Detection of music tempo with beats per minute (BPM) detector

Pedro Macedo
Faculty of Computer and
Information Science
Ljubljana, Slovenia
Email: pf06708@student.uni-lj.si

***Abstract*—The purpose of this project is to classify the *tempo* of each audio file by checking the audio Bits Per Minute (BPM). The implementation will be done in Python by using open source libraries specified for sound analysis.**

I. Introduction

The tempo of a music is a key feature in music that detects the overall speed and rhythm of an audio track. This tempo is typically measured in bits per minute (*BPM*).

The ability to detect and analyse BPM is crucial in several applications, including music recommendation systems, and even fitness apps. Furthermore, this project aims to develop a BPM detector that is capable of analysing audio files to detect the tempo of a music.

Furthermore, *BPM* detection has applications beyond mere analysis. For example, detecting *BPM* of several songs can help a DJ to synchronize the tempo of multiple tracks. By understanding a song's tempo, a wide variety of experiences can be built that improves the user's experiences and preferences.

This project aims to develop a robust and accurate detector that is capable of using the signal processing techniques and available audio analysis libraries to detect the most accurate song BPM and, consequently, its *tempo*. The main goal is to design a system that delivers precise and consistent BPM values across different music genres and, consequently, detects correctly the music's tempo, providing an essential tool for a wide array of musics.

II. Approach

Firstly, the *tempo* detector will set a consistent sample rate. Then, the audio will be divided into smaller time windows in order to detect the *BPM* changes during the song.

To detect the tempo, such as beats, will be identified through onset detection. Onset detection refers to the single instant that marks the beginning of the transient (a noise sound component of short duration and high amplitude). This can be possible by using, for example, the Fourier Transform that will convert the audio signal into a frequency domain, in which the audio signal is broken down into its different component frequencies.

After having all the *BPM* values for each sample window, we can use K-Means clustering to group similar BPM values together. Then, we can use the largest cluster centroid value (this centroid represents the average of a specific cluster BPM values).

Finally, after having the overall BPM value of the audio, the final step is to classify the tempo of the song using a predefined list of musical tempo categories. [1]

III. Technology stack

Since the detector is web-oriented, we will adopt the following technical architecture:

- **Python**: Used to process the audio file and to setup the API for the *web* application. We are also going to use some python libraries, such as, *Pydub*, *SciPy*, *NumPy*, *scikit-learn* and *FastAPI*.
- **React**: Used to create an intuitive, responsive and dynamic web interface.

IV. Evaluation

To ensure that the application is effective and user-friendly, we will conduct tests to see how it behaves in different scenarios. The program must be, not only functional, but also intuitive. Some user will try the final version of the working program and give *feedback* in order to improve the program. We are also using the python library *librosa* to evaluate the program, since this python library also detects the song's tempo by analysing the song. [2]

References

[1] Symphony Nova Scotia, *How do musicians know how fast to play a piece? And why are the terms in Italian?* Available at: https://symphonynovascotia.ca/faqs/symphony-101/how-do-musicians-know-how-fast-to-play-a-piece-and-why-are-the-terms-in-italian/

[2] Vicente G. Reyes, *Getting the tempo of a song using python*, Available at: https://dev.to/highcenburg/getting-the-tempo-of-a-song-using-librosa-4e5b