



School of Information Technology and  
Engineering at the  
ADA University



School of Engineering and Applied  
Science at the  
George Washington University

## MUSIC GENRE DETECTION USING DEEP LEARNING TECHNIQUES

A Thesis

Presented to the Graduate Program of Computer Science and Data Analytics  
of the School of Information Technology and Engineering  
ADA University

In Partial Fulfillment  
of the Requirements for the Degree  
Master of Science in Computer Science and Data Analytics  
ADA University

By  
Ilyas Karimov

April, 2022




## THESIS ACCEPTANCE

This Thesis by: Ilyas Karimov

Entitled: *Music Genre Detection Using Deep Learning Techniques*

has been approved as meeting the requirement for the Degree of Master of Science in Computer Science and Data Analytics of the School of Information Technology and Engineering, ADA University.

Approved:

Dr. Samir Rustamov (Adviser)		28.04.2022
Dr. Abzatdin Adamov (Program Director)		28.04.2022
Dr. Sencer Yeralan (Dean)		28.04.2022

## Table of Contents

<b>1</b>	<b>INTRODUCTION.....</b>	<b>8</b>
<b>1.1</b>	<b>PROBLEM STATEMENT .....</b>	<b>8</b>
<b>1.2</b>	<b>HISTORY.....</b>	<b>8</b>
<b>1.3</b>	<b>SIGNIFICANCE OF STUDY .....</b>	<b>9</b>
<b>1.4</b>	<b>LIMITATIONS OF STUDY .....</b>	<b>9</b>
<b>2</b>	<b>RESEARCH APPROACH.....</b>	<b>10</b>
<b>2.1</b>	<b>DATASET .....</b>	<b>10</b>
<b>2.2</b>	<b>UNDERSTANDING MUSIC.....</b>	<b>11</b>
<b>2.3</b>	<b>LITERATURE REVIEW.....</b>	<b>20</b>
<b>2.4</b>	<b>METHODOLOGY .....</b>	<b>29</b>
2.4.1	FEATURE EXTRACTION FOR THE FIRST SECTION.....	29
2.4.2	FEATURE EXTRACTION FOR THE SECOND SECTION.....	30
2.4.3	ALGORITHMS .....	31
<b>2.5</b>	<b>EXPERIMENTS .....</b>	<b>43</b>
2.5.1	FIRST SECTION (GTZAN DATA) .....	43
2.5.2	SECOND SECTION (CUSTOM DATA).....	44
2.5.3	USER INTERFACE .....	46
<b>2.6</b>	<b>TESTING .....</b>	<b>46</b>
<b>3</b>	<b>CONCLUSION .....</b>	<b>51</b>
<b>4</b>	<b>FUTURE WORK.....</b>	<b>51</b>

## ABSTRACT

The purpose of this study is to compare various methods for detecting the genre of music using machine learning algorithms. The findings are obtained using Neural Network, Convolutional Neural Network, and Recurrent Neural Network - Long Short-Term Memory, and Multi-Layer Perceptron algorithms on two distinct datasets. The first dataset is well-known in the field of music classification; it is named the GTZAN dataset and contains 1000 records. The second dataset is compiled by me and added to the GTZAN collection, which currently has over 2200 songs with vocals and instrumentals combined. To avoid confusing the model, I removed the first section of the MFCC features from the second dataset. For experiments, just MFCC features are utilized, and I write a script to choose the first 30 seconds, modify the bit rate from 41000 to 20500 and extension from .mp3 to .wav, and then extract MFCC features with different segments such as 1,3,6,10, 15 and 30. To avoid confusing the model, I removed the first section of the MFCC features from the second dataset. At the conclusion of the investigation, From the first section, the best result was 85% and 75% training and validation accuracies, which is obtained with only the usage of GTZAN Data. From the second section, I obtained a 62% F1 score and 80% and 62% training and validation accuracies, respectively, using MLP and CNN. However, the best results were obtained with CNN, with segments 15 and 30 achieving 72 and 71% accuracy, and 80% and 75% training and validation accuracy, respectively.

## LIST OF FIGURES

No	Figure Caption	Page
2.1	GTZAN Data	10
2.2	Dataset for 2nd experiment	11
2.3	Time-domain, Waveplot	12
2.4	Frequency domain	13
2.5	Short-time Fourier transform	15
2.6	STFT Visualization	16
2.7	Log-spectrogram in decibels	17
2.8	Mel Spectrogram	18
2.9	Roadmap of MFCC technique	19
2.10	Step-wise MFCC summary	20
2.11	Types of layers	33
2.12	Basic model calculations	34
2.13	Loss function	34
2.14	Sigmoid function	35
2.15	ReLU	36
2.16	CNN steps	37
2.17	Activation Map	37
2.18	Pooling operation	38
2.19	Slide of the Kernel over the Picture	39
2.20	Slide of the Kernel over 3D data	39
2.21	RNN and Feed-Forward NN demonstration	41
2.22	LSTM gates	42
2.23	User Interface	43
2.24	Result	46
2.25	Result in-detail	46
2.26	Result	47
2.27	Result in-detail	47
2.28	Result for “going under” and “can’t find the time”	48
2.29	Result for “barcarole” and bensound	49

## LIST OF TABLES

No	Figure Caption	Page
----	----------------	------

## LIST OF ABBREVIATIONS

Abbreviation	Explanation
MIR	Music Information Retrieval
FFT	Fast Fourier Transform
SFT	Standard Fourier Transform
STFT	Short-time Fourier Transform
dB	Decibels
MFCC	Mel-Frequency Cepstral Coefficients
SVM	Support Vector Machine
C	Chorus
M	Music Box
R	Rock
P	Piano
D	Disco Music
MLP	Multi-Layer Perceptron
LDA	Linear Discriminant Analysis
MSD	Million Song Dataset
MTT	MagnaTagaTune dataset
KNN	K-Nearest Neighbour
ROC-AUC	Receiver Operating Characteristics, Area Under the Curve
PR-AUC	Performance Ratio, Area Under the Curve
KL	Kullback-Liebr
FMT	Fourier-Mellin 2D
GMMs	Gaussian Mixture models
BH	Beat Histogram
PH	Pitch Histograms
MIDI	Music Instrument Digital Interface
VA	Valence-Arousal
MLR	Multiple Linear Regression
SVR	Support Vector Regression
ReLU	Rectified Linear Unit
AI	Artificial Intelligence
GPUs	Graphics Processing Units
CNN	Convolutional Neural Network
MRI	Magnetic Resonance Imaging
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory

**Additional Keywords and Phrases:** automatic music genre detection, music genre classification using deep learning, MFCC features

## 1 INTRODUCTION

### 1.1 Problem Statement

Music affects our language, intelligence, mood, mental clarity, caring, and relaxation, among other things, and research indicates that these effects can be both positive and negative [1].

The term "genre" refers to a type of music traditionally used to designate music as belonging to a particular style or tradition [2]. A basic part of music information retrieval systems (MIR) is music genre categorization, which has grown in relevance and attention with the rise of digital music and yet there has been sufficient progress in the field of automatically classifying musical genres, and the stated classification accuracy is also poor [3]. More than 500 studies on genre classification have been done over the past two decades, beginning in 1994, with varying degrees of accuracy [4].

Despite the fact that genres are already available on the Apple Music and Spotify music platforms for consumers to browse through, song recommendations are based on category tags, and each song can be classified into numerous genres. Nonetheless, for the purposes of this study, I will only take into consideration music that falls into a particular musical genre. In this study, a number of parametric machine learning approaches that were solely based on MFCC features were employed to identify and classify objects.

### 1.2 History

The idea of investigating automatic genre categorization began when music professionals discovered that classifying music into genres was becoming an inordinately time-consuming task [5]. The development of automatic categorization systems became increasingly popular among academics as a result of this [5]. It was necessary to manually categorize a portion of the music library's collection into distinct genres throughout the 1980s. On the other hand, with the introduction of MP3 players onto the market, the classification process underwent a significant transformation. As a result of this invention, which was first launched in the 1990s, Internet storage and bandwidth were more inexpensive than they had ever been [5]. Those moderately priced MP3 devices also allowed for the listening of other commercial music collections and categorizing the music tracks into several categories.

Princeton University researchers George Tzanetakis and Perry Cook published their paper "Musical genre classification of audio signals" in 2002 [6], and they are experts in the field of music genre classification. Their piece was one of the very first to be published on this topic. After this article was published, additional study was carried out on the subject to contribute to its knowledge base. It is possible that they have utilized some of the same research methodologies in other studies as well,



simply for the sake of comparing the outcomes. It has been discovered that other researchers have come up with concepts that are as excellent as or better than their own.

### **1.3 Significance of Study**

The task of automatically classifying music is not a simple one, there are few exact, obvious, and consistent rules for categorizing music by genre or location, making it challenging for both people and computers to categorize music in this way [18]. As it turns out, even establishing a taxonomic framework can be a challenge and considering the measurements for musical, similarity can also be difficult to define and implement and it is generally difficult to categorize music because of these ambiguities [18]. It is imperative that we define music genres, as they are one of the few and most useful instruments, we have at our disposal for appreciating and debating the work of musicians [19]. Using these classifications in a flexible and descriptive manner rather than as a form of rigid division can significantly improve our understanding, recognition, and pleasure of the music that we hear [19]. It is not enough to judge a musical genre purely on the basis of its commercial viability to overlook its significance as a means of categorizing and studying music [20]. The paper [20] mentions that many fans of death metal or rap dress and speak in ways that are distinct from those of other music genres, such as metalcore or hip-hop, meaning when it comes to music it's all about the genre. Psychological studies have indicated that a composition's style can impact listeners' liking for it more than the piece itself [21]. Music appreciation and cognition are heavily influenced by categorization in general, according to additional psychological study [22]. Musicologists and theorists alike can benefit greatly from advances in automatic genre classification [23]. Genre research that links cultural and content-based traits or uses ontologies that may effectively map genre interrelationships can also have significant musicological importance [20].

### **1.4 Limitations of Study**

Training efficient genre classifiers necessitates the acquisition of credible data [20]. Automatic genre categorization has been shown to be hampered by human annotators' inability to reach universal agreement on the classification of musical genres. In addition to a wide range of opinions on how to classify a certain song, there is also a wide range of choices when it comes to genre labels [20]. Few genres have precise definitions, and the information that is given is typically vague and inconsistent from one source to the other [20]. Genres typically cross over significantly, and a single recording can, to varied degrees, be found in more than one and genre ties can be complicated, and some genres are more expansive than others [20]. Because of this, genres typically contain numerous distinct clusters within them [20].

## 2 RESEARCH APPROACH

### 2.1 Dataset

This research is generally separated into two portions; the first section featured tests using the original data, which was the GTZAN data [7, 8]. The second section involved studies using the modified data.

The GTZAN data collection contains photos of the spectrograms of the music tracks, two CSV files, and audio files, among other things. The features of the audio files are contained within these CSV files. For each 30-second-long song, a mean and variance are computed over different features that can be retrieved from an audio source and stored in a single file and the secondary file has the same format as the first, but the songs have been divided into three-second audio files [8]. This way, the amount of data fed to the classification model will be increased 10 times. When it comes to data, more is almost always better.

Generally speaking, the dataset is divided into 10 genres: rock, reggae, pop, metal, jazz, hip-hop, disco, country, classical, and blues, to name a few examples. Each genre is represented by 100 tracks, for a total of 1000 tracks in the dataset overall. Each track is 30 seconds in length and has a resolution of 16 bits (*Fig 2.1*). During the data processing process, I realized that each recording is completely empty of voices. Personally, I did not use all of the data that was made available for my research, but rather only the tracks.

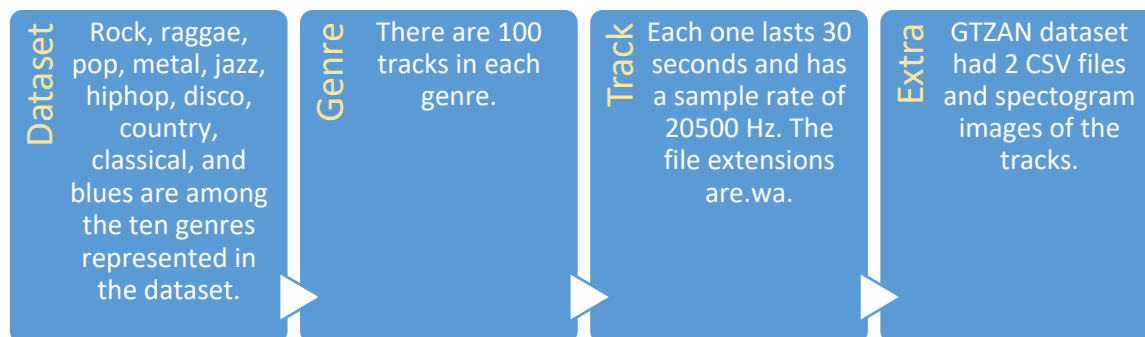


Figure 2.1. GTZAN Data

After that, the GTZAN dataset was used in conjunction with a few selected songs for the second part of this investigation. Among the 11 genres included in the collection are rock, reggae, pop, metal, jazz, hip-hop, disco, country, classical, blues, and mugham. The collection has 2200 records from 11 different genres. Additionally, I added the top 100 songs from each genre from 2000 to 2020, as well as newer songs from each existing genre, to create the collection. This means that I have more than doubled the amount of data available by including one additional genre in the mix.

The new tracks that have been added to the genres comprise both music and vocals. I was under the impression that a more varied and diversified dataset would lead to more accuracy, therefore I included songs with vocals in half of the songs from the other genres. The songs I acquired all had.mp3 extensions and were preserved in their entirety. Additionally, they include a 41000 Hz sampling rate. Later, I have written a python script to convert them to 30-second long 20500 Hz sampling rate .wav tracks (Fig 2.2).

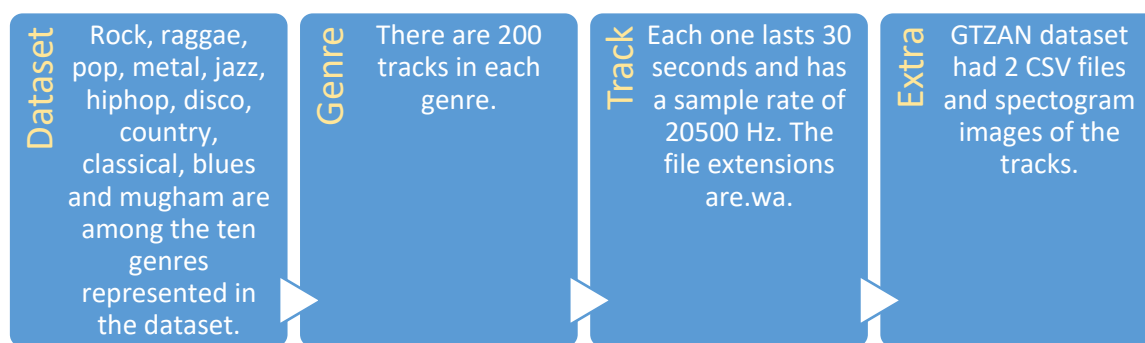


Figure 2.2 Dataset for 2<sup>nd</sup> experiment

One additional genre I have added, which is called Mugham. As a contrast to tasnif and ashik, mugham (also known as Muğam in Azerbaijani) is one of the numerous classical compositions from Azerbaijan that may be heard [9]. In addition to scales, "mugham" modes are related with a library of melodies and melodic fragments that performers employ in the course of improvisation, and Mugham is a composite composition consisting of several elements [10]. The selection of a certain mugham as well as the style of performance is tailored to the occasion [10]. It is commonly related with growing intensity and rising pitches in performance, and it serves as a type of poetic-musical communication between performers and initiated audiences [10].

## 2.2 Understanding music

The examples provided here is based on the first blues track called "blues0000.wav". For the analysis of the music data, I utilized the librosa library. Librosa is a music and audio analysis Python package and to design music information retrieval systems, it supplies the necessary building blocks [11].

The waveform is used to calculate the amplitude of a signal as a function of time and in turn, amplitude quantifies vibration, which generates audible noises (Fig 2.3.) [12]. Most of the time, we're looking at time domain waveforms because they represent sound as a single, continuous vibration, making a variety of audio operations like chopping, reordering, and fading much simpler than they would be with frequency domain waveforms, which are typically displayed as voltage on analog synthesizers and mixing consoles and as values between -1 and 1 in Ableton Live and Logic [12]. We can find time and amplitudes on x and y axis.

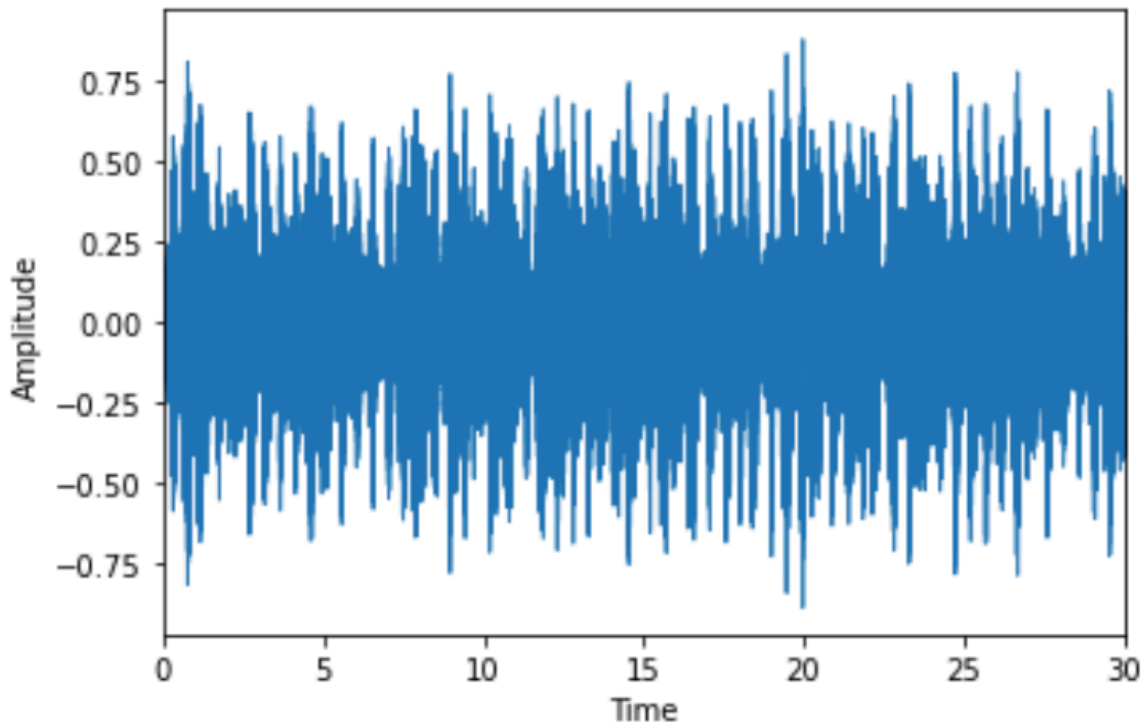


Figure 2.3. Time-domain, Waveplot

With the use of Fourier transforms, the frequency domain is broken down into a collection of discrete frequencies, which is based on the idea that any sound may be described mathematically as the sum of a large number of sine waves (*Fig 2.4.*) [12]. With the use of the Fourier Transform, we can see that any waveform can be rewritten as the sum of its sine and cosine functions, demonstrating that the waveform itself may be thought of as a function or signal having the form a sine curve [13]. Additive synthesizers may be thought of as pushing additive synthesis to the utmost, since a sawtooth wave from a synthesizer can be broken down into the same building pieces (sine waves) as a dog bark and when you use a plugin like Ableton's Spectrum or one of several equalization plugins that display the frequency spectrum of incoming audio, you're actually operating in this dimension [12].

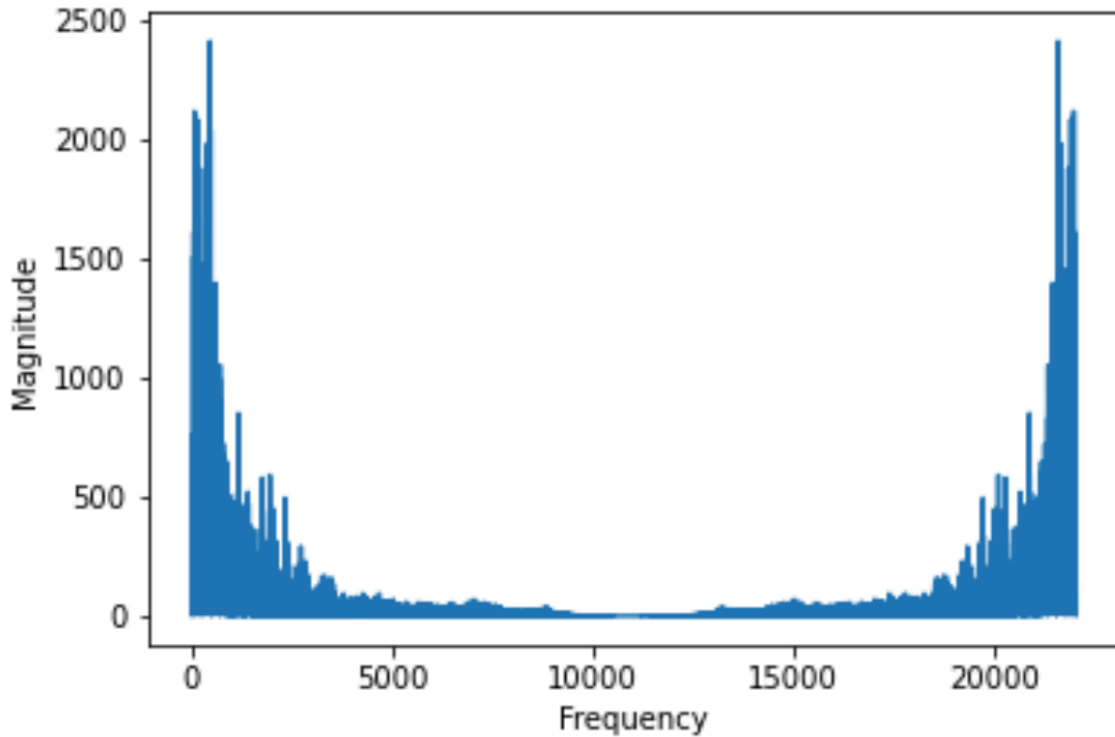


Figure 2.4. Frequency-domain

Taking into account that this graph is symmetric in half, I divided the length of magnitude and frequency by two for the other half of the graph.

Discrete Fourier transform (DFT)  $X[k]$  of an  $N$ -point signal  $x[n]$  is as follows [14]:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk} \quad k = 0, 1, \dots, N-1$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-nk} \quad n = 0, 1, \dots, N-1$$

Where

$$W_N = e^{-j2\pi/N}$$

These transform equations require  $N$  complex multiplications and  $N-1$  complex additions for each term and  $N-1$  complex multiplications for all  $N$  terms [14]. The symmetry properties of DFT can be used to implement in a fast or real-time manner using a fast Fourier transform (FFT) algorithm [14].

Finding a fast DFT implementation can be accomplished in a variety of ways, namely by experimenting with various FFT algorithms and one of them as follows [14]:

$$x_1[n] = g[2n] \quad 0 \leq n \leq N - 1$$

$$x_2[n] = g[2n + 1]$$

$x_1[n]$  and  $x_2[n]$  are two new N-point signals from a 2N-point signal  $g[n]$  with the help of splitting the odd and even parts as above [14]. The overall formula  $x[n]$  will be [14]:

$$x[n] = x_1[n] + jx_2[n] \quad 0 \leq n \leq N - 1$$

To DFT of  $g[n]$ , which is  $G[k]$ , the equation will be [14]:

$$G[k] = X[k] + jX_2[k] \quad 0 \leq k \leq N - 1$$

$$k = 0, 1, \dots, N - 1, \text{ with } X[N] = X[0]$$

and this equation is used, where

$$A[k] = \frac{1}{2}(1 - jW_{2N}^k)$$

and

$$B[k] = \frac{1}{2}(1 + jW_{2N}^k) \quad (9)$$

The complex conjugate property of  $G[k]$ ,  $G[2N - k] = G^*[k]$  is used to find the remaining points [14].

Fourier transformations of a windowed signal are used in the short-time Fourier transform (STFT) [14]. A standard Fourier transform (SFT) offers frequency information averaged over the signal time interval, but an STFT delivers frequency information that is time-localized [14]. STFT is given by these formulas:

$$X_{STFT}[m, n] = \sum_{k=0}^{L-1} x[k]g[k - m]e^{-\frac{j2\pi nk}{L}} \quad (10)$$

$$x[k] = \sum_m \sum_n X_{STFT}[m, n]g[k - m]e^{-j2\pi nk/L} \quad (11)$$

a signal is represented by  $x[k]$ , while  $g[k]$  is an L-point window function and The Fourier transform of the product  $x[k]g[k - m]$  can be understood as the STFT of  $x[k]$  [14]. (Kehtarnavaz, 2008) mentions that a windowed signal's Fourier transforms are used to computer STFT:

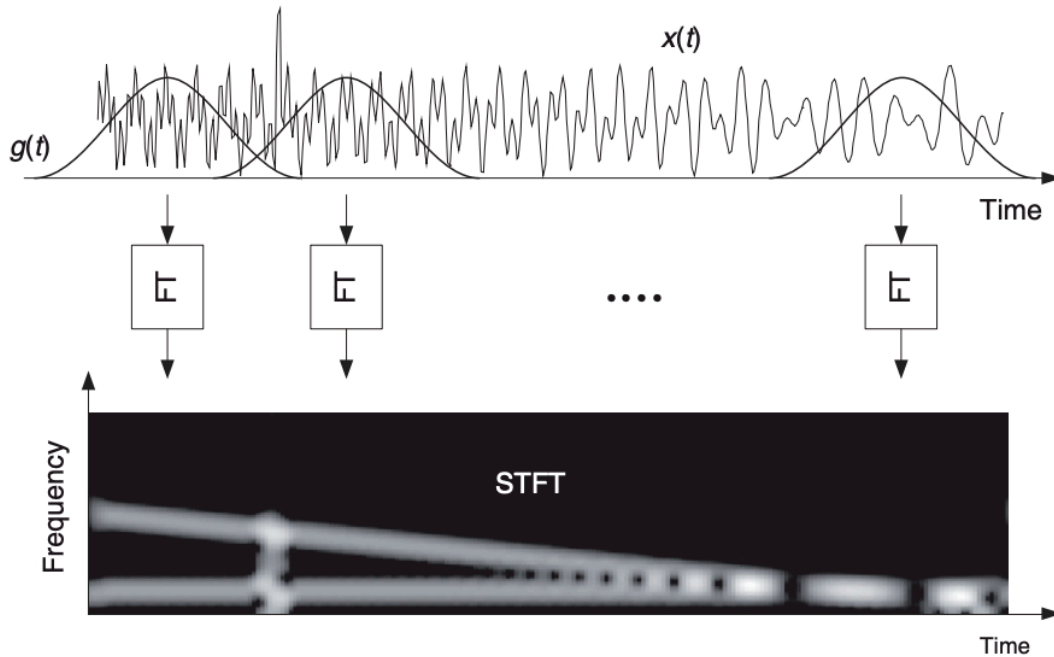


Figure 2.5. Short-time Fourier transform

STFT's time and frequency resolutions are mutually exclusive [14]. The resolution in the time domain is better, but the resolution in the frequency domain is less when using a narrow-width window, and vice versa [14]. This can be done by looking at the spectrogram, which is a plot of STFT intensity with time [14]. When the STFT processor moves the window function  $W[n]$  along the signal  $x[n]$  according to the hop length, it performs the FFT operation on samples within the window, which monitors the frequency over time [16]. The overall process is shown down below [16]:

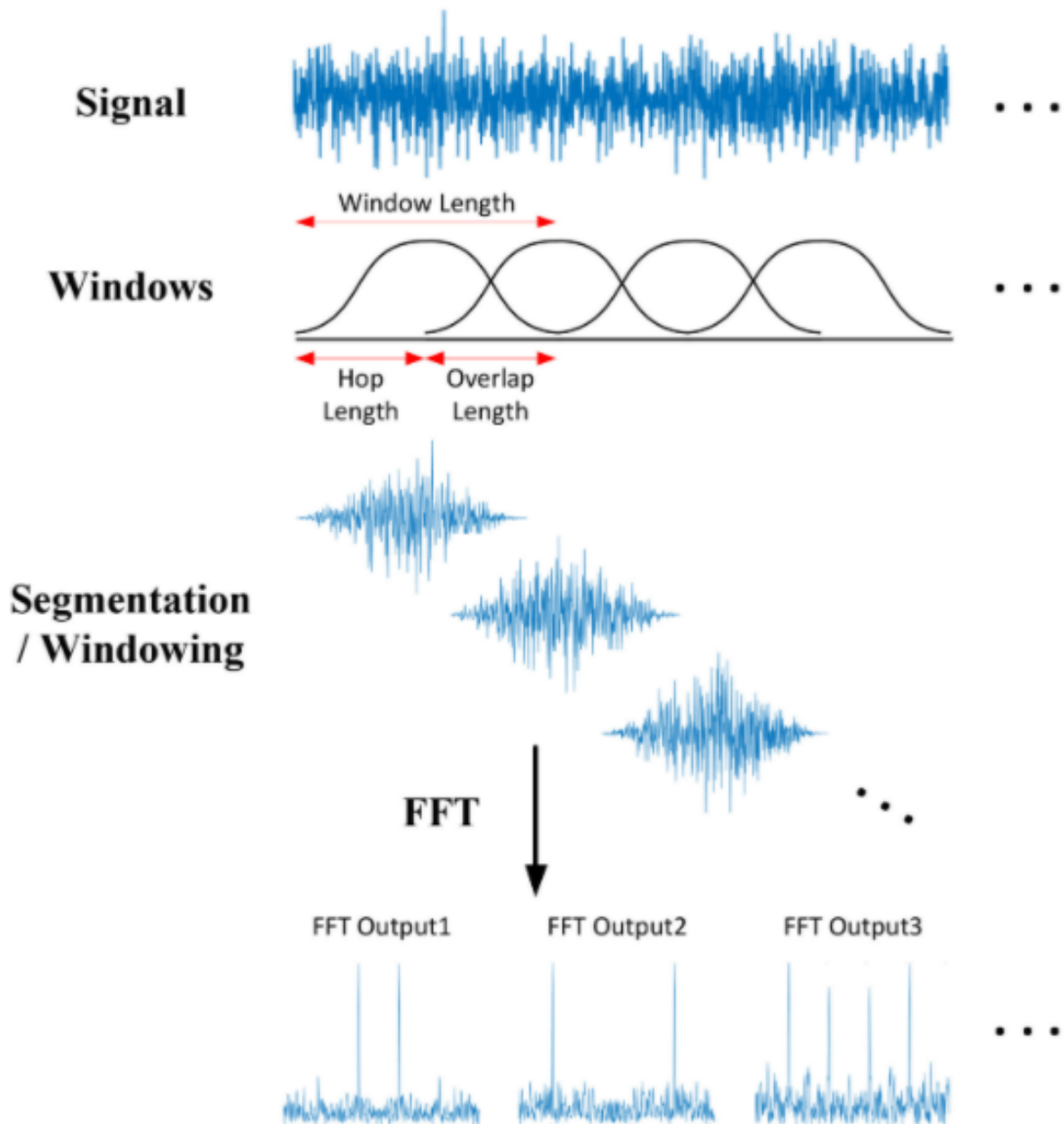


Figure 2.6. STFT Visualization [16]

Afterwards, it was time to understand spectrograms for which I utilized the librosa library's stft (short time fourier transform) function. `n_fft` parameter is the number of samples we're considering for a single fft, `window_size` (window size) is the size of the window we're considering for executing a



single fft, and hop length is how much we're moving each fourier transform to the right. The non-intersecting section of window length is known as hop length [17].

The window size is a measure of both the number of samples and the length of time, and it is the most important metric in the study [15]. The signal's fundamental frequency, strength, and rate of change all influence the window's size and because the FFT size defines how many frequency bands will be reduced to set the frequency resolution, it is a direct consequence of the principles of the Fourier series [15]. The resolution of the analysis is affected by the size of the window [15]. When viewing the spectrogram, a colorbar in decibels (dB) is displayed, which represents the logarithmic function of the magnitude [15] (Fig 2.7.).

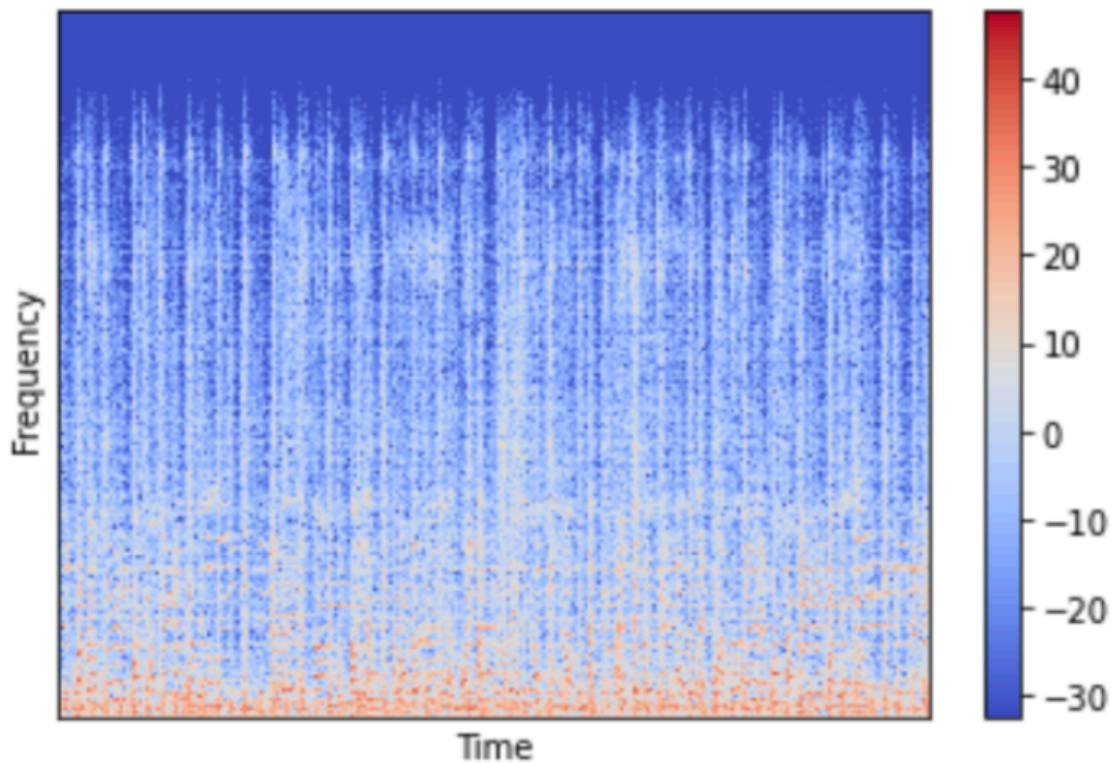


Figure 2.7. Log-spectrogram in decibels

When frequencies are translated into the Mel scale, the result is a Mel-spectrogram [17]. Visualization of the mel-frequency cepstral coefficients (MFCC) and time dependency could be found down below (Fig 2.8).

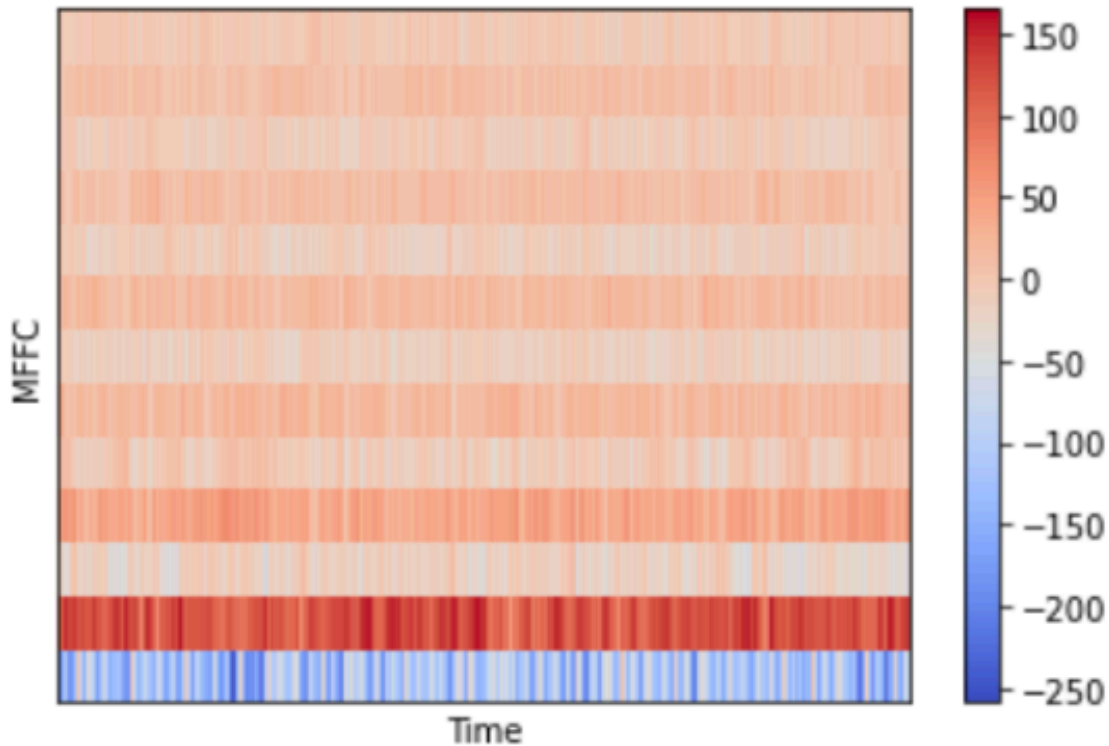


Figure 2.8. Mel Spectrogram

When developing an automated speech recognition system, the first step is to extract features, which is to say, identify the components of the audio signal that are good for deciphering the linguistic content and discard the rest [35]. Automated speech and speaker detection relies heavily on a feature known as Mel Frequency Cepstral Coefficients (MFCCs) [35]. LPCs and LPCC were the major feature type for automatic speech recognition prior to MFCC debut [35]. MFCC introduced a new type of LPC and LPCC (ASR). The steps are given down below how to compute MFCCs towards the end [35] (*Fig 2.9.*):

1. Organize the signal into a series of brief clips.
2. The periodogram estimate of the power spectrum is calculated for each frame.
3. Apply the mel filterbank to the power spectra and add the energy in each filter bank
4. Logarithmize all filterbank energies.
5. Perform a DCT of the log filterbank energy logs.
6. Keep coefficients 2-13 of the DCT and discard the rest of the coefficients.

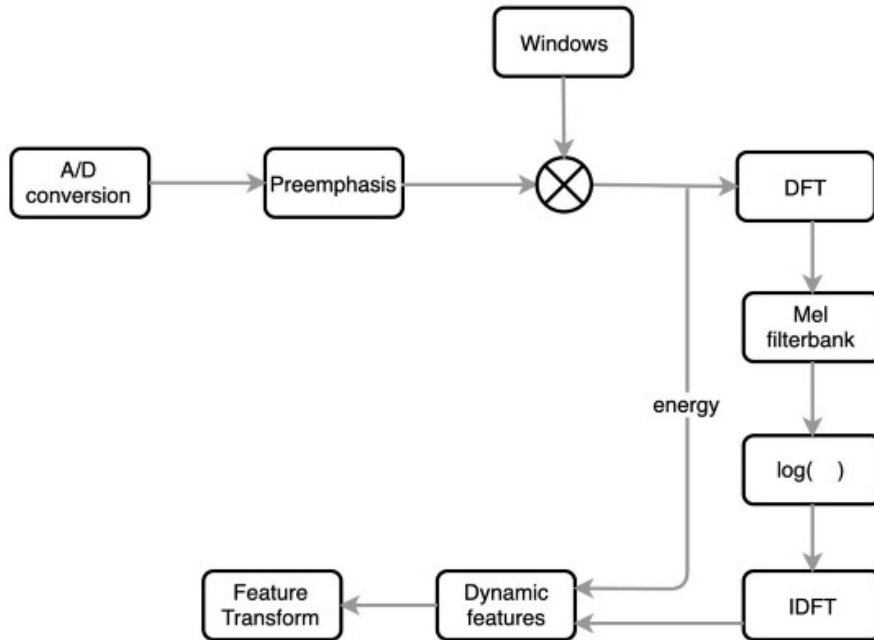


Figure 2.9 Roadmap of MFCC technique

For extracting audio features, MFCC is the most used method [34]. The shape of a person's vocal tract determines the sound they make (including tongue, teeth, etc) [33]. Any sound produced can be adequately described if this shape can be appropriately determined and the vocal tract is precisely represented by the envelope of the speech signal's temporal power spectrum, and MFCC (which is nothing more than the Mel-frequency cepstrum coefficients) accurately depicts this envelope [33].

The way we hear sounds differs from how machines hear them because at lower frequencies, our ears are able to hear more clearly than at higher frequencies [35]. Even though there is only a 100-hertz difference between the noises at 200 Hz and 300 Hz, we can tell them very easily from the ones at 1500 Hz and 1600 Hz and as opposed to this, the resolution of the machine is constant across all frequencies [35]. If you include the ability to hear as a feature in your model, it is likely to perform better [35].

As a result, we'll use the mel-scale to convert the real frequency to a range that humans can perceive as pleasing to the ear [35].

The formula below can be used to convert a frequency expressed in Hertz ( $f$ ) to Mel units [34]:

$$mel(f) = 1127 \ln\left(1 + \frac{f}{700}\right) \quad (12)$$

The MFCC model takes the top 12 coefficients of the signal after applying the IDFT procedures and using the result, we may compute the inverse transform in this step as well [34]. Along with the 12 coefficients, it will use the energy of the signal sample as the feature, which makes 13 features total and the formula for energy could be found down below [34]:

$$Energy = \sum_{t=t_1}^{t_2} x^2[t] \quad (13)$$

The MFCC method also considers the features' first and second order derivatives, resulting in a total of an additional 26 features in the final analysis [34]. The difference in these coefficients between the audio signal samples is used to construct derivatives, which aid in deciphering the transition [34].

Using the MFCC approach, each audio signal sample yields 39 characteristics that are fed into the speech recognition model [34]. The picture shows the overall process we get through to get MFCC features [33]:

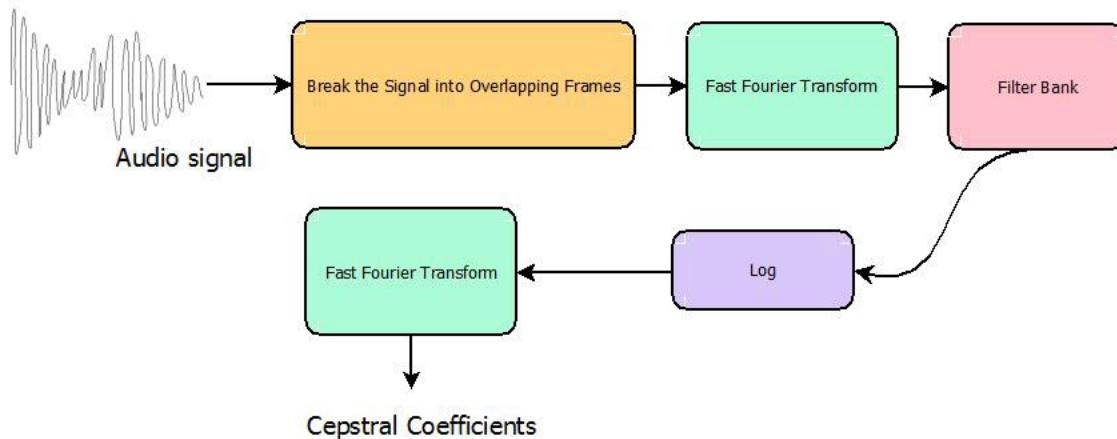


Figure 2.10 Step-wise MFCC summary

### 2.3 Literature Review

This research study [24] provides a straightforward way for music summarization, namely by extracting summary information from various segments of the music data. A support vector machine was proposed in the usual way (SVM). Mel-frequency cepstral coefficients (MFCC) are frequently used as SVM vectors to categorize musical genres. While melody is critical in determining the type of music, it is inconvenient as a feature vector. Because genres are associated with timbre, which is connected to the frequency aspects of sound signals, prior research have made use of MFCC features. Genres are intrinsically linked to the timbre of musical data, which is determined by the frequency qualities of sound impulses.

The paper [24] states that the typical method divides the music data into small frames in order to create vectors of MFCC features. Similar frames are classified as the same genres, and the most

common label is chosen as the summary, which eliminates any potential hints to the accuracy. The proposed strategy is easily implemented by applying the conventional method to various parts of musical material. Each summary is assumed to be equal in length, which means that when two summaries N1 and N2 are added together, the total number of summaries is two. Despite the fact that the preceding strategies used Kullback-Leibler to determine the distances between clusters, the k-means algorithm was chosen. Their dataset contains the following genres: Disco Music (D), Chorus (C), Music Box (M), Rock (R), and Piano (P), which correspond to vocal sounds with instruments (bass and drum), vocal sounds without instruments, instrumental music without vocal sound, vocal sound with musical instruments (bass, guitar, and drum), and music from the piano instrument. There are 350 tracks in total, with each genre featuring 70 16-bit songs at a sample rate of 44.1kHz. They constructed an MFCC in 20 dimensions using 50 ms frame lengths as vectors. The training was validated tenfold. They divided data into training and testing segments in an 85:15 ratio. D, C, M, R, and P have F-values of 92, 99, 98, 93, and 97, respectively. The scores indicate that the accuracy is increased when a single portion of music is used rather than many sections. Later, they compared summaries from the standard and new methods, which had a total duration of 5 and 30 seconds, respectively. Three portions of music were extracted for this paper's methodology. They chose two alternative ratios to allow for a better comparison of the results: one with a 1:1:1 ratio and another with a 5:3:2 ratio. Even though they obtained an F-measure less than 96 percent, the average F-measure for all trials, this indicates that accuracy is not significantly decreased when smaller chunks of music data are put to the task of music genre categorization rather than the entire piece of music. According to research, their strategy almost always outperforms the traditional method.

This study [25] focuses not only on the classification of musical genres, but also on the selection of new music depending on an individual's interests. This is accomplished by the application of digital signal processing and autoencoders. According to the paper, each genre has its own distinct acoustic characteristics, such as rhythm, timbre, density, and pitch, which simplifies genre classification. The Mel Frequency Cepstral Coefficients (MFCC) features were employed to train the CNN-based autoencoder, together with the features of latent space vectors generated by the trained autoencoders. They used the GTZAN dataset, which has 100 tracks with a duration of 30 seconds each. They underwent three distinct processes, including digital signal processing, classification using machine learning techniques, and autoencoder usage. They expressly used the first method for feature extraction, utilizing the PyWavelet and librosa libraries, and extracted 606 features, including the Zero Crossing Rate, Spectral Contrast, Spectral Flatness, Spectral Bandwidth, Spectral Rolloff, Root Mean Square Energy, Mel Frequency Cepstral coefficients, Chroma STFT, Chroma SQT, Tonnetz, Polynomial Features, and Wavelet Transform. Later on, they utilized machine learning methods such as Multilayer Perceptron, Logistic Regression, Random Forest, Linear Discriminant Analysis (LDA), K Nearest Neighbor, SVM, Naive Bayes, and Gradient Boosting. They chose a segment count of six during the MFCC extraction process, and because each track is 30 seconds in length, each segment is 5 seconds in length. They obtained 13x259 as the vector size by increasing the window size and hop length to 2048 and 512, respectively. Following the creation of the autoencoder model, MusicRecNet-like features were utilized. They chose mean-squared error as their loss function in order to achieve a low loss rate when reconstructing images. They employed sigmoid and up sampling layers rather than maximum pooling layers for the activation and decoder. They

may observe how compression affects the findings by using latent space sizes of 64, 128, 256, 512, and 1024. They selected 5128, 570, and 300 random samples for training, testing, and validation, respectively, during the training process. Following the feeding of the dataset to the model, space layer feature vectors are retrieved to prepare for music classification and clustering methods.

They [25] utilized a tenfold cross validation on eight various classification methods and found that the SVM algorithms had the highest accuracy, with an average of 81 percent and a maximum of 88 percent. The Naive Bayes algorithm achieved the lowest accuracy, with an average and maximum of 64.7 and 73 percent, respectively. LDA, on the other hand, displayed the highest accuracy when solely MFCC characteristics are used for classification, with average and maximum accuracies of 75.3 percent and 84 percent, respectively. By comparing the accuracy with the autoencoder approach, the paper concludes that the latent size had no effect on the accuracy. MLP demonstrated the highest accuracy, with an average of 42% and 43% with 512 latent sizes, respectively, showing that autoencoders were ineffective. On the other hand, when it came to the recommendation system, it was recommending various genres.

This investigation [26] was conducted using a classification system for western music. They have mostly concentrated on eleven genres: classical, electronic, R&B, blues, jazz, metal, country, folk, pop, rap, and rock. The article discusses the significance of music auto-tagging on mood and believes it to be a work of categorization and instrument identification. They suggest a transfer learning strategy in this study in response to the inability of the top 50 characteristics to recognize music genres, such as classical music in the Million Song Dataset (MSD) and blues in the MagnaTagATune Dataset (MTT). Transfer learning consists of two components: source and target tasks. After modifying the network to a more particular dataset, the trained neural network model may be reused in the source job, making it accessible for regression and classification tasks. The report illustrates one of the studies by demonstrating how pre-trained CNNs were trained on a large synthesis dataset. According to the article, these features were used in the target task of acoustic piano pedal-on/off classification. For audio tagging, the muscinn library was utilized, which includes several pre-trained CNN models, as well as MTT\_muscinn and two MSD models, namely MSD\_muscinn and MSD\_muscinn\_big. The first model was trained on the MagnaTagATune dataset, while the second models were trained on the Million Song Dataset. Although the architecture of the models is identical, alternative 50-tag vocabularies were evaluated since the datasets used to train these models are different.

Two methods were proposed to adjust the pre-trained models to the target objective, namely the use of MTT\_muscinn and MSD\_muscinn models during feature extraction and the modification of the MSD\_muscinn or MSD\_muscinn\_big model's dense layer to 11 genres rather than 50 tags as during initialization using the model's initialized weights [26]. There are 1100 songs in the dataset, with 100 songs in each genre, 75% of which have been divided for training purposes and the remainder for testing. They sampled four random segments from each song for each epoch, each part lasting three seconds. In the baseline approach, the k-nearest neighbors algorithm (KNN) was utilized as the machine learning model for classification, and parameter optimization was accomplished via grid-search. The prior study's CNN setups were retained. The baseline technique was entitled

Features+KNN, whilst the other method's two models were dubbed CNN MSD and CNN\_MSD big. To measure the performance of three models, ROC-AUC (Receiver Operating Characteristics, Area Under the Curve) and PR-AUC (Performance Ratio, Area Under the Curve) were used (Precision Recall, Area Under the Curve). The AUC of a model with 100% erroneous predictions is 0.0, while the AUC of a model with 100% accurate predictions is 1.0. The greatest performance on evaluation data was obtained when CNN MSD big was used, with ROC-AUC and PR-AUC values of 0.9799 and 0.8938, respectively. Additionally, they conducted a song-by-song analysis. The ultimate genre of the song is selected by majority vote, and they achieved an overall accuracy of 0.8836 percent. The article finds that most easy-to-classify genres are rap, classical, metal, and jazz and concludes that pop and R&B are the most often misclassified genres.

The purpose of this research study [27] is to examine techniques for automatically detecting the genre of music. They argued for experimenting with different components of audio signals' musical surface and rhythmic structure. The term "musical surface" refers to the texture, timbre, and instrumentation of tracks. According to the study, they presented a nine-dimensional feature vector that comprises mean-Centroid, mean-RollOff, mean-Flux, mean-ZeroCrossings, standard-Centroid, standard-RollOff, standard-Flux, standard-ZeroCrossings, and LowEnergy. The aforementioned features may be determined by calculating the mean and standard deviations of a second's worth of "texture" windows, each of which consists of 40 "analysis" windows of 20 milliseconds. Totally, there are 512 samples at a sampling rate of 22050 Hz. The computations were performed using the Short Time Fourier Transform, which can be found simply using the Fast Fourier Transform. Centroid, Rolloff and Flux are the measures of spectral brightness, shape and change respectively. ZeroCrossing is helpful for detecting the level of noise in the signal, and LowEnergy represents the proportion of "analysis" windows with a lower energy consumption than the average energy consumption of "analysis" windows in comparison to "texture" windows. When it comes to the rhythmic structure, it is based on the Wavelet Transform, which analyzes the signals. The Discrete Wavelet Transform is similar to WT with the case of compact representation of signal in time and frequency. The difference between STFT and DWT is that the latter provides high time resolution and low frequency resolution for all frequencies. The DWT is utilized to compute the octave decomposition of the frequency in this case, which is accomplished by the use of full wave rectification, low pass filtering, and down sampling, followed by the computation of the autocorrelation function.

For the classifications, they used data from radio, internet, and disks, and they used a Gaussian classifier with 50 tracks as the training dataset, each lasting 30 seconds [27]. The parameters of each class's multidimensional Gaussian distributions are estimated using the training data. Because there are 15 genres and each genre comprises 50 tracks that are 30 seconds long, the dataset is 6.25 hours long in total. Because they are based on human perception, MFCCs are often used in voice recognition investigations. In this study, the mean and standard deviation of the first five MFCC values were computed using a larger texture window of 1 second. They employed six genres in total for the classification: classic, country, disco, hiphop, jazz, and rock. They've also divided the classic music into four categories: choral, orchestral, piano, and string quartet. Hiphop and classic, with 90 percent and 86 percent accuracy, respectively, were the most accurate genres. Despite the fact that

jazz and rock appear to have distinct genres, the findings are 37 percent and 48 percent, respectively. It turns out that, despite the fact that the genres sound different, according to their model, jazz is most often confused with rock (27 percent). When it comes to classical music classification, choral has proven to be the most accurate, with a 99 percent accuracy rate. Following that are string 4tet, piano, and orchestral, which receive 80 percent, 75 percent, and 53 percent of the vote, respectively.

Due to the heterogeneity of music genres, it is exceedingly difficult to classify them even with the bear ear. This research [28] was conducted in order to classify four major musical genres: pop, classical, metal, and jazz. They processed the audio using an application called Marsyas, which is an open-source software piece of software. Additionally, they have included a database of songs dubbed "GTZAN Genre Collection" in their references. This database contains a total of 1000 songs, 100 of each genre, with each audio file being 16-bit. They've gathered songs from ten different genres that are 30 seconds long and have the music extension .wav. The files have a sample rate of 22050 Hz. The reason for choosing four major genres is that the success rate decreases as the number of genres increases, as indicated in the article. They used 70% of the 400 songs for training and 30% for testing. However, they have pre-processed the music into a.csv file using Python scripts. "The process begins by reading the.csv file into Matlab and extracting the MFCC features for each covariance matrix of the cepstral features, which are then stored as a mel-matrix, effectively modeling the frequency features of each song as a multi-variate Gaussian distribution" REFERENCE VER. Their methodology entails reading the first half of the files as waveforms and then extracting frames at intervals of 20 milliseconds. The Fourier Transform is obtained by multiplying a hamming window and a frame. The following step is to convert frequencies to the mel scale. Humans perceive pitch shifts as linear below 1 kHz and logarithmic above 1 kHz, thus this model simulates this. To arrange the frequency components in a more logical order, they use the Discrete Cosine Transform to approximate the Karhunen-Loeve Transform. Higher frequencies are features that make negligible differences to human perception and offer less information about the music, thus they maintain the top 15 of these 20 frequencies. They used Mel Frequency Cepstral Coefficients (MFCC) to describe their data and apply machine learning algorithms, as earlier research had suggested. By the representation of waveform as a matrix of cepstral features gives them a vector of 15 cepstral frequencies for the number of frames exists in the song.

They further compress this matrix representation by saving the mean vector and covariance matrix of the cepstral characteristics across each 20ms frame as a cell matrix [28]. Modeling the frequencies as a multivariate Gaussian distribution further reduced the processing needs for KL Divergence comparisons. The Kullback-Lieber (KL) Divergence is a crucial computation in their k-NN training that is used to determine the distance between two songs, the article mentions. It's worth noting that they obtain four-dimensional standard orthonormal base vectors with each value signifying a genre (classical, jazz, metal, or pop) and each value having a value of either 1 or 0. They preprocess the input data by combining the mean vector and the top half of the covariance matrix into a single feature vector, stemming in  $15 + (15 + 1) * 15/2$  features for each song. The proportions of training, validation, and testing data are 70, 15, and 15, respectively. They obtain the highest accuracy of 97 percent with classical and pop music and the lowest accuracy of 67 percent with jazz music using DAG SVM. The greatest score obtained using the k-Means algorithm is 93 percent accuracy for



Metal, while the poorest score obtained using the k-Means algorithm is approximately half for Jazz. Other genres fared rather well, with higher than or equal to 88 percent. Jazz is the most difficult genre to detect using k-NN, with an accuracy of nearly two-thirds. Other accuracies above 80. Finally, but certainly not least, NN outperformed all other models. Accuracy for Jazz and Pop was 100 percent, but Metal and Classical were 76 and 88 percent, respectively, making metal the most challenging genre. To further conduct their research, they have transferred images to genres. They have collected images that seemed similar visually, e.g., nature images for the classification of the classical tracks. The input data for NN was pre-processed by merging the mean vector and the upper half of the covariance matrix into a single feature vector. They've also taken the output data and turned it into vectors. For training, validation, and testing, the data was split 70%, 15%, and 15%, respectively. Features extracted from the images helped to transfer them to Fourier-Mellin 2D (FMT). By using the k-Means clustering with the data obtained from FMT. Each of the generated picture clusters was linked to a genre, such that a song's genre and a random image in the corresponding image cluster could be mapped together. There were some intriguing outcomes created by their music-to-picture mapping tool. Songs like Lady Gaga's Poker Face were appropriately classified as Pop by our system. According to the authors, when they mapped the pop genre to a random picture from its related image cluster, they got a pretty fair accuracy.

Because there are so many different sorts of music genres, humans require a system for gathering information about each song's musical elements that is both consistent and unique from the type of genre [29]. One of the literatures compares and discusses the overall method of music genre identification by comparing two important articles about music genre classifications. Distinguishing traits include the rhythm, frequency range, and texture of the sound. Those characteristics are referred to as features, and feature extraction is the process of putting them together. The initial stage in detecting musical styles, regardless of the approach employed, is to train the system, which requires playing examples of songs from various genres and training it about their qualities. Unknown music can only be classified with accuracy if the system has been trained on a large enough training set. It also emphasizes a critical component in the feature selection process. It was accomplished by selecting and modifying them so that music tracks from various genres did not overlap. The neighborhood of a point is crucial for obtaining the right genre categorization for an unknown file when using typical data mining approaches like k-means or Gaussian mixture models (GMMs). First and foremost, this paper will describe and compare two works of musical genre classification research, one of which is Tzanetakis and Cook's Musical Genre Classification of Audio Signals, which is the first notable work in the subject and uses an acoustic analytic technique. Debora C. Correa, Luciano da F. Costa, and Jose H. Saito's Tracking the Beat: Classification of Music Genres and Synthesis of Rhythms uses weighted directed graphs to convey the rhythm of a music recording in a whole new way. The music tracks are presented in waveform so that the track's frequency spectrum can be examined in great detail. For each music recording, a 30-dimensional feature vector is created by integrating the attributes of timbral texture (19 dimensions), rhythmic content (6 dimensions), and pitch content (5 dimensions). Timbral texture refers to the characteristics of music's timbre. The timbral texture isn't immediately apparent in the music recording, but it can be extrapolated based on intuition. The short-time Fourier transform determines the timbral texture

features (STFT). The frequency spectrum of music is investigated in the following way: It shows how different frequency bands and time intervals are important in relation to one another. STFT determines spectral centroid, which is the "frequency below which 85% of the magnitude distribution is concentrated," spectral flux, which is a measure of frequency spectrum change, and time domain zero-crossing, which is a measure of the track's noisiness. The mel-frequency cepstral coefficient, which was developed for automatic speech detection, is also employed (MFCC). MFCCs have been developed on the basis of the STFT, which is aimed at human perception.

The concise representation of the frequency spectrum is the most significant aspect in establishing a musical genre [29]. Only the top five calculated coefficients are used in this approach. The music track is divided into 23ms-long pieces known as analysis windows in order to do feature calculation. Each of the previously mentioned timbral texture properties is only computed once per analysis window. Selecting a short analytical time frame provides for a more consistent frequency, which aids analysis. The disadvantage of using a limited window is that it makes it difficult to generalize about the original music based on the sample supplied. The actual characteristics used for classification on the 1 second texture window are calculated as a running average and variance over the analysis windows contained in that texture window. The greatest peak and the Beat histogram (BH) sum are computed to create a 6-dimensional representation of rhythmic content features that can be used for categorization. Finally, the most crucial components of the pitch are the content aspects. These features are computed in the same way as rhythmic content characteristics are computed. In contrast to the BH, which employs a single bin for all of the tones inside, there are two types of pitch histograms (PH). Each tone's octave is taken into account in the unfolded form, and all of the tones inside it are separated into different bins as a result. The largest peak of the folded histogram and total sum are two of the five aggregate features determined from the two PH versions. Correa, Costa, and Saito discuss a new approach to musical genre classification. The music track is the main focus, as the name suggests. The recordings used for training and testing are encoded using the Music Instrument Digital Interface (MIDI) so that they may be examined more effectively. Unlike waveform formats, this format preserves the entire melody and rhythm information. The rhythm of a song is made up of a series of notes with varied durations. Weighted directed networks (digraphs) with 18 vertices representing the most common note lengths and edges representing the relative popularity of the two-note sequence connected with them are used. By averaging the weight of each edge across all songs in a specific genre, aggregated digraphs for distinct genres can be computed. To begin, the genre digraphs are utilized to extract 15 properties from the graph, such as total vertex degree, and apply PCA to the 18 x 18 matrix that reflects it. Both procedures resulted in a total of 52 dimensions. Tzanetakis and Cook were able to appropriately describe an unfamiliar music recording with a precision of 59% using the ten genres of classical, country, disco, hip hop, jazz, rock, blues, reggae, pop, and metal. In addition, discriminating between music and speech may be achieved with an accuracy of 86%. Correa, Costa, and Saito used the four genres of blues, bossa-nova, reggae, and rock to get an accuracy rate of 85.72%. When comparing the two methods, keep in mind that as the number of choices grows, classification challenges get more challenging. Another

advantage of utilizing MIDI data for tracking the beat instead of waveform data is that it is easier to understand.

In this paper [30], the authors evaluate the feasibility and utility of Gaussian Processes (GPs), which are Bayesian nonparametric models for music genre classification and emotion estimates. These are two significant issues in the field of information retrieval about music (MIR). This field of study has focused on activities that all contribute to the creation of fast music search and recommendation services, intelligent playlist construction, and other visually appealing applications. Typically, researchers limit the number of genres in MIR experiments to roughly ten of the most widespread and clearly distinguishable categories. Each genre classification system is made up of at least two components. These are a feature extractor and a classifier. Several studies advocate that emotion be characterized using continuous multidimensional measures based on low-dimensional spaces in order to solve the issue of assuring consistency in the interpretation of mood categories. Most people are familiar with Russell's two-dimensional Valence-Arousal (VA) space, in which emotions are represented by points on VA's plane. Assumed in this research, music emotion detection involves finding the VA plane point that corresponds to a particular piece of music's emotional content. Using the same music feature representation and two distinct regression models, it is possible to measure both valence and arousal independently. When searching for specific mood-related auditory elements, previous research has failed to find a single one that is most regularly used. Music emotion estimation has shown success with regression models such as Multiple Linear Regression (MLR), Support Vector Regression (SVR), or Adaboost.RT, as well as Multi-Level Least-Squares or regression trees. Again, training data is required for model learning. Identifying moods with VA values that are constant among listeners is even more difficult than identifying genres. This is because people's perceptions of emotion vary widely. It is the purpose of this research to examine the usefulness of Gaussian Methodology to music genre and emotion detection tasks and to assess their efficiency to the present state of the art Support Vector Machines (SVMs). However, researchers needed to do a more complete analysis into the results of our first tests, which indicated that GPs may be an alternative to SVMs. Both the GP and SVM models were examined and compared in this study utilizing two databases of equal size, the same collection of characteristics, and the same system parameters. GPs outperform SVMs in both tasks, according to the results. According to R2 metrics, the GPs were able to improve genre categorization accuracy by up to 11% in all situations, including the notoriously challenging task of estimating Valence rather than Arousal. Based on a song's feature representation, we hypothesize in this work that music emotion recognition may be used to estimate the Valence-Arousal (VA) values of the song. The same training data and accompanying reference VA values are used to train separate Gaussian Process regression (GPR) and support vector regression (SVR) models. The "MediaEval'2013" database was utilized for the music emotion recognition studies. In total, there are 1000 45-second segments culled from 1000 distinct songs. In all, there are 125 songs in each of the following eight genres: Blues (Electronic), Rock (Classical), Folk (Jazz), Country (Country), and Pop (Pop). There were 53-100 different musicians in each genre, ensuring a well-balanced mix of styles. Several annotators have scored each clip on a 9-point scale for Arousal and Valence. An average of 107.9 clips has been annotated by each of the 100 participants. The final Arousal/Valence label for each clip was derived from the mean of the annotator ratings. Before feature extraction, all music audio data had a sample

frequency of 44.1kHz, which was lowered to 22.05kHz. The "standard" collection of feature extraction algorithms we employed in our tests is frequently used in music signal processing investigations. They are MFCC (mel frequency cepstral coefficients), LSP (line spectral pairs), TMBR (timbre features), SCF and SFM (spectral crest factor and spectral flatness measure) and CHR (chromagram). They employed the Marsyas tool, which is capable of extracting any combination of the attributes mentioned previously. The frame size was set to 512 points, which equated to 23.2 msec at a sampling rate of 22050 Hz. There was a complete lack of visual overlap between frames, therefore, for each feature type, they utilized the default dimensionality, which was 13 for MFCC, 18 for LSP, 1 for each timbre feature, and 24 for SFM, SCF, and CHR. A single vector is created for each frame when several features are computed. Afterwards, each window of the vector sequence is separated into 20 vectors and the mean and standard deviation are determined. On top of all of that, we compute window-specific means and standard deviations and combine them into a single vector to represent our whole clip. According to the results of the experiment, including timbral elements into the MFCC had no impact. However, the spectral crest factor (SCF) and spectral flatness measure (SFM) produced substantial improvement while the chromagram and line spectral pairings performance being negatively influenced. Their music was drawn from the well-known GTZAN song collection, which includes 30 second recordings from different types of musical genres, including rock, reggae and rock as well as hip-hop and jazz. Each genre has 100 recordings, for a total of 1000. MediaEval'2013 data for music emotion estimate was used to process all 1000 clips, and the exact same characteristics were retrieved. Once more, each piece of music was represented by a single feature vector, this time made up of statistics at two different levels of the frame level. When it comes to GP, in the kernels scenario, Logistic function was better. The difference between GP and is 2.8 percent, which amounts to 13.6 percent relative error reduction. In the conclusion, they find that the GP and SVM have many comparable qualities. "They are both non-parametric, kernel-based models, and their implementation and usage as regressors or binary classifiers are the same. However, GP are probabilistic Bayesian predictors which in contrast to SVM create Gaussian distributions as their output" mentions the researchers. Using the training data, it is possible to learn new parameters. However, SVM gives sparse solution, i.e. just support vectors are needed for the inference, which might be a bonus when working with big quantity of data. Emotion estimation and genre categorization studies utilizing the MediaEval'2013 music database revealed that generalized linear programming (GP) models regularly outperformed support vector machines (SVMs).

The multi-label genre classification is more important in the music genre classification because music is influenced by an increasing amount of different musical styles [31]. Single genre labels usually assign the meaning of inherent stylistic elements to a music piece, however it increases ambiguity in case of musical pieces having multiple genres. For the multi-label genre categorization, they present an original technique for recognizing musical genre boundaries. A music work is split into genre pieces by genre boundaries. Then multi-label genre classification is obtained by applying single-label genre classification to each segments. Genre boundary candidates are generated, and then genre borders are determined, in the suggested technique. It is possible to generate genre boundary candidates by looking at the highest points on novelty score curves. Novelty score curves is calculated using MFCC, chroma characteristics and rhythmograms. Genre-specific thresholds of MAXSEG, SEGSIZE, and THETA are used to create border candidate music segments. Throughout

the article, a segment is defined as a music sequence. MAXSEG threshold is the maximum number of segments in one music composition. Musical impressions can be ruined by frequent genre shifts, which are common in arrangements. SEGSIZE threshold is the size of a segment. The border identification method is hampered when segments are too short to be useful. This is due to the fact that most music has a repeated pattern, with a basic unit consisting of four musical bars being the most common unit. As a result, a genre shift is quite improbable to occur in less than four bars of music. Because of this, SEGSIZE is defined as  $SEGSIZE \frac{60}{TEMPO} \times 16$ , divided by TEMPO where TEMPO is the tempo of the song in beats per minute. The final threshold, THETA, is defined for computing a boundary from a large number of peaks. "If the novelty scale value at a specific peak is more than this threshold, then the peak will be classed as a border candidate and in the context of MFCC and chroma, the terms "mft," "cht," and "rht" refer to border candidates created by each of these methods" mentions the author. In order to determine a genre's boundary, aspects within the LAG are examined. To put it another way,  $TEMPO * 4$  seconds equate to the LAG value. Since the acoustic properties should all shift at around the same time, this makes sense. That is, if all of the acoustic characteristics are found inside the LAG, then one of the border candidates is picked as a border. According to their first evaluation results, genre shifts are more likely to occur at the earliest border of the border candidates. According to the paper, average precision score has improved 7% with the average precision, by increasing to 0.39.

The classification of musical genres is the subject of yet another study [32]. For the 21st century, listening to a song and then selecting which genre it belongs to is laborious, so in this research, they have proposed a new system for categorizing songs. The classification of songs is based on the number of beats per minute (bpm), loudness, energy, danceability, speechiness, valence, accousticness, and discrete wavelet transform (DWT) properties. Classical and Sufi songs are the two types of music that are analyzed in this study. In order to get results quickly, the database has 200 songs in each of two environments: real time and run time only. The similarity between Indian Sufi songs and Classical music in terms of the utilization of natural songs and speechiness patterns may lead to a slight drop in accuracy. It is possible to adapt this composition to a variety of other musical genres, including Jazz, Rock, Pop, Rap, and so on. When utilized in the music-selling industry, it can help cut costs by automating the process of identifying genres and improve customer satisfaction by eliminating the need for operators to have prior knowledge of different music styles. This requires no prior knowledge. Songs that combine two or more genres, such as Jazz-Rock, Rock-Rap, or Classical-Rock, can be used into this project.

## 2.4 METHODOLOGY

### 2.4.1 Feature Extraction for the first section

For the first section, I utilized the GTZAN dataset, which has 1000 songs in total, 100 songs per genre. There are 10 genres in the dataset. The sample rate of songs is 22050 Hz. Now I will be calculating the MFCC features for the songs. Since each song is 30 seconds, we can find the sample per track, which will be

$$\text{sample per track} = 22050 * 30$$

Now considering that taking MFCC features for 30 seconds would be too long, therefore, we split it into segments, 10, to be precise. Then each segment will be 3 seconds. Then the number of sample per segment would be

$$\text{the number of sample per segment} = 22050 * \frac{30}{10}$$

22050 multiplied by 30 divided by 10 (  $22050 * 30 / 10$ ). Then to find the expected number of mfcc vectors per segment we need to divide it by the hop length, which I took 512 in my case

$$\text{vector size} = \frac{(22050 * 30)}{(10 * 512)} = 130$$

In the end, we do the ceiling of the value and we get 130 vectors, each vector containing 13 MFCC features. This was for a segment, and considering that we have 1000 songs, each containing 10 segments, we will have 10000 of the expected number of mfcc vectors per segment.

#### 2.4.2 Feature extraction for the second section

For the second section, I used the custom dataset I made, which has 2200 songs in total, 200 songs per genre. There are 11 genres in the dataset. The sample rate of songs is 22050 Hz. Now I will be calculating their MFCC features for the songs. Since each song is 30 seconds, we can find the sample per track, which will be

$$\text{sample per track} = 22050 * 30$$

Now for this part, I have used segment 1,3,6,10,15 and 30 to train the model. Therefore, I will be calculating vector size for all:

$$\text{vector size (seg = 1)} = \frac{(22050 * 30)}{(1 * 512)} = 1291$$

$$\text{vector size (seg = 3)} = \frac{(22050 * 30)}{(3 * 512)} = 431$$

$$\text{vector size (seg = 6)} = \frac{(22050 * 30)}{(6 * 512)} = 215$$

$$\text{vector size (seg = 10)} = \frac{(22050 * 30)}{(10 * 512)} = 129$$

$$\text{vector size (seg = 15)} = \frac{(22050 * 30)}{(15 * 512)} = 86$$

$$\text{vector size (seg = 30)} = \frac{(22050 * 30)}{(6 * 512)} = 43$$

This means for segments 1, 3, 6, 10, 15 and 30, I'll have the vector size of 1291, 431, 215, 129, 86 and 43 vectors of 13 MFCC features.

### 2.4.3 Algorithms

The machine learning algorithms I used for this research papers are Multi-Layer perceptron (MLP), Neural Network (NN), Convolutional Neural Network and RNN-LSTM which were Deep Learning Algorithms.

An artificial neural network is a type of algorithm based on the structure and function of a human brain, and it is a subfield of deep learning [36]. Deep learning works best with problems that include analog inputs and outputs and on the other hand, they're not just a few numbers in a spreadsheet, but rather photographs, documents, and audio files [36]. A dataset's input-output correlations and more sophisticated characteristics can be discovered with the help of Deep Learning [40]. Artificial Intelligence (AI) services and apps rely on deep learning to execute analytical and physical activities without the need for human supervision [37]. Modern products and services (including digital assistants, voice-activated TV remotes, and credit card fraud detection) and upcoming innovations alike utilize deep learning technology (such as self-driving cars) [37]. Manual feature engineering, which must be explicitly written by programmers, is required for machine learning algorithms [40]. Deep learning algorithms, as opposed to machine learning, can learn features on their own without the need for user input [40]

In this era of large data, a promising component of deep learning training is the overwhelming of data, but deep learning's triumph over traditional machine learning is also due in large part to the availability of more powerful computational hardware resources, such as graphics processing units (GPUs) [40].

In order to further refine and categorize predictions, deep neural networks use numerous layers of interconnected nodes [37]. Forward propagation refers to the process of advancing computations over a network. It's the hidden layers' job to analyze input data in accordance with an activation function and then pass it on to a higher-level layer [38]. Visible layers are the input and output layers of a deep neural network [37]. There are two layers in a deep learning model: an input layer that accepts data for processing, and an output layer that makes the final prediction or classification [37].

Backpropagation is a technique that employs algorithms like gradient descent to generate prediction errors and then moves backwards through the layers to alter the weights and biases of the function in an effort to train the model [37]. A neural network can generate predictions and fix itself for mistakes using forward and backpropagation together [37]. The algorithm continuously improves

over time [37]. This method was initially developed back in the 1970s, but it wasn't until a seminal publication by David Rumelhart, Geoffrey Hinton, and Ronald Williams in 1986 that its significance was fully understood and appreciated [50].

#### 2.4.3.1 Multi-Layer Perceptron

Even though the Perceptron technique is widely known now, it was originally designed to be an image recognition computer [41]. It derives its name from its ability to perceive, see, and recognize pictures in a manner similar to that of a human [41]. Since multi-layer perceptrons are one of the most effective types of artificial neural networks, the topic is commonly known as "neural networks" or "multi-layer perceptrons" [39]. As a forerunner to bigger neural networks, the perceptron model represents one neuron [39]. Researchers are looking at how basic models of biological brains may be utilized to do complex computer tasks like predictive modeling in machine learning [39]. Instead of creating accurate models of the brain, our objective is to construct strong algorithms and data structures that may be used to represent complex situations [39]. The hierarchical or multi-layered structure of neural networks is what gives them their predictive power [39]. The data structure may integrate lower-order features like lines, groups of lines, and shapes into higher-order features by picking out (learning to represent) characteristics at various scales or resolutions [39]. A total of three layers are involved: the input layer, the output layer, and the hidden layer [40]. This layer receives the signal to be processed from the input source. The output layer is responsible for tasks like prediction and categorization [40]. The MLP's real computational engine is comprised of an arbitrary number of hidden layers positioned between the input and output layers [40].



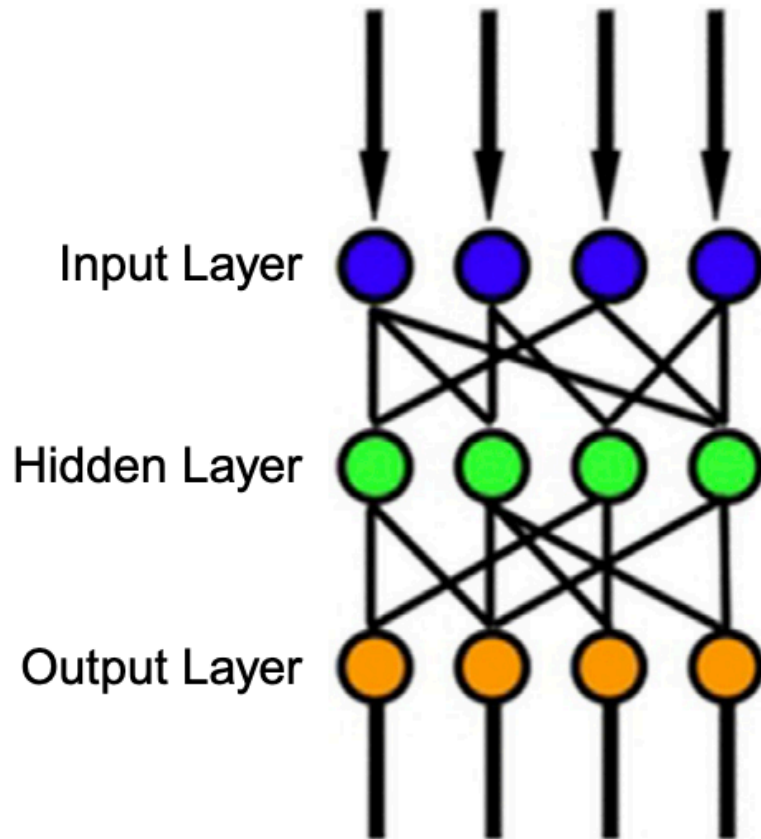


Figure 2.11 Types of layers

Each neuron in the output and hidden layers performs the following calculations:

$$o(x) = G(b(2) + W(2)h(x))$$

$$h(x) = F(x) = s(b(1) + W(1)x)$$

where  $b(1)$  and  $b(2)$ ,  $W(1)$  and  $W(2)$ , and  $G$  and  $s$  are bias vectors, weight matrices, and activation functions respectively.

Warren McCulloch, a neurophysiologist, and logician Walter Pitts, a philosopher, collaborated on a model of the brain in the early 1940s [41]. Given a set of inputs and weights, it was a basic linear model that yielded a positive or negative output and calculated as in Fig 2.12 [41]. This paradigm of computing was named neuron because it attempted to replicate how the brain's most fundamental building unit operated [41]. Each neuron also has a bias, which may be regarded of as an input that

always has the value 1.0, and it must be weighted as well [39]. If there are two inputs to a neuron, it will need three weights [39]. The bias and each of the inputs have their own [39].

Weights are commonly set to random values, although more sophisticated starting procedures are possible [39]. The bigger the weights, the more intricate and fragile the system is [39]. Weights in the network should be kept to a minimum, and regularization methods can be employed to achieve this goal [39].

$$\underbrace{f(x, w)}_{\text{output}} = \underbrace{x_1 w_1 + \dots + x_n w_n}_{\text{inputs}} \quad \begin{array}{l} \text{weights} \\ \swarrow \end{array}$$

Figure 2.12 Basic model calculations

While in Rosenblatt's model, the weighted total of inputs is used to determine whether the neuron fires and generates an output, the weighted sum is not used in Rosenblatt's model [41]. The activation function is represented by the threshold T (Fig 2.13). There is a neuron that produces a 1 if the weighted sum of its inputs is larger than zero [41].

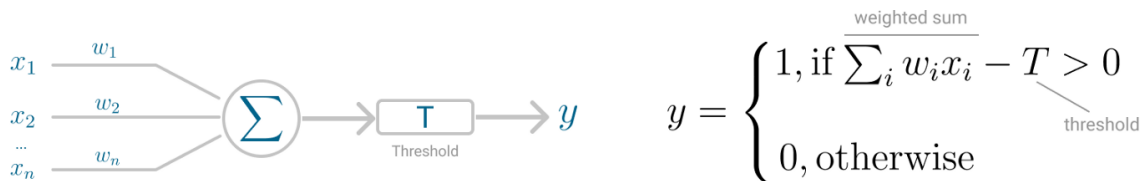


Figure 2.13 Loss function

An activation function, also known as a transfer function, is used to process the summed weighted inputs [39].

The activation function is a straightforward mapping of summed weighted input to the neuron's output [39]. When a neuron is stimulated, the activation function determines its threshold and the intensity of its output signal [39]. Non-linear activation functions have traditionally been utilized in the past, because of this, the network's inputs may be combined in increasingly sophisticated ways, resulting in more complex models [39]. The logistic function, also known as the sigmoid function, which produces a value between 0 and 1 with an s-shaped distribution, and the hyperbolic tangent function, also known as tanh, which produces the same distribution over the range of -1 to +1, were both non-linear functions that were employed [39]. The first Perceptron models employed the sigmoid function, which encodes non-linear functions by mapping any actual input to a value of 0

or 1. Inputs of -1 or -1 can be fed into the neuron, and it will still be able to output 0 or 1 [41] (Fig 2.14).

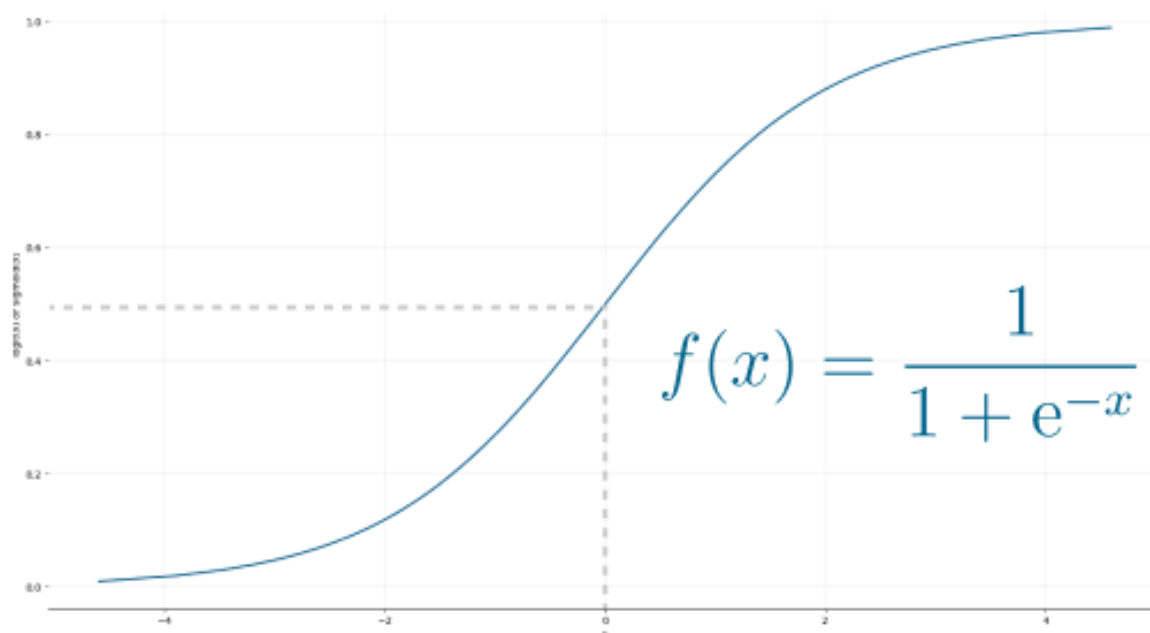


Figure 2.14 Sigmoid function

In the recent decade, the majority of Deep Learning articles and algorithms have used the Rectified Linear Unit (ReLU) as the neuron's activation method [41]. In deep learning models, the Rectified Linear Unit is the most widely employed activation function [42]. If the function is given a negative value, it returns 0, but if it is given a positive value, it returns that value [42]. The formula is:

$$f(x) = \max(0, x)$$

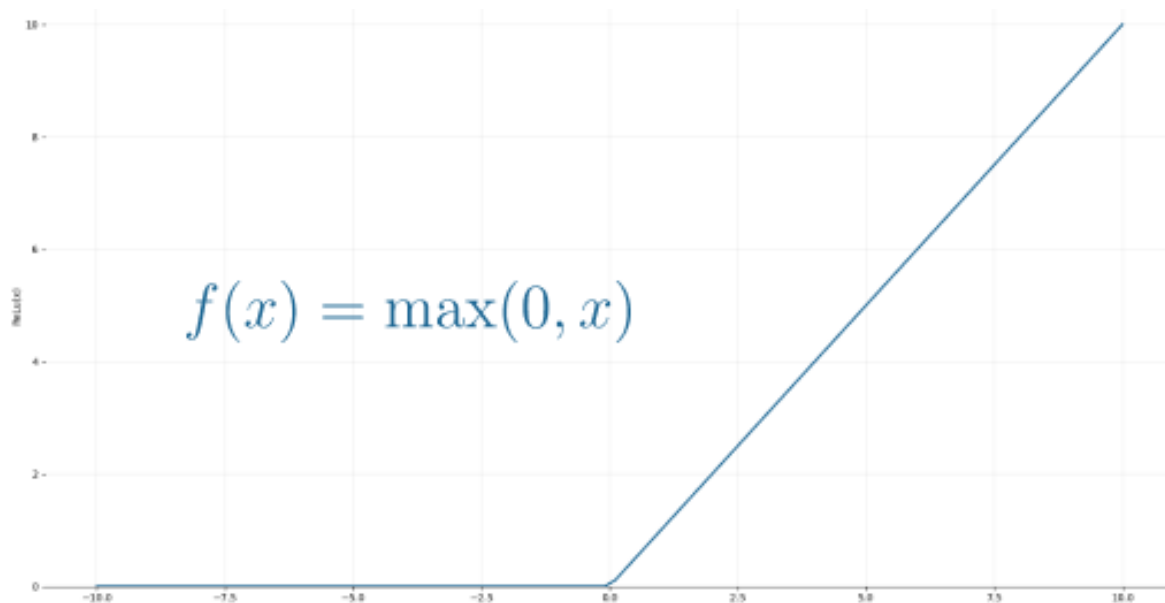


Figure 2.15 ReLU

#### 2.4.3.2 Convolutional Neural Network

There are several types of neural networks but the Convolutional Neural Network (CNN), often known as CNN or ConvNet, is one of the most popular [43]. An MLP-inspired advancement, convolutional neural networks (CNN) are based on biological inspirations [40]. Yann LeCun is the head of Facebook Research and is the inventor of the Convolutional Neural Network, a network design that excels in recognizing objects in images (CNN) [36]. Images may be classified, clustered, and objects can be detected using CNNs [36]. An image in digital form is a representation of visual information in the form of a binary code, and it has a grid-like arrangement of pixels with pixel values that indicate the brightness and color of each pixel [43]. Aside from that, they're used in optical character recognition and Natural Language Processing (NLP) [40]. In addition to pictures, CNNs may also be used to sound, and they are also commonly used in text analytics and in graph data using graph convolutional networks [40]. Compared to its baseline algorithms, CNN's state-of-the-art art efficiency has made it a success in many sectors [40].

Convolutional, pooling, and fully linked layers are all common in CNNs (*Fig 2.16*) [43]. Filters, sometimes called as kernels, are used to identify characteristics in CNN [40]. For a filter to work, it has to be taught to identify certain traits [40]. Each filter concatenates with it to generate an activation map, which is smaller than the image itself (*Fig 2.17*) [44]. An element-wise product and sum is what the filter achieves by performing the convolutional operation between two matrices [40]. A neuron's output may be equated with every component of the activation map, which means that each neuron is linked to a tiny local region of the input picture, and the filter's size is proportional to that region [43]. There are also shared parameters across all neurons in an activation map [43]. By

minimizing the amount of duplication in the input feature, the training of the CNN is accelerated and as a result, the amount of RAM the network uses are likewise lowered [40]. When using max pooling, input data is passed through a window, and the data with the highest value in the window is gathered and used in the output matrices [40]. Concatenating several convolution layers and max pooling operations improve the algorithm's efficiency for feature extraction [40]. The data is transformed into a feature vector by going through an MLP after it has been processed via these deep layers to create feature maps [40]. High-level reasoning occurs in this layer, which is called a fully-connected Layer [40]. The entire CNN is fed by the input layer [49]. It typically represents the picture's pixel matrix in a neural network for image processing [49].

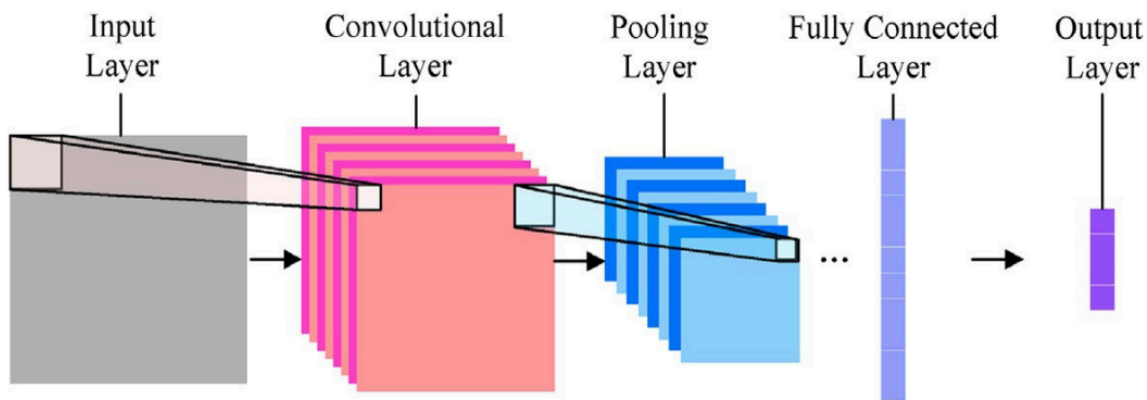


Figure 2.16 CNN steps

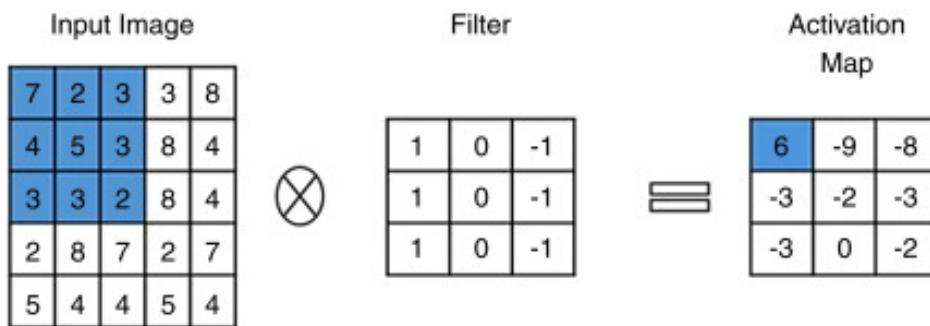


Figure 2.17 Activation Map

The first layer of a Convolutional Neural Network is always a Convolutional Layer [45]. The CNN's fundamental building piece is the convolution layer [43]. Convolutional layers apply a convolution operation to the input, passing the result to the next layer [44]. A convolution converts all the pixels in its receptive field into a single value [44]. To blur and sharpen pictures as well as execute other

operations, convolutions have been employed for a long time in image processing [45]. Local connection patterns are enforced by CNNs (e.g., by increasing the sharpness of edges and embossing them) on neurons in neighboring layers [45]. In terms of computing burden, it bears the lion's share of the network [43]. Two matrices, one of which is known as a kernel and the other as a limited receptive field matrix, are used in this layer to execute a dot product [43]. The kernel is smaller than an image, but it contains more information [43]. Kernel height and width will be modest, but the depth will extend to all three channels if the picture is made up of three channels [43].

Between each convolutional layer, a pooling layer is common [46]. The representation is down sampled by the pooling layer, which minimizes the number of parameters and computations [46]. A summary statistic of neighboring outputs is used by the pooling layer to substitute the network's output at certain spots [42]. This reduces the representation's spatial size, which in turn reduces the computation and weights required [42]. Max and average pooling functions are available [46]. Each slice of the representation is treated independently during the pooling phase (Fig 2.18) [42, 46]. The largest value from the Kernel-covered area of the picture is returned by Max Pooling [47]. Average Pooling, on the other hand, returns the average of all the values from the Kernel-covered region of the picture [47]. When it comes to max pooling, summarizing areas is all that is required to reduce the size of an input image [47]. A grid, which is the pool size, and a stride must be selected in order to achieve maximum pooling [47]. Stride is a filtering parameter that affects the amount of movement in the picture or video being processed by the neural network [48]. Neural networks may be trained to move one pixel, or unit, at a time, by setting their stride to 1 [48]. Rather than a fraction or decimal, stride is often adjusted to an integer since it has an effect on encoded output volume [48].

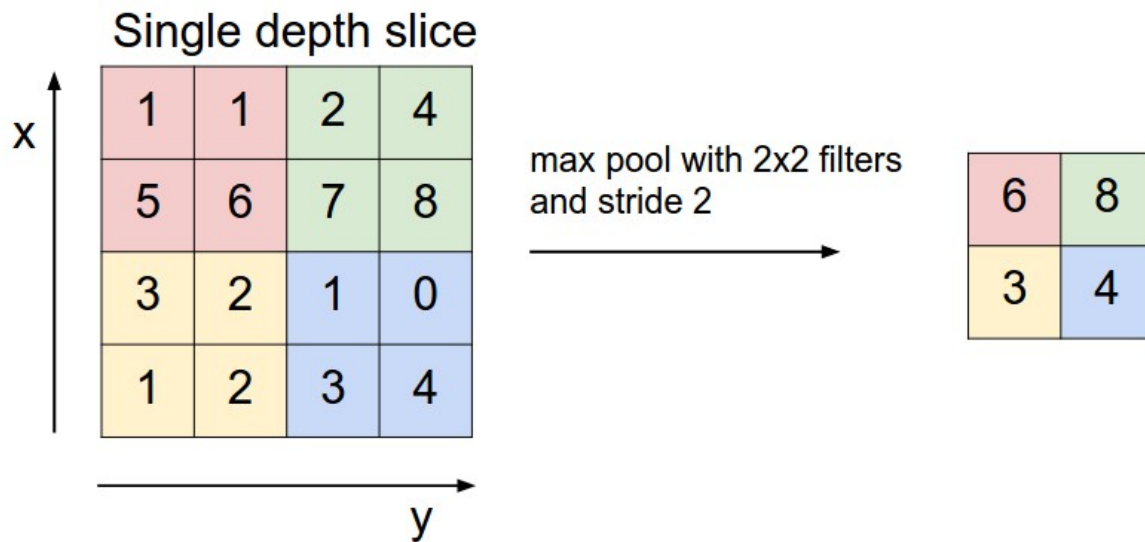


Figure 2.18 Pooling operation

If we have an activation map of size  $W \times W \times D$ , a pooling kernel of spatial size  $F$ , and stride  $S$ , then the size of output volume can be determined by the following formula [43]:

$$W_{out} = \frac{W - F}{S} + 1$$

The probabilities of each class are contained in the output of the fully-connected layer [40]. The last fully-connected layer is called the “output layer” and in classification settings it represents the class scores. The output that has been categorized has the highest likelihood [40]. The back propagation of gradients is used to update and optimize the algorithm's weights [40].

For classification, a fully-connected layer needs just a one-dimensional vector from a three-dimensional layer to suit its input, which called the flatten layer [52]. As an illustration, a  $5 \times 5 \times 2$  tensor might be transformed into a 50-dimensional vector [52]. Features retrieved from a picture by previous convolutional layers are now ready to be classified [52]. These characteristics are classified using the softmax function, which needs a one-dimensional input [52]. As a result, it's critical that the top layer be able to flatten [52].

Convolution Neural Network (CNN) refers to a 2D CNN that is utilized for picture categorization [51]. However, 1 and 3-dimensional Convolution Neural Networks (CNNs) are real-world applications of Convolution Neural Networks [51].

Since its debut in Lenet-5 design, this has been the standard Convolutional Neural Network. Typically, Conv2D is used to images (Fig 2.19) [51]. The 2D CNN gets its name from the fact that the kernel moves in two dimensions on the data, as seen in the accompanying graphic.

Conv1D is a kernel that glides along one axis [51]. One dimension of kernel sliding is required just for Time-Series data [51]. 1D CNNs may also be used to analyze audio and text data because they can be represented as a time series [51]. For a number of applications, such as the classification and early diagnosis of personal biomedical data, structural health monitoring, anomaly detection and identification in power electronics and electrical motor fault detection, 1D CNNs have recently been proposed and immediately achieved state-of-the-art performance levels [53]. Because 1D CNNs only conduct 1D convolutions, a real-time and low-cost hardware implementation is also possible, which is a big benefit, such as scalar multiplications and additions [53].

In Conv3D, the kernel slides in 3 dimensions (Fig 2.20) [51]. Such as data from Magnetic Resonance Imaging (MRI) scans [51]. The brain, spinal cords, internal organs, and a slew of other structures may all be studied using MRI data [51]. Additionally, 3D data may be generated by merging many X-ray images collected from different angles throughout the body [51]. A CT scan is an example of this [51]. We can categorize and extract characteristics from this medical data using Conv3D [51]. Video is another example of 3D data. A video is nothing more than a collection of still images strung

together in time [51]. Because it contains spatial properties, we can use Conv3D on video as well [51]

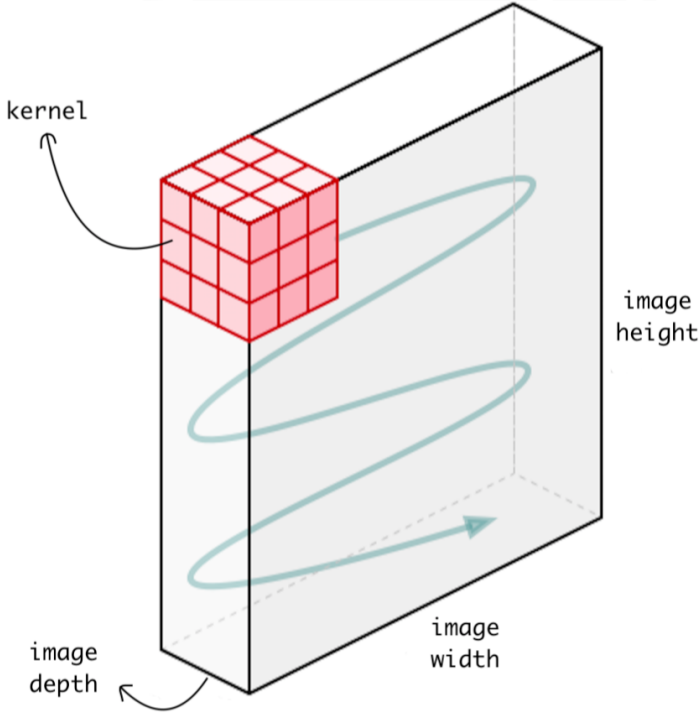


Figure 2.19 Slide of the Kernel over the Picture



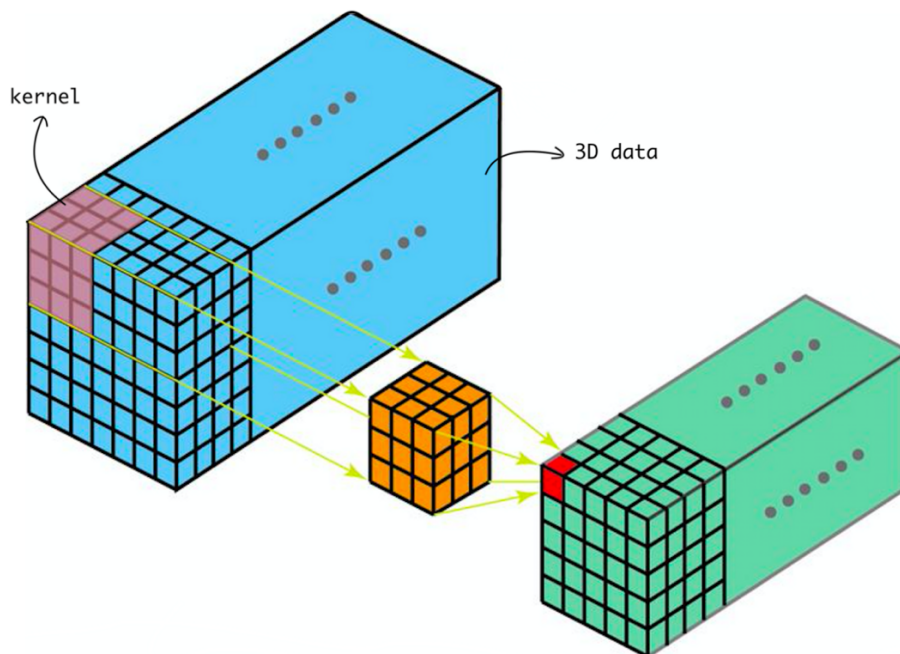


Figure 2.20 Slide of the Kernel over 3D data

The main advantage of employing a CNN is that it can extract spatial characteristics from the data using its kernel, whereas other networks cannot [51]. CNNs, for example, are highly good at detecting edges, the distribution of colors, and other spatial features in an image, making them ideal for image classification and other spatially based data [51].

#### 2.4.3.3 Recurrent Neural Network (RNN) and Long Short-Term Memory Network (LSTM)

There have been recurrent neural networks around for a while now and they were first developed in the 1980s, but their full potential has only just been realized [54]. RNN's internal memory helps them to recall critical information about the input they received, allowing them to be quite accurate in forecasting what's going to happen next [54]. For this reason, they're the algorithm of choice for a wide range of sequential data, including time series, voice, text, financial data, audio, video, and a wide range of other media and they have the ability to better grasp a sequence and its context [54].

In a feed-forward neural network, information only flows from the input layer to the output layer in one way, meaning the data is sent from one node to the next in a straight line, with no intermediate stops [54]. In a recurrent neural network (RNN), data is fed into a recursive loop and decisions are based on both current information and what it has already learnt from past inputs [54].

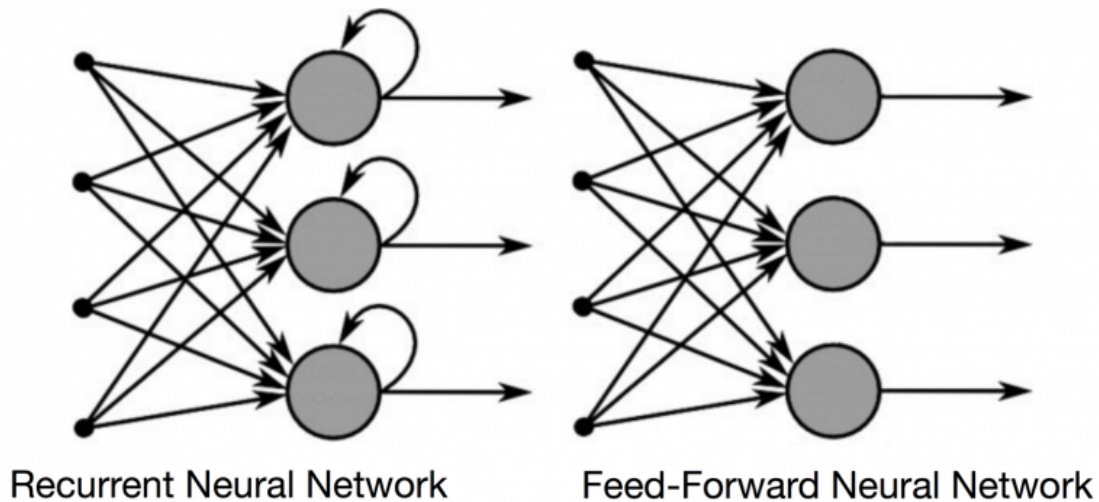


Figure 2.21 RNN and Feed-Forward NN demonstration

The word "Azerbaijan" is fed to a feed-forward neural network, which analyses the word character-by-character and by the time it reaches the character "e" it has already forgotten about the characters "A" and "z", therefore this form of neural network can't anticipate what the following character will be [54]. A Recurrent neural network, on the other hand, has an internal memory that allows it to recall these characteristics [54]. Looping back into the network, it generates output and replicates it. Recurrent neural networks combine the present with the past [54]. As a result, an RNN uses the present and recent past as inputs [54]. A RNN can achieve things other algorithms can't because the sequence of data carries critical information about what is to come next [54].

The Long Short-Term Memory Network (LSTM), a form of recurrent neural network developed by Jurgen Schmidhuber, is the father of another famous technique that, like MLPs and CNNs, scales with model size and dataset size and can be trained via backpropagation [36]. RNNs can retain inputs for a long time with the help of LSTMs [54]. This is due to the fact that LSTMs have a memory, much like a computer's memory [54]. The LSTM has a built-in memory that it may use to read, write, and erase data [54].

It's possible to think of this memory as a gated cell, in which case the cell selects whether or not to store or erase information dependent on the relevance the information is assigned (i.e. whether it opens the gates or not) [54]. Weights, which are also learnt by the algorithm, are used to allocate

importance [54]. In layman's terms, this implies that it gets smarter as it goes along, figuring out what information is critical and what is not [54].

Input, forget and output gates are all three gates you have in an LSTM (Fig 2.22) [54]. This group of gates decides whether or not to allow fresh input (input gate), erase information because it isn't relevant (forget gate) or allow it to affect output at the current timestep (output gate). A RNN with its three gates is depicted in the following image [54].

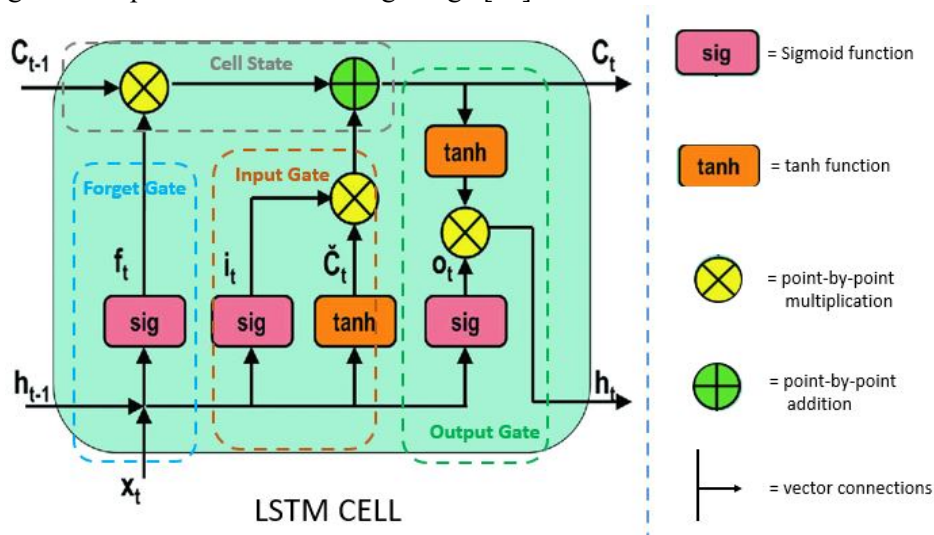


Figure 2.22 LSTM gates

## 2.5 Experiments

Depending on the experiment, I may use more or fewer segments, hidden layers, or filters, epochs and different loss metrics, therefore, I will be mentioning in each experiment. For all experiments, I have used Adam optimizer, Sparse Categorical Cross Entropy loss function and Softmax activation function. I've conducted over 60 experiments and models, but I'll highlight the most significant ones here.

### 2.5.1 First Section (GTZAN Data)

The first experiment was to feed the neural network with data. The number of segments was ten in this case, indicating that 3 seconds of music was used to extract MFCC features. I used an 80/20 split between training and testing data. As a result of using the "relu" activation method with three hidden layers with filters 512, 256, 64, the output layers now had 10 filters each due to their 10 labels. The model had a training accuracy of 99 percent and a validation accuracy of 66 percent after 150 epochs, indicating that it was overfitting.

I dropped one hidden layer from the same neural network and trained it for 50 epochs to achieve 72 percent training accuracy and 53% validation accuracy in the second experiment. I used a learning rate of 0.0002 in both experiments.

Later on, I switched to CNN, which had the identical number of segments. I split the training, validation, and testing datasets by 70%/15%/15%. Later, I used Conv2D, three layers with 32 filters and a kernel size of (3,3), and Max pooling 2D with a height and width of (3,3) and strides of (2,2), and batch normalization in each layer. The rate of learning was decreased to 0.0001. There are 64 filters in the dense layer, and all activations are "relu" in hidden layers. This model achieved a training and validation accuracy of 70% and 60%, respectively.

The next model was done with 1 segment size, meaning MFCC features were extracted for each 30 seconds. Using the same model features, it resulted in 70% and 55% training and validation accuracies, meaning

The next model was the same with dropout of 0.3 in the last layer, which resulted in 85% and 75% training and testing accuracies respectively.

### **2.5.2 Second Section (Custom Data)**

One important experiment was done with MLP with regularization 0.001, with 3 hidden layers 350, 150, 50 adaptive learning rate is, 500 max iterations. I got the average F1 62%.

I experimented with LSTM with 64 and 32 kernel sizes, each with one dense layer, for this model. I used a dropout of 0.3 in the dense layer and obtained training and test accuracy of 80% and 68 %, and 64 %, and 55 %, respectively.

I also experimented with 1D CNN by splitting training, validation and testing data 60%, 20% and 20%. I added two Conv1D 16, 32 and 64 filters with 3 kernel\_size two times (three blocks). The pool size and stride size are 2 and 1. In each block, I used 0.2 dropout in each block. The learning rate is 0.0001.

For this model, I used a dataset with a single segment. I divided the training, validation, and testing datasets by 70%, 15%, and 15%, respectively. The dense layer was increased to 11 due to the increased number of genres. This model achieved a training precision of 91% and a validation precision of 49%, respectively.

Another model I created contained ten segments and achieved accuracy of 80%, 10%, and 10% in training, validation, and testing, respectively. In the final output layer, I used three convolutional layers with a learning rate of 0.0001 and a dropout of 0.3. It resulted in a 70% accuracy for training

and a 66% accuracy for validation over 60 epochs. Accuracy in training and validation is 86 percent and 57%, respectively.

Later, I attempted a model with five segments, three hidden layers with filter sizes of 16, 32, and 64, and a default learning rate of (0.01), which resulted in 73 percent and 67 percent training and validation accuracy, respectively, after 90 epochs.

In another model, increasing the learning rate from 0.01 to 0.001 while maintaining the model helped achieve 79 and 69 percent accuracy in 90 epochs, respectively.

I changed some parts of the dataset in subsequent models; I deleted sections where all MFCC feature values are 0.0, which helped my model improve.

In simple models with several hidden layers and filter size less than 64, the models still performed poor such as 74% and 49%, and 65% and 60% training and validation accuracies.

I split the data into training, validation, and test segments by 60%, 20%, and 20%, respectively, in this model. I had three hidden layers with filter sizes of 32, 128, and 128. Following flattening, one dense layer with 256 filters was created, followed by an output layer with 11 filters. The rate of learning is set to 0.001. In the final three layers, I used dropout 0.3. The model achieved a training and validation accuracy of 90% and 75% after 100 epochs, respectively.

Another model with six segments underperformed, resulting in 68 percent and 71% training and validation accuracy, respectively, due to the high number of dropouts. This model contained four hidden layers with 256, 128, 64, and 32 filters, respectively. After flattening the layer and adding one dense layer with filter 16, the output layer with 11 outputs was created. Each layer had a 40% dropout rate. This could be why it performed poorly.

With 0.001 and 32, 32, 64, and 128 filters with a learning rate of 0.001 and dropout in the final two dense layers, 91 percent and 73%, respectively, were achieved in 150 epochs.

The same model with an additional hidden layer and filter 256 achieved 87 percent and 74% accuracy in training and validation, respectively.

I recently experimented with MFCC with 15 segments (2 seconds) with training, validation, and testing split as 70%/15%/15%. With 16, 32, the rate of learning is 0.001%, 32 (dropout =0.3), 64 (dropout =0.3), 100 epochs and got 72% training and 71% validation.

The last model I experimented with MFCC with 30 segments (1 second) with the parameters and got 80% and 75% training and validation accuracy.

### 2.5.3 User Interface

I have made an api with fastapi as a UI. Through writing `python3 -m uvicorn server:app --reload` in the terminal, it will open postman application.

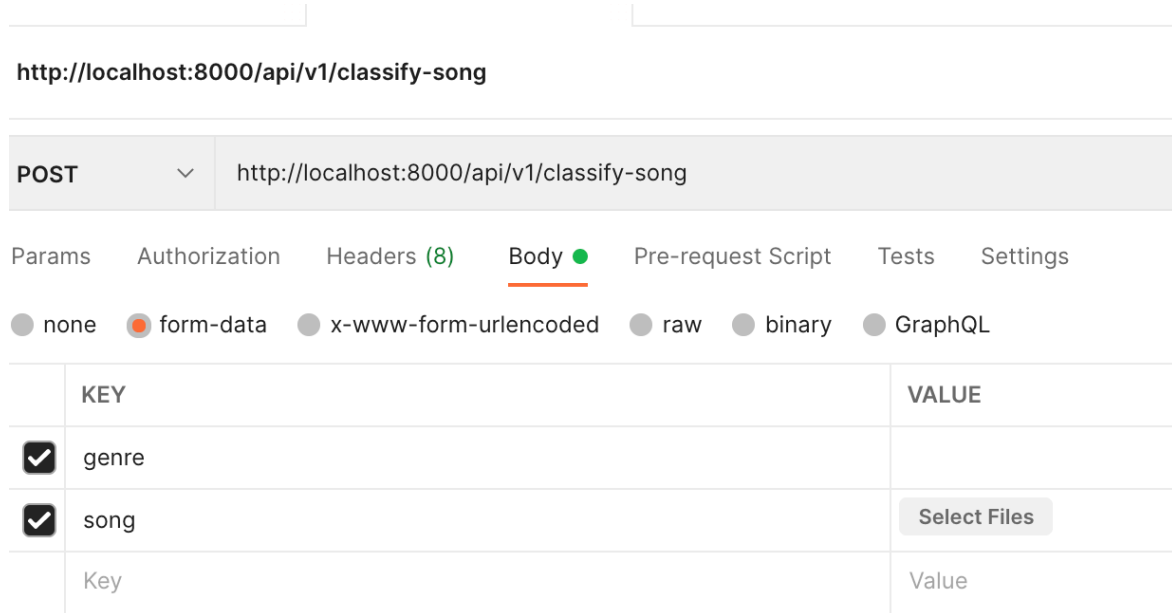


Figure 2.23 User Interface

Genre is the target genre the song I will upload for the testing and song is the song for the testing.

### 2.6 Testing

I use the model where I got the training and testing accuracies 72% and 71%. It will have 14 segments. The songs I have tested are not included in my original dataset.

Let's test some of the songs. The test song is Segah by Agakarim Nafiz, which is Mugam. The result is written here: For better understanding, I also print the predicted and targeted genres in terminal.

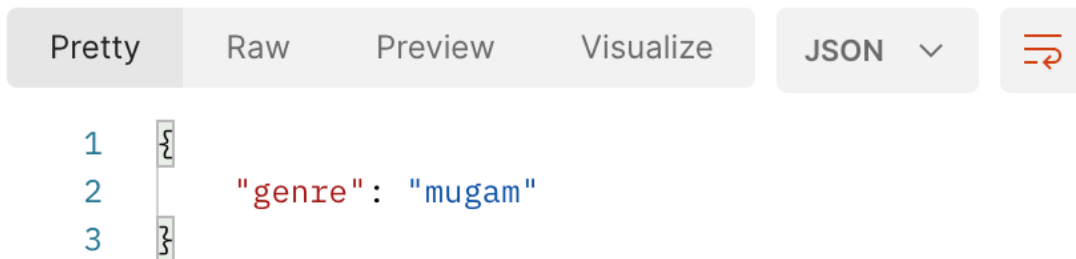


Figure 2.24 Result

Out of 14 segments, 9 was mugham, 2 was classical and surprisingly 3 was pop (Fig 2.25).

```
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 6 mfcc: classical
target: mugam
predicted: 0 mfcc: pop
target: mugam
predicted: 0 mfcc: pop
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 0 mfcc: pop
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 4 mfcc: mugam
target: mugam
predicted: 6 mfcc: classical
target: mugam
predicted: 4 mfcc: mugam
{'mugam': 9, 'classical': 2, 'pop': 3}
Genre is: mugam
```

Figure 2.25 Result in details

Another song I test is positions by Ariana Grande which is a pop song.

KEY	VALUE
<input checked="" type="checkbox"/> genre	pop
<input checked="" type="checkbox"/> song	pop_Ariana_Grande_-_Positions.mp3 <input type="text" value="x"/>
Key	Value

Figure 2.26 Input

Pretty Raw Preview Visualize JSON

```
1 {
2   "genre": "pop"
3 }
```





```

target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 0 mfcc: pop
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 0 mfcc: pop
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 7 mfcc: rock
target: rock
predicted: 0 mfcc: pop
{'rock': 11, 'pop': 3}
Genre is: rock

target: country
predicted: 10 mfcc: jazz
target: country
predicted: 9 mfcc: country
target: country
predicted: 9 mfcc: country
target: country
predicted: 7 mfcc: rock
target: country
predicted: 9 mfcc: country
target: country
predicted: 9 mfcc: country
target: country
predicted: 9 mfcc: country
target: country
predicted: 9 mfcc: country
target: country
predicted: 6 mfcc: classical
target: country
predicted: 9 mfcc: country
target: country
predicted: 9 mfcc: country
target: country
predicted: 6 mfcc: classical
target: country
predicted: 6 mfcc: classical
target: country
predicted: 10 mfcc: jazz
target: country
predicted: 6 mfcc: classical
{'jazz': 2, 'country': 7, 'rock': 1, 'classical': 4}
Genre is: country

```

Figure 2.28 Result for “going under” and “can’t find the time”

target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 10 mfcc: jazz	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 1 mfcc: metal	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 4 mfcc: mugam
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 7 mfcc: rock
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 3 mfcc: blues
target: classical	target: jazz
predicted: 6 mfcc: classical	predicted: 10 mfcc: jazz
{'classical': 12, 'jazz': 1, 'metal': 1}	{'jazz': 11, 'mugam': 1, 'rock': 1, 'blues': 1}
Genre is: classical	Genre is: jazz

Figure 2.29 Result for “barcarole” and bensound

### 3 CONCLUSION

The purpose of this study is to present deep learning algorithms for music genre detection. The first segment is devoted to GTZAN data, which consists of 100 songs divided into ten genres. The dataset has already been created, and I have trained using several segments, such as 1 and 10, by using NN, CNN, and LSTM. The second portion of the investigation was carried out using a custom-made dataset. This dataset was compiled by me from several sources containing the top 100 songs from 2000 to the present day for a period of two decades. I more than increased the quantity of songs and introduced a new genre, Mugam, which is traditional music from Azerbaijani culture. The following segments have been used in this research: 1, 3, 6, 10, 15, and 30. The greatest score I could get for the first phase of the task was 85 percent training accuracy and 75 percent validation accuracy, respectively. For the second, I came up with a variety of possible findings, but I ultimately decided on the models with 72 percent and 71 percent training and testing accuracy for 15 segments, and 80 percent and 75 percent training and testing accuracy for 30 segments. Based on the results of the tests, I discovered that the most frequently misunderstood genres are metal and rock, pop and hip hop, all of which have small distinctions.

### 4 FUTURE WORK

Using a variety of approaches, this investigation might be carried on indefinitely. It is possible to extract spectrograms of the current tracks together with their labels. The next step will be to train the model with different parameters, possibly with cross validation as well, to determine which parameters produce the best accuracy. This will be accomplished with the use of Convolutional Networks and perhaps, with RNN-LSTM. In some of the literature evaluations, they have also used a variety of approaches and attributes, such as gaussian distributions, chroma, pitch, and tempo (speed). That would be an excellent addition to the current dataset, as it would likely increase its precision while also perhaps improving its accuracy. The advancement of this research could be to identify the genres that are similar such as pop and hip hop, or rock and metal. When it comes to song structure, hip hop is more complex than pop, with a lot of rapping and scratching, as well as free verses and beatbox [56]. When compared to hip hop music, pop songs typically feature fewer lyrics and shorter song lengths [56].

### REFERENCES

- [1] McCraty, R., Barrios-Choplin, B., Atkinson, M., & Tomasino, D. (1998). The effects of different types of music on mood, tension, and mental clarity. *Alternative therapies in health and medicine*, 4(1), 75-84.
- [2] Samson, Jim. "Genre". In *Grove Music Online*. Oxford Music Online. Accessed March 4, 2012.
- [3] 2003. Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. Association for Computing Machinery, New York, NY, USA.
- [4] Sturm, B.L.: The State of the Art Ten Years After a State of the Art: Future Research in Music Information Retrieval. *Journal of New Music Research* 43(2), 147–172 (2014)
- [5] Feature extraction for musical genre classification mus15. (n.d.). Retrieved April 15, 2022, from

<https://hpac.cs.umu.se/teaching/sem-mus-15/Merkelbach.pdf>

- [6] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *Speech and Audio Processing, IEEE Transactions on*, 10(5):293–302, Jul 2002.
- [7] Data sets - marsyas. (n.d.). Retrieved April 16, 2022, from <http://marsyas.info/downloads/datasets.html>
- [8] Olteanu, A. (2020, March 24). GTZAN dataset - music genre classification. Kaggle. Retrieved April 16, 2022, from <https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification>
- [9] "Intangible Cultural Heritage – The Azerbaijani Mugham". Archived from the original on 2011-07-04. Retrieved 2009-06-04.
- [10] Hunt, Ken. "Alim Qasimov and the domino principle". [kenhunt.doruzka.com](http://kenhunt.doruzka.com). Archived from the original on 11 December 2012. Retrieved 26 June 2014.
- [11] Librosa. *librosa*. (n.d.). Retrieved April 16, 2022, from <https://librosa.org/doc/latest/index.html>
- [12] An introduction to the frequency domain - blog: Splice. *Blog Splice*. (n.d.). Retrieved April 17, 2022, from <https://splice.com/blog/frequency-domain-introduction/>
- [13] Bevelacqua, P. (n.d.). Fourier transforms. *Fourier Transform*. Retrieved April 18, 2022, from <https://www.thefouriertransform.com/>
- [14] Kehtarnavaz, N. (2008). Frequency Domain Processing. *Digital Signal Processing System Design*, 175–196. doi:10.1016/b978-0-12-374490-6.00007-6
- [15] Window size. *Introduction - Window Size*. (n.d.). Retrieved April 19, 2022, from <https://support.ircam.fr/docs/AudioSculpt/3.0/co/Window%20Size.html#:~:text=The%20window%20size%20represents%20a,and%20changes%20of%20the%20signal>.
- [16] Jeon, H., Jung, Y., Lee, S., & Jung, Y. (2020). Area-efficient short-time Fourier transform processor for time-frequency analysis of non-stationary signals. *Applied Sciences*, 10(20), 7208. <https://doi.org/10.3390/app10207208>
- [17] Maheshwari, H. (2021, September 10). Terms you need to know to start speech processing with Deep Learning. *Medium*. Retrieved April 19, 2022, from <https://towardsdatascience.com/all-you-need-to-know-to-start-speech-processing-with-deep-learning-102c916edf62#:~:text=Hop%20length%20is%20the%20length,portion%20of%20the%20window%20length>.
- [18] Herrera-Boyer, P., Klapuri, A., & Davy, M. (n.d.). Automatic classification of pitched Musical Instrument sounds. *Signal Processing Methods for Music Transcription*, 163–200. [https://doi.org/10.1007/0-387-32845-9\\_6](https://doi.org/10.1007/0-387-32845-9_6)
- [19] Preston Cram. (2021, December 1). Are Music genres important? 3 ways they improve our experience with music. *Preston Cram*. Retrieved April 19, 2022, from <https://www.prestoncram.com/post/are-music-genres-important-3-ways-they-improve-our-listening-experience#:~:text=Music%20genres%20provide%20us%20with,music%20on%20a%20personal%20level>.
- [20] McKay, Cory & Fujinaga, Ichiro. (2006). *Musical Genre Classification: Is It Worth Pursuing and How Can It be Improved?*.
- [21] North, A. C., & Hargreaves, D. J. (1997). Liking for musical styles. *Musicae Scientiae*, 1(1), 109–128. <https://doi.org/10.1177/102986499700100107>
- [22] Tekman, H. G., & Hortacsu, N. (2002). Aspects of stylistic knowledge: What are different styles like and why do we listen to them? *Psychology of Music*, 30(1), 28–47. <https://doi.org/10.1177/0305735602301005>
- [23] C. McKay, and I. Fujinaga. "Automatic Music Classification and the Importance of Instrument Identification," in *Proceedings of the Conference on Interdisciplinary Musicology*, 2005.
- [24] Y. KIKUCHI, N. AOKI and Y. DOBASHI, "A Study on Automatic Music Genre Classification Based on the Summarization of Music Data," 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC), 2020, pp. 705-708, doi: 10.1109/ICAIIIC48513.2020.9065046.
- [25] Y. Atahan, A. Elbir, A. Enes Keskin, O. Kiraz, B. Kirval and N. Aydin, "Music Genre Classification Using Acoustic Features and Autoencoders," 2021 Innovations in Intelligent Systems and Applications Conference (ASYU), 2021, pp. 1-5, doi: 10.1109/ASYU52992.2021.9598979.
- [26] B. Liang and M. Gu, "Music Genre Classification Using Transfer Learning," 2020 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2020, pp. 392-393, doi: 10.1109/MIPR49039.2020.00085.

- [27] George, T., Georg, E., & Perry, C. (2001, October). Automatic musical genre classification of audio signals. In Proceedings of the 2nd international symposium on music information retrieval, Indiana (Vol. 144).
- [28] Music genre classification - stanford university. (n.d.). Retrieved April 19, 2022, from <http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf>
- [29] Musical genre Classification - hpac.cs.umu.se. (n.d.). Retrieved April 19, 2022, from <https://hpac.cs.umu.se/teaching/sem-mus-15/reports/Muellers.pdf>
- [30] K. Markov and T. Matsui, "Music Genre and Emotion Recognition Using Gaussian Processes," in IEEE Access, vol. 2, pp. 688-697, 2014, doi: 10.1109/ACCESS.2014.2333095.
- [31] H. Nakamura, H. Huang and K. Kawagoe, "Detecting Musical Genre Borders for Multi-label Genre Classification," 2013 IEEE International Symposium on Multimedia, 2013, pp. 532-533, doi: 10.1109/ISM.2013.108.
- [32] Masood, Sarfaraz. (2014). Genre classification of songs using neural network. 10.1109/ICCCT.2014.7001506.
- [33] Nair, P. (2018, July 27). The dummy's guide to MFCC. Medium. Retrieved April 19, 2022, from <https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd>
- [34] MFCC technique for speech recognition. Analytics Vidhya. (2021, June 13). Retrieved April 19, 2022, from <https://www.analyticsvidhya.com/blog/2021/06/mfcc-technique-for-speech-recognition/>
- [35] Crypto. Practical Cryptography. (n.d.). Retrieved April 19, 2022, from <http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/>
- [36] Brownlee, J. (2020, August 14). What is deep learning? Machine Learning Mastery. Retrieved April 14, 2022, from <https://machinelearningmastery.com/what-is-deep-learning/>
- [37] By: IBM Cloud Education. (n.d.). What is deep learning? IBM. Retrieved April 14, 2022, from <https://www.ibm.com/cloud/learn/deep-learning#:~:text=Deep%20learning%20is%20a%20subset,from%20large%20amounts%20of%20data.>
- [38] Luhanawal, V. (2020, October 24). Forward propagation in Neural Networks-simplified math and code version. Medium. Retrieved April 22, 2022, from <https://towardsdatascience.com/forward-propagation-in-neural-networks-simplified-math-and-code-version-bbfcfe6f9250>
- [39] Brownlee, J. (2020, August 14). Crash course on multi-layer perceptron neural networks. Machine Learning Mastery. Retrieved April 14, 2022, from <https://machinelearningmastery.com/neural-networks-crash-course/>
- [40] Abirami, S., & Chitra, P. (2020). Energy-efficient edge based real-time healthcare support system. The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases, 339–368. doi:10.1016/bs.adcom.2019.09.0
- [41] Chen, J. (2022, February 8). Neural network definition. Investopedia. Retrieved April 14, 2022, from <https://www.investopedia.com/terms/n/neuralnetwork.asp#:~:text=A%20neural%20network%20is%20a,organic%20or%20artificial%20in%20nature.>
- [42] Dansbecker. (2018, May 7). Rectified linear units (ReLU) in Deep learning. Kaggle. Retrieved April 22, 2022, from <https://www.kaggle.com/code/dansbecker/rectified-linear-units-relu-in-deep-learning/notebook>
- [43] Mishra, M. (2020, September 2). Convolutional Neural Networks, explained. Medium. Retrieved April 22, 2022, from <https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939>
- [44] Mostafa, S., & Wu, F.-X. (2021, July 23). Diagnosis of autism spectrum disorder with convolutional autoencoder and structural MRI images. Neural Engineering Techniques for Autism Spectrum Disorder. Retrieved April 22, 2022, from <https://www.sciencedirect.com/science/article/pii/B978012822822700003X>
- [45] What is a convolutional layer? Databricks. (2020, May 15). Retrieved April 23, 2022, from <https://databricks.com/glossary/convolutional-layer>
- [46] Ke, Q., Liu, J., Bennamoun, M., An, S., Sohel, F., & Boussaid, F. (2018, May 25). Computer vision for human-machine interaction. Computer Vision for Assistive Healthcare. Retrieved April 23, 2022, from <https://www.sciencedirect.com/science/article/pii/B9780128134450000058>
- [47] Balachandran, S. (2020, March 8). Machine learning - max & average pooling. DEV Community. Retrieved April 23, 2022, from <https://dev.to/sandeepbalachandran/machine-learning-max-average-pooling-1366#:~:text=There%20are%20two%20types%20of,image%20covered%20by%20the%20Kernel.>

- [48] DeepAI. (2019, May 17). Stride (machine learning). DeepAI. Retrieved April 23, 2022, from <https://deepai.org/machine-learning-glossary-and-terms/stride#:~:text=Stride%20is%20a%20component%20of,over%20the%20image%20or%20video>.
- [49] Zhang, & Wang, & Xu, Dongdong & Chen,. (2019). Research on Scene Classification Method of High-Resolution Remote Sensing Images Based on RFPNet. *Applied Sciences*. 9. 2028. 10.3390/app9102028.
- [50] Nielsen, M. A. (1970, January 1). Neural networks and deep learning. Retrieved April 23, 2022, from <http://neuralnetworksanddeeplearning.com/chap2.html>
- [51] Verma, S. (2022, April 5). Understanding 1D and 3D convolution neural network: Keras. *Medium*. Retrieved April 23, 2022, from <https://towardsdatascience.com/understanding-1d-and-3d-convolution-neural-network-keras-9d8f76e29610>
- [52] CNN Explainer. Polo Club of Data Science @ Georgia Tech: Human-Centered AI, Deep Learning Interpretation & Visualization, Cybersecurity, Large Graph Visualization and Mining. (n.d.). Retrieved April 23, 2022, from <https://poloclub.github.io/cnn-explainer/>
- [53] Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2020, November 14). 1d convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*. Retrieved April 23, 2022, from <https://www.sciencedirect.com/science/article/pii/S0888327020307846>
- [54] Donges, N. (n.d.). A guide to RNN: Understanding recurrent neural networks and LSTM Networks. *Built In*. Retrieved April 23, 2022, from <https://builtin.com/data-science/recurrent-neural-networks-and-lstm>
- [55] Julita. (2011, February 20). Difference between hip hop and pop. *Difference Between Similar Terms and Objects*. Retrieved April 23, 2022, from <http://www.differencebetween.net/miscellaneous/difference-between-hip-hop-and-pop/>