TALLINN UNIVERSITY OF TECHNOLOGY
SCHOOL OF ENGINEERING
Department's title

FEATURE EXTRACTION TECHNIQUES IN MUSIC INFORMATION RETRIEVAL

TUNNUSTE ERALDAMINE MUUSIKATEABE OTSINGUS

MASTER THESIS

Üliõpilane: ZAYN KHASHANA

Üliõpilaskood: 165595MAHM

Juhendaja: OLEV MARTENS, PROFESSOR/HEAD OF RESEARCH GROUP
DMITRY SHVARTS, RESEARCH SCIENTIST

Tallinn 20..

(*On the reverse side of title page*)

AUTHOR'S DECLARATION

Hereby I declare, that I have written this thesis independently.
No academic degree has been applied for based on this material. All works, major viewpoints and data of the other authors used in this thesis have been referenced.

16 MAY 2020

Author: ZAYN KHASHANA
/ZAYN /

Thesis is in accordance with terms and requirements

"....." ..... 20….

Supervisor: ….....
/signature/

Accepted for defence

"....."..... 20… .

Chairman of theses defence commission: .....
/name and signature/

Non-exclusive Licence for Publication and Reproduction of Graduation thesis[1]

I, Zayn Khashana hereby

1. grant Tallinn University of Technology (TalTech) a non-exclusive license for my thesis Feature Extraction Techniques in Music Information Retrieval

supervised by

Olev Martens / Dmitry Shvarts

1.1 reproduced for the purposes of preservation and electronic publication, incl. to be entered in the digital collection of TalTech library until expiry of the term of copyright;

1.2 published via the web of TalTech, incl. to be entered in the digital collection of TalTech library until expiry of the term of copyright.

1.3 I am aware that the author also retains the rights specified in clause 1 of this license.

2. I confirm that granting the non-exclusive license does not infringe third persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

---

[1] *Non-exclusive Licence for Publication and Reproduction of Graduation Thesis is not valid during the validity period of restriction on access, except the university`s right to reproduce the thesis only for preservation purposes.*

Zayn (*signature*)

16 May 2020 (*date*)

Mechatronics Department

THESIS TASK

Student: Zayn Khashana , 165595MAHM

Study programme, MASTERS OF MECHATRONICS /MAHM02/13

main speciality:

Supervisor(s): OLEV MARTENS / POFESSOR, HEAD OF RESEARCH GROUP

Consultants: DMITRY SHVARTS / RESEARCH SCIENTIST

…… (company, phone, e-mail)

Thesis topic: *Feature Extraction Techniques in Music Information Retrieval*

(Estonian) Tunnuste eraldamine muusikateabe otsingus

Thesis main objectives:

1. Determining better performing alternatives for audio feature extraction.
2. Testing these audio feature extractors with noisy signals.
3. Effect of filter banks in the process of audio feature extraction.

Thesis tasks and time schedule:

No	Task description	Deadline
1.		
2.		
3.		

Language: English **Deadline for submission of thesis:** 18 MAY 2020

Student: ZAYN KHASHANA / 165595MAHM

/signature/

Supervisor: Olev Martens

/signature/

Co-supervisor: Dmitry Shvarts

/signature/

Head of study programme: Mart Tamre .....

/signature/

Terms of thesis closed defence and/or restricted access conditions to be formulated on the reverse side

CONTENTS

PREFACE

As a musician and engineer, it is much interesting for me to write about a topic in music information retrieval, and most likely to continue deepening my knowledge in. As music has a role in our daily routine, feature extraction techniques can ultimately improve the way computers listen to music which will definitely impact our music listening using the many available digital platforms.

In all possible ways, I thank everyone who supported and advised me during my study period, teachers, friends and study mates.

My supervisors in this work, Professor Olev Martens and Mr.Dmitry Shvarts, thanks for your assistance and advice. Professor Mart Tamre and all Mechatronics department teachers, thanks as you made our study much enjoyable and fulfilling.

Last but not by any way least, My Family, father, mother, sisters and brother, for supporting me all the time, I love you so much.

List of abbreviations and symbols

Hz	=	Hertz
Khz	=	Kilo Hertz
MFCC	=	Mel frequency cepstral Coefficient
Ms	=	Millisecond
RMSE	=	Root Mean Square Error
GTCC	=	Gammatone Cepstral coefficients
FT	=	Fourier Transform
DFT	=	Discrete Fourier transform
MIR	=	Music information retrieval
BFCC	=	Bark frequency cepstral coefficients
LSTM	=	Long short-term memory

INTRODUCTION

In recent years, audio data and music resources became enormous enough that it urged the necessity to develop systems to manage it and make its information retrieval easier for users as every one of us is considered a music listener, have our own musical style and taste, also our moods change which will impact the type and genre of music we listen to on a daily basis, we want to spend short time to find the music that fits our mood and environment, some of us who are musicians and composers want to publish their music on the web in order to be listened to, so we go search for platforms that help us to do so. Luckily that the emerging science of music information retrieval came to help achieve these tasks through many musical platforms that are available today whether to listen or publish music like Spotify, SoundCloud, etc., these platforms requires intelligent algorithms to process this large amount of musical data being published monthly and give the desired outcome we expect.

As the field of music information retrieval is vastly growing in the last two decades, researchers are addressing the problems that arise and work to find better performing solutions or alternatives to existing ones, development for recommendation systems or music searching platforms are always being discussed. One of the main tasks that these systems depend on is the genre classification of music, based on the results of this task comes the results of other systems, a powerful music recommendation system or application won't give you the musical style you prefer unless it is powerful enough to classify music genres with high accuracy, this task has been performed with different set of features and usually the development go towards improving the learning algorithm, however feature extraction and selection is the most important pre-processing step before learning, when selecting feature extractors, it is important that these extractors are powerful enough to extract relevant information from the audio signals that are distinctive descriptors of the music genres, so the learning algorithm can distinguish between them easily.

Humans are naturally good in classifying audio with good accuracy and some studies had been made on this topic, so in order to achieve classification accuracy closer to that of the human ear, it is required to build a learning model that gets same accuracy level or even higher that have similar properties to our hearing system. In this study the task of music genre classification will be addressed from the perspective of finding the closest biomimicry feature extractors that are able to classify audio as human ear does, the accuracy of the classification model is a direct indication of the used feature extractors, MATLAB software will be used to develop a classification model to test the feature extractors.

1. MUSIC INFORMATION RETRIEVAL

Music as an artistic global activity that has different styles and cultural taste in which millions of pieces are being produced every year, also with the advancements of the digital audio and storage systems and its huge musical content raised the need to develop intelligent systems to manage this enormous database and develop it to an extent to minimize our interaction with it and the time spent searching for a specific content. The science of music information retrieval came to prominence as it is facilitating everything, we need to manage digital audio databases whether recommendation systems, search engines and even producing and publishing, these services became available and easily accessible with various platforms and application in the recent years like YouTube, Spotify, etc., however as this science of information retrieval from music is still considered an infant science and there is much room for development and problems that arise are being addressed and discussed.

1.1 Motivation

All tasks in this field is based on the idea of extracting relevant and correlated information from music and incorporate it with automated algorithms with the help of various digital representations for musical pieces and high processing devices. Signal processing techniques and methods to extract this information from the audio signals provides the ability to improve it whether by denoising or applying some techniques to improve sound clarity, then using this extracted information and incorporate it with machine learning algorithms that were developed recently. The development of effective tools to manage, edit and produce audio, it is required to have sufficient information about the context and content of music and audio data available in various databases. Context of audio is the indication to which it belongs from a large audio collection. For example, two male singers in a pop music song would be similar while any information about the composition of an audio signal is referred to its content. For example, if you know a particular part of a classical song is played by a violin solo [1].

The reason I decided to research such topic is to make an empirical study on audio feature extractors that mimics the biological auditory system. For example, MFCC is the state of the art that is been used in the previous years that makes it able for machine learning algorithms to distinguish between audio files, it mimics and scales the non-linear way humans hear and classify music, however the question is that there are other feature extractors that may have a better distinctive features and can better scale and mimic the human ear, for example using gammatone filter banks or bark frequency

cepstral coefficients and others. In this study these features will be discussed and evaluated by a classification model with different types of audio signals to determine which of them are the best performing for each type of audio signals.

1.2 MIR problems & Applications

Music information retrieval is involved in almost any digital interaction we experience with music, with many producers, composers and listeners, it's important to use these applications to fulfill our queries related to music. There is two types of applications in this field, music retrieval which is involved in finding user's queries whether searching for a song, an album or a preferred genre. Michael A. Casey [2] introduced a good explanation on different types of music retrieval systems, one is mentioned " high specificity systems", this type when the system retrieve an approximate or match the characteristics of the query issued by the user or able to capture the exact content of musical data. For example, a user wrote a song name for an artist, then the retrieval system returns back the same exact song the user asked for or a close approximation to it, but systems that retrieve musical content that have low matching characteristics of that given by the user are called " low specificity". For example, a query that was given requiring a specific song, the system gives back songs that are in the same genre, so there are low matching criteria between the query and the genre. In this section will explained the most required applications and problems in the field of MIR.

1.2.1 Applications in music retrieval

Audio fingerprinting It is a form of representation of an audio signal in which it contains information specific to the signal meant to be represented, it captures features and audio descriptions highly correlated with the song. An example for this technology that we use is Shazam as the user records a segment of an audio and its algorithm works to identify the whole song. The main concept of this technology is to pick specific distinctive features that makes this recording unique from other audio available in the database in order to have a high identification accuracy, and this is done by means of signal processing techniques to analyze the time-frequency domain of the audio or common known techniques to purify the signal from a background noise, along with machine learning algorithms, it is possible to build a robust accurate audio fingerprinting system [3].

Cover song identification A song cover means another modified or somehow altered version to the original song which may be structured differently with using different tempo, timbre, instruments played and key arrangement. This task has captured much

attention recently from researchers to understand the underlying scientific principles in this area of research and its relevance with different concepts as music similarity and music cognition. By combining some concepts from music similarity, cultural aspects and music cognition, we can get a clear image of how identification of song covers can be done, usually an accurate measure to determine how accurate an identification system is, to look for the human perception of similar tasks. We can identify many different covers and correlate it to one song by picking out some common features and general representation of the song, we don't know exactly what these distinctive features are, but we can correlate different covers together even with different vocals and instruments. There is a proposal that if the cover has high correlation with the song, then a short segment of this song can be enough to identify it, then timbre can be considered as a distinctive signature and is an important contributor in the similarity measures [4], but the main measure is the original song and other versions to be compared to it and the match happens when the common features between them are highly correlated.

Query by humming This is a type of music retrieval widely used systems in which it is given an input in the form of a short segment of song hummed by the user without having any metadata about the song. It works by extracting certain features from the input audio and compare these features to the features of music in musical databases, then it displays the songs that their features match the input. The efficiency of the system is assessed by its ability to process the input query, analyze and extract its features in a short time and with minimal user interaction. Factors that affect the result of the query is that the user may sing this song out of tone or the identification system in not powerful enough to match the similarity of the query to the music in the database.

1.2.2 MIR problems

As the interest increases in the field of MIR since it was introduced, it still has much room for research and development and it still didn't reach the level to fully manage the huge broadening of digital audio data. Despite the latest advancements in artificial intelligence and different machine learning algorithms, there still a missing gap between music cognition in humans and machines, human evaluation is needed in all tasks in MIR. Another problem that exists is that existing data available for research are inadequate thus it is important to determine all available sources of data involved in music and improve their metadata and make them ready for researchers, also state their legality to be used and provide online repositories. To be mentioned that MIR researches received some criticism of being impractical on large scales, many of researches that have been done are only applicable for small scale data and are

impractical with large scale data, but the evolving of computational powerful tools can definitely help to apply them to large scale data [5].

1.3 Music & audio representations

There are millions of musical data available in digital databases that have different forms representing music, it can be in the form of text as we see as lyrics of the song and this type of formats is textual, also it can be poster of new music albums or audio formats that are used in recordings, another form is music instrument digital interface which is known as MIDI, these are different representations for music and audio in general which are required to be acquainted to, especially for studies and researches in music information retrieval.

1.3.1 Score representation

This form is the most common representation of music in which musical symbols denoting the rhythms, chords and melody of the piece meant to be played, its common name is sheet music. The instructions a musician needs to make the performance is encoded in the sheet music, these instructions are information like note onsets, key signature, duration, pitch and dynamics. This information encoded in the representation can only be read by a musician who has the knowledge to give out the music as states in the sheet. Below figure 1.1 is an example of sheet music for a song name "Just for you" for the Italian pianist Giovanni Marradi.

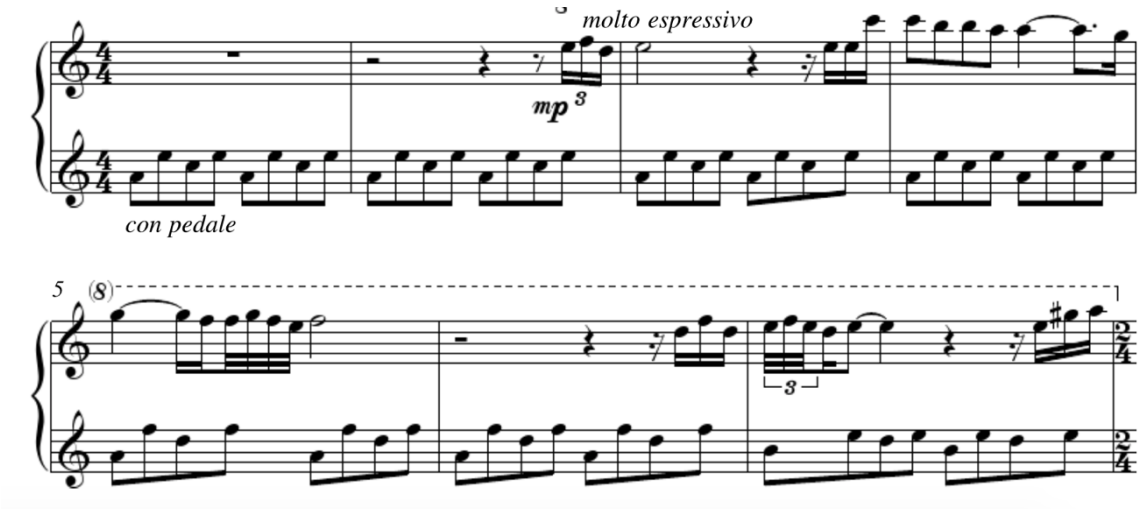

Figure 1.1 Sheet music representation for a contemporary classical song taken from [6]

1.3.2 Waveform and sound representation

Waveform is a representation used to display audio or sound waves, it displays the changes in the amplitude that occurs over time, it is a visualization tool to show the audio, waveform with lower amplitudes indicates a low-pitched sound or soft, while higher amplitudes indicates louder or higher-pitched sounds. Audio signals are generated when an object vibrates, human's voice is generated when vocal chords vibrates, piano sound generates when its strings vibrate by the act of hammering it, the generated sound waves travel through air causing the air molecules to oscillate, these oscillations cause rapid displacements of air particles in the form of compression and rarefaction, then these waves when they hit ear drums it causes certain nerves to vibrate and generate an electrical signal to the brain which is then perceived by humans as sound, or sound recorder or a microphone receives these waves and translates it into the intended sound. These waves behave in a specific manner that determines its periodicity and frequency, this part will be discussed in chapter 3 in details.

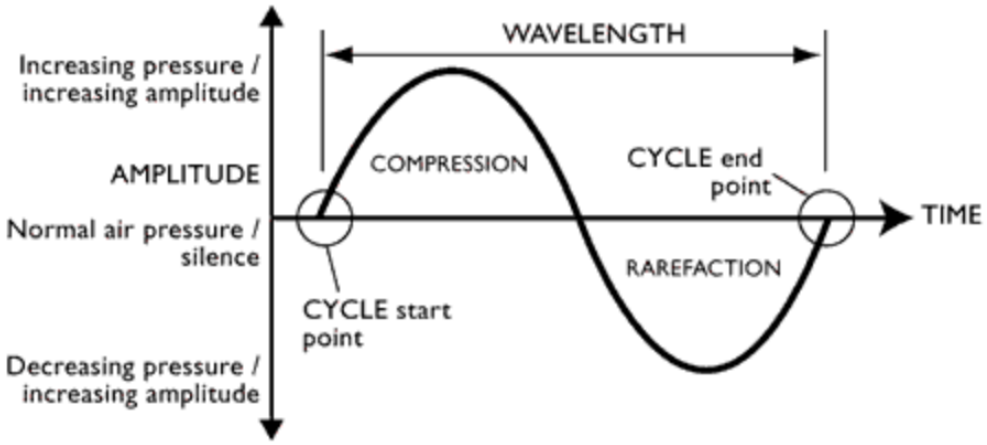

Figure 1.2 shows the sound waveform and the way it is formed, picture taken from [7]

Speech and music waveforms

Understanding the difference between each of these signals is important in to understand whether speech signal analysis or music signals analysis. Figures 1.3, 1.4 below shows the difference between speech and music signals waveform. On a fundamental level music and speech are considered similar and they are being processed and analysed similarly, also they share common features. From the point of frequential standpoint, the range of fundamental frequency for adult male is from 85-180 Hz, while for adult female is from 165-225 Hz, so we can say human voice is ranged from 85-225 Hz ("Voice frequency", n.d), while music can have different

frequency ranges but it must be within the human audible frequency range from 20Hz to 20 Khz, then on a functional level both speech and music are different. The nature of speech and music signals also tend to be disrupted usually by silences, these silences are rather more frequent to occur in speech than in music. A study showed that there are approximately three silences in a sentence, while in music, silences do occur usually but the transients (which occur at the start and end of silence) in speech are more noisy than the transients associated with music [8]. Audio signals are considered to be non-stationary signals and the reason because of the frequent changes in the frequency content, in music the pitch doesn't change much compared to that of speech that are continuously changing. Also, formants in music that occur because of some acoustic properties of the musical instruments used show more stability than formants in speech.

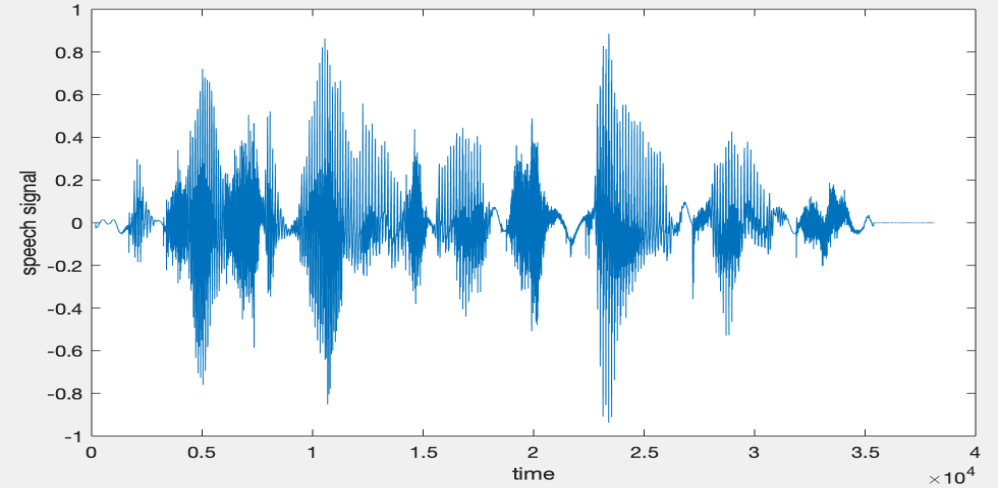

Figure 1.3 shows a speech waveform

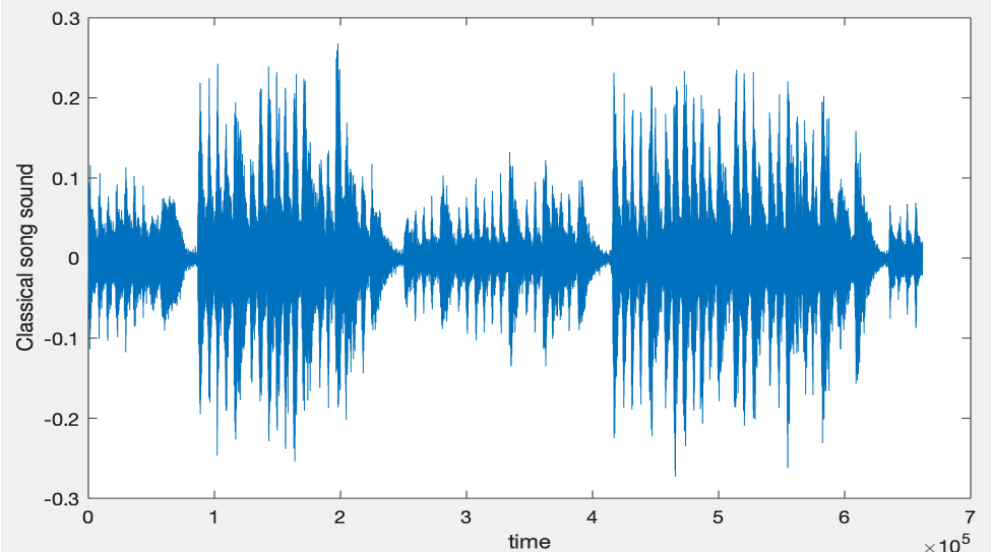

Figure 1.4 shows a classical song waveform

1.4 Software used, audio samples used and list of features

MATLAB software will be used in this study, and the reason is because of the availability of different tools for audio signal analysis and different libraries that will be used for feature extraction process, also for features evaluation, a deep learning model will be made in MATLAB, it offers a wide range of machine learning applications that can be programmed.

Audio samples / Dataset

There is a dataset called GTZAN, which consists of 10 music genres and each genre contains 100 songs, each song is 30 seconds which is a good fit to test features on it. It contains music with vocals and non-vocal music, which will be separated and analysed individually, also a noise will be added to both, the reason is to test the performance of the features used in case of vocal, non-vocals and noised signals. This dataset was used in different studies before for classification of music genres using many machine learning algorithms. However, the main task in this study is not to classify genres but to determine the best performing audio extractors that are much closer to the human auditory system. The accuracy of the final classification of the algorithm developed is a direct indication of the performance of the feature extractors. The higher the accuracy of classification with a specific set of feature extractor, the better it is than another set of a different feature extractor, the closer it is to human perceptual auditory system.

1.5 Objectives and tasks

The goal intended to be reached from the study is to test the effects of auditory feature extractors or potential audio extractors that may hold a better approximation for the human ear and can have better audio classification accuracy and exhibit more noise robustness than conventional feature extractors. The reason is that the human ear has an amazing signal processing and audio analysis properties, so testing and developing audio feature extractors that acts as a biomimicry model that have similar biological properties for example masking property or other properties. In this empirical study audio feature extractors will be discussed and explained and how the signal processing process of audio differs from one to another for each of them, also their applications in real world industries. The results from this study is a deterministic approach to know what best feature extractor performs better for a specific type of audio data like music with vocals, non-vocal music and which is more noise robust.

2 RELATED WORK

2.1 State of the art in MIR

In this chapter discussions will include some important aspects to deepen the understanding of the tasks in MIR and music representations. Most of the research that was done before in music information retrieval tasks pointed out at creating new features to extract information from audio signals or using a different combination of pre-existing features to improve results or improve the currently existing machine learning algorithms to get better results, however there is noticeable lack for an empirical study or a deterministic approach that points the drawbacks of current feature extraction techniques. The notion to perform analysis on any kind of audio, the aim always is to achieve results close enough to our natural judgements as humans, for example a task like audio classification of music genres, we all can listen to a song or even a part of it and give a judgement to which genre it belongs, now we want computer to do the same with accuracy close to ours and here comes the role of using robust feature extraction techniques that works in a similar way to our ears with all of its magnificent audio analysis, the human ear has masking properties and can still recognise sounds, voices and songs in the presence of an external noise in the environment which gives it a biological noise robust properties, it also have a great filtering effects, a detailed description of its properties will be mentioned in next discussions. The purpose of conducting this research is to test and implement different features that have common or close properties to human ear to extract information from audio signals that can be used in many MIR tasks. Also to mention that each task has different set of features, sub-tasks and pre-processing process, there is many tasks in the field of music information retrieval to be considered like beat tracking, pitch tracking, instrument recognition, mood detection and many others [9]. The main focus will be on classification and to determine which are the best performing feature extractors that are more noise robust and efficient in different environmental conditions.

2.1.1 Music Similarity

To describe music, there is some characteristics that should be mentioned such as melody, lyrics or musical instruments played, to generalize this concept, the audio content of a song when it is similar to the audio content of another song, then it is considered similar, the audio content of a song holds its descriptive properties, humans are able naturally to identify similar patterns in music we listen, like melody or repetitive

rhythm or repetitive pattern that happens within one song or different songs, a music genre has different songs that share similar patterns or feel like, for example romantic songs share similar properties and give us similar feelings like tranquillity and romance, from this we conclude that similarity can be within the same song or a different song have similar adjectives, which can be defined as local similarity and global similarity [10], their main working measures is to detect the changes in the signal or repeated patterns because the biggest challenge in music similarity is to be define a measurement criterion that determines similar audio, and this involves picking accurate musical descriptors that acts as a deterministic similarity measure to use for all musical pieces and gives a result of a collection of similar audio files together. So performing such a similarity analysis involves computing a similarity matrix as proposed by Jonathan Foote [11], Similarity measures can differ depending on the task, for example in music genre classification, we need to define each genre's unique descriptors that makes it distinctive from other genres and this will be explained in the next section.

2.1.2 Music Genre Classification

In this part explained the closest work that was done on this task and how a different approach can improve its results. The task of classifying music into genres became a necessity since the availability of enormous amount of large database of musical recordings that needs to be categorized, thousands of songs are being published every month on different platforms, these platforms needs to have a classification system deployed so users can find their preferable genre either for live streaming or buying, the genre of a music is defined as its own style or category that a song belongs to. Despite there is only few genres that are clear for all cultures, there still some genres that are perceived differently in different cultures, still humans have a natural ability to classify music just by listening to 25 seconds of a song as investigated in [12]. And this raised the question of how humans gives such a judgement to which genre a song belongs to, then it was found that it's about the musical properties that a genre has is what makes it distinctive from another one, so researchers wanted to develop a framework or baseline to which songs can be classified and there it is not a consensus by everyone it still prone to have inaccurate classifications as musicians or listeners can disagree with a particular classification [13]. There is two ways to build a classification algorithm, a supervised in which you pre-label the genres and train the algorithm to learn from this data then classify newly provided songs, the other is unsupervised in which you provide the unlabelled dataset of songs but have to determine features that distinguish between genres or characteristics of each genre that makes it different than

the rest and this was determined as melody, harmony, rhythm and sound, these four properties of music can be effectively used to classify music however it is still prone for misclassification for others, and the final judgement is according to people.

A good classification system will ease the hassle of finding the favourite musical style to listen to and this task is one of the most important and required tasks in music information retrieval that even other tasks depends on it, for example recommendation system or browsing system needs a pre-labelled musical database so it can give you results based on your listening history in case of recommendation system or give out a list of the genre you want in case of a browsing system.

Many machine learning algorithms have been developed to classify musical datasets like support vector machine, neural networks, K nearest neighbours, gaussian mixture model and gradient boosting trees. Figure 2.1 shows the process of genre classification.

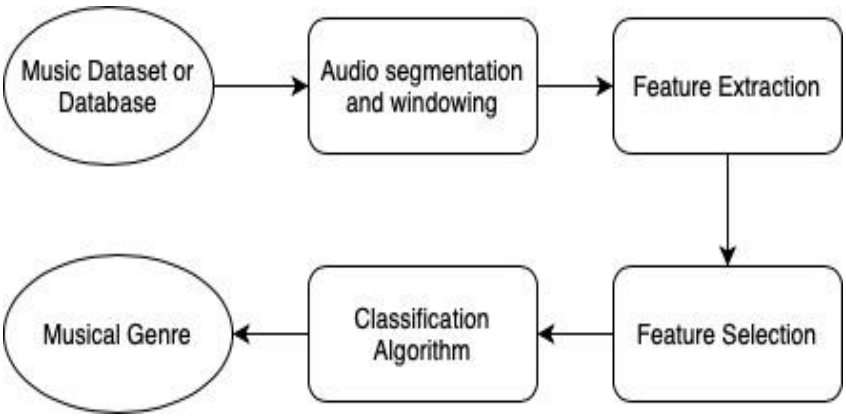

Figure 2.1 The process of music genre classification

In 2002 George Tzanetakis and Perry Cook, they agreed that the statistical properties associated with instruments being commonly used in the genre along with its rhythmic structure can act as a distinctive descriptor for each genre [14], for example classical songs have things in common like instruments used and tempo and other features that gives almost similar feelings different than listening to dance music like techno with high tempo and loudness, so in their proposal they implemented features like MFCC and features that can represent these instruments used and texture of the songs to categorize them to genres and they evaluated these features by developing different classifiers using machine learning and the classification accuracy they achieved was 61% while it is 70% as a human accuracy to classify the same genres [15], they extracted features from the audio signals using MFCC like most classification algorithms, later came new algorithms like support vector machines achieved more than 90%, then

when neural networks came to prominence it proved its efficiency in classification because of its great pattern recognition abilities and multi-layers that gave it an advantage over other classification algorithms, specially convolutional neural networks that are able to accurately pick similarities between many images (in this case images of spectrograms and MFCC) [16]. However to note that these algorithms focus to improve accuracy of the algorithm itself and they use the same feature extractors without regard to the robustness of these features, while the classification accuracy is tremendously affected if the audio contains noise, also audio signals with vocals is different than without, depending on MFCC only is not that sufficient feature extractor and maybe not the closest representation for audio signals from human perceptual perspective, that's the reason this study aims to test different features that may be more noise robust and be more effective with different types of audio signals that can determine which and why these set of features can be used.

2.2 Music Recommendation Systems

The increasing amount of data and thousands of songs being published every month makes it difficult for a user whether a listener or buyer to find their musical style, that's why recommender systems were developed to facilitate for all users to find what they look for in a short amount of time, these systems have access to all musical database up to date, it even makes music discovery easier for those interested in a specific style, at the same time with the developments of machine learning algorithms made it easier for these systems to learn from the listening habits or the behaviour of the user and recommend them relevant songs based on their favourite genre or even based on their mood with less effort, and that is the correlation that combines music information retrieval with recommender systems, so the preliminary step for developing a powerful recommender system is to first perform accurate MIR tasks whether genre or mood classification, so feature extraction techniques that extracts relevant information from the audio signals is what can determine the efficiency of the recommender system [17], as it predicts what a user may like based on the music analysis results from their previously listened music, almost anywhere online we go to buy something or listen we find a list of recommendations appeared that is based on our search history and how long the user spent listening to a specific musical genre. There are many applications that merges MIR and recommendation tasks, the role of MIR is to give a clear analysis for the audio signals by extracting features and recommendation systems find similarities between the user's listening history and the music content out there, an example of that is the automatic playlist generation in Spotify or music apps, we find personalised playlists in our Spotify accounts that is made based on our listening history

and its main algorithm depends mostly on MIR by analysing the audio content stored in our history and the results are being matched with similar audio that exists in the web which is called " static playlist generation" according to Peter Knees & Markus Schedl in their book music similarity [18], another type they talked about called "dynamic playlist adaptation" which depends on explicitly on the user's choices and preferences.

Many researches proposed the idea of generating playlists by the notion of having a huge collection of music and computing the similarities between songs and generating the playlists based on it, but a disadvantage for this is that the user maybe interested in listening to another genre or have different mood that requires different kind of music and then comes the role of taking in consideration the explicit user's choice. In the next section will be explained the latest techniques used in making such systems.

2.2.1 Collaborative Filtering

This type of recommender system is considered as the most popular recommender systems, it works by exploiting the information collected about user's likes and interests and gathering data about their listening habits, clicks they made and their online activity and it is called " implicit rating" because user's didn't intentionally provide it, another type called " explicit rating" because customers intentionally provide an explicit feedback for the service like the music they listened and it is usually in the form of a rating scale [19]. but mainly the performance of all types of recommender systems depends on the amount of data so it can be able to give accurate recommendations. There are two types of the collaborative filtering that will be explained next.

User-based filtering This type is considered a memory based method and its works by assuming that users who share similar interests and rated items similarly or purchased it, then they are most likely to agree on the same items in the future, its mostly logical decisions, also with the advancements in machine learning algorithms been deployed, it gives satisfactory results, it uses k-nearest neighbours to find existitng users who have recorded preferences and rating who are similar to the one required to give the recommendation to, this recommendation technique is considered as easy and accurate to implement but its drawbacks that many people won't rate items also the insufficient amount of users data, so it makes it a difficult task to recommend something for a user with no previous data which is known as " the cold start problem", a good practical solution to solve this issue is to ask few questions to the users and give recommendations according to their answers.

Item-based filtering A different approach is applied in this method in which the algorithm focuses on the similarities between item instead of users as the user-based,

and it sounds logical that if a person likes an item and this item is similar to another one, then the person is more likely to be interested in that other item, its drawback is the same as for all collaborative filtering method which is the data with sufficient users and their activities.

2.3 Music Data Mining

The availability of enormous musical datasets, songs, artists and music production firms that publish millions of songs continuously urged the need to develop techniques to manage these datasets. Music data mining methods allowed to analyse, perform and improve tasks different tasks in MIR, and as known that the accuracy of these task's results is affected by how accurate relevant information have been extracted from audio signals, it is also affected by how efficient these data mining techniques in organising and preparing these data before performing the tasks, whether music genre classification, emotion detection or recommendation system, these all need a pre-prepared data with accurate relevant information, in this section will be discussed the state of the art or the latest techniques and most developed used in music data mining, however as this field is a wide research area, for this reason a well selected topics and techniques will be explained to act as a reference and assist in understanding this field and how to proceed in it.

2.3.1 Music Metadata

It is a description for the information contained in a musical recordings allowing it to be shared and distributed, these data can be the song name, artist name, name of the album, data of release, etc, its importance lies in facilitating performing tasks on the music data, for example a dataset that has accurate metadata about the genres inside along with name of the artists, name of the songs and numbers of its audio files will make easier to perform a task like music genre classification or making a recommendation system so the users can spend less time finding what they want [20]. One of the most used metadata formats is " ID3" and mostly used with mp3 audio files and it helps in storing all relevant information to the audio file making it easier to extract all these information when needed, another method which is online music metadata base as mentioned in music data mining book for T.Li & G.Tzanetakis [21] which helps streaming and listening applications like Spotify and Soundcloud to extract the information of the music the users listen to, examples of these are MusicBrainz and MP3tag.

Acoustic Features These are the elements the an audio is composed of, it's the description you can give to a song you listen regarding its melody or rhythmic structure, instruments used to play it, in which these elements have different frequencies and their combination gives out what we call a song, this section will list the main features that are used to describe audio and which will definitely help in understanding the next chapter which is about feature extraction in time and frequency domains.

Table 2.1 Acoustic features

Feature	Definition
Melody	A set of consecutive rhythmic tones that form the main component of the song
Harmony	It is the superposition of tones and melodies that are simultaneously occurring over time
Key Signature	Arrangement of a set of notes that consists of flats and sharps that form the music composition and are used as an indication to which keys should be played
Tempo	The speed at which the music is played and is measured in beats per minute
Rhythm	The continuous repetition of a musical pattern with its variation as it moves over time
Intensity	It is the measure of amplitude of the vibrations coming out from the sound, can be range from soft to loud
Pitch	The perceived frequency of sound specially the fundamental, it starts as low from the left of the piano and increases to higher pitch as you move to the right of the piano's keyboard
Timbre	It is the colour of the music or is quality that makes the listener gives a judgement of which musical instrument being played
Acoustics	Is the analysis and study of sound considering the external effects applied on it

2.4 State of The Art in Dimensionality Reduction

The aim we always aspire to achieve is high accuracy results in tasks we want to do, so addressing all the steps that affect accuracy is crucial to tackle problems associated with it, since we deal with large audio data that have high dimensionality then reducing these data into lower dimensions without sacrificing its relevant information, or by other means the data after reduction should have the relevant data we need to perform our task is a good contribution to the final's accuracy of our task, there always a threshold where the number of features reaches then the performance begin to decrease as shown in figure 9. Tasks like audio classification if performed with high dimensional data, the result won't be satisfactory and the model's performance will decrease [22], and the reason is because of a phenomenon called " curse of dimensionality " which states that as the number of dimensions increase, this makes the data exists in a larger volume of space than it used to be and it increases exponentially making it difficult to generalize and this causes one of the most well-known problems called " overfitting", so by applying dimensionality reduction techniques, it eases the hassles associated with high dimensional data, in the next discussion, state of the art of best performing dimensionality reduction techniques that can be applied to audio data will be explained and how it can affect the performance of the classification model.

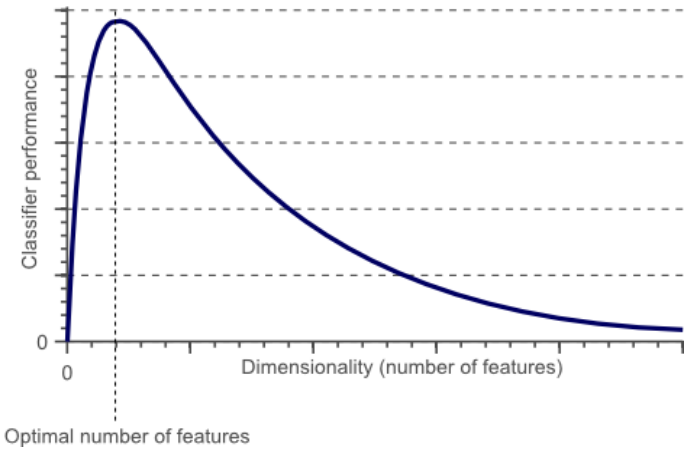

Figure 2.2 shows the curse of dimensionality [23]

In figure 9 a description of how number of features can have an adverse impact on the performance of the classifier, after certain number of features " which are enough for the classifier to be able to generalise the problem and learn the data well ", the performance will start decreasing and as the features increase the model will begin to have overfitting to a degree that the model will begin to learn irrelevant data from the extra features such as noise, so an indispensable step is to perform feature selection before training the algorithm.

2.4.1 Feature Selection

Having a large amount of data that is needed to perform the intended task requires pre-processing steps in order to achieve decent results, classification algorithms performed on large audio data are highly impacted by the techniques used in pre-processing and feature extraction, audio data has high dimensions or attributes that we need to extract, but it's so important to be cautious with selection of these features by determining the best performing attributes and discarding the low or negative relevant ones from the data, it is shown that this step highly impacts the results of the model [24]. The main reason of not using all features in the data is to overcome the negative impact it have on the performance, this impact can avoid overfitting which significantly reduce the accuracy of the model, also more features mean more time needed for training and more computational power, also from a logical point of view, what if you want to make a classification system to classify music based on emotions, then I gave you an audio dataset that contains the country of origin of each song or date of birth of the artist, do you think these information is relevant to the emotion the song gives, the answer is absolutely not, emotions associated with a song comes from a set of features that makes us feel the way we do when we listen, these features can be like timbre, for example violin with piano induces a state of love, romance and tranquillity, bass and drums are good to dance on, and so on, that's why in the next discussion will be explained the state of the art in feature selection methods and dimensionality reduction techniques.

Types of Feature Selection Methods

1- Filter based

This method depends on statistical measures to rate features and give each a score in which we select the best ranked ones depending on their score, there are different types of this method will be discussed below.

Variance Threshold It shows an indication for the features that have fixed values and don't change with observations and shown to have small contribution to the output.

Univariate Selection This is a proposed features selection method that can be applied to data to better our understanding of their features, avoid overfitting and improve model generalisation, it works by performing statistical measures on all features in a dataset and determine the best performing features and the strength of their relationship with the outcome we want in our task, so this method is actually good to

know which will be the most contributing features to the accuracy of the task, examples are Pearson's correlation and chi-squared [25].

Correlation Method The main task of feature selection is to select features that performs well and contribute to the outcome of the task, features that are highly correlated to each other don't add new information to the data and it's just redundant, and this is the method to remove them which improves the accuracy of the model. Each feature is given a correlation coefficient that indicates the level of its correlation to other features, this coefficient varies in arrange between -1 to 1, when its value is zero this means no correlation, while value of 1 means that there is a strong correlation [26].

2- Wrapper based

This method depends developing a machine learning algorithm that evaluate different sets of features and choose the one that has the best performance, this is an iterative process and despite its effectiveness it has a disadvantage of being computationally exhausting, figure 2.3 shows how the algorithm works.

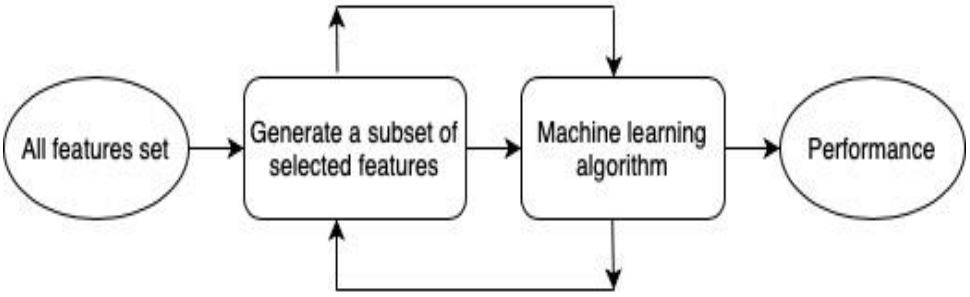

Figure 2.3 Wrapper-based method algorithm [25]

3- Embedded Method

In this approach the feature selection and the learning algorithm are interacted in which both previous methods are combined, so for example in case of a classification algorithm, normally we perform feature selection and then train our model but in this case they both are being done at the same time. Using this method has many advantages as they exhibit more accuracy, faster and more robust to develop overfitting while model training [27].

Feature Importance A technique that rates all features in a dataset, a feature that has a high rate means it has high contribution to the output of the intended task, by this method we can discard irrelevant features without exerting much computational power due to high dimensionality. It is implemented using a trained classifier like decision tree that ranks the features [28].

2.4.2 Principal Component Analysis

As mentioned in previous discussions the importance of dimensionality reduction for achieving good classification accuracy, in this section will be explained the importance of the principal component analysis (PCA). It's one of the most common linear dimensionality reduction techniques available in recent years that proved its efficacy to improve the classification process and shorten the computational time and the reason is that it minimizes the features or dimensions of the data while capturing most of the relevant information contained within, the way it works by capturing components from the original data and the result is in the form of principal components in such a way that the first component contains most of the variance in the data and the next component contains the rest while keeping low correlation between both and so on [29]. An example of its application, an audio classification algorithm that has high dimensional audio data, after we extract the features from the data we can't feed it to the classification algorithm because we get very low accuracy, so now we need to apply PCA that works by first normalising the data as a preliminary step that rescale the data to be normally distributed, and the reason it is so important because for example if we have an audio data in which one of its features let's say length of the song which is in seconds and which is less than a second feature which is the sampling rate in hertz, what will happen is that PCA will count on the component that gives the higher variance as always the first principal component contains the highest variance, so scaling has an important role in order to make PCA to not account for such inaccuracies [30], after performing data normalisation, a covariance matrix should be computed on these normalised data, then perform eigen decomposition by computing the eigenvectors and their eigenvalues of the covariance matrix, the values of the resulted eigenvalues are a direct indication of how good or more correlated the new subspace will be, the target is to get a subspace of highly correlated data to the original data, since eigenvalues resulted from the eigenvectors gives us information about the length and the magnitude of these eigenvectors, if we see that the eigenvalues have close magnitude to each other, this is a good indication of a good subspace, also eigenvectors that contain high magnitude eigenvalues are more important and contain more information, also those eigenvectors with low magnitude eigenvalues or equals to zero don't contribute much to the data and good to remove them [31].

3 BIOLOGICAL & ARTIFICIAL AUDIO FEATURES

This chapter illustrates the perception of sound in the auditory system and its components from a psychoacoustic point of view, this in turn helps to understand the process of music perception in the human ear that gives an understanding how we can apply these mechanisms on an artificial signal processing models that serves as a biomimicry model to give the optimum digital signal processing results. Systems that imitate biological models shows the most accurate results and to develop algorithms that have the ability to process music signals as humans we must understand the hearing mechanisms and how it can be applied in artificial models, To keep it simple, The main concept is to understand how can we make the machine hear and how do we hear and shorten the gap between them.

3.1 How do we perceive music?

Firstly, music is defined as a set of combination of tones in a harmonic way to deliver a feeling or an emotion to its listener. It has its own features which are perceived by our auditory system. The act of hearing is a sensory and perceptual event that is induced by the propagation of a wave from the sound source, this propagation is composed of multiple waves that has frequency, amplitude and phase, since all frequencies carries information, these information are the features that are decoded by the auditory system and transferred to certain parts in the brain where it is perceived. The human ear has the ability to localise sound at discrete time by using timing analysis, spectral information of the signal, correlation which shows the periodicity of the signal and pattern matching ("Sound localisation", n.d), It also has a non-linear response to sounds of different intensity levels within a range of 20 Hz to 20 Khz.

Sound as a wave can be described in the time and frequency domains, the time domain sound is in the form of multiple consecutive oscillations that are changing over time while in the frequency domain it can be described as a spectrum which has an amount of vibration at each individual frequency, if we have the spectrum information of a sound in the frequency domain, we can calculate the time domain information and vice versa if we have the time domain information, we can calculate the frequency domain information [32]. There is a concept in psychoacoustics known as critical band and auditory filters which was introduced by Harvey fletcher in 1933 and refined in 1940, it describes the frequency bandwidth of the auditory filter of the cochlea "The organ in the inner ear that perceives sound ("Critical band", n.d), however the idea of these filters are theoretical or mathematical that describes the behaviour of the frequencies in the

cochlea over a bank of auditory filters that overlap naturally. In fig 3.1, shows the concept of critical bandwidth and auditory filters.

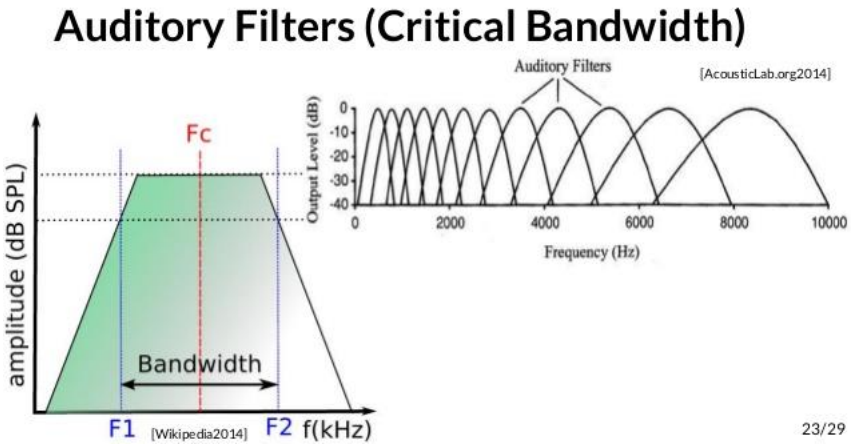

Fig 3.1. Critical bandwidth within auditory filters

The concept of consonance and dissonance happens when two frequencies are so closely spaced that they lies in the same auditory filter, for example A note that has a frequency of 440 Hz and A# (A sharp note) that has 466.164 Hz ("A# Musical note", n.d), a critical band at 400 Hz is going to be around 100 Hz wide which means that these two frequencies are so closely separated that they will fall within one filter and are going to interfere with each other. In contrast two other frequencies for example A of 220 Hz and E of 329.628 Hz which appears that they are far enough apart to activate two separate auditory filters.

3.2 Audio feature extraction

The process of extracting information from sound using its features is called feature engineering, these features contains information that we need to extract in order to perform a wide range of applications in the areas of music recommendation systems, speech processing, bioacoustics, etc , for example building a music recommendation system to recommend music to users depending on their musical style or preferences requires to perform genre classification, doing such a task requires to perform extraction of specific audio features that are able to distinguish these genres from each other, these features can be extracted in two domain, time domain or frequency domain, so we need first to be able to decode the information in the audio signals and the better the accuracy of the extraction process, the higher the accuracy of the result, in the next parts of this chapter we will go through these features and which acts as better

representatives for the music signals [33]. There are two kinds of features to be extracted, low level features which are not directly interpreted by humans which we refer to as tempo, amplitude and pitch while high level features are considered to be highly interpretable by humans and they are derived from low-level features, these extracted features are represented in an interpretable domain either time or frequency then transferring them to a high level features like a classification system that uses the low dimensional or low level features to classify music genres [34].

3.2.1 Audio signals representation

Audio signal is a representation of sound that changes over time, the range of audible frequencies to humans is known to be from 20 Hz to 20 Khz ("Audio signal", n.d), it is important to know the type of the signal we are dealing with, a signal can be characterized by being periodic or aperiodic, continuous or discrete, stationary or non-stationary, however audio signals in general are considered to be non-stationary because to define a signal as stationary means that its frequency content is constant over time like that of a sine wave which has a specific frequency value that doesn't change over time, however this is not the case for audio signals as they are a superposition of multiple sine waves and each has its own frequency content, the stationarity of the signal can only be determined from the frequency domain and not related to the time domain, for example a sine wave of 10 Hz as shown in fig.3.2 and its magnitude spectrum as shown in fig.3.3, the amplitude of the sine wave changes over time while it has the same frequency content over the same period of time [35].

The sine wave can be represented by the following equation,

$$x(t) = A\sin(\omega t + \varphi) \qquad (3.1)$$

Where A - the amplitude.

ω - the angular frequency.

φ – the phase.

The sine wave below in figure 3.1 has an important property as it reserves the same wave shape when intervened with another sine wave of the same frequency and different magnitude and phase like waves of an audio signal.

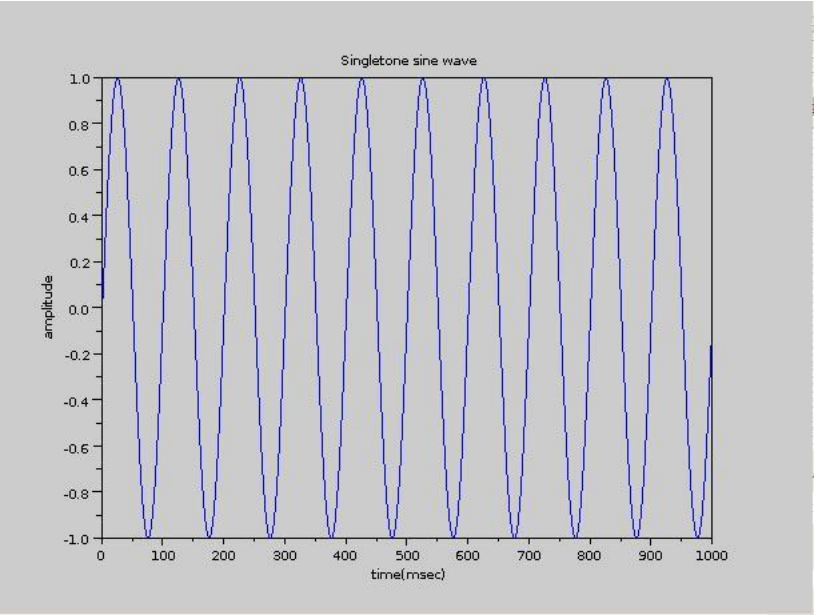

Figure 3.2 sine wave of 10 Hz sampled with a sampling frequency of 1000 Hz

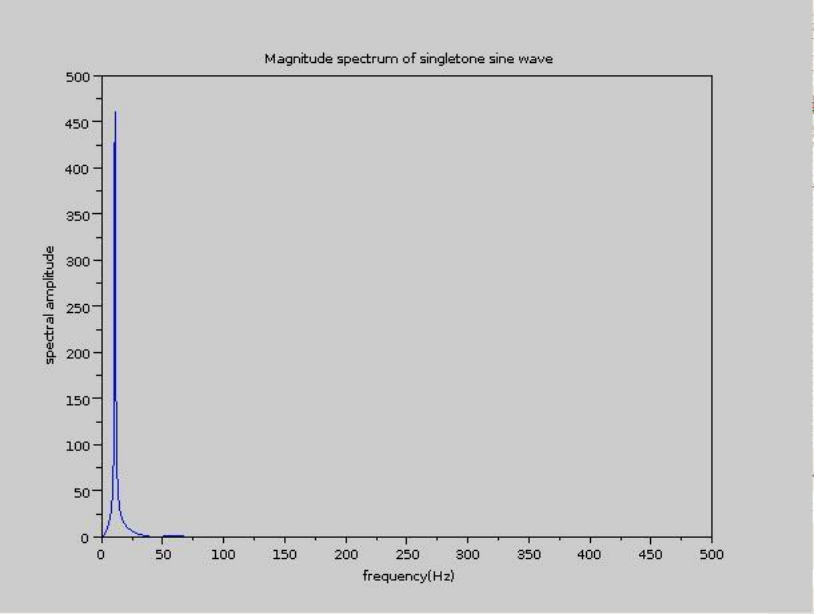

Fig 3.3 Magnitude spectrum of sine wave in figure 1

3.2.2 Periodicity of audio signal

A signal is said to be periodic if it repeats with the same pattern over a regular period of time, the periodicity of a signal can be described in below equation (3.2)

$$f(x) = f(x + p) \qquad (3.2)$$

Where p – periodicity of signal x

The fundamental harmonics of this signal can be calculated as follows

$$f_o=\frac{1}{p} \quad (3.3)$$

The fundamental frequency is known to be the lowest frequency of periodic waveform denoted as f_o and from which the harmonic spectrum is composed and known as "Harmonics ". It is the multiples of the fundamental frequency, for example if the fundamental is 220 Hz, then its harmonic spectrum or as called harmonics will be 440 Hz, 660 Hz, and so on. According to Fourier analysis is that for a function to have harmonic spectrum it must be periodic and for this reason we need to check the periodicity of the signal which is called " periodicity analysis " and there is two known methods to perform using autocorrelation analysis and Fourier [36]. For example, in music information retrieval tasks like beat and tempo detection, Humans have a natural cognitive ability to detect beats and tempo of music without difficulty, however the challenge here is to automate this biological process into a programmable process to a system that has a large scale datasets of various music styles, the extraction process becomes difficult due to the fact that beat and tempo are not explicitly defined because of the complex structure of the rhythm, from here comes the necessity of periodicity analysis and two proposed methods will be discussed.

Fourier tempogram It is a two-dimensional representation of an audio signal displaying the variation of pulse strength over time which shows the intensity of the estimated periodicity given in beats per minute (BPM) over time. The tempogram works by detecting sudden changes in the input audio signal which can be determined from the note onsets, so the more accurate the onset detection techniques the better the results of the tempogram [37], from here comes the need to use a novelty function because of its ability to detect such changes in the properties of the signal such as its spectral content or energy. There is two known types of novelty functions that can be used in this task, energy based and spectral based novelty functions.

Energy-based novelty functions When a note is played on a musical instrument, at this time instance there is an instant increase in the energy of the signal, for example hitting a piano key, or playing a note in violin, this sudden increase in the energy of the signal can be computed by the energy novelty function [38].

To explain this process mathematically, if we have a discrete-time signal x and apply a discrete window function w of bell shaped function where its center is at time zero square

which is moving along the signal x to determine the local sections, then the local energy of signal x with respect to the applied window function w denoted by E_W^X is given by

$$E_w^x(\mathrm{n}) := \sum_{m=-M}^{M} | x(n+m)\, w(m) |^2 = \sum_{m \in Z} | x(m)\, w(m-n)|^2 \qquad (3.4)$$

Where x – Discrete-time signal,

w – Discrete window function,

E_w^x(n) – Energy of the signal x,

m - Non-zero samples of the window function,

n – Number of samples the window function shifts from.

As a nature of human perception of sound is non-linear and to be able to scale it we need to use a logarithmic scale, also we need to put into account that analysis of music or an audio file contains parts of low and high energy, so being able to capture the most of it is an important aspect and by applying logarithm to the values of energy from equation (3.4) then the low energy parts will be clearly audible even if it is overlaid by high energy musical event. The resultant of the energy-based novelty function after applying the logarithm is as below equation (3.5).

$$\Delta_{Energy}^{Log}(\mathrm{n}) = | \log (E_w^x(\mathrm{n+1}) - \log (E_w^x(\mathrm{n})) |_{\geq 0} = | \log \frac{E_w^x(\mathrm{n+1})}{E_w^x(\mathrm{n})} |_{\geq 0} \qquad (3.5)$$

One drawback of this logarithm energy values that as it makes all parts of the signal both high and low energy clearly visible, it will also act as an amplifier to noise if the signal is distorted by noise, so it's important to denoise the signal before computing its energy function, another drawback of this approach that detecting onsets of musical events that have multiple music instruments played at the same time are hard to detect, for example a song that has violin, piano, drums and some other effects like strings, instruments with higher intensities will mask the lower ones, however a good observation is that each of them has different frequency content then comes the need to detect these changes in frequency and the spectral based novelty function that measure these changes will be introduced in the next discussion.

Spectral-based novelty function As known as spectral flux, it is based on transforming the signal into time-frequency domain to be able to detect the frequency changes across the signal, it is known to be sensitive to spectral fluctuations or transients and this is important as it gives it an advantage over the energy based function as it can detect the onsets of musical events much clearer specially when

multiple instruments being played at the same time, or by others means it can overcome the masking effect of louder instruments over the quieter ones [39]. Computing the spectral flux is a representation of the amount of changes in the spectrum of the signal over time, it can captured by measuring the differences between two consecutive STFT frames by computing each frame's power spectrum, the frame is a segment of the signal that we determine, then its equation (3.6) is as follows [40].

$$SF = \sum_{n=1}^{n} (D_t (n) - D_{t-1}(n))^2 \qquad (3.6)$$

Where D_t – Normalised frequency distribution in frame t.

From the above equation (3.6) we see the computation of spectral flux and how it can be measured, it is considered as a frequency domain representation and categorized as low level feature descriptor, it is used frequently in tasks that require speech or onset detection [41], an implementation for an audio signal that represents a 30 seconds of the classical song " The Four Seasons, Op. 8: Concerto No. 2 For Violin In G Minor, Rv 315 "Summer " in MATLAB to show the spectral flux is as below.

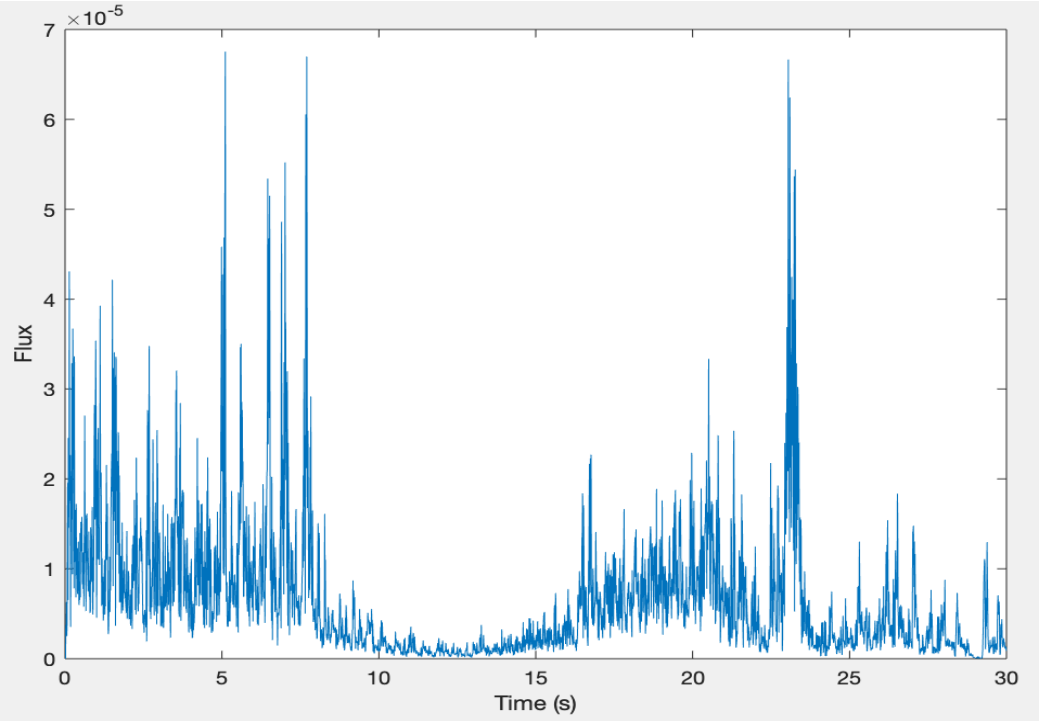

Fig 3.4 Spectral flux of the mentioned song showing the change of its frequency content over time

Autocorrelation method This is a common method known for checking the periodicity of the signal, it is a measure of the similarities between the original audio signal and

shifted versions of it when the signal is delayed over time, it is inspired by the natural periodicity analysis of the human ear [42]. Many tasks in music information retrieval uses autocorrelation to detect the periodicity of audio signals in music specially if it is distorted by noise, also to identify the missing fundamental frequency of sound which is known as " Missing fundamental" according to ("Autocorrelation", n.d), below will be shown an example in Matlab on finding periodicity of a signal using autocorrelation.

3.3 Time-domain features

Audio signals has time-domain representations and frequency domain representations, however most tasks needs frequency domain analysis to reveal the signal's information for detailed analysis which can't be seen in the time domain, the accuracy of the tasks depend mostly on these features, in the time domain these features are directly extracted from the audio signal. Most audio features either show time or frequency information but not both, in the next section wavelets will be explained how it can show information in both domains which can be a better alternative than showing only time domain information.

3.3.1 Zero-Crossing Rate

It is a measure of how frequent the sign of the signal changes or the frequency in which the signal changes its sign crossing the zero point from the positive to the negative or vice versa. It is used in many tasks in speech processing and music information retrieval like musical instrument recognition and music genre classification due to its simplicity, can also measure the noise level in audio signals, when zero crossing rate values are high, this is an indication of the existence of noise, another application is that it can detect the presence of speech in audio signal or not, it is computed using below equation (3.7) according to [43], let's assume that $x_i(n)=0,1,\ldots,N-1$, to be the samples of the i^{th} frame.

$$Z(i) = \frac{1}{2N} \sum_{n=0}^{N-1} \text{sgn } x_i(n) - \text{sgn } x_i(n-1) \qquad (3.7)$$

Where sgn $x_i(n)$ = 1, $x_i(n) \geq 0$ or -1, $x_i(n)<0$.

The results from the above equation (3.7) can show the high frequency content of a signal if the sign frequently changes, also can do periodicity check for the signal, because as periodic signals repeat the same pattern over time, then variations in the values of zero crossing rate can be an indication that the signal is aperiodic.

3.3.2 Energy

The energy of the signal indicates the loudness of the sound as it shows the amplitude of the sound signal and can be computed directly from the time domain representation [44],

$$E = \sum_{n=0}^{n} x[n]^2 \qquad (3.8)$$

Where x[n] is the input signal

N is the length of the signal

3.3.3 Root mean square energy

As known as "RMS "is calculated directly from the energy of the signal as following equation (3.9)

$$RMS = \sqrt{\frac{1}{N} \sum_{n} | x(n) |2} \qquad (3.9)$$

Where x[n] is the input signal

N is the length of the signal

RMS is a measure for the strength of the signal's amplitude, which is an indication of its loudness, so as the signal's amplitude is constantly varying the RMS can calculate its average [44].

3.4 Frequency-domain features

Most of tasks in audio signal analysis requires transformation of given audio signals into the frequency domain, the reason is simply that time domain don't hold that much information of the given signals to perform the needed tasks, time domain represents the changes happens to an audio signal over time while frequency domain represents the changes of the audio signal as a function of frequency and since sounds are an audible properties of humans that have band limit in which it is perceived, so in order to be able to make machines hear sounds the way we do, we have to decode the content of the audio signal the way human ears do, then transforming signals to its corresponding frequency values is the best way we can make algorithms that analyse sounds as human auditory system does. While most audio signal analysis techniques focus on either showing the information within the signal in time or frequency domains but not both, then it becomes an advantage for discussing ways to represent signals in

both domains simultaneously, then comes the role of wavelets which will be discussed later in this chapter.

3.4.1 Fourier transform

Fourier transform is made on the purpose to show the frequency information of a signal by displaying its frequency values and where it lies over a range of frequencies, any change happens in one of the domains consequently happens in the other domain (time or frequency), however it is noticed that the frequency information of a given signal don't show the time it happened then came the need to find a way to show the time information, Dennis Gabor came with the idea that instead of considering the whole signal, just consider a segment of it by fixing a window function which is a non-zero function inside a specific interval and zero outside of it, and it is multiplied by each segment of the signal as it moves over the entire signal [45], by this way we can get the frequency content for each segment, below will be shown an example of applying " hamming " window to a 30 seconds audio signal of a pop song done in Matlab.

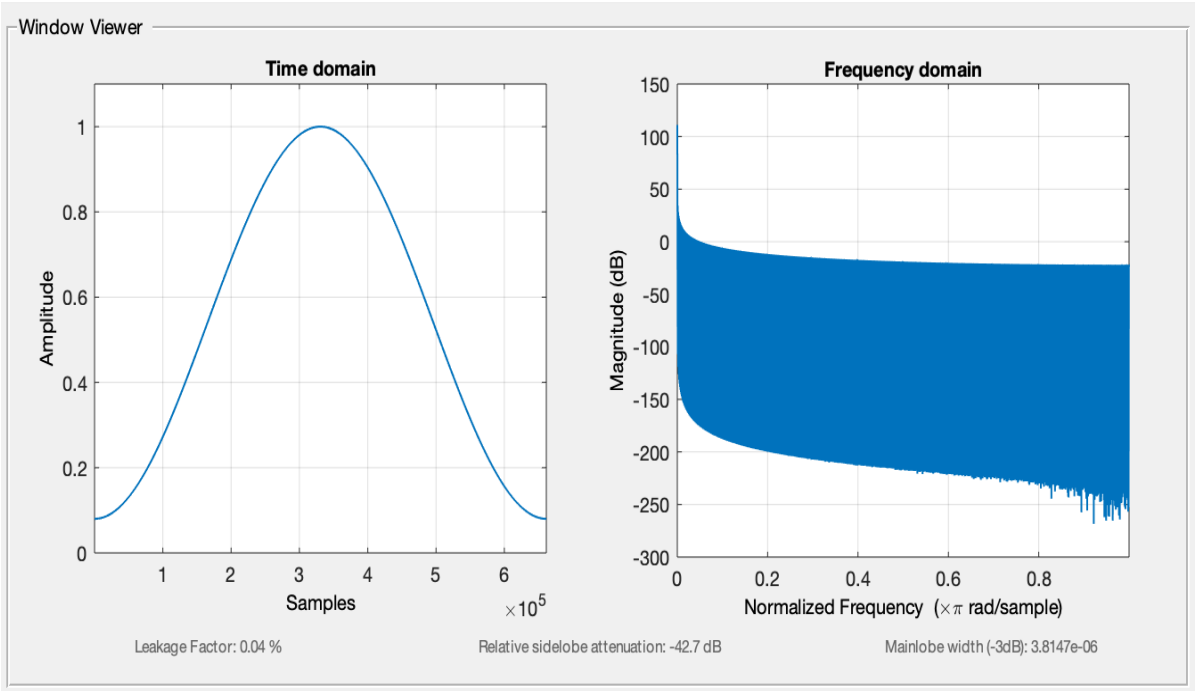

Fig 3.5 shows a hamming window applied to a 30 seconds audio signal and displays it in the time and frequency domains

However, it is important to note that as application of a window function gives the advantage of displaying the frequency content of a segment of a given signal and this raises the question of what length this applied segmentation should be? A good answer for this question is that a proper window size should contain as much as possible

information of the frequency content and time information for the given signal, while this seems difficult according to heisenbug uncertainty principle which states that it's impossible to have a window function that achieves good frequency resolution without affecting time localisation, because selecting longer window sizes will give better frequency resolution but will affect time localisation and to achieve good time resolution will affect the frequency, also the research showed that no amount of overlap can improve the results, so the most proper solution is to compromise and have an optimal selection of time-frequency resolution without sacrificing much of the information in the signal [46]. Another point to account for is the stationarity of the signal, most audio signals are in continuous change, for example a song being played, one note of it is considered stationary but as more notes being played along with different instruments, this leads to a change in the signal characteristics such as timbre and frequency, and eventually becomes non stationary, so a solution is to apply the window function of size equal to the length of the stationary part, a wider window size will get a more confused frequency content, that will make it difficult to distinguish single notes, also FFT theorem assumes that the frequency information across the signal is not changing or repeating over a periodic pattern, otherwise the results of the analysis would be inaccurate, so to avoid this, small window sizes applied to short segments of the audio signal would suffice to get accurate frequency representation [47].

Spectral leakage Another drawback of Fourier transform, so as FT supposes that the signal behave in a periodic manner, while most audio signal don't stay too long periodic and eventually changes, so as discussed in previous section is to segment the signal to frames and analyse each frame separately as the signal tend to be more stationary within this frame, then apply the window function, the result after windowing will the frequency representation of that audio segment but with some leaked spectral information from the edges of the applied window function, a proposed solution to this problem that helped to mitigate the effects is to apply a window functions of " Hann" or " Hanning", it showed that it smoothens these discontinuities at the edge of the window making it behave in a more periodic manner as discussed in section 2.2 in [41].

As shown below in figure 3.6, the spectral leakage because of windowing, also it is worth to mention that every task requires reconsideration of which window type , length and overlap between consecutive segments, a task that requires detecting speech in a noisy environment will have different window parameters than a task for classifying music genres [48].

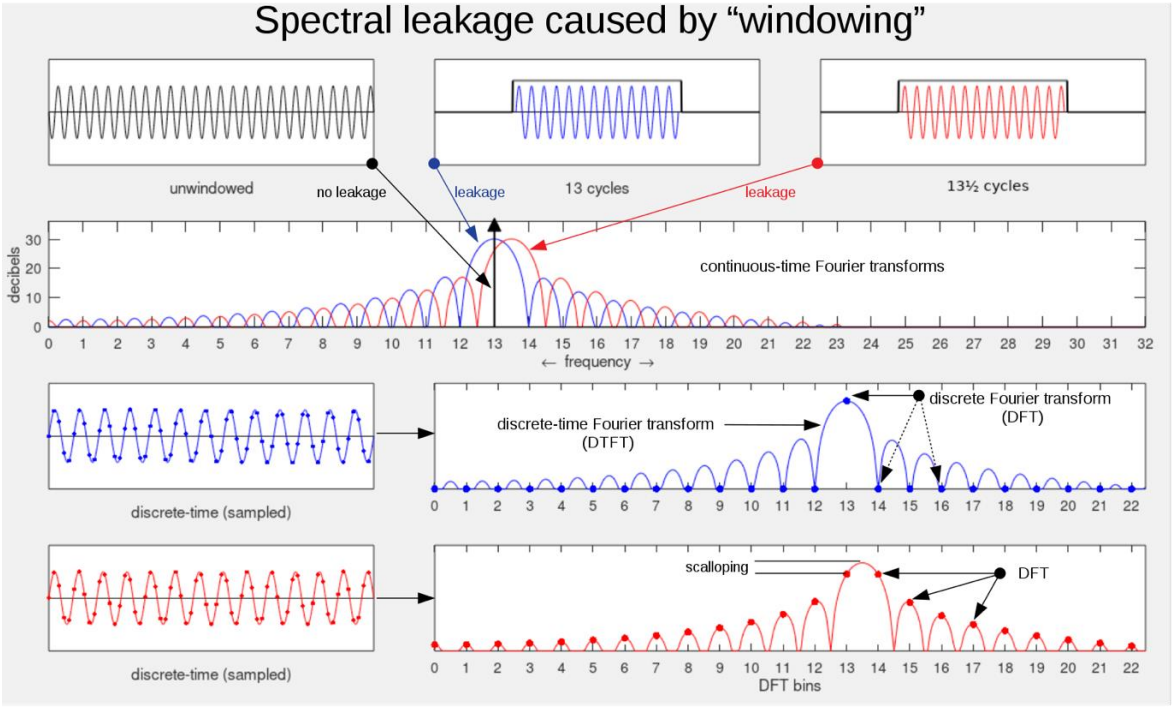

Figure 3.6 shows spectral leakage happened after applying a window function (“Spectral leakage”, n.d)

3.4.2 Wavelet Analysis for Audio Signals

The main purpose of audio analysis is to be able to get an accurate representation of an audio signal without losing much of its information in the analysis process, so when we have a signal to be analysed, we need to find the optimum way to retain as much information for this signal in both frequency and time domains, Fourier transform don’t offer this advantage, FT decomposes the signal as sum of cosine and sine waves, it gives us a good frequency resolution with sacrificing its time localisation, we end up having the frequency information but the time of its occurrence remains hidden in the frequency spectrum. Another drawback of Fourier transform is that it gives good results for stationary signals but when it comes to non-stationary signals that have their frequency content changes over time, it doesn’t give the expected results, when short-time Fourier transform came to light and promised to offer a solution for good time localisation, it was found that it didn’t give the ideal expected performance and the reason is that the window size used in the analysis remains the same all over the signal in time and frequency [49]. Here comes the role of wavelets that is able to overcome these disadvantages and gives good time-frequency resolution even for non-stationary audio signals like that of music, this will be explained in the following discussion and how we can use wavelets in music analysis and pointing its properties. However, there is many wavelets that had been developed for different purposes, so it’s important to

keep the focus on specific kind that serves us to get accurate results for our task, as we are analysing audio signals then we will study the properties of wavelets that give the desired result in analysing music [50].

The word wavelet is defined as a little wave that have an oscillatory behaviour that decays rapidly unlike the sinusoids, wavelets have a finite duration, short time Fourier transform are unadjustable and the window size used in the analysis remains the same all over the signal, if the window is so narrow, we will have a very good time resolution but we lost most of the frequency content, and if the window is very wide, then we captured all the frequency content but lost the time localisation.

The Discrete Wavelet Transform Recently, wavelet analysis applications have arose due to the need to find an effective way to extract information from non-stationary signals as music with better time-frequency resolution rather than the conventional short-time Fourier transform and its associated drawbacks that were mentioned in the previous discussion (3.4.2). Also, their ability to be adjusted to parameters that fits the needed task. In wavelets we can control its parameters, it consists of a scaling parameter that determines how wide or narrow the window is and a translation parameter that slides the window along the signal over its duration (time), then to get the frequency information without losing any, make a wider window that covers the whole signals to pick out the lower frequency content as it exists during the whole signal duration, and then use smaller windows to pick out high frequencies, because low frequencies are not localised in time then can be done with a wider window, while high frequencies are localised in time so can be captured with smaller windows, and these are similar properties to the human auditory system that have similar time-frequency resolution characteristics [51]. There are two types of wavelet transformation, the continuous and the discrete, we will go through the discrete here as it has a denoising properties for audio signals.

The DWT can be described by the following equation,

$$W(j,k) = \sum_j \sum_k x(k)\ 2^{-j/2} \psi\ (2^{-j}\ n-k) \qquad (3.10)$$

Where $\psi(t)$ is a the mother wavelet,

j,k are for scale, translation.

The process of computing DWT is similar to discrete multirate filter banks, it works by filtering the signal through low pass and high pass filters, this signal decomposition will result in determining the signal sub-bands, then half of the samples are neglected after sampling as the Nyquist-Shannon sampling theorem states, after that the low sub-band

is iteratively filtered by the same previous decomposing technique to result in much narrower sub-bands, and always the length of the coefficients in each sub-band equals to half of number of the coefficients of the previous stage, this technique has been proven to be effective as it will give us the large coefficients that resembles the part of the signal we need to capture and low coefficients are more likely to contain noise and uncorrelated frequencies of the signal so it has denoising properties [52]. This type of transformation provides a logarithmic decomposition for frequencies which imitates the way humans perceive frequencies and this property make it act as biomimicry model, the closer the machine's perception of music to humans, the more accurate it is, and this because it provide a multiresolution decomposition which was difficult to achieve with Fourier methods, so now for example if we have an audio signal that resembles a song, which is considered as non-stationary signal, it will contain high frequencies and low frequencies, we use wide windows to capture the low ones and compressed windows to capture the high ones [53].

A MATLAB explanation for the above process of DWT for a common pop song "Emotions for Mariah Carey" to illustrate its audio analysis process, the song will be segmented into 30 seconds and will be decomposed according to DWT as below figure.

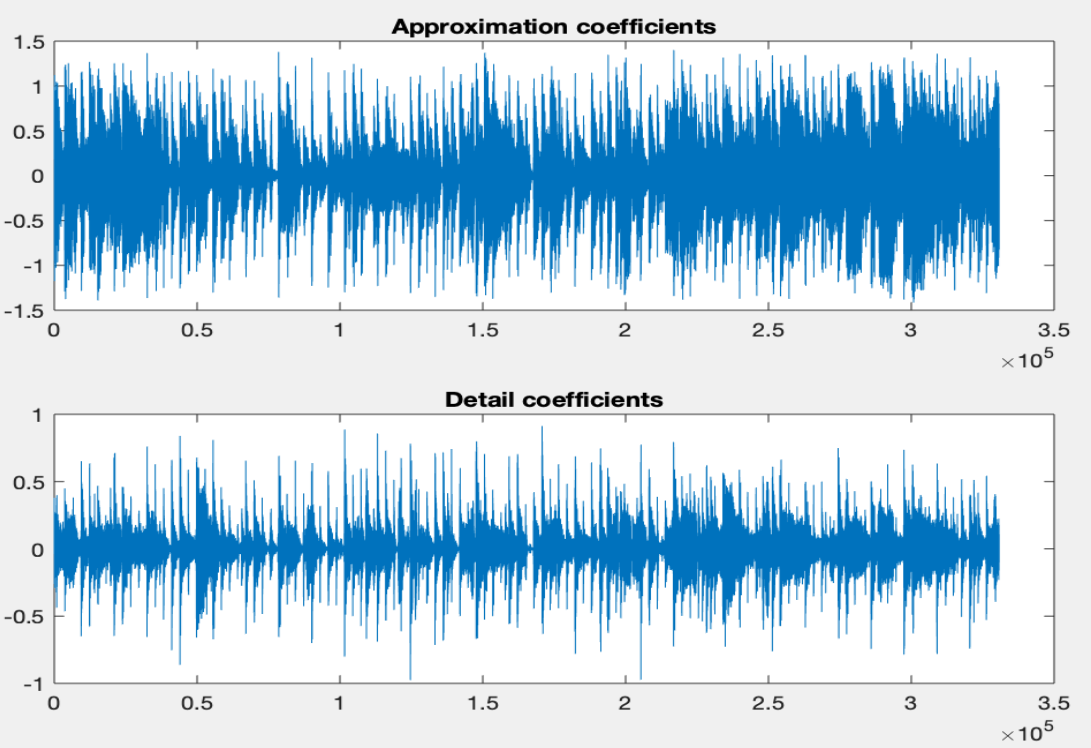

Figure 3.7 The DWT signal decomposition into approximation and detail coefficients

The above figure (3.7) shows the audio signal decomposed simultaneously into detail coefficients as it passed through a high pass filter and approximation coefficients as it passed through low pass filter as shown in figure (3.8).

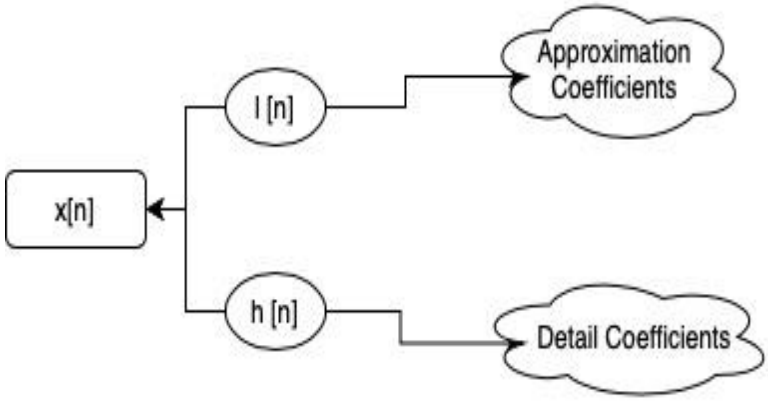

Figure 3.8 showing signal x[n] passing through low-pass filter l[n] and high-pass filter h[n]

The output of the low-pass filter will go through another decomposition through another high and low pass filters with half the cut-off frequency of the previous one according to ("Discrete wavelet transform", n.d), then the output of the second decomposition will be,

$$Y_{low}[n] = \sum_{k=-\infty}^{\infty} x[k]\, l[2n-k] \qquad (3.11)$$

$$Y_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]\, h[2n-k] \qquad (3.12)$$

Next is to decompose the approximation coefficients into two decompositions simultaneously, and this is the second stage of multiresolution analysis that is supposed to be done by the DWT, it follows an iterative filtering process to give narrower sub-bands in which the number of coefficients of each sub-band equals half that of the coefficients of the previous decomposition. The denoising and compressing properties of the DWT lies in giving the possibility to capture the part of the signal you are interested in in the form of large magnitude DWT coefficients and the noise are comes out in the form of smaller DWT coefficients and a denoising example will be explained below on how DWT denoises a signal [52]. So the wavelet analysis is considered to have an efficient zooming effect that contribute to capture even the small details of the signal represented as high frequencies at specific time and can disregard other non-interesting parts of the signal as discussed in [50].

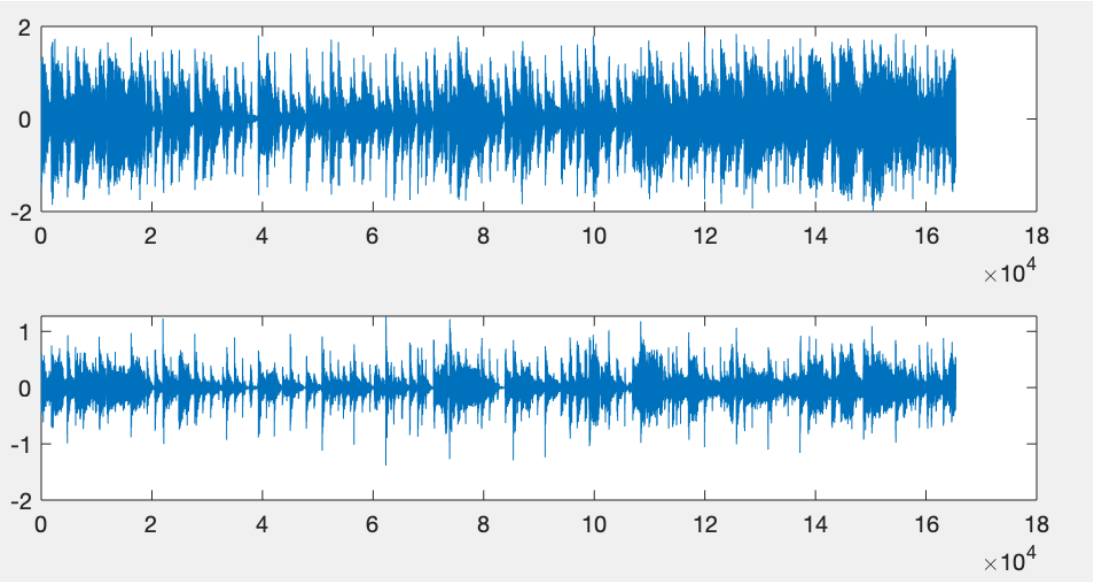

Figure 3.9 shows the decomposition of the approximation coefficients as it passed through high-pass and low-pass filters

Denoising of an audio signal Noise is regarded as random frequencies that exists in the signal and can disturb the analysis process and interfere with the important frequencies that needs to be captured from the signal, then signal denoising is a common approach that is been applied to signals that exhibit noisy behaviour and wavelets have denoising properties, below denoising example was done using MATLAB with the audio file of 30 seconds for the same song used in fig 3.7,

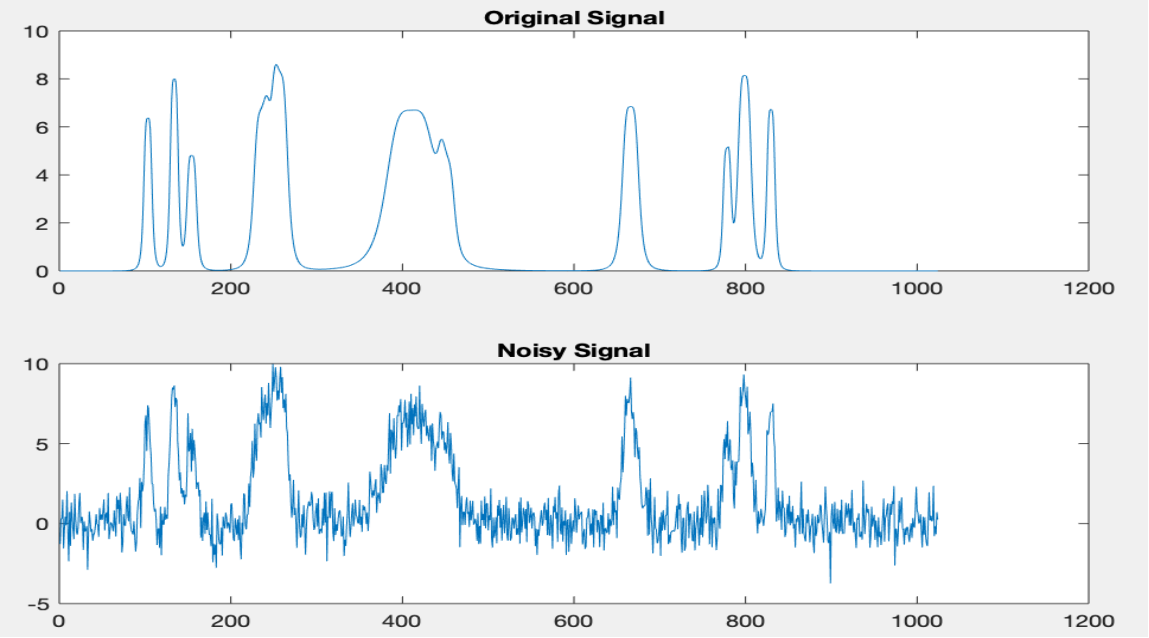

Figure 3.10, original signal of a pop song distorted with additive Gaussian white noise

Figure 3.10 is an example explaining the denoising process using wavelets, an audio signal has been distorted by additive Gaussian white noise, then will use wavelets to denoise it to test its denoising properties, it is done by maximizing or minimizing specific frequencies, as noise has different frequencies than the original frequency range of the signal, then wavelets can distinguish them by what is called " Wavelet thresholding ", it gives us two output coefficients, one is of large magnitude which carry all the main and essential features of the original signal and the second is of small magnitude coefficients which are regarded as noise and can be removed without adverse effects on the original signal [54]. This procedure is shown in figure 3.11 for denoising the audio signal,

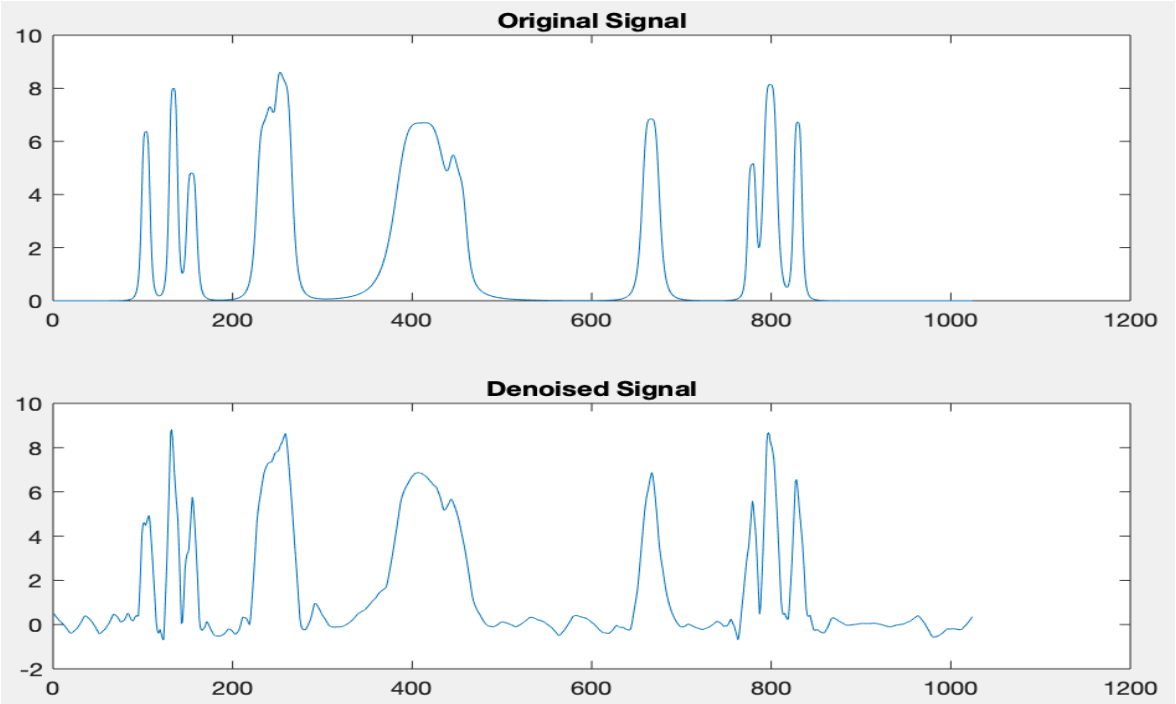

Figure 3.11 Signal denoising using wavelets

In figure 3.11 wavelets used to denoise the signal and as shown the denoised signal is exhibiting similar behaviour as the original signal.

3.4.3 Mel Frequency Cepstral Coefficients

As known as " Mfcc", It is considered as a way for sound visualization, inspired by the way humans perceive sounds, as humans understand sounds non-linearly, then it is required to scale these sound on a logarithmic scale so that machines can understand the way human ear do [55], for this reason Mel scale is used because of its ability to scale sounds logarithmically which is close to perceptual auditory system. It was used first for speech recognition systems and lately it gained popularity in music classification tasks and since then it is been used in a wide range of music information retrieval applications [56], MFCC's computation procedure will explained in details in the following discussion along with its advantages and drawbacks.

Feature extraction of audio signals is the preliminary step for any music information retrieval task, these extracted features are the distinctive description for these signals, it's a decoding process for the information within these signals so we can decompose them to perform specific tasks like music genre classification, music recommendation system or speech recognition. MFCC are set of features that transform the frequency of the signals into a human perceptual like scale, the process of extracting these coefficients is like a transformative approach for the linearly spaced frequencies of the signal which is in Hz into the Mel scale which is non-linear "logarithmic " [57].

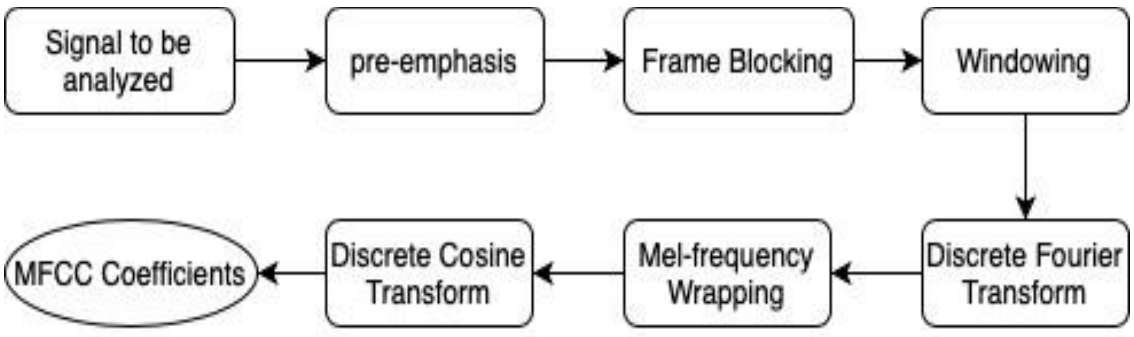

Figure 3.12 shows the process for computing the Mel-frequency cepstral coefficients

Pre-emphasis The importance of this step lies in amplifying high frequencies and boosting the overall spectrum of the signal, sometimes audio signals experience a kind of decay at high frequency regions that were supressed, that needs to be accounted for to get accurate analysis results [58],

It is performed by applying a high pass filter to the input signal as follows,

$$Y(t) = x(t) - \alpha x(t-1) \qquad (3.13)$$

Where Y(t) is the output of the signal.

α is the filter coefficient (its value ranges from 0.90 to 1).

X(t) is the input signal.

Frame blocking This is the second step to be done next in the computation of MFCC, an audio signal of period t is changing over time and is considered non-stationary, so doesn't make sense to apply Fourier analysis as it is applied only when assuming the signal is in a stationary condition even if not completely, it can be considered as quasi-stationary, so to tackle the non-stationarity of audio signals is to divide it into short segments of equal length and analyse them separately, the common segment size used in audio signal processing is 20-40 ms of 50% overlapping, this is based on the assumption that signals tends to have stationary behaviour with short period of time, large segments will change this to be non-stationary also too short won't give us enough samples to have a spectral representation for the audio signal [59].

Windowing Each audio segment should be multiplied by the window function, so if hamming window is applied, it fulfils the Fourier analysis assumption for staionarity of the signal and reduces the spectral leakage [58], it can be represented by the following equation,

$$w[n] = 0.54 - 0.46 \cos \frac{2\pi n}{N-1} \qquad (3.14)$$

Where $0 \leq n \leq N-1$

Discrete Fourier Transform This step involves applying the discrete Fourier transform to each audio segment with the hamming window, so we will get an output of the magnitude spectrum of frequencies of each frame, the DFT can be computed by the following equation,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}} \qquad (3.15)$$

Where $0 \leq k \leq N-1$

N is the number of points

The reason hamming window is applied while performing the DFT for the audio segments is that it makes the peak of the frequency response values is more sharper and distinct which makes it easier to distinguish from other responses, while without using the signal experiences a kind of discontinuity at the edges of the frame which makes the peak of the frequency response appear as blurred [60].

Mel-frequency wrapping The next step after getting the frequency response values, is to visualize them on a scale to be understood, as mentioned before that the way we humans perceive sounds is logarithmically, so in order to have accurate results in our audio analysis tasks, we need to give machines the ability to perceive sounds the way we do, from this point, the Mel-scale and filter banks comes to have an important role that will be explained in the following discussion.

Mel scale was originally developed as a way to visualize audio in a similar manner that mimics human auditory system, after some experiments to scale human perception of audio, it was found that below 1000 Hz can be perceived linearly and the scale's spacing is in linear form, while above 1000 Hz it is converted to a logarithmic spacing, the formula below is used to convert Hz to mels and vice versa is,

$$F_{mel} = 2595 \log_{10} (1+\frac{f}{700}) \qquad (3.16)$$

Where f is the frequency in Hertz

F_{mel} is the frequency in mels

The power spectrum for each frame that was computed from the step will be multiplied by a set of filter banks (20-40) filters, the standard is 26 [61], in a triangular shape in which these filter banks are placed in an equally spaced way on the Mel scale by using the above equation (3.16) to place them on the scale and this way converts the audio in Hz to mels which mimics the human ear, the spacing of filter banks starts narrower and gets wider as frequencies increase and then it gets us an estimate of the amount of energy that occurs at these different frequency regions on the scale [61], the shape of the filter bank used of course has a direct impact on the result of the extracted frequencies, in which each filter bank gives a different spectral estimation that is a direct reflection for the properties of the applied filter. The triangular shape is the most common used one, hanning filter shape was also used before, some researches showed gammatone filters can provide better performance in case of noisy signals [62], this will be experimented in the following discussions.

The output of the mel spectrum can be represented by the following equation (3.17) as mentioned in [63],

$$S(m) = \sum_{k=0}^{N-1} [\, | X(k) |^2 H_m(k) \,] \qquad (3.17)$$

Where $0 \leq m \leq M - 1$

M is the number of the triangular mel filters

$H_m(k)$ is the weight applied to the k^{th} energy spectrum bin that contributes to the m^{th} output band, the below figure (3.13) shows how the filter banks are placed on the mel scale.

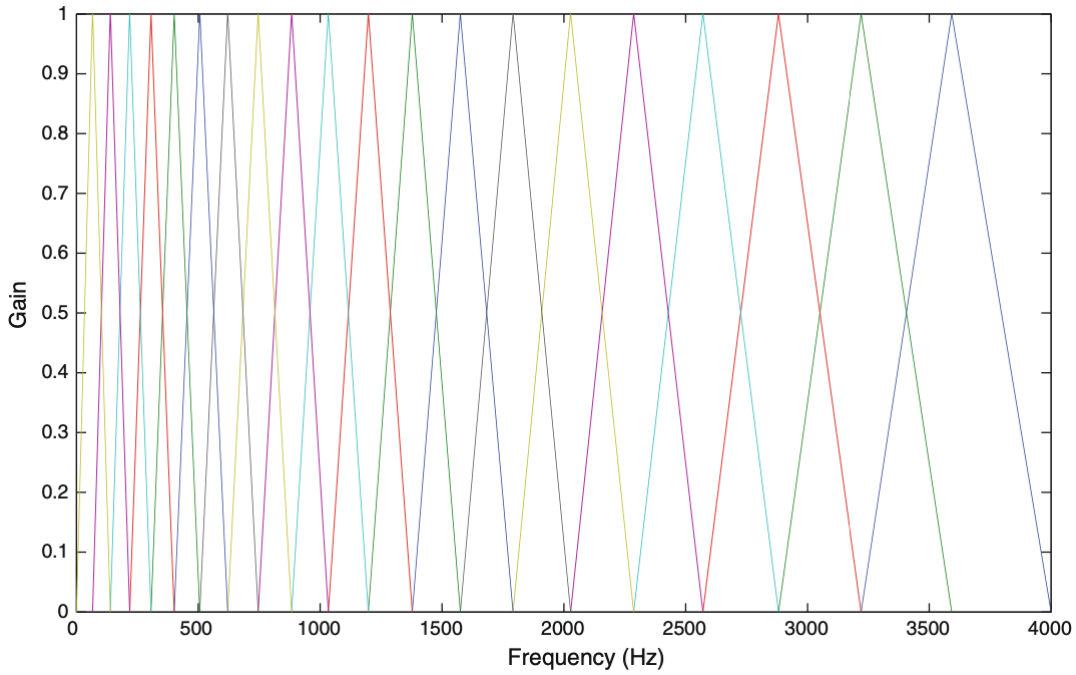

Figure 3.13 shows triangular filter banks placed on Mel scale as explained in [63]

The resultant of the above step will get us the energy spectrum of the mel frequencies resulted from applying the triangular filter banks, then we have to take the log for each of these energies and this is inspired by human's ear sensitivity for hearing, that has a non-linear behaviour, to mimic this behaviour we apply the log, it is also showed that this step makes it more robust to very quiet and very loud sounds, and it significantly affect the accuracy of speech recognition [64].

Discrete Cosine Transform As known as " DCT"**,** this is the last step required to get the cepstral coefficients, from last step we got the log of the energies of the filter banks which are 26 log energies from the 26 applied filters, these energies are of low order coefficients and high order coefficients, most of the important information we need are

encoded within the lower order coefficients which contains the vocal tract information and most tasks like speech recognition or music information retrieval tasks as genre classification requires only these low order coefficients, so we pick only the first 13 coefficients and drop the rest "high order coefficients". As all of the 26 coefficients were found to be highly correlated because of the overlapping between the applied filter banks, the more overlapped the filters, the more the filter's energies are correlated, and vice versa, this resulted in some issues while using it to perform tasks, so the advantage of applying DCT lies in its ability to decorrelate these coefficients and separate them so we can keep only the first 13 coefficients we need [64]. The final Mel frequency cepstral coefficients are computed from the following equation,

$$C(n) = \sum_{m=0}^{M-1} log_{10}\,(s(m))\cos\left(\frac{\pi n(m-0.5)}{M}\right) \qquad (3.18)$$

Where n= 0,1,2,….., c-1

C(n) The cepstral Coefficients.

An example for extracting the Mel-frequency cepstral coefficients from a rock song "The stone roses-Elephant stone), implemented in MATLAB with 40ms window length and 50% overlap.

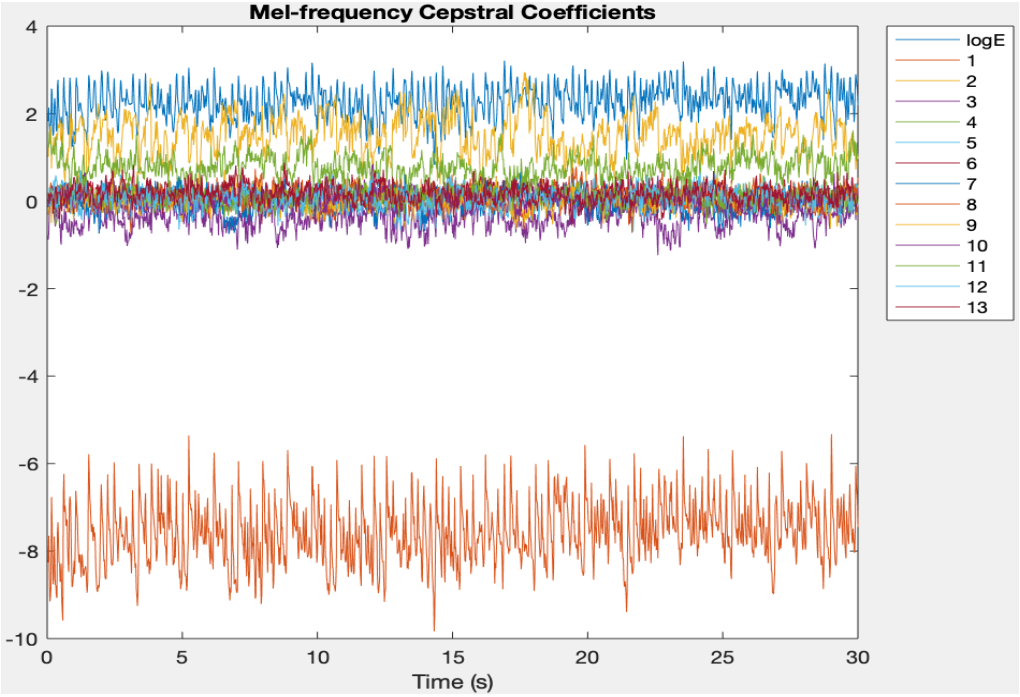

Figure 3.14 shows 13 MFCC extracted from a rock song

3.4.4 Gammatone Cepstral Coefficients

Feature extraction of audio signals is a pre-processing step before performing any task on audio data, it is the way that discriminates between each file of these data. The most conventional and common way for feature extraction is the MFCC that was explained in the previous section, in this section a different approach will be discussed for extracting the information within the audio signals. MFCC is considered to have perceptual properties like that of human ear and that was explained previously, however there are other auditory filters that have another way to mimic the same behaviour and may give better performance with analysis of audio signals or some type of them, for example, MFCC maybe good with audio that contains no vocals, but is not performing good with vocal audio or noisy signals. It is important to study other auditory filters that have good potential to scale the way humans listen to music and have similar properties like masking. Gammatone filter banks shows that it can mimic the impulse response of the auditory nerves of the human hearing according to study [65]. The impulse response of the gammatone can described by below equation,

$$g(t) = at^{n-1} \cos(2\pi ft+\varphi)\, e^{-2\pi bft} \qquad (3.19)$$

Where a = peak value

t^{n-1}= Time onset

Exponential for calculating decay and bandwidth

f = Center frequency of the filter

φ= The phase

The computation of GTCC is resembles that of MFCC but the filter banks are instead replaced with gammatone filters, the reason is that these filters mimics the magnitude characteristics of the auditory filters in the human ear [66], its impulse response can be computed from the above equation (3.19). The human ear behaves in a non-linear way as discussed in the beginning of chapter 3, so filter banks that exhibit non-linear behaviour can somehow and have similar frequency characteristics can be used to be as an artificial human ear and gammatone filters have these properties.

The similarities between the gammatone filters and human ear in the analysis of audio comes from that the bandwidth of the gammatone filters are estimated in relevance with the critical band of the human ear that fall within the center frequency of these filters, since "Harvey fletcher" in 1933 introduced the concept of critical band, he made

an assumption that the auditory filters in the cochlear part of the human ear is made of rectangular bandwidth filters which he found that it can very closely approximate the exact human hearing capabilities, he then concluded that the auditory filter's bandwidth is the result of equivalent rectangular bandwidth " ERB " at the center frequency f_c [67], using below equation,

$$ERB = 24.7 + 0.108\, f_c \qquad (3.20)$$

Where f_c = Center frequency

Equation (3.20) resulted in the highest values of quality factor, as abbreviated Q factor, which is used for evaluating the efficiency of filters in the selection of frequencies among a specific range, a high Q factor indicates that the filter is highly selective [68]. The reason for using the ERB scale is its logarithmic response that mimics the human ear, below is a figure to show the way the gammatone filters are placed on the ERB scale, taken from [69].

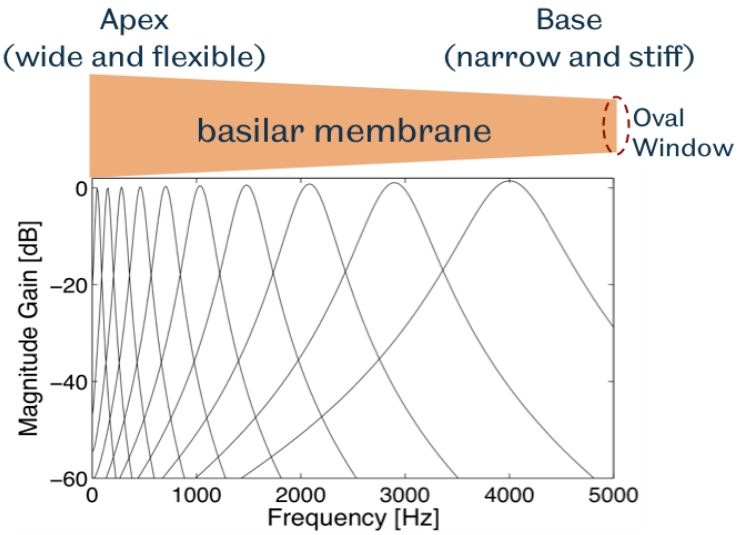

Figure 3.14 shows the frequency response of the gammatone filter banks in a way that their center frequencies is equally spaced between 50 Hz and 4 Khz

3.4.5 Bark frequency cepstral coefficients

A different approach for emulating the human's hearing capabilities. Bark scale is a psychoacoustic measure that was introduced by Eberhard Zwicker in 1961, the way it works is that it mimics the frequency response of the human hearing which lies within 24 critical bands by using the frequency warping scale known as bark scale. The most common scale in speech and audio processing is the Mel scale which can extract mel frequency cepstral coefficients from an audio signal, its computation process was

explained previously in section 3.4.3, however its performance degrades with existing noise in the environment. The auditory system has a natural masking property, bark scale approximates the perceived spectrum of hearing into 24 critical bands and this makes it a potential closer artificial representor for the human hearing. The bark frequency cepstral coefficients computation is much similar to that of MFCC with just changing the mel scale filter to that of bark scale filter, then the result is the resulting cepstral coefficients extracted from the original signal. The bark frequency can be computed using below equation [70],

$$B(f) = 13 \arctan(0.00076f) + 3.5 \arctan\left(\frac{f}{7500}\right)^2 \quad (3.21)$$

Where f = Frequency in Hz

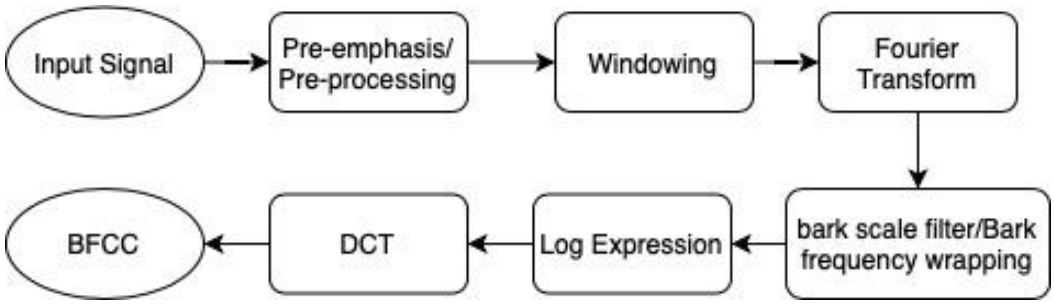

Figure 3.15 shows the computation process of BFCC [70]

4 Features evaluation and testing

In the previous chapter, feature extractors were discussed and their potential to improve the feature extraction process of information from audio signals, which are highly dimensional and performing feature extraction significantly lowers these dimensions while keeping the most relevant ones to our task.

The following discussion will be focused on the evaluation model that had been made to test the auditory feature extractors, a deep learning model developed in MATLAB for classifying different music genres using the features intended, the classification accuracy " in percentage" is a direct indication of the used feature performance. For example, while comparing the features classification results the higher the percentage the better this feature is for classifying the type of signals used as an input signal.

4.1 Types of signals

The features will be tested with different types of signals to determine which are the most noise robust and which have higher performance. A set of audio files without vocals "Just music "will be used and another set with vocals "Music with speech", these sets of audio files will be interrupted with noise, each of these sets will be independently tested using each of the auditory feature extractors. The music dataset consists of 200 audio files of two genres, each genre is 100 audio files, and the vocals dataset consists of 300 audio files of 3 genres, each is 100 files, audio files are mono channel and sampled at 22050 Hz.

The amount of audio files used have no direct impact on the performance of the feature extractors. The reason is that the classification percentage of each of these feature extractors will all be compared together to determine which has the highest percentage and consequently performs better, for example if Mel frequency cepstral coefficients were extracted for a specific dataset and resulted in 65% and gammatone cepstral coefficients were extracted for the same dataset and resulted in 60%, and bark spectrum coefficients resulted in 55%, then this means that for this type of audio data (music with vocals), MFCC has better performance than GTCC & bark spectrum, and the same procedure will be followed with other types of datasets and after noise disruption. This way it can be figured out which are the best feature extractors that performs better than others and are closer to the human ear which was found to be 70%, then the closer

the accuracy of a feature extractor to this threshold, the closer it is to human auditory perception.

4.2 Deep learning model for classification

The classification model is made in MATLAB, it is a method for testing the performance of the audio feature extractors. Firstly, the dataset is loaded, then split to training and validation sets with ration of 80% training and 20% validation. Extraction of features will be performed separately on each of the two sets, as mentioned before, that audio sets are highly dimensional with many redundant information, so feature extraction keeps the highly relevant information while discarding the others. The data used in this model is considered to be small and deep learning models will show lower performance compared to machine learning models when small data is used, but in this study the focus is to compare the feature extractors and determine the best performing with different types of audio data.

The learning model workflow starts with accessing the audio files then reading them, MATLAB will save them in what is called " cell arrays", these audio files are labelled and it is always preferred that each label contains sufficient amount of audio files so that the neural network can learn about each label and pick the patterns and features that distinguish each label from the other. The neural network was designed using LSTM with an adjustable number of hidden units, sequence input layer, fully connected layer, SoftMax layer and classification output layer. The number of sequence input layers corresponds to the number of features or dimensions that have been extracted from the audio files. For example, MFCC extraction will extract 13 coefficients that contains the most highly relevant information to feed to the network, pitch corresponds to one additional layer and so on, the architecture of the network used is taken from here [71]. The size of the fully connected layer corresponds to the number of labels of the audio data entered as input.

After designing the neural network, training options needs to be specified, the optimizer is set to " Adam" , mini-patch options to shuffle and made to be "every-epoch", and validation data is fed with the validation dataset's cell array, plots are allowed to be displayed to see the accuracy percentage of the classification model. The model is ready to be trained and display the results, however some modification can be done like number of hidden units in the LSTM, and the model itself is subjective to many trial and error and modify different parameters in order to get the desired results, but in this study as its main subject to compare and determine best performing features not to get high accuracy percentage of classification, then current parameters should suffice.

4.2.1 Results

The first dataset to be fed to the network consists of 200 audio files of two music genres, each genre is 100 audio files, sampled at 22 Khz, and it is of non-vocal audio data " just music", each song is segmented into 30 seconds.

The feature extractors used in the training will be only the ones meant to be tested, by this way, the final accuracy classification percentage can all be given to the feature extractor selected without any contribution from other additional feature extractors that can improve the accuracy as spectral centroid, spectral flux and pitch.

Training with Mel spectrum

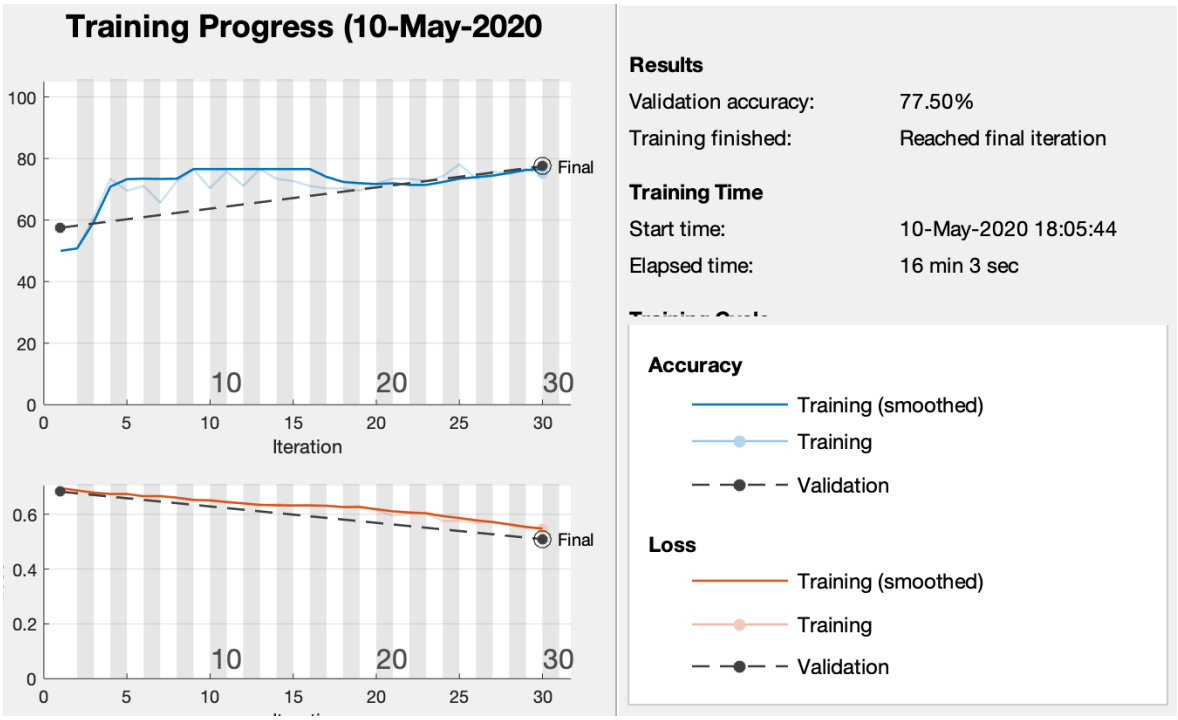

Figure 4.1 shows the training results using Mel spectrum for audio classification

Adding noise

Noise will be added to test the feature extractor robustness, noise can be due to environmental factors or any distortions that may happen and disrupt the audio signals. The amount of noise added to the dataset is the same with all feature extractors.

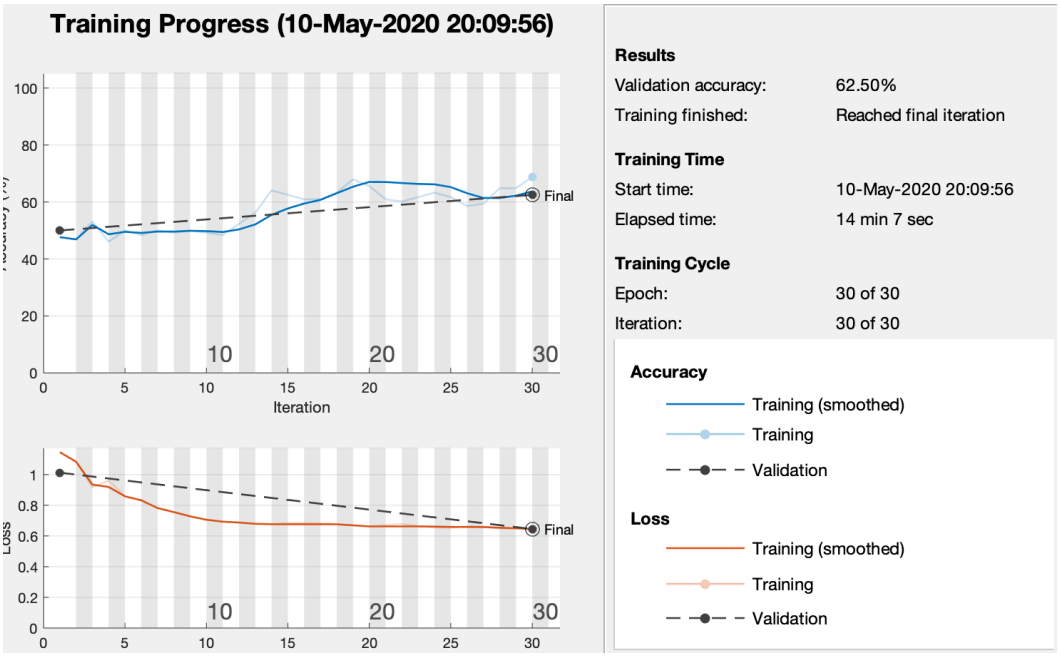

Figure 4.2 shows the training results with Mel spectrum and adding noise

Training with erb spectrum "Gammatone cepstral coefficients"

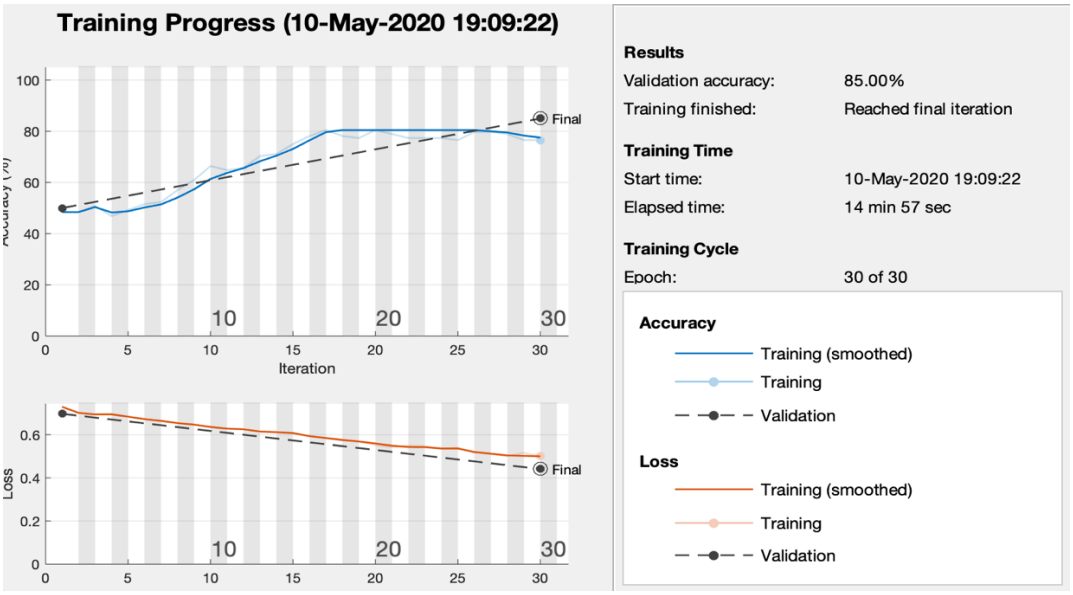

Figure 4.3 shows the training results with erb spectrum

Adding noise

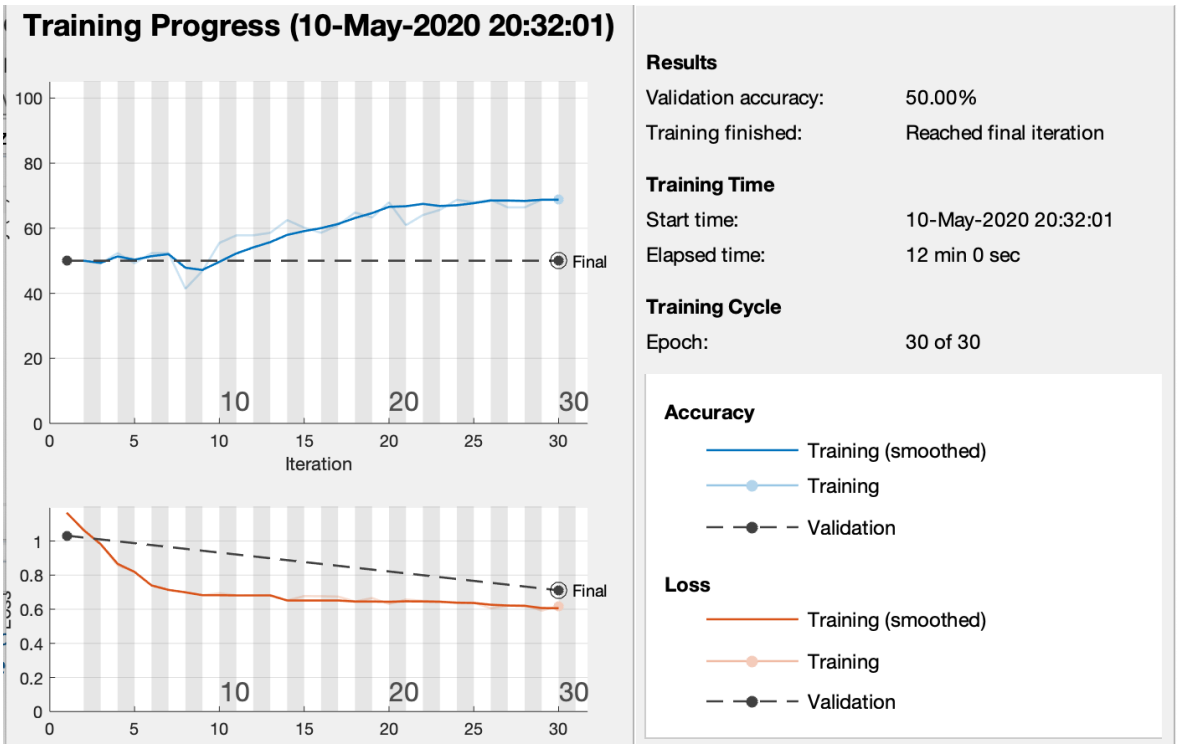

Figure 4.4 shows the training results with erb spectrum and added noise

Training with bark spectrum

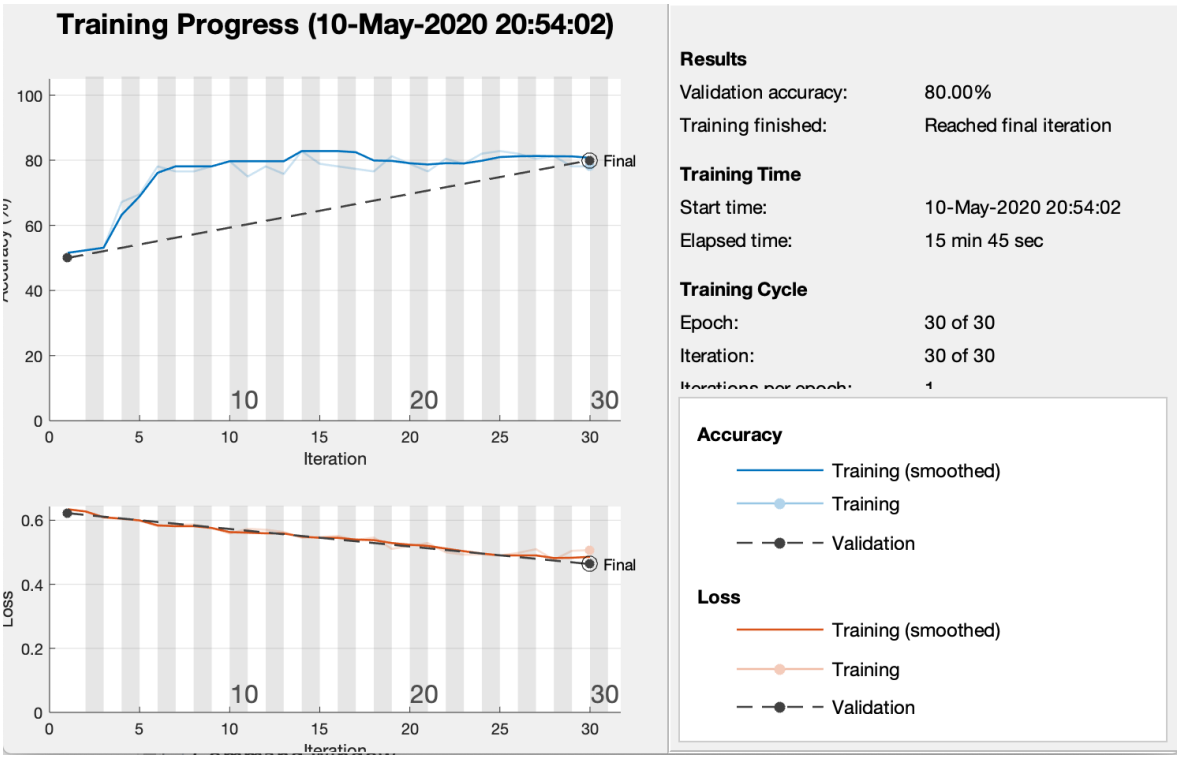

Figure 4.5 shows the training results with bark spectrum

Adding noise

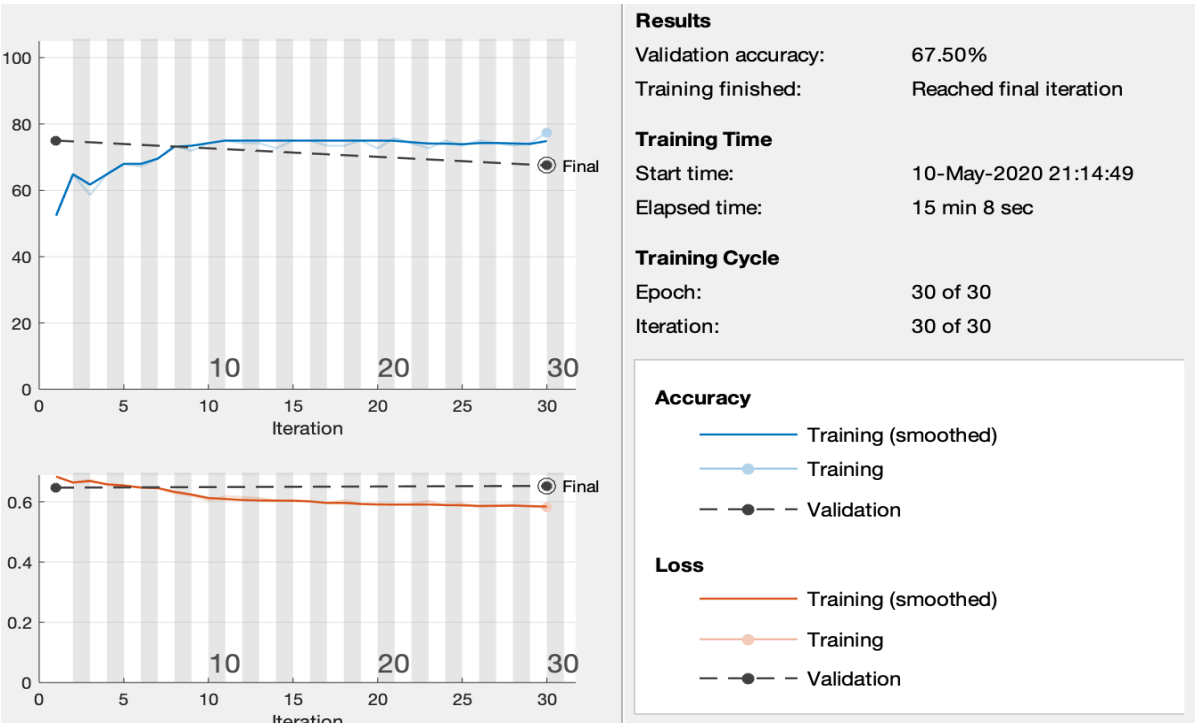

Figure 4.6 shows the training results with bark spectrum and added noise

The second dataset which consists of 200 audio files of two music genres but with vocals is to be fed to the neural network with same feature extractors and conditions applied to the previous dataset.

Training with Mel spectrum

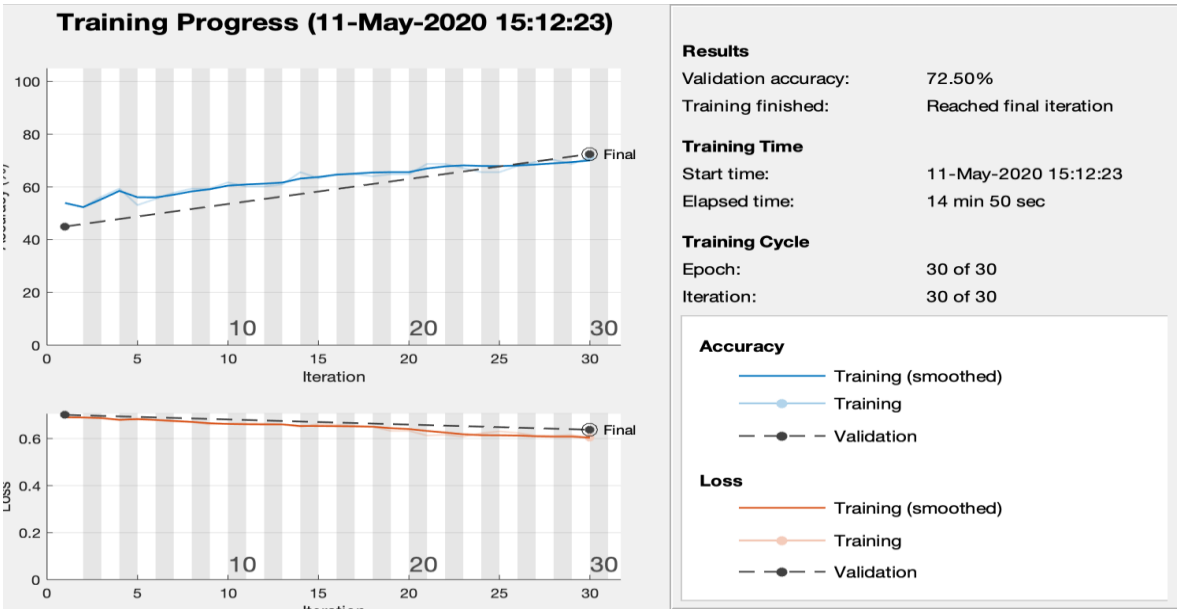

Figure 4.7 shows the training results with Mel spectrum with the vocals dataset

Adding noise

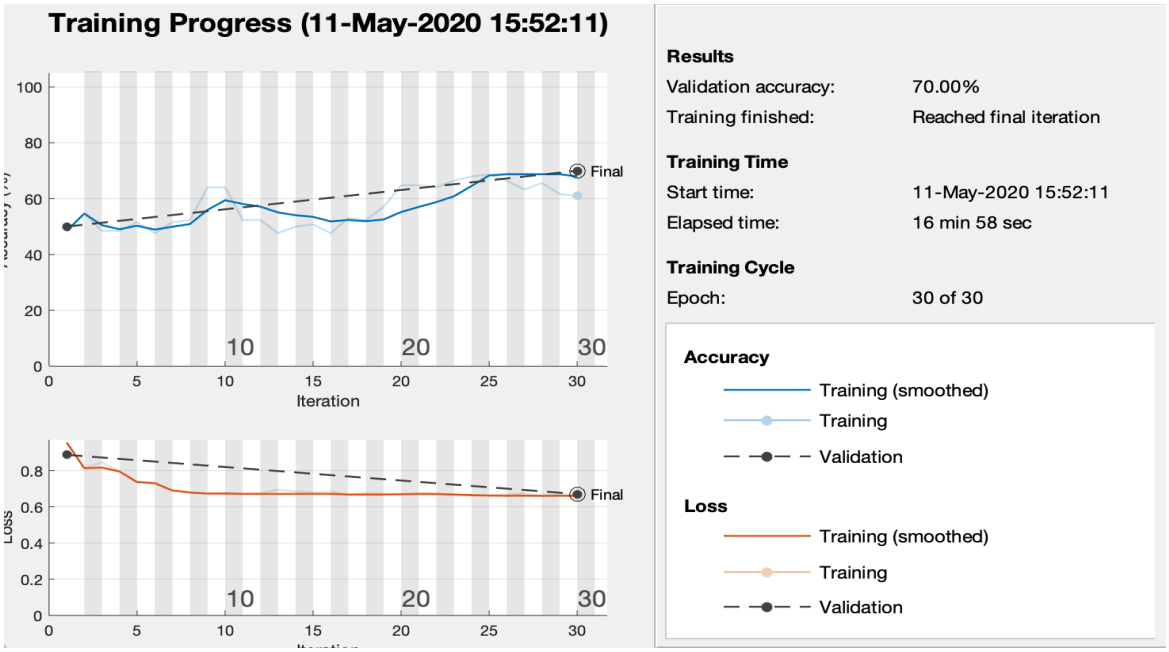

Figure 4.8 shows the training results with Mel spectrum of the vocals dataset and added noise

Training with erb spectrum

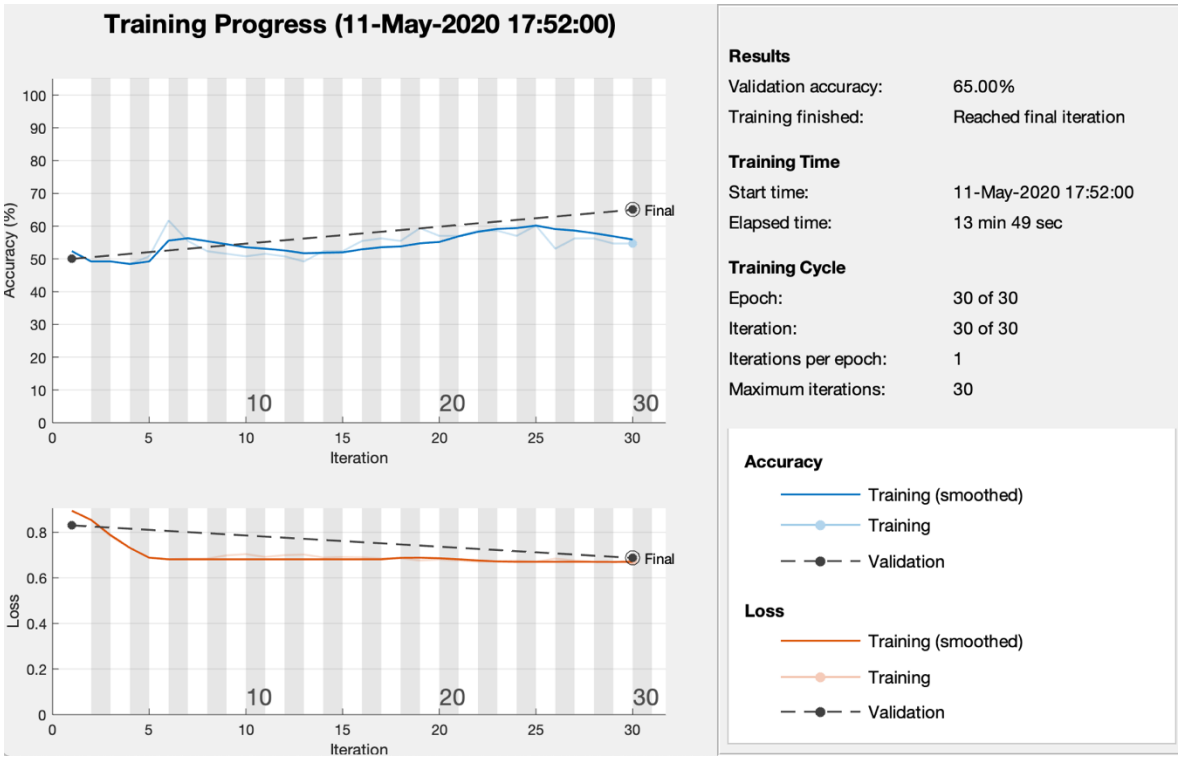

Figure 4.9 shows the training results with erb spectrum of the vocals dataset

Adding noise

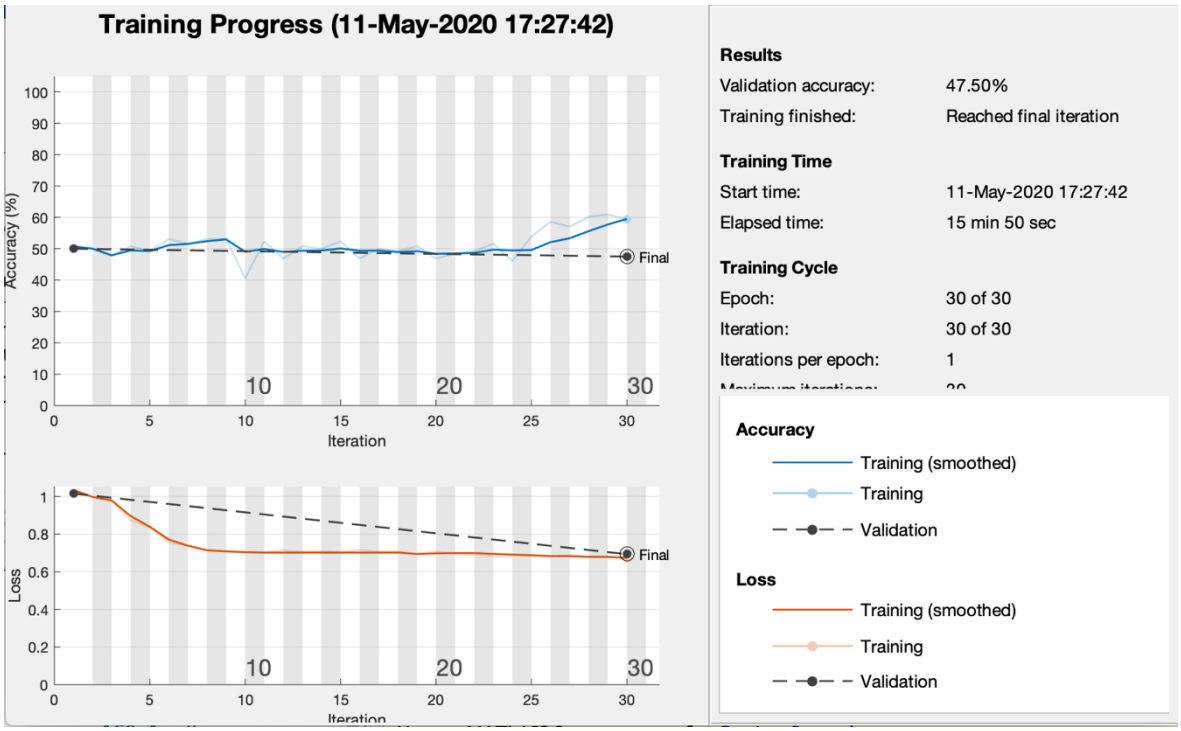

Figure 4.10 shows the training results with erb spectrum of the vocals dataset and added noise

Training with bark spectrum

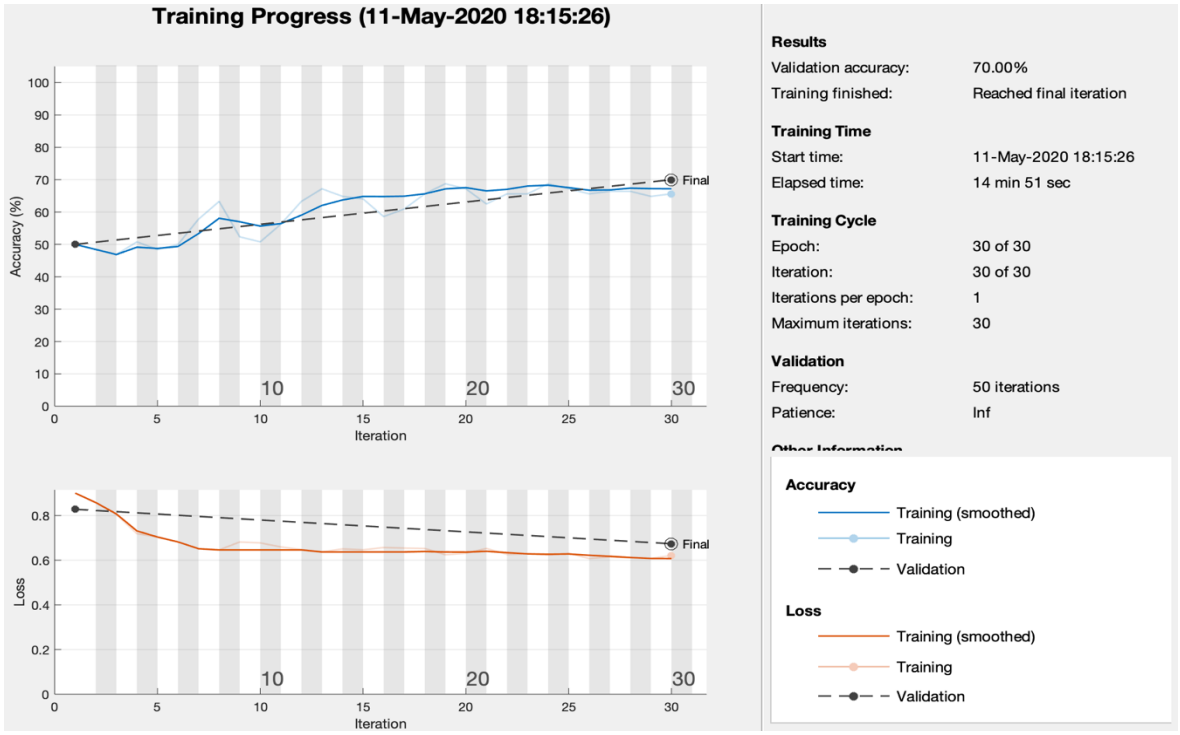

Figure 4.11 shows the training results with bark spectrum of the vocals dataset

Adding noise

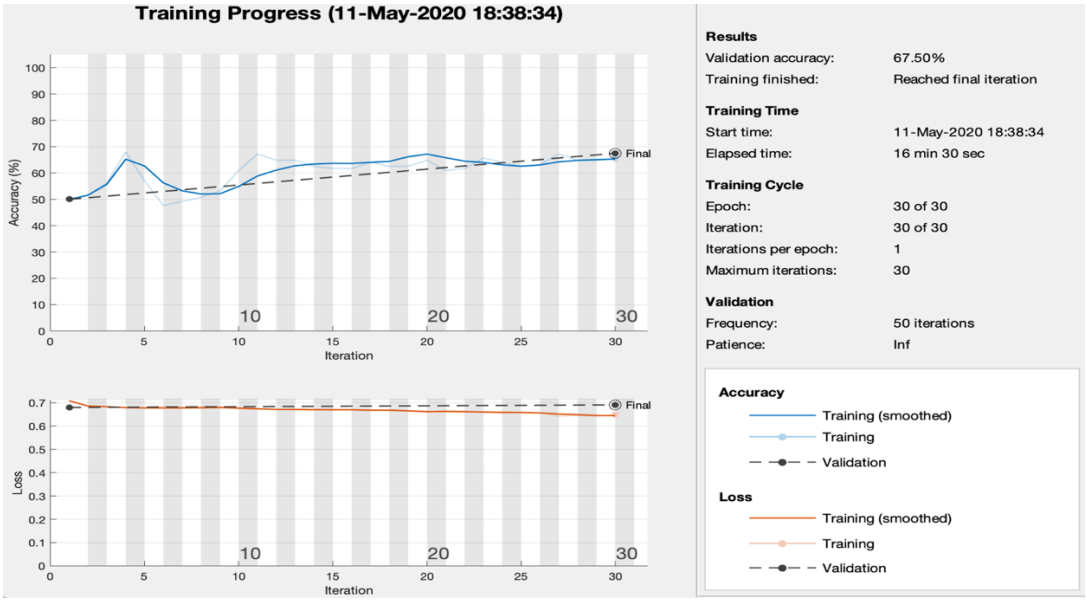

Figure 4.12 shows the training results with bark spectrum of the vocals dataset with added noise

The third dataset is mixed between the first and the second dataset, which means it contains 200 audio files of both music vocals and non-vocal music. The model training will be run with the same conditions applied to the previous datasets, and same value of noise.

Training with Mel spectrum

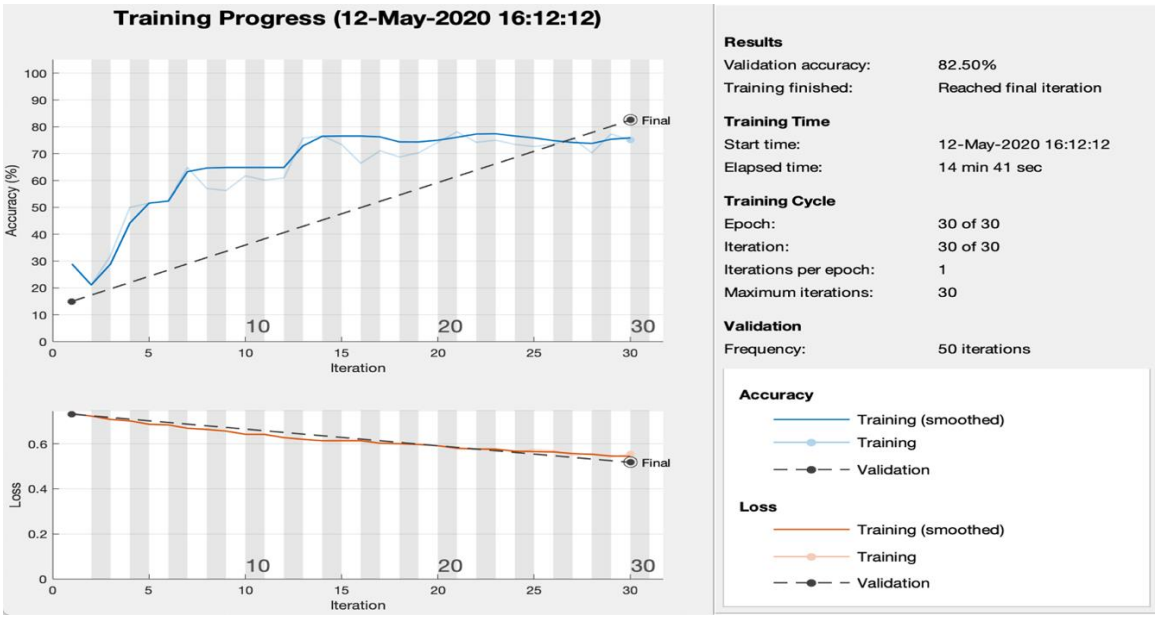

Figure 4.13 shows the training results with mel spectrum of the mixed dataset

Adding noise

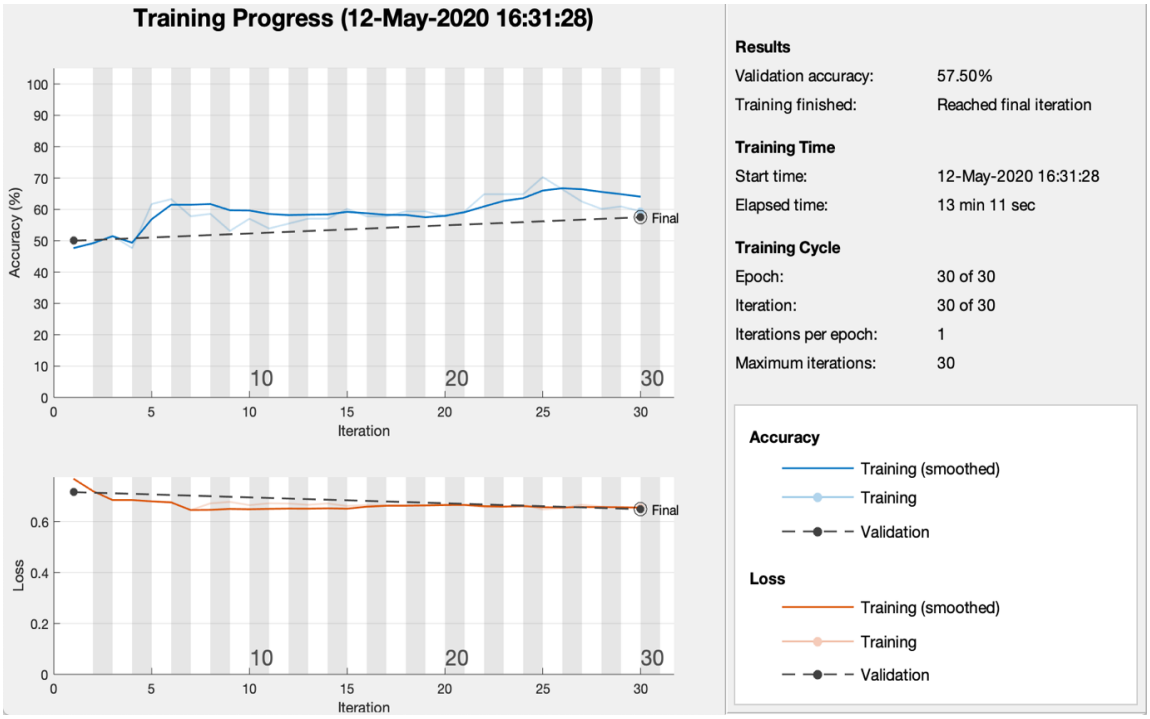

Figure 4.14 shows the training results with mel spectrum of the mixed dataset with added noise

Training with erb spectrum

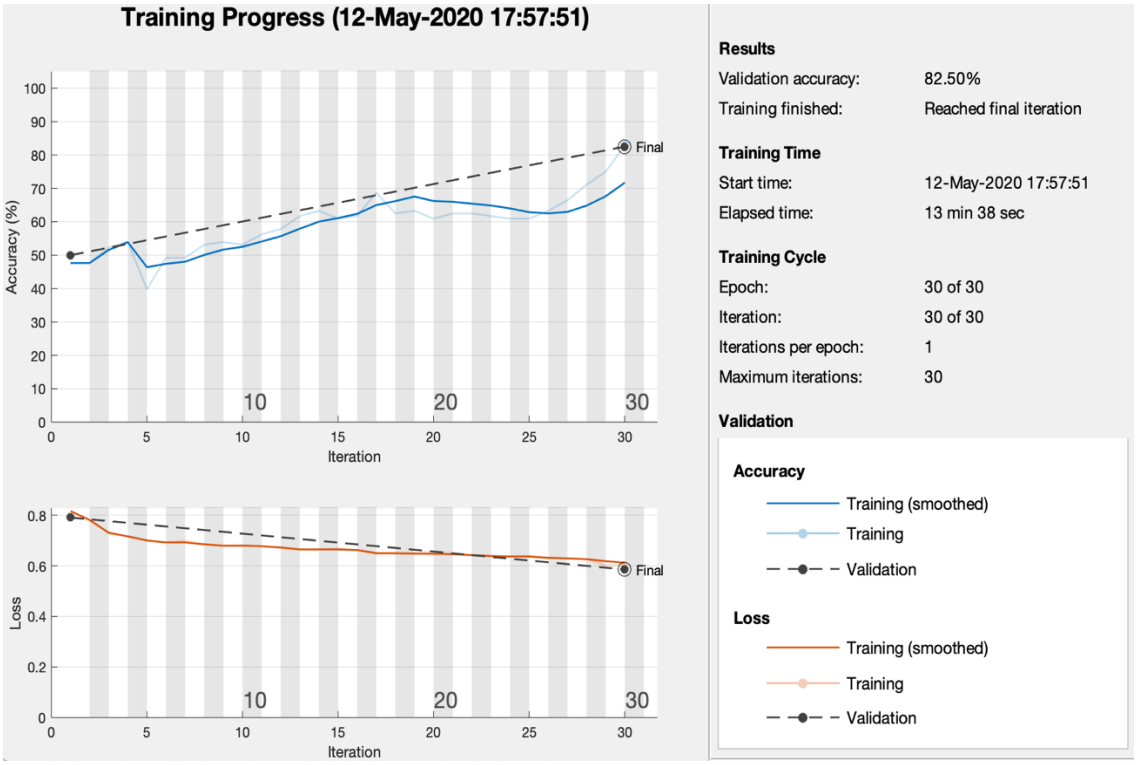

Figure 4.15 shows the training results with erb spectrum of the mixed dataset

Adding noise

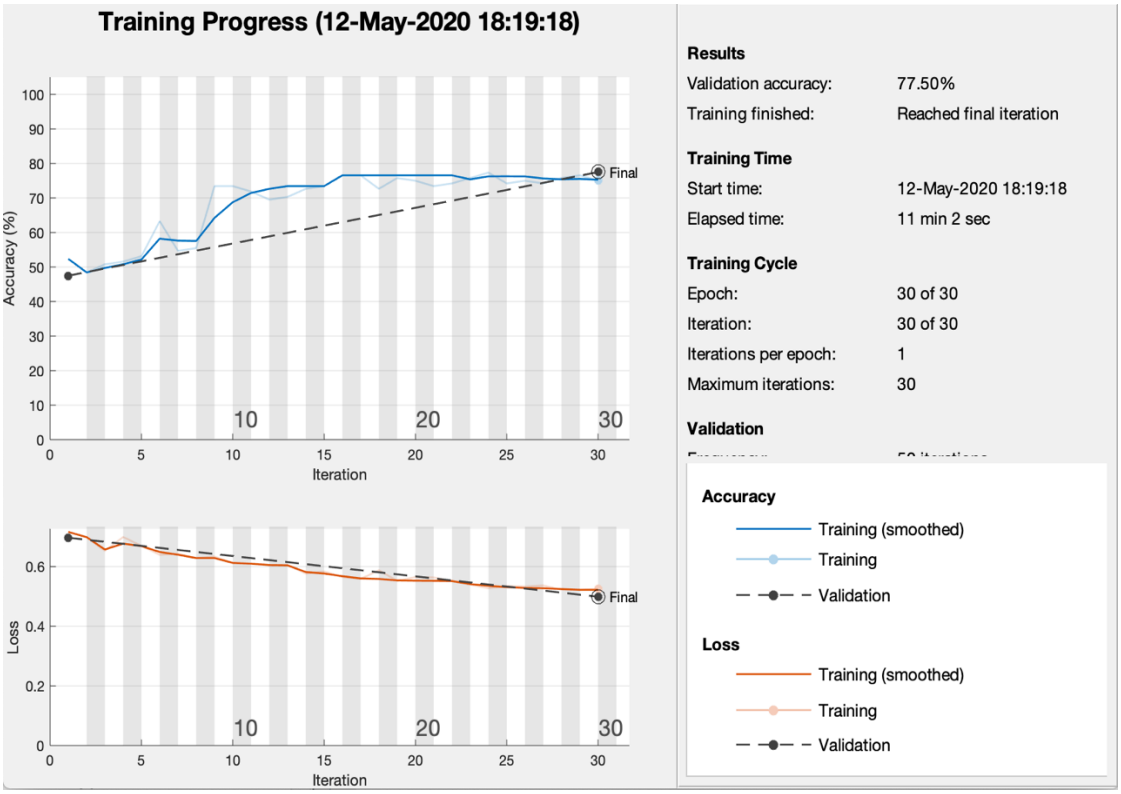

Figure 4.16 shows the training results with erb spectrum of the mixed dataset with added noise

Training with bark spectrum

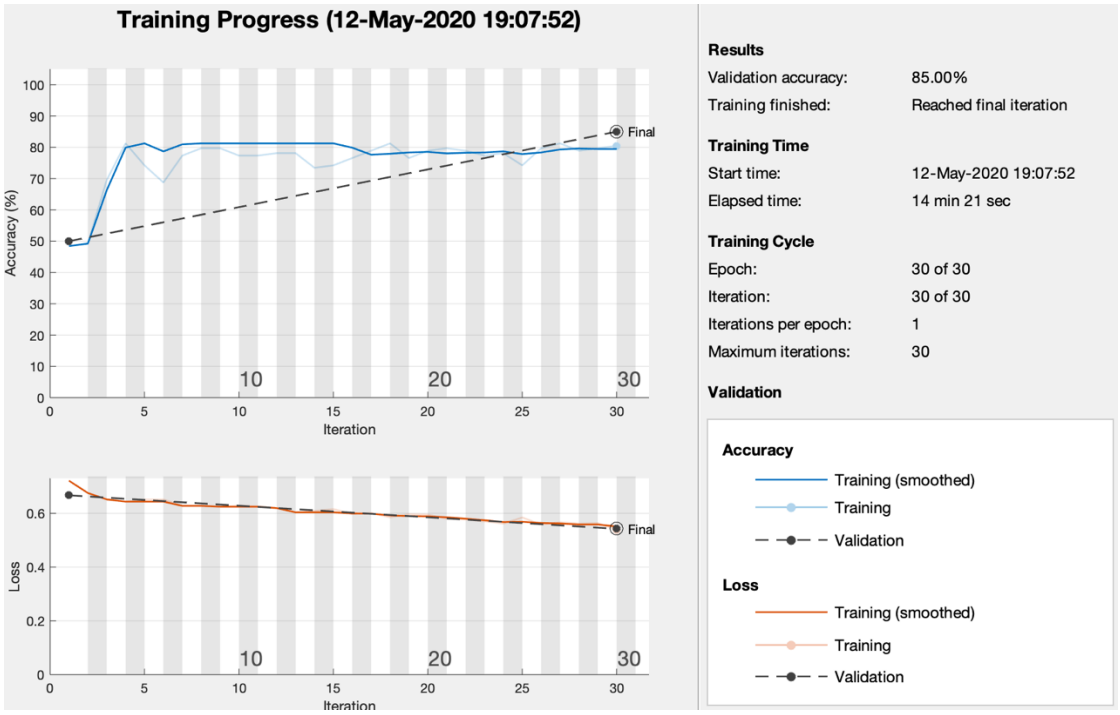

Figure 4.17 shows the training results with bark spectrum of the mixed dataset

Adding noise

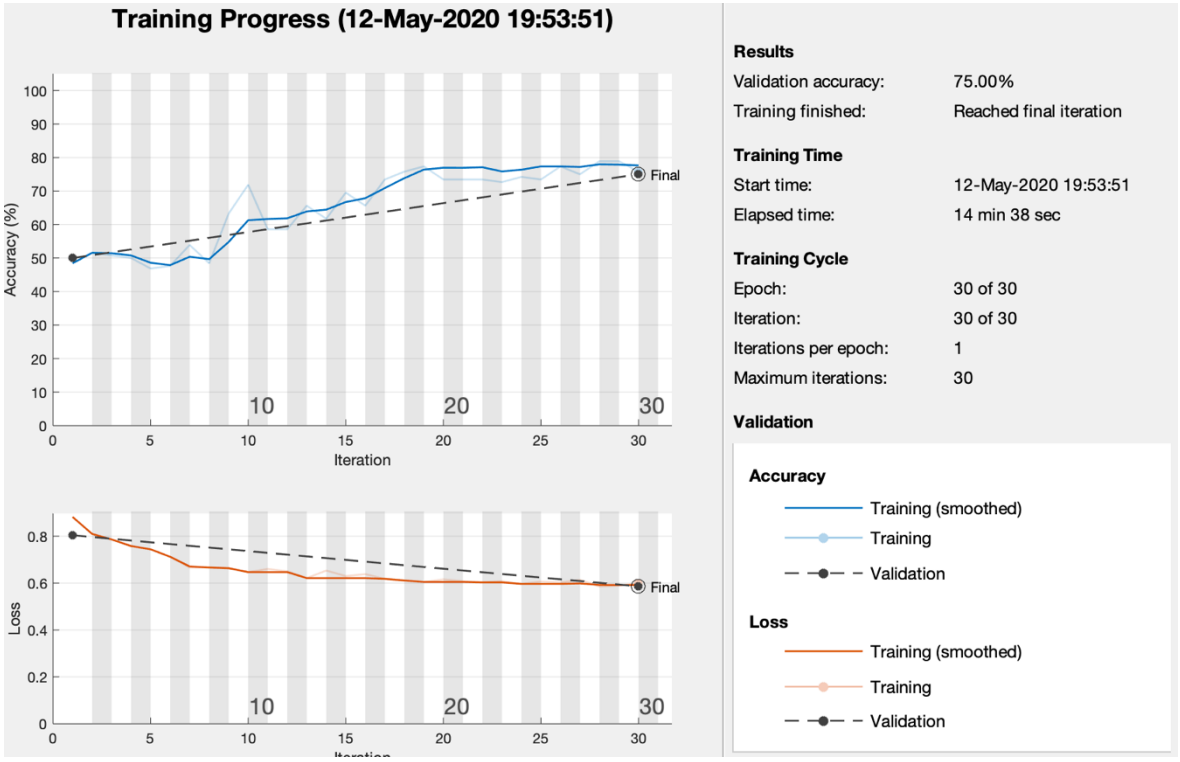

Figure 4.18 shows the training results with bark spectrum of the mixed dataset with added noise

4.3 Conclusion

The deep learning model acted as an evaluation method to test the performance of the auditory feature extractors. The aim from this experiment to find the feature that best approximates or gives the most closer representation to the human ear, for this reason noise were added to all audio files in each dataset to test its robustness and masking properties, also considering different types of audio signals were taken in consideration as there is music with vocals and non-vocal music. Music with vocals has an added speech signal to the audio itself which alters its spectral range of frequency and this in turn affects the performance of the feature extractor, also considered adding noise that may occur due to any environmental conditions. After model training the following results were found, the results are in percentage that indicates the ability of the model to classify the audio files into genres.

Table 4.1 Classification accuracy of audio feature extractors by deep learning model

Feature extractor	Non-vocal music dataset	Non-vocal music dataset with noise	Vocal music dataset	Vocal music dataset with noise	Mixed dataset of the previous two datasets	Mixed dataset with noise
Mel spectrum	77.50%	62.50%	72.50%	70%	82.50%	57.50%
Erb spectrum	85%	50%	65%	47.50%	82.50%	77.50%
Bark spectrum	80%	67.50%	70%	67.50%	85%	75%

The table above shows the classification percentage resulted from the deep learning model training. It reflects the ability of the feature extractors to classify songs that were given as input audio signals to the model. These feature extractors work by extracting the most relevant information in the audio signals decreasing its high dimensionality and feeding to the classification model. The accuracy results showed a high accuracy for the erb spectrum with non-vocal music classification while low when disrupted by noise, this means it gives great results if the audio signals were denoised before applying it, but it won't perform well in case of real time classification and there exists noise due to environmental conditions. The Mel spectrum showed a moderate and can be said acceptable performance in both classification for non-vocal music and noise robustness, and it is commonly used in music industries, then comes the bark scales that showed good classification performance makes it better than the Mel spectrum and higher noise robustness which gives it an advantage over the erb and Mel spectrums. The second testing phase that have the vocal music dataset, Mel spectrum's accuracy surpassed both the erb and bark spectrums and also in noise robustness, making it a good choice for vocal music, or speech recognition applications. The third testing phase that have the mixed dataset of combined vocal and non-vocal music, which makes sense of most real audio datasets that exists out there and makes it easier to use in different application than to split to vocal and non-vocal. The bark spectrum performed the best as feature extractor than both Mel and erb spectrums and also for noise robustness, making it the best choice to apply for real time audio or large audio datasets. This result makes it the closest approximation to the human ear combining its noise robustness properties and classification accuracy and the reason as mentioned in section 3.4.5 its structure of 24 bands that corresponds to the 24 critical bands of the human hearing.

SUMMARY

Audio data is vastly growing with millions of songs being published every year, then became the need to develop algorithms and robust systems to deal with is inevitable. Music information retrieval is the science that deals with such audio data and retrieve information from it then feed it to machine learning algorithms to perform specific tasks. In the last decade these algorithms became so much developed to efficiently manage and do different MIR tasks, however there still many research areas that researchers pointed out for development and these are being discussed every year in the international society of music information retrieval "ISMIR". One of the most important steps in MIR tasks is the feature extraction. As mentioned before that audio data are highly dimensional data and most of these dimensions or features are considered to be redundant, so feeding the raw audio data directly to algorithms is not going to get good results, for this reason comes the necessity to only extract the most relevant features from the data, and this is done through the feature extraction process. The feature extraction process involves applying some auditory feature extractors with signal processing, the efficacy of these feature extractors depends on how effective they are in extracting the relevant information from the audio signals even if these signals have been exposed to noise due to environmental or other factors. The most commonly used is the Mel frequency cepstral coefficients which scales the frequency response of the audio signals on a non-linear scale, and this is inspired by the way humans listen to sounds. We listen to sounds linearly for sounds that are below 1000 Hz but above, we perceive it logarithmically. There are different types of psychoacoustical scales that approximates audio in a similar way human do, but the question is to what extent? then testing different feature extractors with real audio is the way to see which will perform better, specially that using mel scale was found to give poor performance in the presence of noise, and this disadvantage will be a challenge in case it is needed to process audio data in real time and there exists a noise due to any environmental conditions. There are two other psychoacoustical scales that also have similar properties to the human ear and may hold the key to a better performance, those are the erb scale and bark scale. A deep learning model was made to classify music genres, and the spectrum of these audio files were extracted using the three psychoacoustical scales to see which performs better and more noise robust, also different types of audio signals were used (music with vocals, non-vocal music and mixed). The results showed that bark scale is the best to approximate human hearing and have the best noise robustness properties, this can make it as a better alternative to use in music genre classification without need to separate the datasets into vocals and non-vocals which would be a difficult task in processing large amounts of audio data.

LIST OF REFERENCES

[1] G. Tzanetakis and G. Tzanetakis, '1.3 in Manipulation, analysis and retrieval systems for audio signals', in *Princeton University, Princeton, NJ*, 2002, p. 198.

[2] M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, 'Content-based music information retrieval: Current directions and future challenges', *Proc. IEEE*, vol. 96, no. 4, pp. 668–696, 2008, doi: 10.1109/JPROC.2008.916370.

[3] Chirp, 'Audio fingerprinting — what is it and why is it useful?', 2018.

[4] and P. H. Joan Serra, Emilia Gomez, 'Advances in music information retrieval', in *Advances in music information retrieval*, 2010, p. 423.

[5] M. Schedl, E. Gómez, and J. Urbano, '059-Music-Information-Retrieval-Recent-Developments-Applications.Pdf', vol. 8, no. 2, 2014, pp. 127–261.

[6] G. Marradi, 'Giovanni Marradi Just for You', 2014. [Online]. Available: https://www.scribd.com/document/162968969/Giovanni-Marradi-Just-for-You. [Accessed: 23-Apr-2020].

[7] M. Ottewill, 'Planet Of Tunes - Sound waveform diagrams', 2015. [Online]. Available: http://www.planetoftunes.com/sound-audio-theory/sound-waveform-diagrams.html#.XqHN1m5S9o5. [Accessed: 23-Apr-2020].

[8] J. Wolfe, 'Speech and music, acoustics and coding, and what music might be "for"', *Proc. 7th Int. Conf. Music Percept. Recognition, 2002*, pp. 10–13, 2002, doi: 10.1.1.294.8001.

[9] Steve Tjoa, 'why_mir', *2015*. [Online]. Available: https://musicinformationretrieval.com/why_mir.html. [Accessed: 30-Mar-2020].

[10] J. U. Emilia Gómez,Markus Schedl, 'Music InformationRetrieval:Recent Developments andApplications', 2014.

[11] J. Foote, 'Visualizing Music and Audio using Self-Similarity'.

[12] R. . Perrot, D., and Gjerdigen, 'Scanning the dial: An exploration of factors in the identification of musical style. In Proceedings of the 1999 Society for Music Perception and Cognition pp.88', 1999.

[13] I. S. Stepan Evstifeev, ‘Music Genre Classification Based on Signal Processing’, *2015*. [Online]. Available: moz-extension://31f58944-9ef0-d44d-8113-7a8bfa126874/enhanced-reader.html?openApp&pdf=http%3A%2F%2Fceur-ws.org%2FVol-2277%2Fpaper28.pdf. [Accessed: 02-Apr-2020].

[14] G. E. George Tzanetakis, Perry Cook, ‘Automatic Musical Genre Classification Of Audio Signals’, 2002.

[15] M. Dong, ‘Convolutional Neural Network Achieves Human-level Accuracy in Music Genre Classification’, Feb. 2018.

[16] E. J. P. Arianna A. Serafini, Derek A. Huang, ‘Music Genre Classification’, 2019.

[17] D. S. Alexander Gunawan, ‘Music Recommender System Based on Genre using Convolutional Recurrent Neural Networks - ScienceDirect’. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050919310646. [Accessed: 06-Apr-2020].

[18] M. S. Peter Knees, ‘9.3 Music Similarity and Retrieval’, in *313*, .

[19] S. Luo, ‘Introduction to Recommender System - Towards Data Science’, 2018.

[20] LANDR, ‘Metadata for Musicians: What It Is and Why It’s Vital | LANDR Blog’, 2019.

[21] T. Li and G. Tzanetakis, ‘Music Data Mining’, 2011, pp. 3–42.

[22] L. J. P. van der Maaten, ‘Dimensionality Reduction: A Comparative Review’, 2019.

[23] O. Source, ‘The Curse of Dimensionality in Classification’, *2014*. [Online]. Available: https://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/. [Accessed: 11-Apr-2020].

[24] R. Shaikh, ‘Feature Selection Techniques in Machine Learning with Python’, 2018. [Online]. Available: https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e. [Accessed: 14-Apr-2020].

[25] Haitian Wei, ‘Feature Selection Methods with Code Examples - Analytics Vidhya - Medium’, 2019.

[26] A. Shetye, ‘Feature Selection with sklearn and Pandas - Towards Data Science’,

2019. [Online]. Available: https://towardsdatascience.com/feature-selection-with-pandas-e3690ad8504b. [Accessed: 16-Apr-2020].

[27] Y. Charfaoui, 'Hands-on with Feature Selection Techniques: Embedded Methods', 2020.

[28] S. Lahoti, '4 ways to implement feature selection in Python for machine learning | Packt Hub', 2018. [Online]. Available: https://hub.packtpub.com/4-ways-implement-feature-selection-python-machine-learning/. [Accessed: 15-Apr-2020].

[29] K. Yin, 'Step-by-Step Signal Processing with Machine Learning: PCA, ICA, NMF for source separation, dimensionality reduction', 2019.

[30] 2011 Pedregosa et al., JMLR 12, pp. 2825-2830, 'Importance of Feature Scaling — scikit-learn 0.22.2 documentation', *Pedregosa al., JMLR 12, pp. 2825-2830, 2011*, 2019.

[31] S. Raschka, 'Implementing a Principal Component Analysis (PCA)', 2014.

[32] N. R. C. (US) C. on D. D. for I. with H. Impairments, R. A. Dobie, and S. Van Hemel, 'Basics of Sound, the Ear, and Hearing', 2004.

[33] G. S. ; M. G. ; T. Friedrich, 'Fast Audio Feature Extraction From Compressed Audio Data - IEEE Journals & Magazine', 2011. [Online]. Available: https://ieeexplore.ieee.org/document/5784292. [Accessed: 25-Feb-2020].

[34] A. Lerch, *An introduction to audio content analysis: Applications in signal processing and music informatics*. 2012.

[35] 'Non-Stationary Nature of Speech Signal (Theory) : Speech Signal Processing Laboratory : Biotechnology and Biomedical Engineering : Amrita Vishwa Vidyapeetham Virtual Lab'. [Online]. Available: http://vlab.amrita.edu/?sub=3&brch=164&sim=371&cnt=1104. [Accessed: 26-Feb-2020].

[36] M. Müller, *Fundamentals of Music Processing*. 2015.

[37] M. Tian, G. Fazekas, D. A. A. Black, and M. Sandler, 'On the use of the tempogram to describe audio content and its application to Music structural segmentation', in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2015, vol. 2015-August, pp. 419–423, doi:

10.1109/ICASSP.2015.7178003.

[38] M. Müller, 'novelty_functions', *Fundamentals of Music Processing*, 2015. [Online]. Available: https://musicinformationretrieval.com/novelty_functions.html. [Accessed: 02-Mar-2020].

[39] M. Müller, 'C6S1_NoveltySpectral', *Fundamentals of Music Processing*, 2015. [Online]. Available: https://www.audiolabs-erlangen.de/resources/MIR/FMP/C6/C6S1_NoveltySpectral.html. [Accessed: 06-Mar-2020].

[40] Theodoros Giannakopoulos, 'Spectral Flux - an overview | ScienceDirect Topics', *Introduction to Audio Analysis*. [Online]. Available: https://www.sciencedirect.com/topics/engineering/spectral-flux. [Accessed: 07-Mar-2020].

[41] P. Knees and M. Schedl, *Music similarity and retrieval*. 2013.

[42] J. Hsu, C. Liu, and A. L. P. Chen, *Music Data*, vol. 3, no. 3. 2001.

[43] T. Giannakopoulos, 'Zero Crossing Rate - an overview | ScienceDirect Topics', *Introduction to Audio Analysis*, 2014. [Online]. Available: https://www.sciencedirect.com/topics/engineering/zero-crossing-rate. [Accessed: 10-Mar-2020].

[44] M. Müller, 'energy', *Fundamentals of Music Processing*, 2015. [Online]. Available: https://musicinformationretrieval.com/energy.html. [Accessed: 11-Mar-2020].

[45] P. R.hill, *Audio and speech processing with Matlab*. 2018.

[46] Paolo Prandoni and Martin Vetterli, 'SIGNAL PROCESSING', 2008, p. 388.

[47] Ian McLoughlin, 'Applied Speech and Audio Processing', 2009.

[48] F. Alías, J. C. Socoró, and X. Sevillano, 'A review of physical and perceptual feature extraction techniques for speech, music and environmental sounds', *Appl. Sci.*, vol. 6, no. 5, 2016, doi: 10.3390/app6050143.

[49] F. B. Claudia Schremmer, Thomas Haenselmann, 'WAVELETS IN REAL-TIME DIGITAL AUDIO PROCESSING: ASOFTWARE FOR UNDERSTANDING WAVELETS

IN APPLIED COMPUTERSCIENCE', 2015. [Online]. Available: moz-extension://31f58944-9ef0-d44d-8113-7a8bfa126874/enhanced-reader.html?openApp&pdf=http%3A%2F%2Fciteseerx.ist.psu.edu%2Fviewdoc%2Fdownload%3Fdoi%3D10.1.1.23.9490%26rep%3Drep1%26type%3Dpdf. [Accessed: 15-Mar-2020].

[50] A. Popescu, I. Gavat, and M. Datcu, 'Wavelet analysis for audio signals with music classification applications', in *2009 Proceedings of the 5th Conference on Speech Technology and Human-Computer Dialogue, SpeD 2009*, 2009, doi: 10.1109/SPED.2009.5156187.

[51] P. C. George Tzanetakis, Georg Essl, 'Audio Analysis using the Discrete Wavelet Transform'. [Online]. Available: moz-extension://31f58944-9ef0-d44d-8113-7a8bfa126874/enhanced-reader.html?openApp&pdf=https%3A%2F%2Fsoundlab.cs.princeton.edu%2Fpublications%2F2001_amta_aadwt.pdf. [Accessed: 16-Mar-2020].

[52] MATLAB, 'Types of Wavelet Transforms', 2016.

[53] F. Bömers, 'Wavelets in real time digital audio processing : Analysis and sample implementations', no. May, pp. 1–119, 2000.

[54] Mathworks, 'Wavelet Denoising - MATLAB & Simulink'. [Online]. Available: https://www.mathworks.com/help/wavelet/ug/wavelet-denoising.html. [Accessed: 19-Mar-2020].

[55] D. Gartzman, 'Getting to Know the Mel Spectrogram - Towards Data Science', 2019. [Online]. Available: https://towardsdatascience.com/getting-to-know-the-mel-spectrogram-31bca3e2d9d0. [Accessed: 21-Mar-2020].

[56] PeterAhrend, 'Music Genre Classification Systems [1ex] - A Computational Approach | Enhanced Reader', 2006.

[57] M. S. P. Knees, 'B.S Music similarity and retrieval', in *Music similarity and retrieval*, 2013, p. 1125.

[58] H. Fayek, 'Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between | Haytham Fayek', 2016. [Online]. Available: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html. [Accessed: 23-Mar-2020].

[59] K. K. W.́ojcicki Kuldip K. Paliwal, ames G. Lyons, 'Preference for 20-40 ms

window duration in speech analysis', 2015.

[60] Roger Jang, '12-2 MFCC'. [Online]. Available: http://mirlab.org/jang/books/audioSignalProcessing/speechFeatureMfcc.asp?title=12-2 MFCC. [Accessed: 25-Mar-2020].

[61] O. Resource, 'Mel Frequency Cepstral Coefficient (MFCC)', *2013*. [Online]. Available: http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/. [Accessed: 26-Mar-2020].

[62] G. K. Liu, 'Evaluating Gammatone Frequency Cepstral Coefficients with Neural Networks for Emotion Recognition from Speech', Jun. 2018.

[63] K. S. R. and A. K. Vuppala, 'MFCC Features', *2014*. [Online]. Available: moz-extension://31f58944-9ef0-d44d-8113-7a8bfa126874/enhanced-reader.html?openApp&pdf=https%3A%2F%2Flink.springer.com%2Fcontent%2Fpdf%2Fbbm%253A978-3-319-03116-3%252F1.pdf. [Accessed: 26-Mar-2020].

[64] B. L. Zheng-Hua Tan, *Automatic Speech Recognition on Mobile Devices and over Communication Networks - Google Books*. 2008.

[65] P. Pertila, 'Gammatone filter banks & Mel-frequency cepstral coefficients Introduction', *Tut*, 2015.

[66] J. M. Liu *et al.*, 'Cough signal recognition with gammatone cepstral coefficients', in *2013 IEEE China Summit and International Conference on Signal and Information Processing, ChinaSIP 2013 - Proceedings*, 2013, pp. 160–164, doi: 10.1109/ChinaSIP.2013.6625319.

[67] W. H. Abdulla, 'Auditory based feature vectors for speech recognition systems', *Adv. Commun. Softw. Technol.*, no. January 2002, pp. 231–236, 2002.

[68] E. Tutorial, 'Active Band Pass Filter - Op-amp Band Pass Filter', 2015. [Online]. Available: https://www.electronics-tutorials.ws/filter/filter_7.html. [Accessed: 27-Apr-2020].

[69] N. Ma, 'An Efficient Implementation of Gammatone Filters'.

[70] T. S. Natarajan Sriraam and P. G. C. M, 'Pre-term Neonates Cry Pattern Recognition Using Bark Frequency Cepstral Coefficients', 2020, pp. 335–338, doi: 10.1109/icatiece45860.2019.9063769.

[71] MATLAB, 'Long Short-Term Memory Networks - MATLAB & Simulink', *2020*. [Online]. Available: https://www.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html. [Accessed: 10-May-2020].

[72] MATLAB, 'Classify Sound Using Deep Learning - MATLAB & Simulink', 2019. [Online]. Available: https://www.mathworks.com/help/audio/gs/classify-sound-using-deep-learning.html. [Accessed: 13-May-2020].

APPENDIX

MATLAB deep leaning model code used to test the features, the design parameters of the network was used by help of this example [72].

```
% load the genre dataset

location = fullfile('/Users/zaynamed/Desktop/MATLAB Testing/Dataset/Mixed Genres');

ads = audioDatastore(location, 'IncludeSubFolders', true,...
     'LabelSource', 'foldernames' );

% Split to training and test data:

 [traindata,testdata]=splitEachLabel(ads,0.8);

 lentraindata= length(traindata.Files);

 % read the audio files for train data:

 for i=1:lentraindata

    [data{i},fs]=audioread(cell2mat(traindata.Files(i)));

 end

 % Adding noise to the train dataset to disrupt the audio signals

  for r=1:lentraindata

    noisydatatrain{r}= cell2mat(data(r))+ (randn(size(data(r)))*(0.02));

end

 % read the audio files for test data:

 lentestdata= length(testdata.Files);
```

```
 for j=1:lentestdata

     [dataTest{j},fs]=audioread(cell2mat(testdata.Files(j)));

 end

 % Adding noise to the test dataset to disrupt the audio signals

  for t=1:lentestdata

    noisydatatest{t}= cell2mat(dataTest(t))+ (randn(size(dataTest(t)))*(0.02));

end

 % extract the features for train dataset:

 aFE = audioFeatureExtractor( ...
   "SampleRate",fs, ...
   "Window",hamming(round(0.03*fs),"periodic"), ...
   "OverlapLength",round(0.02*fs), ...
   "SpectralDescriptorInput","barkSpectrum", ...
   "spectralCentroid",true, ...
   "spectralSlope",true);


featuresTrain=cellfun(@(x)extract(aFE,x),noisydatatrain,"UniformOutput",false);

Ytrain=traindata.Labels;
labelsValidation=testdata.Labels;

% equalizing the number of rows of the mfcc coeficcients of the train data:

lenFeaturesTrain= length(featuresTrain);

for k=1:lenFeaturesTrain

   Processfeatures=featuresTrain{k};
```

```
    Processfeatures=Processfeatures(1:2989,:);
    FeatTrain{k}= Processfeatures;
end
% Convert 1*n array to n*1 array:
FeatTrain=permute(FeatTrain,[2 1]);
% convert cells in the above array 800*1 with cells (800*13) into (13*800):
lenfeatTrain= length(FeatTrain);
for u=1:lenfeatTrain
    processfeatTrain= FeatTrain{u};
    processfeatTrain= permute(processfeatTrain,[2 1]);
    readyfeaturesTrain{u}= processfeatTrain;
end
readyfeaturesTrain= permute(readyfeaturesTrain,[2 1]);
%Extract features for Validation set:
featuresTest=cellfun(@(x)extract(aFE,x),noisydatatest,"UniformOutput",false);
featuresTest=permute(featuresTest,[2 1]);
% Convert the array of features to have only first 2990 coefficients.
lenfeaturesTest= length(featuresTest);
for n=1:lenfeaturesTest
```

```
    usefeatures=featuresTest{n}

    usefeatures=usefeatures(1:2989,:);

    FeatTest{n}=usefeatures;

end

%Convert the test features into 200*1 array instead of 1*200 cell:

FeatTest=permute(FeatTest,[2 1]);

% Convert cells in array 200*1 ( 800*13) to (13*800):

lenFeatTest= length(FeatTest);

for r=1:lenFeatTest

    useFeatTest= FeatTest{r};

    useFeatTest=permute(useFeatTest,[2 1]);

    readyfeatures{r}= useFeatTest;

end

readyfeatures=permute(readyfeatures,[2 1]);


%Define and Train the Network

layers = [ ...
    sequenceInputLayer(2)
    lstmLayer(125,"OutputMode","last")
    fullyConnectedLayer(numel(unique(Ytrain)))
    softmaxLayer
    classificationLayer];
```

```
options = trainingOptions("adam", ...
    "Shuffle","every-epoch", ...
    "ValidationData",{readyfeatures,labelsValidation}, ...
    "Plots","training-progress", ...
    "Verbose",false);

net = trainNetwork(readyfeaturesTrain,Ytrain,layers,options);
```