

Speech Processing Final Project

Note Recognition from an Audio File



Project Guide: **Dr. Jagmohan Chauhan**

Group Member:

- Neil Dave (23PGAI0049) – Neil.d23pgai@jioinstitute.edu.in
- Yash Turakhia (23PGAI0037) – yash.t23pgai@jioinstitute.edu.in

Table of Content

Abstract.....	3
Motivation.....	3
Background	4
Dataset Description.....	4
Methodology.....	5
Fourier Transform.....	5
Note Frequency Chart.....	6
Process Flow.....	7
Results.....	8
Conclusion.....	10

Abstract

The primary goal of the project is to develop an algorithm that detects the principal or predominant note in a sound file. Through this project, we showcase an algorithm that takes an audio wav file as input, performs analysis, and produces the note and frequency of the note as output in form of a graph. We then match the received frequency with pre-defined list of frequency to detect note from the file. It uses Fourier Transform to extract the frequency domain features of the audio file, which are used to detect the peak frequency. The Fourier Transform is a mathematical technique that converts a time-domain signal into a frequency-domain signal. It decomposes a signal into its constituent frequencies and their amplitudes, which helps to identify the frequencies in the signal. To determine the peak frequency, we use a peak detection algorithm that looks for maxima in the frequency domain features. The highest local maximum is considered the peak frequency. The peak frequency is then matched with a predefined databases of notes and their frequencies to determine the note.

Motivation

Music has always been an integral part of human culture, and people have been producing and enjoying music for thousands of years. However, music production and analysis have become more sophisticated in recent times, and digital audio technology has made it easier to create, record, and analyse music. One of the most important aspects of music analysis is note detection, which involves identifying the predominant note or notes in a piece of music. Note detection has various applications, including automatic transcription, music recognition, and instrument tuning.

The aim of this project is to develop an algorithm that can detect the note in a sound file. The algorithm is designed to be simple and easy to implement, while still providing accurate results. The motivation for this project is to provide a tool for music enthusiasts and professionals that can help them quickly identify the notes in a piece of music.

Background

Sound is a wave that propagates through a medium, such as air or water. A sound wave can be represented as a function of time and amplitude, where time represents the horizontal axis, and amplitude represents the vertical axis. The frequency of a sound wave determines its pitch, and the amplitude determines its volume. In musical terms, a sound wave with a high frequency corresponds to a high-pitched note, and a sound wave with a low frequency corresponds to a low-pitched note.

Understanding frequencies is essential for musicians, sound engineers, and producers as it enables them to effectively manipulate and regulate the sound in a recording or live performance. By adjusting the frequencies of various instruments and sounds, a balanced and melodious mix can be achieved that is pleasing to the human ear.

Moreover, having knowledge of frequencies can assist in detecting and resolving problems with a recording, such as unwanted resonances or unclear low-end frequencies. For instance, acoustic guitars or vocals recordings may contain resonant noises that could interfere with a take. Even drummers can benefit from coordinating notes to frequencies by tuning their drums and percussion pieces, which can result in a more harmonious relationship with the song's key signature.

Dataset Description

We have used various musical note wav or mp3 file to predict the note of the music. There are various note musical files used such as C0, C1, C4, D, E, G, etc. The music file datasets used in this project are taken from this website (<https://freesound.org/>).

Methodology

The project was achieved using python script. We have defined a Python script with below libraries used for implementation:

- **numpy (as np):** This is a popular library for scientific computing in Python. It provides support for multi-dimensional arrays, mathematical functions, and tools for working with arrays.
- **math:** This is a built-in Python module that provides mathematical functions.
- **wave:** This module in the Python standard library provides support for reading and writing WAV files.
- **os:** This is a module in the Python standard library that provides a way of interacting with the file system.
- **struct:** This is a module in the Python standard library that provides functions for converting between Python values and C structs represented as Python bytes objects.
- **matplotlib.pyplot (as plt):** This is a plotting library for Python. It provides functions for creating a variety of plots, including line plots, scatter plots, and histograms.

Fourier Transform

We use Fourier Transformation to detect note pitch. The Fourier Transform is a mathematical technique used to convert a signal from the time domain to the frequency domain. In signal processing, the Fourier Transform is used to analyse a signal and identify its frequency components. It decomposes a signal into its constituent frequencies and their amplitudes, allowing us to understand the signal's frequency content. This is important because many real-world signals, including audio signals and images, can be more effectively analysed, and processed in the frequency domain. The mathematical representation is as follows:

$$F(\omega) = \int f(t)e^{-i\omega t} dt$$

Where,

$F(\omega)$ is the frequency domain representation of the signal **$f(t)$**

$f(t)$ is the time-domain signal.

ω is the frequency variable in radians per second

i is the imaginary unit, where $i^2 = -1$

\int represents integration over the time domain.

Using a mathematical formula, the Fourier Transform operates by transforming a signal from the time domain to the frequency domain. The formula takes the time-domain signal as input and produces a frequency-domain representation of the signal as output. This frequency-domain representation is often called the Fourier Spectrum or the Power Spectrum.

The Fourier Transform can be computed using different methods, such as the Fast Fourier Transform (FFT) algorithm, which is a computationally efficient implementation of the Fourier Transform. The FFT algorithm is commonly used in digital signal processing applications because it can compute the Fourier Transform of a signal quickly and accurately.

Note Frequency Chart

A note is a fundamental element in music that represents a sound with a specific pitch. It is associated with a letter name and a frequency expressed in Hertz (Hz), which indicates the number of vibrations per second. In the Western chromatic scale, there are 12 semitones ranging from C0 at approximately 16.35 Hz to B10 at around 31,608 Hz. However, the human ear can only perceive frequencies between 20 Hz and 20,000 Hz, with frequencies above 8,000 Hz being referred to as upper-harmonics. These frequencies are not considered fundamental. Musicians use octave ranges to classify and identify audible notes.

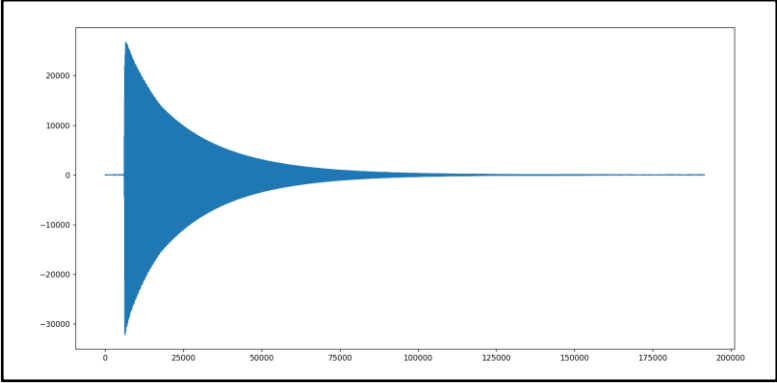
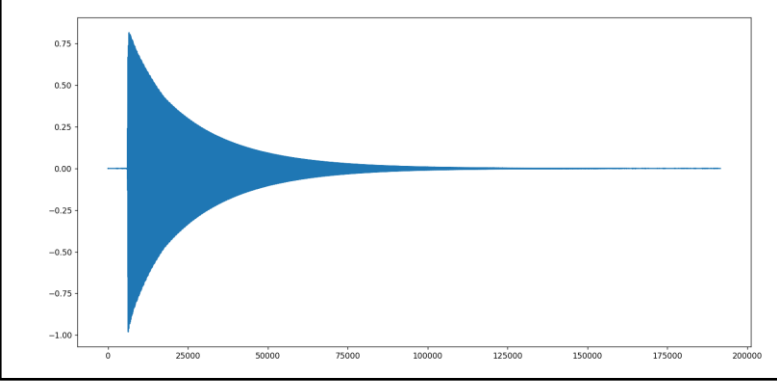
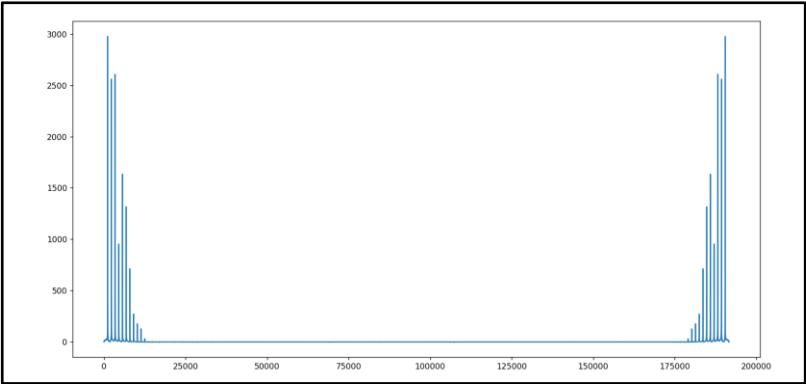
NOTE	OCTAVE 0	OCTAVE 1	OCTAVE 2	OCTAVE 3	OCTAVE 4	OCTAVE 5	OCTAVE 6	OCTAVE 7	OCTAVE 8
C	16.35	32.7	65.41	130.81	261.63	523.25	1046.5	2093	4186.01
C#/Db	17.32	34.65	69.3	138.59	277.18	554.37	1108.73	2217.46	4434.92
D	18.35	36.71	73.42	146.83	293.66	587.33	1174.66	2349.32	4698.63
D#/Eb	19.45	38.89	77.78	155.56	311.13	622.25	1244.51	2489.02	4978.03
E	20.6	41.2	82.41	164.81	329.63	659.25	1318.51	2637.02	5274.04
F	21.83	43.65	87.31	174.61	349.23	698.46	1396.91	2793.83	5587.65
F#/Gb	23.12	46.25	92.5	185	369.99	739.99	1479.98	2959.96	5919.91
G	24.5	49	98	196	392	783.99	1567.98	3135.96	6271.93
G#/Ab	25.96	51.91	103.83	207.65	415.3	830.61	1661.22	3322.44	6644.88
A	27.5	55	110	220	440	880	1760	3520	7040
A#/Bb	29.14	58.27	116.54	233.08	466.16	932.33	1864.66	3729.31	7458.62
B	30.87	61.74	123.47	246.94	493.88	987.77	1975.53	3951.07	7902.13

Process Flow

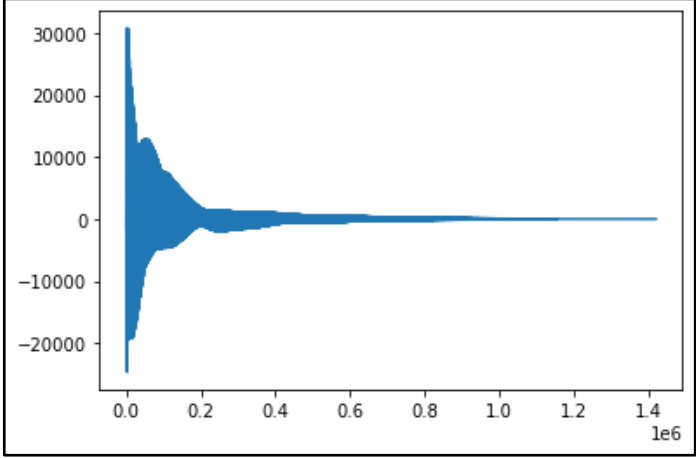
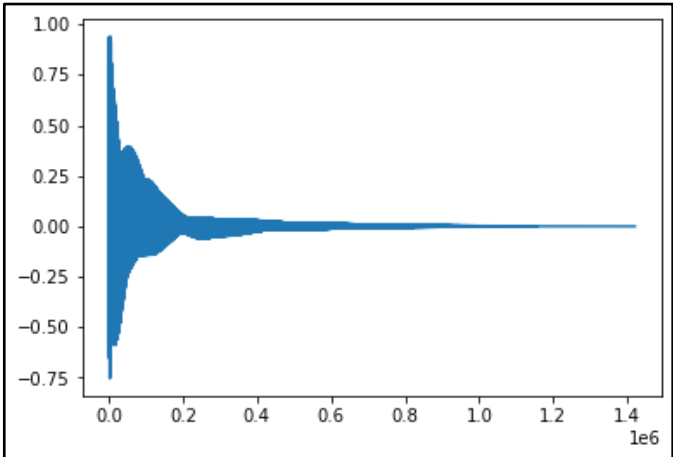
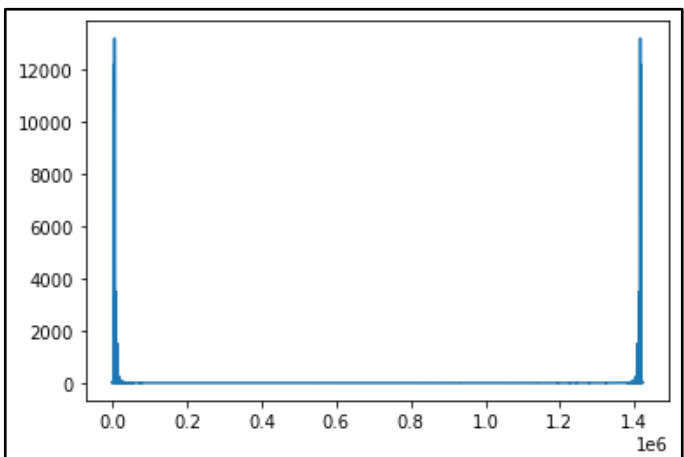
1. The Script takes an audio file as an input: The audio file is passed through python script for predicting the note of audio file.
2. The audio file is read and stored as a NumPy array: The audio file is read using a library like "wave" and stored as a NumPy array, which is a widely used numerical computing library for Python.
3. The sound wave and Fourier transform of the audio file are plotted: The sound wave is plotted using the "matplotlib" library, which is a plotting library for Python. Additionally, the Fourier transform of the sound wave is calculated using the "numpy.fft" function and plotted using "matplotlib".
4. Peak detection is performed to determine the frequency of the note present in the audio: A peak detection algorithm is applied to the Fourier transform to identify the highest peak, which corresponds to the predominant frequency of the audio. This frequency is then used to determine the note present in the audio.
5. The frequency is mapped to the closest note in a pre-defined frequency database: A frequency database is created, which is a NumPy array that maps frequencies to their corresponding note names. The detected frequency is then mapped to the closest note in the frequency database using a simple algorithm that matches the frequency to the nearest note frequency in the database.
6. The script output returns the name of the note that is being detected from the input.

Results

Input d2.wav file

Raw Music File	 <p>The waveform shows a sharp initial peak followed by a gradual decay over time. The y-axis represents amplitude from -30000 to 20000, and the x-axis represents time from 0 to 200,000 samples.</p>
Output After Applying Normalization	 <p>The waveform shows the signal after normalization, with the y-axis scaled from -1.00 to 0.75. The overall shape remains the same as the raw file, but the amplitude is reduced to fit within the normalized range.</p>
Output (Fourier transformation of the sound file as a graph)	 <p>The spectrogram displays frequency components over time. The y-axis represents frequency from 0 to 3000 Hz, and the x-axis represents time from 0 to 200,000 samples. Two distinct clusters of high-frequency energy are visible at the beginning and end of the file.</p>
Detected Note	<div data-bbox="713 1832 1248 1942">Detected Note = D2</div>

Input e2.wav file

Raw Music File	
Output After Applying Normalization	
Output (Fourier transformation of the sound file as a graph)	
Detected Note	<div data-bbox="828 1883 1244 1955">Detected Note = E2</div>

Conclusion

Through this project, we try build a speech signal processing techniques to detect and recognize musical notes from a given audio signal. The project was build using Python. The approach used in this project is based on the concept of spectral analysis of the audio signal, where the frequency domain of the signal is analysed to identify the musical notes present in it. The project employs techniques like the Fast Fourier Transform (FFT) to identify the pitch of the notes. The performance of the project has been evaluated on a set of audio files containing different types of musical notes, and the results have been reported in this report for few samples of the complete dataset.