFCC coefficients are then computed by applying the discrete cosine transform (DCT) to the Mel-filtered values. The resulting coefficients provide a compact representation of the speech signal that captures its frequency spectrum shape, pitch, and timbre. (see Section 2.3)

To derive the whole calculation lets now assume that :

- The duration of the audio clip, T = 1.5 sec (fixed)
- The sampling rate, sr = 22050 Hz
- The frame size, t_{frame} = 25 ms = 0.025 sec
- The hop size, H = 10 ms = 0.01 sec

We can proceed similarly as follows:

First, we calculate the number of samples N in one frame. This is done by multiplying the frame duration by the sampling rate:

$$N = t_{frame} \cdot sr = 0.025 \text{ sec} \cdot 22050 \text{samples/sec} = 551 \text{ samples} \tag{4.5}$$

Next, we calculate the number of frames. We do this by dividing the total duration by the hop size:

$$n_{frames} = \frac{T}{H} = \frac{1.5 \text{ sec}}{0.01 \text{ sec/frame}} = 150 \text{ frames} \tag{4.6}$$

Note: we do not use the frame size in the denominator because we are assuming each frame starts after a hop length of 10 ms, not taking into account the frame size.

Therefore, for an audio signal of 1.5 seconds sampled at a rate of 22050 Hz, we can obtain about 150 frames with a frame duration of 25 ms and a hop length of 10 ms. Thus, in each frame, we have $0.025s \cdot 22050Hz$ equals to 551 samples.

By implementing STFT as detailed in Equation 2.12 on each of the 150 frames — each comprised of 551 samples — we can derive the power spectrogram. This spectrogram consists of 551 squared Fourier coefficients per frame, which capture the power associated with each frequency within the frame (refer to Equation 4.2). The resulting output is a 2D matrix P of dimensions 551×150, where $p_{n,k}$ represents the frequency power of frequency k distributed across the frame $n \in 1, 2, \ldots, 551$.

Following this, we convert each frame's power spectrogram to the Mel scale by utilizing Equation 4.4. Once we've achieved the Mel spectrogram for each frame, we convert the Mel coefficients to the decibel scale to simulate the human ear's non-linear response. Consequently, we end up with Matrix S, which shares dimensions with P, but where $S_{n,k}$ represents the frequency power to the Mel scale of frequency k distributed across the frame $n \in 1, 2, \ldots, 551$.

Finally, the DCT is applied to the n^{th} frame of the log of the Mel spectrogram S, generating the MFCCs for that frame:

$$MFCC_{n,k} = \sum_{m=0}^{M-1} \log(S_{n,m}) \cos\left[\frac{\pi}{M}(m + 0.5)\, k\right] \tag{4.7}$$

In Equation 4.7, n, k, and m are respectively, the frame, MFCC, and Mel filter indices with M being the total number of Mel filters. The outcome is a $K \times N$ MFCC matrix, with N frames and K retained MFCCs, forming the final set for the audio clip.

We can observe the obtained MFCC over time frame of the signal in Figure 4.5.

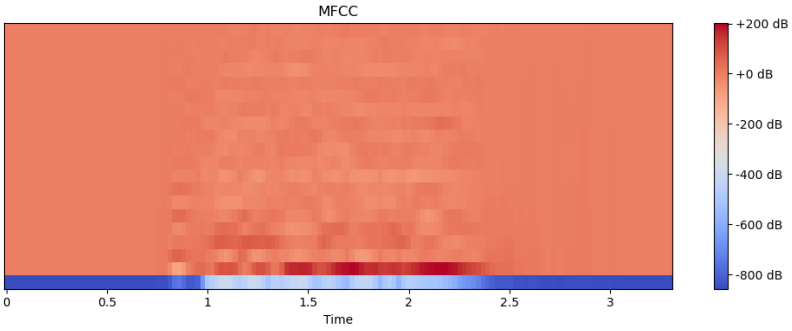


FIGURE 4.5: MFCC plot

This MFCC coefficient representation is a more compact and robust representation of the spectral content of the signal.

4.4.4 Root Mean Square Energy

The Root-Mean-Square (RMS) Energy is a crucial measure in audio signal processing. Unlike onset detection that focuses on detecting the beginnings of sounds, RMS Energy aims to measure loudness, a significant feature for detecting events in an audio signal. The robustness of RMS Energy against outliers makes it reliable for identifying new events, such as the entry of a new instrument or spoken words, particularly when audio is segmented.

The process for calculating RMS Energy is straightforward. As we move through the audio waveform in segments (or windows), we square the amplitudes within each window and sum these squared values. After completing this step, we divide the resulting sum by the length of the frame. The final step involves taking the square root of the resultant value, providing us with the RMS Energy for that specific window.

Mathematically, the RMS Energy is computed using the following formula for an audio signal $x[n]$ with N samples:

$$RMS = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x[n]^2} \tag{4.8}$$

The variable x[n] represents the amplitude of each data point within a frame, while N denotes the total number of samples present in the frame. The RMS Energy is always a positive value, where larger values of RMS signify a louder

or more potent signal. It should also be noted that the RMS value can be impacted by noise within the signal, as well as silent portions where x[n] equals zero.

4.5 Features set

In our approach, we aim to retain as much information as possible from our audio data. We do this by extracting features that include various aspects of the sound characteristics. Specifically, we choose to extract 20 Mel-frequency cepstral coefficients (MFCCs) to capture the spectral envelope of the audio signals, 128 Mel power spectrograms to reflect the frequency power distribution of the signals, and 1 zero crossing rate (ZCR) and 1 root mean square (RMS) to capture time domain features.

Therefore, based on Equation 4.7 we can obtain mean MFCC such that :

$$\overline{MFCC}_k = \frac{1}{N} \sum_{n=0}^{N-1} MFCC_{n,k} \tag{4.9}$$

In this equation, $\overline{MFCC}_k$ is the mean of the k^{th} MFCC coefficient across N frames, N being the total number of frames. The exact same principle is applied on Mel power spectrogram.

Although these features are by nature dependent on the temporal structure of the audio signals, for the purposes of input dimension of neural of our classification model, we transform these time-dependent features into a format that can be processed efficiently. To do this, we use a mapping operation using the mean function. This operation transforms our MFCCs and Mel spectrogram features into one-dimensional arrays, effectively condensing the time dimension while preserving the essential characteristics of the sound. Similarly, the ZCR and RMS, initially feature vectors are average and become scalar values. Table 4.2 depicts the extracted feature dimension.

TABLE 4.2: Dimensionality of Extracted Features

Feature	**Number of Dimensions**
MFCC	20
ZCR	1
RMS	1
Total	22

This step enables us to build a data structure compatible with the input specifications of our models, while preserving as much of the original audio information as possible.

In the next section, we analyse ways of reducing the dimensionality of the data in order to eliminate irrelevant variables. This is done using a feature selection technique as well as the well-known Principal Component Analysis (PCA).

4.6 Feature Selection

Feature selection is a step that can often improve models' performance or increase their accuracy in a more stable way. In addition, when dealing with high-dimensional data, it mitigates the effect of the curse of dimensionality by reducing the number of input variables. Furthermore, reducing dimensionality helps to minimize overfitting, thereby encouraging the train of more generalist models. It is also worth noting that reducing the dimensionality of the data helps to shorten model learning times, which can be particularly beneficial when training models with a large number of parameters. Finally, variable selection helps to make the model more interpretable, by eliminating redundant or irrelevant variables. This facilitates understanding of the model and its ability to be explained, factors of growing importance in many areas of application of machine learning.

In order to optimise our machine learning models, we will explore two variable selection methods: Principal Component Analysis and step-wise Feature Selection (SFS). These approaches will be tested and compared to determine which is best suited to our context.

4.6.1 Principal Component Analysis

Principal Component Analysis (PCA), is often considered as the bedrock of dimensionality reduction. Its relevance spans across scientific fields and has retained its efficacy since it has been proposed by Pearson in 1901.

Namely, PCA aims to project samples onto the axes that account for the maximum variance in the data. This process leads to data being mapped into subspaces that capture the greatest possible variance from the original dataset. Therefore, it preserves the discriminative power of features and decrease the bad effects of high-dimensional data. So, the goal consists in mapping the initial matrix X_n of dimension D, containing $n \in N$ observations to dimension

$M < D$ by preserving the maximum variability between features. This mapping is executed by projection operator u, the unit vector that has the property $u^T \cdot u$ equals to one.

We suppose X_n normalized in order to have all features at the same scale :

$$\begin{aligned}\frac{1}{N}\sum_{i=1}^{N}(u^T X_i - u^T \bar{X})^2 &= \frac{1}{N}\sum_{i=1}^{N}[u^T(X_i - \bar{X})]^2 \\ &= \frac{1}{N}\sum_{i=1}^{N} u^T(X_i - \bar{X})(X_i - \bar{X})^T u \\ &= u^T S u\end{aligned}$$

where $u^T S u$ is the projected covariance matrix.

As stated before, we aim to find projection vector u of the covariance matrix that maximize $u^T S u$. To do so, we use Lagrangian multiplier :

$$\max \; u^T S u \quad \text{s.t.} \quad u^T u = 1$$

Thus, we must satisfy the following condition :

$$\frac{d}{du} u^T S u + \lambda(1 - u^T u) = 0 \tag{4.10}$$

$$\leftrightarrow \; S u = \lambda u \tag{4.11}$$

$$\leftrightarrow u^T S u = \lambda \tag{4.12}$$

Hence, we obtain the eigen decomposition of the covariance matrix of X_n with eigenvectors u of dimension D and eigenvalues λ.

In order to perform the projection of X_n in $M < D$ dimension, we use the M eigenvectors obtained in Equation4.12 corresponding to the M highest eigenvalues λ. Therefore, new coordinates of X_n projected into dimension M are obtained such as:

$$PC = X \cdot \lambda \tag{4.13}$$

A useful property of the covariance matrix factorization is that the sum of all eigenvalues is equal to the total variance explained. Thus, it is possible to assess how much variance is kept when we project data in dimension M such that :

$$\frac{\sum_{m=1}^{M} \lambda_m}{\sum_{n=1}^{D} \lambda_n} \tag{4.14}$$

We can derive from 4.14 the notion of scree plot that allow us to visualize the cumulative variance explained as much as we increase in dimension of projection.

A typical cumulative variance explained threshold of 90% is applied in order to determine the projection dimension of the data. By analyzing the scree plot of features extracted in the previous section in Figure 4.6, we can asses how many principal components we should have to assure the trade-off between dimension and cumulative variance.

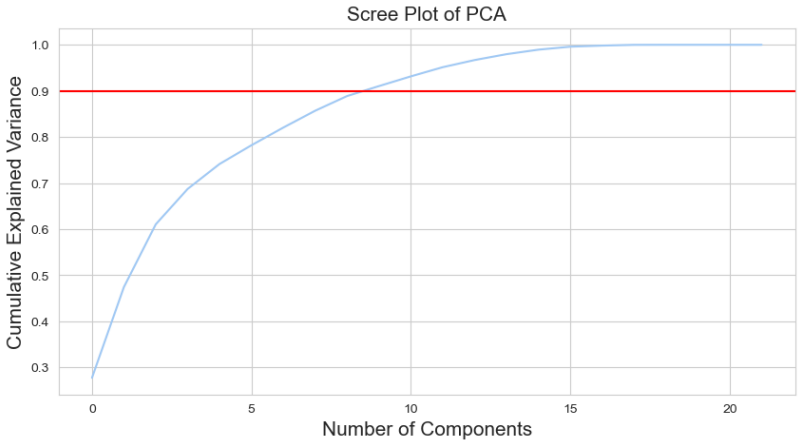


FIGURE 4.6: Scree plot PCA

4.6.2 Forward features selection

Forward Feature Selection (FFS) is a method that iteratively selects the most relevant variables for a machine learning model. This technique improves prediction accuracy by eliminating irrelevant or redundant variables. The steps involved in executing this variable selection technique are :

1. **Initialization**: Start with empty set of selected features
2. **Iteration**: For each variable not yet selected, a model is fit and evaluated based on its performance.
3. **Feature Selection**: a variable is added when it brings the greatest gain in model performance.
4. **Termination**: Repeats the iterative phase of the algorithm until no additional variable improves model performance.

Pseudo code of this technique is available in Appendix ??

Due to the heuristic nature of this variable selection, the algorithm is often run several times and with several model fits to ensure that the subset of variables proposed as output is generalist and stable.

The classifier we have chosen to use for this algorithm is k-NN with $k = 3$. Thus, FFS will provide us with the set of features most relevant for classification.

In our case, we perform a 5-fold cross-validation (CV) to ensure that the procedure was performed in a stable manner. In addition, each training and test set was reset between each CV adjustment in order to obtain the most generalized feature selection procedure possible.

Figure 4.7 shows how the model performed with each combination of the features. The set of features that gives the best model performance was returned. In fact, FFS tells that all the features give the highest accuracy on 5 cross-validation. Therefore, at this step, it can be concluded that all the features contribute significantly to the model's performance.

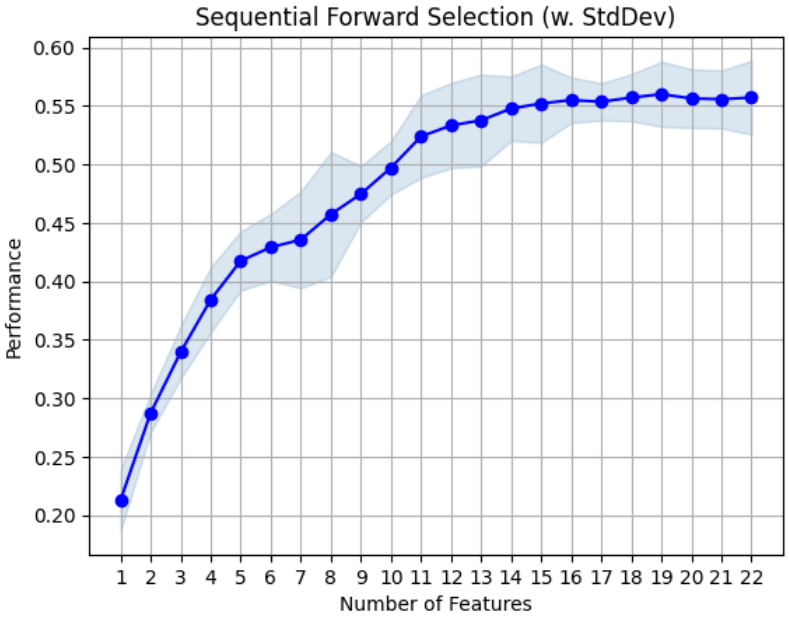


FIGURE 4.7: FFS whole features

4.7 Model Development

In this section, we describe the models used to best classify emotions on the basis of the extracted features and whose dimension we have potentially reduced in the previous two sections.

In this context, we consider the use of the Support Vector Machine (SVM) as a linear classifier in order to evaluate the performance of this model, but also to determine whether the data are linearly separable. To fully understand the implications of this classification technique, we detail the optimisation problem that gives the hyperplane equation, the decision frontier of the classifier. We review the different variants of SVMs, such as the soft margin, the kernel trick and the methods available to make this binary classification problem applicable to a multiclass classification case.

We then present the multilayer perceptron (MLP), a non-linear classifier from the field of deep learning. We propose this model because it has already proved its effectiveness in a wide variety of study domains. Indeed, it is a highly versatile and scalable algorithm that offers great freedom in its architecture. Finally, we discuss the method used to find the best hyperparameters for each model in order to obtain the best classification performance.

After each model development, we explain the choices we made regarding the model's metaparameters and how we defined them. This section marks the end of the methodology part and will close with a summary of what has been undertaken.

4.7.1 Support Vector Machine

Support vector machines (SVM) are used for classification and regression. They have a strong presence in the field of emotion classification in human voice due to their ability to handle complex data and find non-linear relationships between acoustic features and emotions expressed in voice. One advantage of SVM is their ability to efficiently handle high-dimensional data and avoid overfitting. The fact that SVM are able to work with nonlinear data is particularly important in the classification of emotions in the human voice, where acoustic features are often nonlinear. SVMs have been shown to be highly accurate in classifying speech emotions, with higher classification rates than other classification algorithms, such as neural networks and decision trees. Moreover, SVM are easy to interpret, making them useful in domains where transparency and explicability of models are essential, such as our scope of interest. In this thesis, we will explore the use of SVM for emotion classification in human speech and analyze the advantages and limitations of this algorithm.

Theory

We have N observations in our training set and for each observations i, we have a vector x_i of dimension D and its class as $y_i = -1$ or $+1$. Our data are of the form :

$$\{x_i, y_i\} \text{ where } i = 1...L, y_i \in \{-1, 1\}, x \in \mathcal{R}^D \tag{4.15}$$

At this stage, we assume that our data is linearly separable. This means that we can draw a line in the plane (D = 2) which correctly separates all observations x_i for which $y_i = -1$ from the observations x_i for which $y_i = 1$. When $D > 2$, we speak of a hyperplane of the graph $x_1, x_2, ..., x_D$ which respects the properties.

The equation of this hyperplane can be written as $w \cdot x + b = 0$ with :

- w the unit vector normal to the hyperplane
- $\frac{b}{||w||}$ the perpendicular distance between the intercept and the hyperplane.

Both parameters b and w are crucial in estimating the best hyperplane in R^D given our data. In order to understand the implications of these two parameters and to visualize how we will constrain the optimization problem, Figure 4.8 presents the graph of a hyperplane :

We can observe in Figure 4.8 d_1 and d_2 that are nothing but the margin. In SVM, the margin is a measure of the separation between the decision boundary

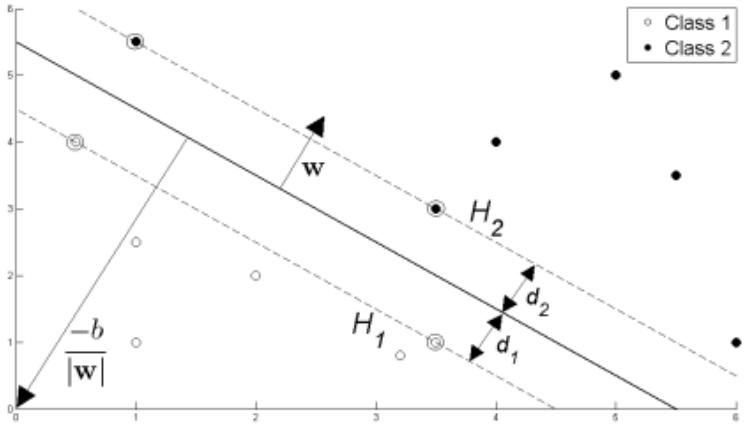


FIGURE 4.8: SVM: hyperplan fitting

(hyperplane) and the closest data points from each class. It represents the distance between the hyperplane and the support vectors, which are the data points closest to the decision boundary. It allows us to support that the strength of the SVM method which is not only to draw a line (or a hyperplane) which classifies the data correctly but to define the largest margin (d1 and d2 on the graph) possible around the line (the hyperplane).

To determine the equation of the line (or hyperplane) in an SVM, we project the data points x_i onto the unit vector w. In order to obtain a classifier, we want a relationship that satisfies the following conditions:

$$x_i \cdot w + b \geq +1 \quad \text{for } y_i = +1 \; x_i \cdot w + b \leq -1 \quad \text{for } y_i = -1 \tag{4.16}$$

via y_i which gives us the label of x_i we can obtain equations of the so called support vector :

$$y_i \left(x_i \cdot w + b\right) - 1 \geq 0 \quad \forall i \tag{4.17}$$

In Figure 4.8 we can see the support vectors which are the points located closest to the decision boundary drawn by the hyperplane. Their equations are such that :

$$\begin{aligned} x_i \cdot w + b \geq +1 \quad for\ H_1 \\ \\ x_i \cdot w + b \leq -1 \quad for\ H_2 \end{aligned} \tag{4.18}$$

Moreover, it can be shown that the margin separating the hyperplane from the support vectors is equal to $\frac{1}{|w||}$.

In this way, we wish to maximize this margin under the constraint that we classify the labels correctly. Thus, maximizing $\frac{1}{||w||}$ can be summarized as minimizing ||w||. Hence our optimization problem is such that :

$$\min \frac{1}{2}|\mathbf{w}|^2 \quad \text{such that} \quad y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \quad \forall i \tag{4.19}$$

Here, the term $\frac{1}{2}||\mathbf{w}||^2$ has no effect on the determination of the optimal value of w but allows a clearer notation later.

We solve this optimization problem via a Lagrangian. We obtain the following expression including the Lagrangian multipliers:

$$L = \frac{1}{2}||w||^2 - \sum_{i=1}^{L} \alpha_i [y_i(x_i \cdot w + b) - 1] \tag{4.20}$$

By taking the partial derivatives for w and b respectively, we end up with:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{L} \alpha_i y_i x_i = 0 \Rightarrow w = \sum_{i=1}^{L} \alpha_i y_i x_i \tag{4.21}$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{L} \alpha_i y_i = 0 \Rightarrow \sum_{i=1}^{L} \alpha_i y_i = 0 \tag{4.22}$$

In Equation 4.22, we can see that the weight vector w is a sum of our sample vectors xi. However, not all sample vectors contribute to the weight vector. Only the sample vectors xi that lie on the margin will have a non-zero value for α_i These non-zero values indicate the importance of the support vectors in defining the decision boundary. By considering these support vectors, we can understand their influence on the overall classification process.

At this point, we substitute value for w by those we obtained at 4.22 and replace it in Equation 4.22. We can now derive :

$$\begin{aligned} L &= \frac{1}{2} \times \left(\sum_{i=1}^{L} \alpha_i y_i x_i \right) \left(\sum_{j=1}^{L} \alpha_j y_j x_j \right) - \sum_{i=1}^{L} \alpha_i y_i x_i \cdot \left(\sum_{i=1}^{L} \alpha_i y_i x_i \right) - \sum_{i=1}^{L} \alpha_i y_i b + \sum_{i=1}^{L} \alpha_i \\ &= \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i=1}^{L} \sum_{j=1}^{L} \alpha_i \alpha_j y_i y_j \; x_i \cdot x_j \end{aligned} \tag{4.23}$$

The last part of Equation 4.23 shows that all the maximization problem depends on the dot product of pairs of samples with $x_i \cdot x_j$.

$$\max_{\alpha} \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{L} \alpha_i \alpha_j y_i y_j x_i \cdot x_j \tag{4.24}$$

under the constraints $\alpha_i \geq 0$ and $\sum_{i=1}^{L} \alpha_i y_i = 0$.

We can use a quadratic programming (QP) solver to tackle this optimization problem, which is a convex quadratic problem. By resolving this, we are able to determine the optimal values of α_i, from which we can compute w and b with the formulas above.

Non-linearly separable data

In a hard margin SVM (previous section), the goal is to find a hyperplane that perfectly separates the data of two classes with maximum margin. This means that there are no data points that lie within the margin. Mathematically, this can be formulated as follows:

$$\min ||w|| \quad \textit{subject to} \quad y_i(w \times x_i + b) \geq 1 \forall i \tag{4.25}$$

where x_i is the feature vector of training example i, y_i is the class label (1 or -1), w is the weight vector, b is the bias, and $||w||$ is the L2 norm of w.

However, in many cases, the data are not linearly separable and it is not possible to find a hyperplane that perfectly separates the data of two classes. This leads to a violation of the hard margin condition, and thus to an infeasible solution for the optimization problem.

The assumption of data linearity made in the previous section often does not hold when we deal with field data such as our voice samples. We can see in Figure 4.9 that it is indeed impossible to linearly separate the data. Let's suppose we attempt to find parameters w and b via the constrained optimization detailed in the previous section, we will not obtain convergence and thus we will not be able to find the values of w and b. A typical example of non-linearly separable data is in Appendix 4.

To solve this problem, we can introduce some flexibility into the model by allowing some margin violation. This leads to a soft margin SVM. In a soft-margin SVM, the goal is to find a hyperplane that separates the data from two classes with as large margin as possible, while allowing some misclassification. Mathematically, this can be formulated as follows:

$$\min \frac{1}{2} \cdot ||w||^2 + C \cdot \sum(x_i) \quad subject\ to \quad y_i(w \cdot x_i + b) \geq 1 - x_i \tag{4.26}$$

where x_i is the error margin for training example i, C is a regularization parameter that controls the trade-off between margin maximization and classification error minimization, and $\sum(x_i)$ is the sum of the error margins over all training examples.

The parameter C controls the trade-off between model complexity and generalizability. If C is small, the model will be more tolerant of classification errors and have a higher generalization ability, while if C is large, the model will be more complex and have a lower generalization ability.

In conclusion, the hard margin SVM is an algorithm that searches for a hyperplane that perfectly separates the data from two classes, while the soft margin SVM allows some margin violation to allow for a more robust and generalized solution.

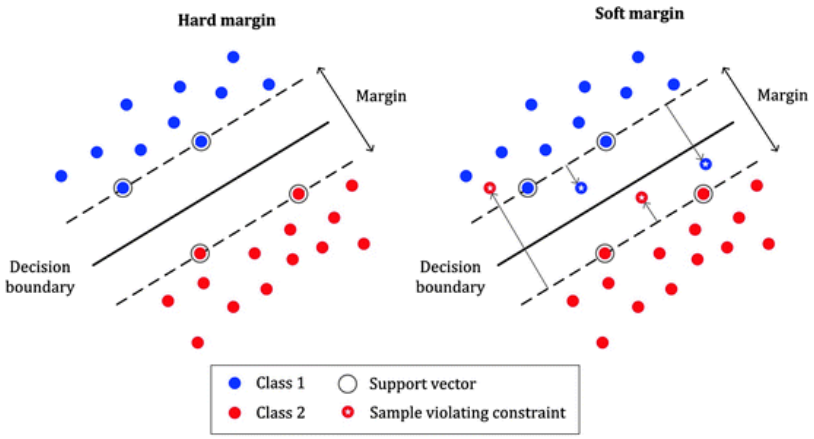


FIGURE 4.9: SVM : hard margin vs soft margin

Kernel Trick

The kernel trick is a powerful concept in machine learning, especially in support vector machines (SVM), that allows us to transform our data into higher-dimensional feature spaces without actually computing the coordinates of the data in that space. This is achieved through the use of kernel functions, which measure the similarity between pairs of data points in the higher-dimensional feature space.

Mathematically, the kernel trick can be described as follows. Let us say we have a set of input data points $x_1, x_2, \ldots, x_n$ with corresponding labels

$y_1, y_2, \ldots, y_n \in -1, +1$. We want to find the optimal hyperplane that separates the data points of different classes. This can be done by solving the following optimization problem:

$$\min_{w,b} \frac{1}{2}||w||^2 \quad \text{subject to } y_i(w \cdot x_i + b) \geq 1, \forall i = 1, 2, \ldots, n \tag{4.27}$$

where w is the weight vector and b is the bias term of the hyperplane. The constraints ensure that each data point is classified correctly with a margin of at least 1.

The main idea behind the kernel trick is to replace the dot product $w \cdot x$ with a kernel function $K(x, x')$ that implicitly represents the input data points in a higher-dimensional feature space. More precisely, we can rewrite the optimization problem as follows:

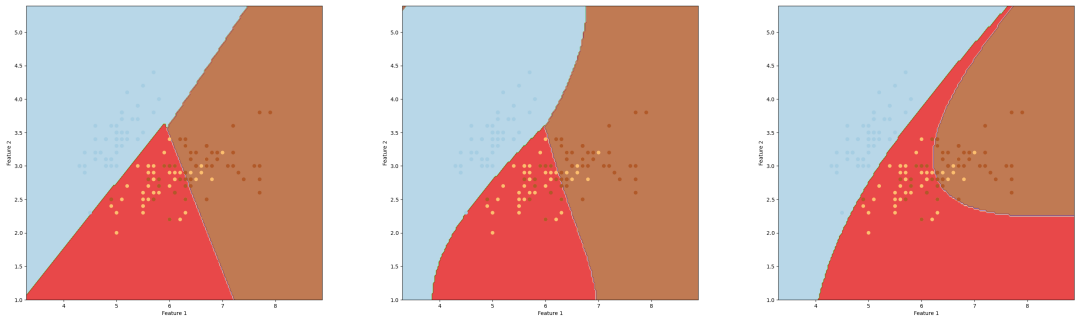
$$\min_{\boldsymbol{\alpha}} \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1}^{n} \alpha_i \tag{4.28}$$

subject to $\sum_{i=1}^{n} y_i \alpha_i = 0$ and $0 \leq \alpha_i \leq C, \forall i = 1, 2, \ldots, n$, where C is a hyperparameter that controls the trade-off between maximizing margin and minimizing classification error.

Among the popular kernel functions $K(.)$ that allow to project data, we have :

- **Linear kernel** : $K(x_i, x_j) = x_i^T \cdot x_j$: When vectors x_i and x_j are standardized, this kernel calculates the cosine similarity between the vectors x_i and x_j. In other words, it measures the angle between the two vectors. Hence, if the data are very similar, they will be projected close to each other and vice versa, allowing them to be linearly separated. We can see in Figure 4.10a the decision boundary of data that initially were not linearly separable but become so after the projection in a higher dimensional space using the linear kernel.
- **Radial Basis Function** (RBF): $K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$: In this case, $\|x_i - x_j\|^2$ is the square of the Euclidean distance in the input space and γ is a factor that weights the rate at which the similarity measure decreases. As Figure 4.10b shows, the decision boundary becomes non-linear with the RBF kernel.

- **Polynomial kernel**: $K(x_i, x_j) = (x_i^T x_j + c)^d$: Similar to the linear kernel, we calculate the cosine similarity but raise it to a degree d and add a constant term c. This kernel makes it possible to model much more complex projections of the data (see Figure 4.10c).



(A) Linear kernel (B) RBF kernel (C) Polynomial kernel

FIGURE 4.10: Illustration of different kernels decision boundary.

The key advantage of using the kernel trick is that we can now handle data that is not linearly separable in the original feature space by projecting it onto a higher-dimensional feature space where it may become linearly separable. Furthermore, the kernel function implicitly defines a feature space, which can be infinite-dimensional, without the need to explicitly compute the coordinates of the data points in that space. This makes the kernel trick computationally efficient and allows us to use SVMs on large datasets.

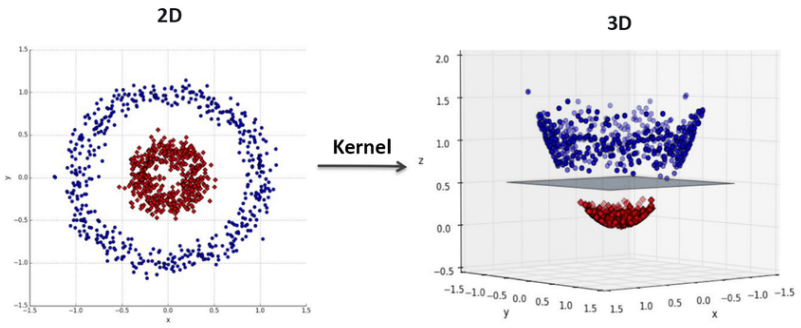


FIGURE 4.11: SVM : kernel trick

In this case, the plot shows the original dataset in 2D (the x and y axes), where the two classes (represented by different colors) are mixed together and cannot be separated by a straight line. However, when the kernel trick is applied, the dataset is transformed into a higher-dimensional space (represented by the z-axis in the 3D plot) where the classes become separable by a plane.

The reason for this is that the kernel trick maps the original dataset to a higher-dimensional space where the classes can be separated by a hyperplane.

In the 3D plot, this hyperplane is represented by the flat surface that separates the two classes. However, in the original 2D space, a hyperplane cannot be used to separate the classes because the data is not linearly separable in that space.

Multi class classification

So far we have considered the SVM as a binary classifier. However, we want to classify n types of emotions for $n > 2$. In order to extend our binary classification model to a multi-class classification model, there are two strategies for splitting a multi-class classification into multiple binary classification problems such that:

- **One-vs-One (OVO)**: For each class, we fit a model to all the other classes, one by one. If we have $n = 4$ classes: ['A', 'B', 'C', 'D'], the binary classification problems are:
  - Fit A vs B
  - Fit A vs C
  - Fit A vs D
  - Fit B vs C
  - Fit B vs D
  - Fit C vs D

  This makes 6 binary classification problems. We can therefore generalize the number of binary classifiers to be trained, denoted B, given a number of classes n such that $B = \frac{n \cdot (n-1)}{2}$.

- **One-vs-Rest (OVR)**: For each class, we fit a single model which classifies whether the observation belongs to class C or to the other classes. Consequently, the number of binary classifiers to fit is equal to the number of classes. If we have 3 classes ['A', 'B', 'C'], the binary classification problems are as follows:
  - Fit A vs [B, C]
  - Fit B vs [A, C]
  - Fit C vs [A, B]

  We opt for the OVR strategy as it is less computationally intensive.

4.7.2 Artificial Neural Networks

An Artificial Neural Network (ANN) is a computing structure inspired by the way the human brain works. It is made up of several artificial neurons, called "units", organised in layers.

4.7.3 Perceptron

The perceptron is a supervised learning technique. It is a simple form of artificial neural network where inputs (i.e variables) are weighted, summed and passed through an activation function to produce an output. The aim of training the perceptron is to adjust the input weights to improve the accuracy of the predictions.

Figure 4.12 depicts how the process works:

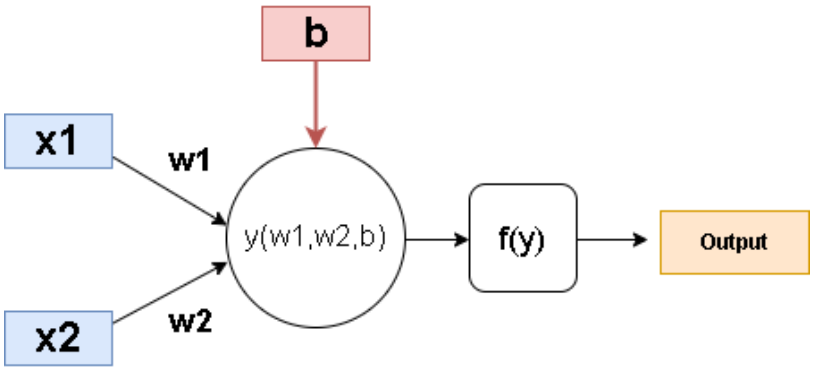


FIGURE 4.12: ANN: Perceptron

1. **Inputs** (x_1, x_2) : A perceptron receives multiple inputs, with each input representing a different feature of the dataset.

2. **Weight and bias** (w_1, w_2, b) : Each input is assigned a weight, and a bias is added. These parameters are learned during the training process.

3. **Linear combination** y : The inputs are multiplied by their respective weights and then summed, with the bias added. The result is a weighted sum of the inputs.

4. **Activation function** $(f(y)$: The weighted sum is passed through an activation function, which transforms the sum into an output

5. **Outputs** : The output z of the activation function is the final output of the perceptron. This output can be used as a prediction in a supervised learning task.

Formally, we first have:

$$y = \sum_i w_i \cdot x_i + b \tag{4.29}$$

The linear combination of inputs, weights and bias. Then, we express the output such that:

$$z = f(y), \tag{4.30}$$

$$= f\left(\sum_i w_i \cdot x_i + b\right). \tag{4.31}$$

where :

- i varies on all inputs.
- w_i is the weight associated with input x_i.
- $f(y)$ is the activation function.
- b is the bias.
- z is the final output.

To conclude on the perceptron, To summarise the perceptron, it is a fundamental concept for understanding how more sophisticated neural networks work. It presents the fundamental idea of assigning weights to inputs, aggregating them and applying an activation function to generate an output. This process is a key element in the understanding of neural networks.

However, the simplicity of the perceptron has limitations in terms of the complexity of the relationships it can model in the data. In order to overcome these limitations and extend the modelling possibilities, we are now introducing an extension to the perceptron: the multilayer perceptron (MLP).

By adding additional layers of neurons (the hidden layers), the MLP makes it possible to model more complex relationships and solve more demanding tasks than a simple perceptron could. We'll now take a closer look at how this more elaborate structure works, building on the understanding we gained with the perceptron.

4.7.4 Multi Layer Perceptron

The multilayer perceptron (MLP) can be seen as an extension of the simple perceptron. Whereas a perceptron has only one computational 'layer', the MLP organizes several perceptrons into layers to form a more complex neural network.

An MLP is typically organized in layers:

- **input layer**: which receives input data ("features"). Each unit corresponds to a feature.
- **hidden layer(s)**: Each unit in a hidden layer is connected to all the units in the previous layer.
- **output layer**: which produces the final result of the network. The number of output units depends on the problem to be solved. For example, for a binary classification task, there could be a single output unit, whereas for a multiclass classification task, there would be one unit per class.

Each neuron takes as input a linear combination of the outputs of all the neurons in the previous layer, to which a bias is added. This combination is then passed through a non-linear activation function.

Mathematically, if we have a neuron with n inputs, the weights associated with these inputs are $w_1, w_2, \ldots, w_n$, the inputs are $x_1, x_2, \ldots, x_n$ and the bias is b, then the output y of the neuron before activation is calculated as follows:

$$y = w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_n \cdot x_n + b \tag{4.32}$$

We then pass y through an activation function f to obtain the neuron's final output:

$$z = f(y) \tag{4.33}$$

There are several choices for the activation function, the main ones depicted in Figure 4.13 are as follows:

- **ReLU (Rectified Linear Unit)**: The ReLU function is defined as $f(x) = max(0, x)$. This function lets positive values pass unchanged, while it blocks negative values (by returning 0). The simplicity of this function allows faster calculations, while introducing a non-linearity that allows the neural network to model complex relationships.

- **Sigmoid**: The sigmoid function is defined as $f(x) = \frac{1}{1+e^{-x}}$. It transforms each input into a range of values between 0 and 1, which can be useful for binary classification problems.
- **Tanh (Hyperbolic tangent)**: Tanh, is defined as $f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$. It transforms each input into a range of values between -1 and 1, providing negative outputs when the input is negative.

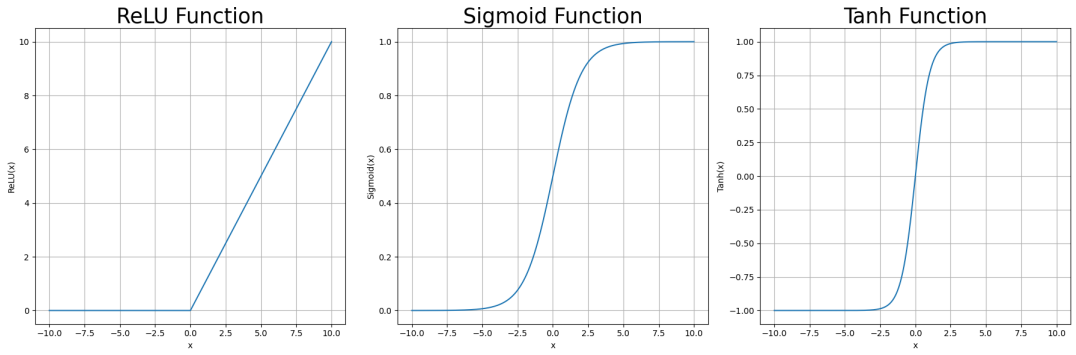


FIGURE 4.13: ANN: activation function

Now that we have defined the structure of the MLP, we can develop its automatic learning mechanism.

MLPs are generally learned in two stages: forward propagation and back-propagation.

Step 1: Forward propagation

Forward propagation involves calculating the outputs of the network for a given set of inputs. Inputs are transmitted from the input layer to the output layer through hidden layers.

We denote $x_j^{(l)}$ the output of the j-th node of the l-th layer and then $x_j^{(l)}$ is given by :

$$x_j^{(l)} = f\left(\sum_{i=1}^{N} w_{ij}^{(l)} x_i^{(l-1)} + b_j^{(l)}\right) \tag{4.34}$$

where:

$w_{ij}^{(l)}$ is the weight of the connection between the i-th node of the $(l-1)$-th layer and the j-th node of the l-th layer, $b_j^{(l)}$ is the bias of the j-th node of the l-th layer, f is the activation function.

Step 2: Backpropagation

Backpropagation is the stage where the MLP learns by adjusting its weights and biases. The idea is to adjust the weights and biases in a way as to minimise the error between the actual outputs of the network and the desired outputs.

To do this, we use an error measure, such that the categorical cross-entropy loss function, given by :

$$L_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}) \tag{4.35}$$

where :

- L_{CE} is the categorical cross-entropy loss.
- N is the total number of instances in the dataset.
- C is the number of classes in the dataset.
- y_{ij} is the actual class label (ground truth) of the i-th instance for the j-th class. It is either 0 or 1, depending on whether instance i belongs to class j or not.
- $\hat{y}_{ij}$ is the predicted probability that the i-th instance belongs to the j-th class, as given by the model.
- The logarithm is the natural logarithm (base e).

Next, the gradient descent algorithm is used to adjust the weights and biases. For each weight $w_{ij}^{(l)}$, the update is given by :

$$w_{ij}^{(l)} = w_{ij}^{(l)} - \eta\frac{\partial E}{\partial w_{ij}^{(l)}} \tag{4.36}$$

where η is the learning rate.

Similarly, for each bias $b_j^{(l)}$, the update is given by :

$$b_j^{(l)} = b_j^{(l)} - \eta\frac{\partial E}{\partial b_j^{(l)}} \tag{4.37}$$

These forward propagation and backpropagation steps are repeated until the error is sufficiently small, or until a maximum number of iterations has been reached.

Therefore, this learning process seeks to minimize the classification error on the basis of the difference observed between the prediction and the true value after the feed forward step. Then the backprogation corrects the weights proportionally to the error. However, this model is so good at fitting any pattern that it will often overfit the training data to the detriment of the prediction made on the test data. However, there are regularization techniques that can prevent the model from overfitting. We present three such techniques:

- **Dropout**: this technique consists in randomly disconnecting a certain proportion of neurons in a layer during a training iteration. In this way, the model cannot systematically udpate weight values. This considerably reduces overfitting.
- **L1 regularization**: this method comes from linear models and works by adding a penalization term to the cost function. This term consists of the sum of the absolute values of the weights. This leads to weight values that are generally smaller or even shrinked to 0.
- **Early stopping**: consists of stopping the learning process when the model starts to overfit the training data. This overfitting is detected when the loss function of the test data starts to increase again. In addition to its regularisation function, the early stopping method helps to reduce the computational cost by stopping the learning process before the model overfits the training data.

This section concludes our discussion on the theoretical models that we've mobilized in this thesis. After establishing the theoretical foundations in the previous chapter, we will now transition into a more practical approach. We will delve into preprocessing methods, the optimization of extracting features from the human voice signal, and the results obtained using the presented models.

Chapter 5

Applications and Results

Now that we have set up the theoretical part, we can begin the implementation phase, bridging the gap between theory and practice. This chapter aims to showcase the implementation of the models, from raw data to classification results, and to explain the decisions made to achieve the most consistent results possible.

To draw generalist conclusions, we combine the RAVDESS and SAVEE datasets (see Section 4.1), which are the most representative in terms of gender balance and the number of different speakers). Also, having two distinct datasets allows us to generalize more or less about the types of sound samples. The purpose of this maneuver is to achieve a model with the best possible performance that can function at this performance level in various application scenarios. RAVDESS and SAVEE datasets are discribed in Section 4.1.

As previously detailed in Section 4.3, preparing raw data derived from the human voice signal requires the following steps: framing, windowing, and normalization. We therefore assume that these steps have been applied to the data from RAVDESS and SAVEE. In that same Section , we outlined the methods used to extract features from these data preparation steps. Continuing in Section 4.4, we identified 22 extracted variables, including the first twenty meaned MFCC over time, zero-crossing rate, and root mean square of each audio sample. These constitute the input variable set for our models in this application phase. In the following section, we will apply the two models (SVM and MLP) introduced in the previous section, explain our strategy for defining the model hyperparameters and present the results obtained. We will offer a critical reflection on the results from each model.

Moreover, we will not limit ourselves to simply classifying emotion labels. We will use the valence-arousal model (see section 3.2.3) to project emotions

in 2D. This allows us to break free from the rigid constraints of emotion categories and visualize potential emotions that do not cluster well. Thus, we can discard extremely noisy emotions and retain emotions that classify well. In addition, this perspective will offer a new angle of attack to be compared with classification models.

5.1 SVM

As outlined in Section 4.7.1, SVM has several hyperparameters that need to be tuned, which greatly influence its classification performance. Our aim is to determine the best hyperparameters for our task such that they are as general as possible. This means that we don't want them to be optimal for one fold and then underperform in others; instead, we desire consistent performance regardless of the fold.

To recap, the SVM's hyperparameters include:

- C: This controls the trade-off between maximizing the margin and minimizing classification errors.
- Kernel function: This refers to the function that projects points onto higher dimensional space. Options include polynomial, radial basis function (RBF), and linear.
- Gamma: A factor that weighs the rate at which the similarity measure decreases.
- Degree: When using the polynomial kernel, this represents the degree of the polynomial function.

To determine the best values for these hyperparameters for our model, we use the Grid Search strategy, which we detail below (see 0).

5.1.1 Grid Search algorithm

Algorithm 1 GridSearchCV

```
best_params ← null                          ▷ Initialization of the best parameters
best_score ← infinity                       ▷ Initialization of the best score
for param_set in param_grid do              ▷ Loop over parameter combinations
    current_model ← create_model_with_params(model, param_set)   ▷ Create model with current parameters
    current_score ← validate_model(current_model, data, labels)  ▷ Evaluate the current model
    if current_score < best_score then      ▷ If this score is better than the previous best
        best_score ← current_score
        best_params ← param_set
return best_params, best_score              ▷ Return the best parameters and their score
```

To ensure that our choice of hyperparameters generalizes well across different portions of the dataset and is not biased towards a specific subset, we performed a cross-validated Grid Search. Specifically, we used a 10-fold cross-validation on 20% of the entire dataset reserved for validation. This means that the grid search algorithm would not only search for the best hyperparameters, but would also evaluate their performance across different folds of the validation set, ensuring a more robust and generalized model.

The set of hyperparameters obtained is in Table 5.1.

Parameter	Value
C	50
gamma	0.1
kernel	rbf

TABLE 5.1: Optimal SVM Hyperparameters

We note that a non-linear kernel (RBF) was chosen. Additionally, the training phases with a linear kernel were computer intensive and yielded poor results (less than 30% accuracy). This suggests that the input data are not, a priori, linearly separable. We obtained a relatively high C value, which reinforces the non-linear trend of the data.

We obtained the following results depicted in Table 5.2.

	Precision	Recall	F1-Score	Support
angry	0.80	0.81	0.80	53
disgust	0.62	0.69	0.65	42
fear	0.69	0.71	0.70	56
happy	0.73	0.68	0.71	53
neutral	0.79	0.74	0.76	46
sad	0.67	0.69	0.68	49
surprised	0.80	0.74	0.77	47
accuracy			0.73	346
macro avg	0.73	0.72	0.73	346
weighted avg	0.73	0.73	0.73	346

TABLE 5.2: SVM classification report

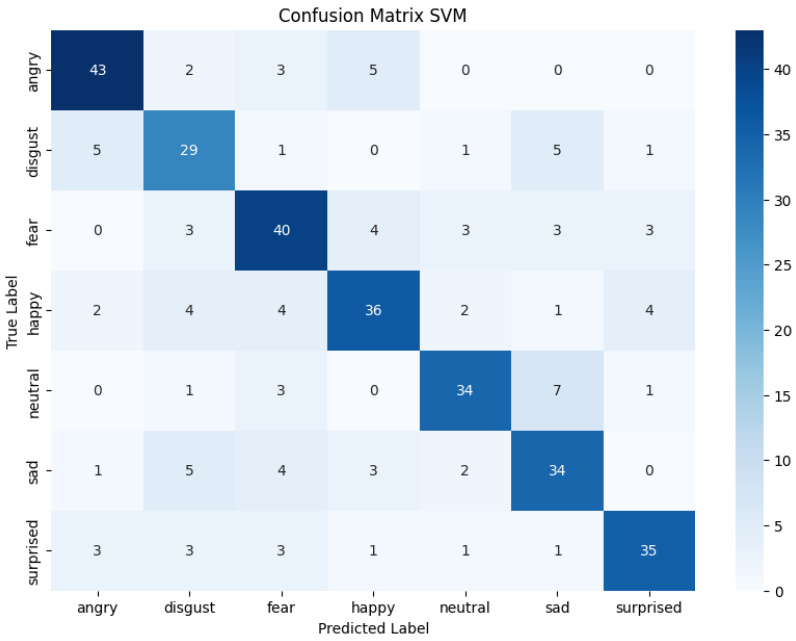


FIGURE 5.1: Confusion matrix SVM

The SVM model achieved an accuracy rate of 73%, which is fairly good for emotion classification tasks.

"Disgust" was the most challenging emotion to classify, hinting at possible overlaps or insufficient distinct features in the data for this emotion. Conversely, the emotions "angry" and "surprised" were the best-classified, indicating the model's adeptness at recognizing these emotions.

Also, notable classification errors were evident. For instance, out of 41 observations labeled "happy," 5 were misclassified as "angry". Additionally, roughly 10% of "fear" samples were mistakenly labeled "happy".

5.2 MLP

In emotion classification using neural networks, the architecture and design decisions are crucial. This section will first outline our chosen Multi-Layer Perceptron (MLP) structure, detailing layer choices, activation functions, and efforts to combat overfitting. We'll discuss the training phase, including our preferred loss function, optimizer, and evaluation metrics. This leads to a visual representation of the MLP's performance and a brief comparison with the Support Vector Machine (SVM).

We then pivot to the One-vs-Rest strategy using MLP. Here, each emotion is classified against all others, providing a unique angle to classification. By comparing this method with the standard MLP, we aim to understand their respective benefits and efficiencies. This section aims to pinpoint the optimal configuration for emotion classification, offering insights for future research.

5.2.1 Regular MLP

Following numerous experiments, we settled on an architecture comprising 5 hidden layers. For both the input layer and the hidden layers, we chose the ReLU activation function, with 256 units per layer. For the output layer, the softmax activation function was selected, with 7 neurons, corresponding to the 7 classes we aimed to predict. The architecture can be seen in Table 5.3.

Given that neural networks tend to overfit the training set, we were meticulous in regularizing the weight updates of the model during the training phase. This included implementing a dropout rate of 0.2 between each layer, an L2 penalty term of 0.002, and an early stopping condition based on the loss value of the validation set. All these measures were adopted to prevent the model from fitting the noise present in the dataset. We selected the categorical cross-entropy loss function (see Equation 4.7.4, with the Adam optimizer, and the accuracy metric for evaluation.

Also, during the training phase, we employed a batch size of 128 for feedforward before backpropagating the error term. This yielded to results presented in Table 5.4:

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 256)	5888
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 256)	65792
dropout_1 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 256)	65792
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 256)	65792
dropout_3 (Dropout)	(None, 256)	0
dense_4 (Dense)	(None, 256)	65792
dropout_4 (Dropout)	(None, 256)	0
dense_5 (Dense)	(None, 7)	1799
Trainable params		270,855
Non-trainable params		0

Table 5.3: MLP Architecture

	Precision	Recall	F1-Score	Support
angry	0.59	0.67	0.63	48
disgust	0.61	0.77	0.68	53
fear	0.65	0.58	0.61	60
happy	0.62	0.65	0.64	46
neutral	0.77	0.53	0.63	43
sad	0.64	0.61	0.62	46
surprised	0.67	0.66	0.67	50
accuracy			0.64	346
macro avg	0.65	0.64	0.64	346
weighted avg	0.65	0.64	0.64	346

Table 5.4: MLP classification report

Thanks to the sequential nature of neural network training, we can monitor the loss function of both the training and validation sets over the course of iterations. This visual inspection allows us to assess if the model is overfitting to the training data. Another visual metric of model performance is the ROC curve. This assists not only in evaluating our model's performance but also in understanding the trade-off between true positives and false positives.

A variety of model architectures were explored with the MLP. Finding the right balance between training set accuracy and validation accuracy was challenging. The final architecture of our MLP model was more complex compared to other configurations. It marginally underperformed when contrasted with

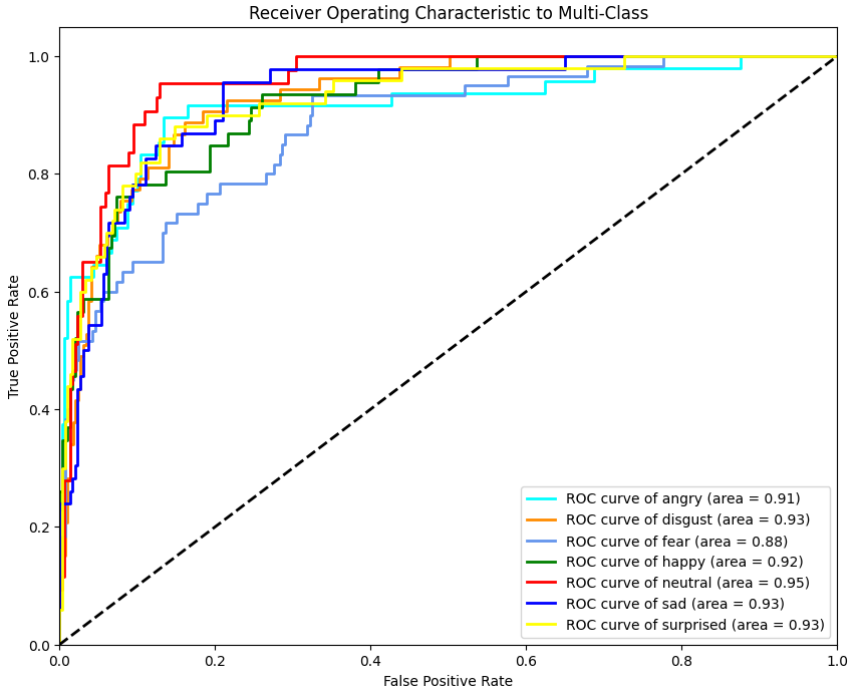


FIGURE 5.2: Receiver Operating Characteristic to Multi-Class

the SVM with an RBF kernel, possibly due to the low-dimensional input space. Notably, even with some surprising outcomes, the training phases of the MLP were robust, not exhibiting signs of overfitting. This hints at a potentially better generalization capacity beyond the datasets RAVDESS and SAVEE.

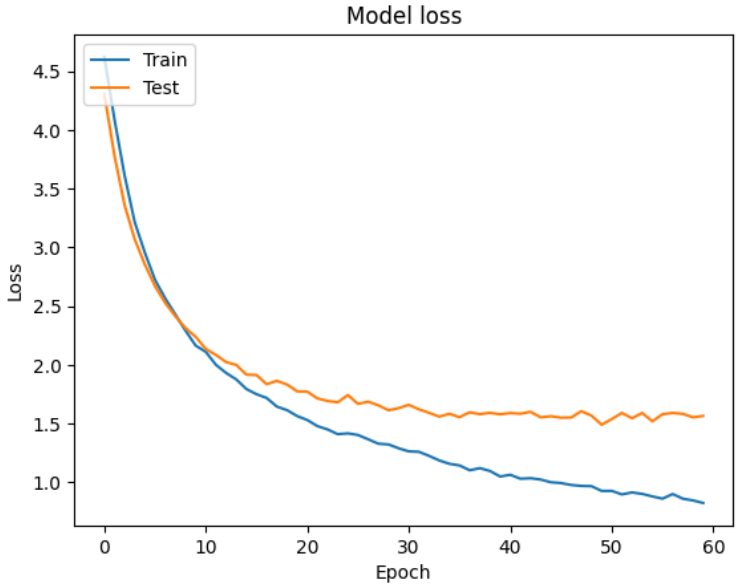


FIGURE 5.3: Model loss

5.2.2 One vs Rest strategy with MLP

Based on our observations in the SVM section (refer to Section 5.1), it is evident that SVM outperforms the basic MLP. Consequently, we aim to compare the SVM classifier with an MLP model designed for per-class classification, employing the "class i versus all" strategy.

Considering the class imbalance inherent during the training phase of each model, we opted to weight the observations of class i. This weighting is done in proportion to the number of elements of class i relative to the count of elements in the other classes.

The methodology employed is as follows:

For the i^{th} MLP model:

- Positive samples originate from class i.
- Negative samples are drawn from the remaining classes.

When classifying an observation x:

The observation x is passed through each of the N MLP models, producing N outputs:

$$O_i = MLP_i(x) \quad \text{for} \quad i = 1, 2, \ldots, N$$

Here, O_i signifies the confidence score (or probability) that x is a member of

class i.

Subsequently, the decision rule applied is:

Assign x to the class with the most significant confidence score:

$$\text{Class}(x) = \underset{i}{\text{argmax}}\, O_i$$

The magnitude of the prediction can be deduced from O_i; a larger value of O_i indicates a stronger confidence by the MLP that x pertains to class i.

The results obtained are presented in Table 5.5.

	Precision	Recall	F1-Score	Support
angry	0.69	0.69	0.69	48
disgust	0.80	0.68	0.73	53
fear	0.71	0.57	0.63	60
happy	0.58	0.65	0.61	46
neutral	0.69	0.84	0.76	43
sad	0.56	0.59	0.57	46
surprised	0.62	0.66	0.64	50
accuracy			0.66	346
macro avg	0.66	0.67	0.66	346
weighted avg	0.67	0.66	0.66	346

TABLE 5.5: MLP OVR classification report

The results from the MLP one-vs-rest approach indicate a modest performance increase of approximately 2%. Although there were noticeable improvements in recall and F1-score, these improvements might be attributed to the use of weighted classes.

However, a significant consideration is the computational demand. The necessity to fit seven distinct models is considerably resource-intensive. Balancing this computational overhead against a marginal increase in accuracy raises questions about the feasibility and practicality of this approach in specific scenarios. While the method offers certain benefits, its suitability should be carefully evaluated, especially if computational resources are constrained.

5.3 labels projection in Valence-Arousal model

In addition to the categorical emotion model, literature review reveals the presence of another representation known as the circumplex model. This model maps an emotion in a two-dimensional space, particularly along the axes of valence and arousal.

Although we were intrigued by this approach, we encountered a challenge: our datasets, such as RAVDESS and SAVEE, were not labeled appropriately for this method. Notably, RAVDESS lacks a specific label for the "calm" emotion. Given this limitation, we made an assumption to treat the "neutral" emotion as a proxy for "calm." This is underpinned by the circumplex model (see Figure 3.2) which posits that both "calm" and "neutral" share zero arousal, though they exhibit slight differences in valence. However, a workaround was conceived by assigning coordinates to the pre-labeled emotions in our dataset based on the circumplex model's structure (refer to Figure 3.2). These coordinate assignments are detailed in Table 5.6.

To project the emotions into this 2D space, we opted for the Multi-Layer Perceptron (MLP) model. The architecture mirrored the one illustrated in Table 5.3. Notably, this approach leaned towards regression rather than classification. As a consequence, we employed the Mean Squared Error (MSE) as the loss function. The Mean Squared Error (MSE) for n samples is defined in Equation 5.1:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5.1}$$

Where:

- n is the total number of samples.
- y_i is the actual value of the i^{th} sample.
- $\hat{y}_i$ is the predicted value of the i^{th} sample.

Given the potential noise associated with some emotions, an attempt was made to filter out and retain only those emotions that clustered most coherently. The resulting projection for the seven emotions is visualized in Figure 5.4. For clarity, Voronoi regions were depicted for each projected class, and data points were color-coded based on class membership.

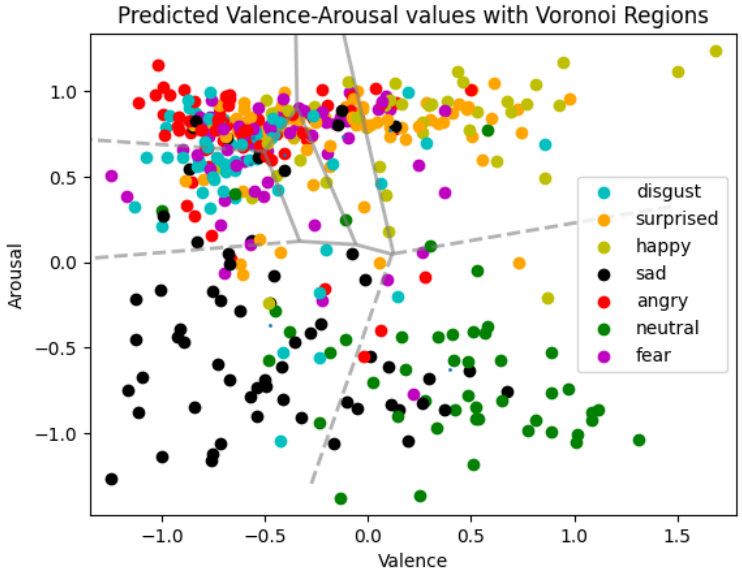


FIGURE 5.4: projection of 7 classes

An examination of Figure 5.4 shows a distinct overlap among the classes "fear", "angry", and "surprised". On the other hand, the regions associated with emotions "sad", "happy", and "calm" appeared to be well-spaced with minimal overlap. It was further observed that the overlap was predominantly along the valence axis as compared to the arousal axis. Because of what we saw, we did another projection by retraining the same model. This time, we only looked at four emotions: "angry", "happy", "sad", and "calm". You can see the results in Figure 5.5.

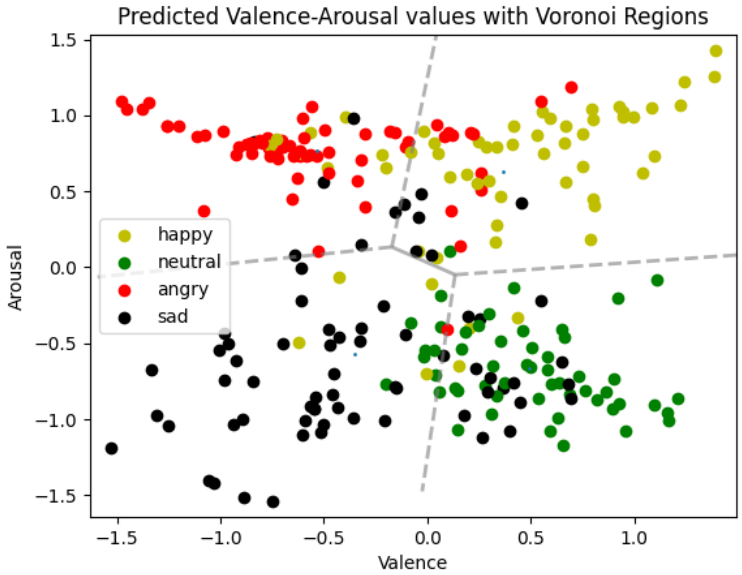


FIGURE 5.5: 4 classes projection

Figure 5.5 presents a more distinct demarcation of the Voronoi zones. Notably, minimal overlap was detected on the arousal axis. However, "calm" and "sad" displayed slight overlaps on the valence axis, a characteristic shared by "angry" and "happy" as well. Such findings suggest that in audio samples, arousal or excitement is distinctly depicted compared to the valence of emotions (i.e., positive versus negative emotions).

The reader will find the Table 5.6. summarizing the positions attributed to each label. Note that *neutral* has been removed.

Emotion	**Valence**	**Arousal**
neutral	1	-1
angry	-1	1
disgust	-1	0.7
fear	-0.7	1
happy	1	1
sad	-1	-1
surprised	0	1

TABLE 5.6: Mapping of emotions to arousal-valence coordinates.

When we changed from 7 labels to 4, it became easier for our models (SVM, MLP, MLP with OVR strategy) to make predictions. Here's the results obtained:

- SVM : The SVM did much better with 4 labels, getting right 83% of the time. This means it found it simpler to tell the difference between the four groups.
- Regular MLP (Multi-Layer Perceptron): This is a type of neural network. It was correct 75% of the time after we reduced to 4 labels. This isn't as high as SVM, but it's still pretty good.
- MLP OvR (One-vs-Rest): Here, we train the model to look at one group at a time and see how it's different from the others. It did slightly better than the regular MLP, being right 76% of the time.

In short, going from 7 labels to 4 made our models work better and give more accurate results.

The use of the Valence-Arousal model provides a different angle to understanding and modeling emotions in voice. It's known that assigning coordinates to each label might seem random, but there weren't many options.

The results weren't great when trying to fit all emotions. However, the picture becomes clearer when focusing on the four distinct emotions: calm, angry, sad, and happy. The Voronoi regions almost directly align with the Valence-Arousal axes, which is an interesting observation.

Also, in our effort to understand emotions using the Valence-Arousal model, some important points must be mentioned about our methods. The placing of emotions on this model was done by us manually. Because of this, there can be some personal views in where we put each emotion. This way of placing emotions hasn't been tested with detailed scientific methods. It's a starting point to see if we can find any interesting patterns, but it's important to use caution when looking at our results. We recommend thinking of our placements as an initial idea, not a final conclusion. Perhaps in future research, more detailed methods can be used to confirm or change our placements and provide a clearer understanding of emotions in this model.

In conclusion, while the traditional categorical model offers a straightforward understanding of distinct emotional states, the Valence-Arousal model offers a more fluid and continuous representation. The choice between them depends on the specific requirements of the analysis or application in question.

Chapter 6

Conclusion

At the beginning of our project, we wanted to see how machines could understand our feelings when we talk. We know that our voices can show our emotions. We imagined a world where machines can "hear" and understand our feelings. This could change things like how fast help comes during an emergency call or how companies understand their customers.

Our studies gave us some surprising results. We found that SVM did better than MLP and MLP OVR. We thought SVM was just a simple model to compare with, but it did really well. This teaches us to always question what we think we know. Another thing we found was about the Arousal-Valence model. We couldn't see clear results, but we feel if we had better data, it could work well.

A lot of experts say that RNN or CNN are the best models for understanding emotions. But for us, SVM was the best. Maybe it's because of the different kinds of data we used. We wonder if other studies got good results because they only used one kind of data. When we looked more into the valence-arousal model, it seemed that feelings are easier to see on the arousal side than on the valence side.

But our study wasn't perfect. The main problem was our data. It wasn't varied enough. Also, the way we set things up was based on what we thought, not on other data that could have helped.

For those who want to do this kind of work in the future, there are some things they can look at. They might study how changing the hop length, window size, sequence length, or number of MEL filters affects results. Also, maybe it would be better to use real emotions from people instead of acted ones. And

there's also the idea of looking at other models, like valence-arousal-dominance.

To finish, a lot has changed since we started our project. Now there are things like Microsoft's Vall-E, ChatGPT, and Bard that can understand and show emotions. As we move into the future, machines that can understand how we feel will be very important. They can help us in many ways and make our lives better.

.1 Appendix

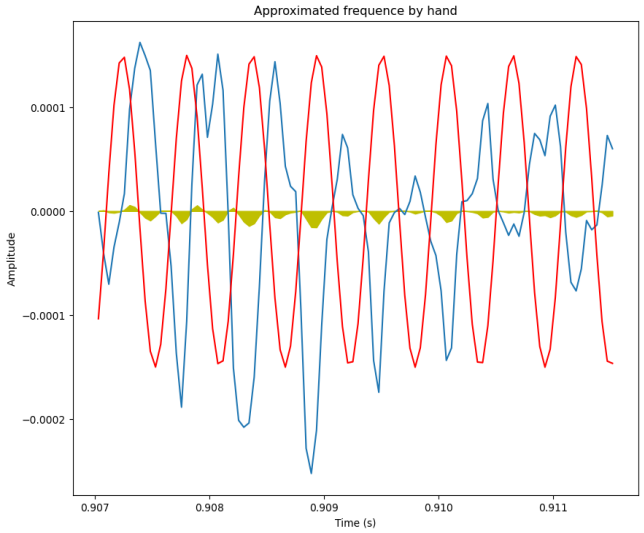


FIGURE 1: FT: synthetic signal not matching the basic one

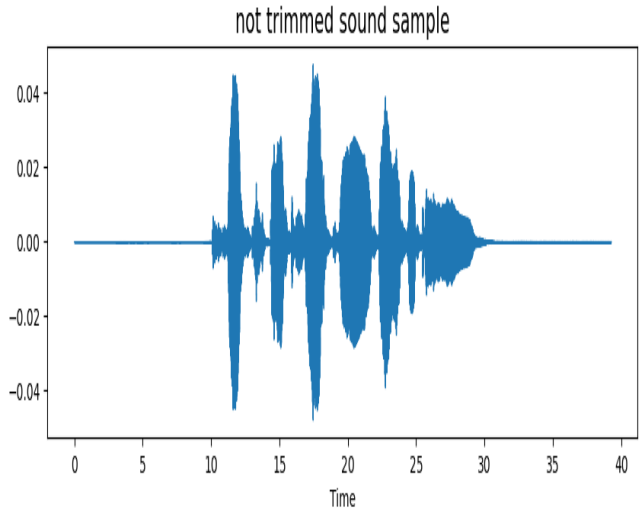


FIGURE 2: Data : Sound sample before trimming

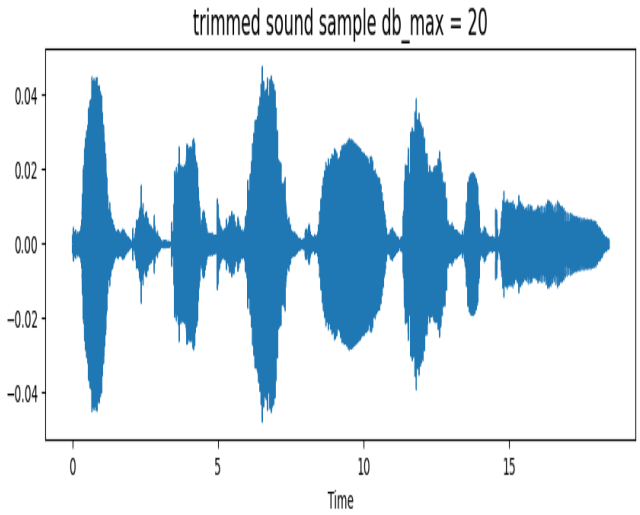


FIGURE 3: Data: Sound sample after trimming 20dB

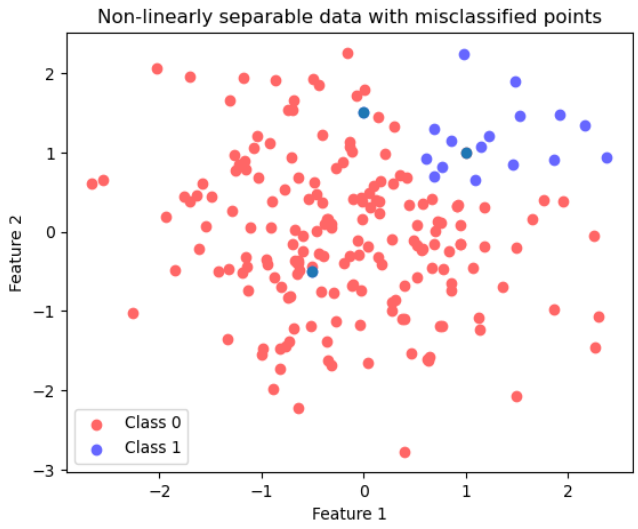


FIGURE 4: SVM: Non-linearly separable data

Algorithm 2 Forward Feature Selection

```
1: S ← {}                                          ▷ Initialization of the selected feature set
2: for i = 1 to n do                               ▷ Loop over the number of features
3:     maxScore ← −∞
4:     bestFeature ← −1
5:     for feature in Features do                  ▷ Loop over the features
6:         if feature not in S then                ▷ If the feature is not already selected
7:             currentScore ← evaluate(S ∪ {feature})   ▷ Evaluate the current feature set plus the new feature
8:             if currentScore > maxScore then     ▷ If this score is the best so far
9:                 maxScore ← currentScore
10:                bestFeature ← feature
11:    S ← S ∪ {bestFeature}                       ▷ Add the best feature to the selected feature set
```

Bibliography

Akçay, M. B. and K. Oğuz (2019). "Speech Emotion Recognition: Emotional Models, Databases, Features, Preprocessing Methods, Supporting Modalities, and Classifiers". In: *Speech Communication* 2682. DOI: 10.1016/j.specom.2019.12.001.

Akçay, Mehmet Berkehan and Kaya Oğuz (2020). "Speech emotion recognition: Emotional models, databases, features, preprocessing methods, supporting modalities, and classifiers". In: *Speech Commun.* 116, pp. 56–76.

Amjad, A., L. Khan, and H. Chang (2021). "Effect on speech emotion classification of a feature selection approach using a convolutional neural network". In: *PeerJ Computer Science* 7, e766. DOI: 10.7717/peerj-cs.766.

Ancilin, J. and A. Milton (2021). "Improved speech emotion recognition with Mel frequency magnitude coefficient". In: *Applied Acoustics* 179, p. 108046. DOI: 10.1016/j.apacoust.2021.108046.

Aouani, H. and Y. B. Ayed (2020). "Speech Emotion Recognition with deep learning". In: *Procedia Computer Science* 176, pp. 251–260. DOI: 10.1016/j.procs.2020.08.027.

Busso, C. et al. (2008). "IEMOCAP: Interactive emotional dyadic motion capture database". In: *Journal of Language Resources and Evaluation* 42.4, pp. 335–359.

Bąk, H. (2016). "A Different Look at Emotion Processing Models". In: *Emotional Prosody Processing for Non-Native English Speakers*, pp. 67–77. DOI: 10.1007/978-3-319-44042-2_4.

Cannon, W.B. (1927). "The James-Lange theory of emotions: A critical examination and an alternative theory". In: *The American Journal of Psychology* 39.1/4, pp. 106–124.

Dahake, P. P., K. Shaw, and P. Malathi (2016). "Speaker dependent speech emotion recognition using MFCC and Support Vector Machine". In: *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*. DOI: 10.1109/icacdot.2016.7877753.

Fan, W. et al. (2021). "LSSED: A Large-Scale Dataset and Benchmark for Speech Emotion Recognition". In: *ICASSP 2021 - 2021 IEEE International*

Conference on Acoustics, Speech and Signal Processing (ICASSP). DOI: 10.1109/icassp39728.2021.9414542.

Hess, U. (2017). "Emotion Categorization". In: *Handbook of Categorization in Cognitive Science*, pp. 107–126. DOI: 10.1016/b978-0-08-101107-2.00005-1.

Hess, Ursula and Pierre Thibault (2009). "Darwin and emotion expression". In: *American Psychologist* 64.2, pp. 120–128. DOI: 10.1037/a0013386.

James, W. (1884). "II.—WHAT IS AN EMOTION?" In: *Mind* os-IX.34, pp. 188–205. DOI: 10.1093/mind/os-ix.34.188.

Ke, Xianxin et al. (2018). "Speech emotion recognition based on SVM and ANN". In: *International Journal of Machine Learning and Computing* 8.3, pp. 198–202.

Khalil, R. A. et al. (2019). "Speech Emotion Recognition Using Deep Learning Techniques: A Review". In: *IEEE Access* 7, pp. 117327–117345. DOI: 10.1109/access.2019.2936124.

Livingstone, S. R. and F. A. Russo (2018). "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English". In: *PLOS ONE* 13.5, e0196391. DOI: 10.1371/journal.pone.0196391.

Ma, Wing-Kin, Tsung-Han Hsieh, and Chong-Yung Chi (2010). "DOA Estimation of Quasi-Stationary Signals With Less Sensors Than Sources and Unknown Spatial Noise Covariance: A Khatri–Rao Subspace Approach". In: *IEEE Transactions on Signal Processing* 58.4, pp. 2168–2180. DOI: 10.1109/tsp.2009.2034935.

MSP-Podcast (n.d.). https://ecs.utdallas.edu/research/researchlabs/msp-lab/MSP-Podcast.html. Accessed: 2023-05-08.

Posner, J., J. A. Russell, and B. S. Peterson (2005). "The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology". In: *Development and Psychopathology* 17.03. DOI: 10.1017/s0954579405050340.

PS, Sreeja and G Mahalakshmi (2017). "Emotion models: a review". In: *International Journal of Control Theory and Applications* 10.8, pp. 651–657.

Russell, J. A. and A. Mehrabian (1977). "Evidence for a three-factor theory of emotions". In: *Journal of Research in Personality* 11.3, pp. 273–294. DOI: 10.1016/0092-6566(77)90037-x.

Sharma, G., K. Umapathy, and S. Krishnan (2020). "Trends in audio signal feature extraction methods". In: *Applied Acoustics* 158, p. 107020. DOI: 10.1016/j.apacoust.2019.107020.

Sonmez, Y. U. and A. Varol (2019). "New Trends in Speech Emotion Recognition". In: *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*. DOI: 10.1109/isdfs.2019.8757528.

Sugan, N. et al. (2018). "Performance Comparison of Different Cepstral Features for Speech Emotion Recognition". In: *2018 International CET Conference on Control, Communication, and Computing (IC4)*. DOI: 10.1109/cetic4.2018.8531065.

Toyoshima, I. et al. (2023). "Multi-Input Speech Emotion Recognition Model Using Mel Spectrogram and GeMAPS". In: *Sensors* 23.3, p. 1743. DOI: 10.3390/s23031743.

Wani, T. M. et al. (2021). "A Comprehensive Review of Speech Emotion Recognition Systems". In: *IEEE Access* 9, 47795–47814. DOI: 10.1109/access.2021.3068045.

Yin, H., V. Hohmann, and C. Nadeu (2011). "Acoustic features for speech recognition based on Gammatone filterbank and instantaneous frequency". In: *Speech Communication* 53.5, pp. 707–715. DOI: 10.1016/j.specom.2010.04.008.

Zhao, J., X. Mao, and L. Chen (2019). "Speech emotion recognition using deep 1D & 2D CNN LSTM networks". In: *Biomedical Signal Processing and Control* 47, pp. 312–323. DOI: 10.1016/j.bspc.2018.08.035.

Zheng, F., G. Zhang, and Z. Song (2001). "Comparison of different implementations of MFCC". In: *Journal of Computer Science and Technology* 16.6, pp. 582–589. DOI: 10.1007/bf02943243.

Zhou, X., J. Guo, and R. Bie (2016). "Deep Learning Based Affective Model for Speech Emotion Recognition". In: *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE, pp. 486–493. DOI: 10.1109/uic-atc-scalcom-cbdcom-iop-smartworld.2016.0133.

Özseven, T. (2019). "A novel feature selection method for speech emotion recognition". In: *Applied Acoustics* 146, pp. 320–326. DOI: 10.1016/j.apacoust.2018.11.028.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
Faculté des sciences
Place des sciences, 2 bte L6.06.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/sc**