Music Identification through Audio Fingerprinting

Yashraj Kakkad - AU1841036
Prayag Savsani - AU1841035
Harshil Mehta - AU1841010
Yashil Depani - AU1841005

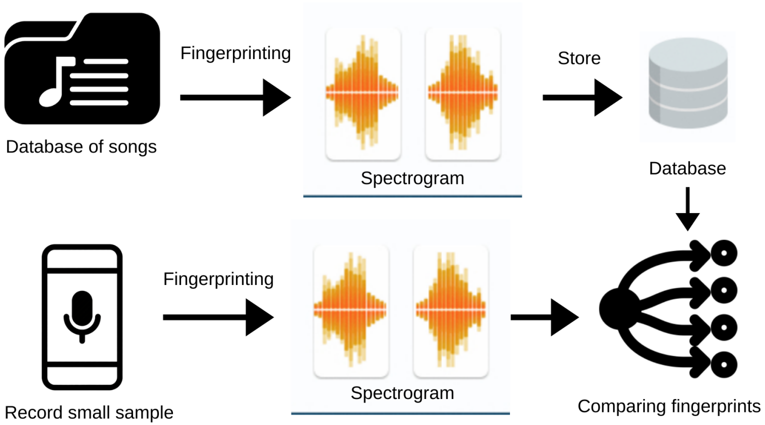
Guided by: Prof. Ashok Ranade

Ahmedabad University

October 20, 2019

Objective

- Extract small chunks of each song and fingerprint the song.
- Store the chunks in an appropriate database schema.
- Fingerprint a small audio sample given by user and identify the song being played.



Sampling

- Music is typically sampled at 44.1 kHz.
- This is because of a theorem by Nyquist and Shannon which requires $f_d \geq 2f_{max}$.
- Maximum sound frequency is of course 20 kHz which leads our sampling rate to be 44.1 kHz.

Problems with sampling rate

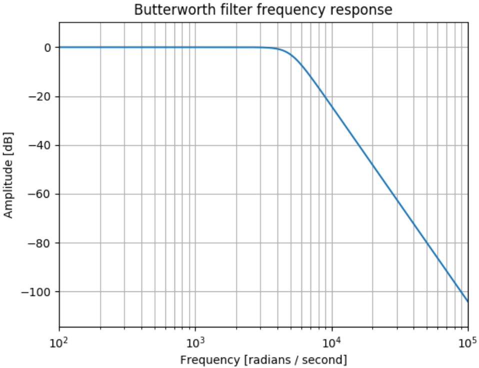
- Performing Fast Fourier Transform on a few hundred songs takes days at such a high sampling rate.
- Therefore we downsample the audio by a factor of 4.
- And as a result, the maximum sound frequency in our audio sample changes to 5 kHz.
- Would it cause any issues?

The song is not the same

- Turns out that the most important part of a song (to us) is below 5 kHz.
- Therefore, for the sake of Fast Fourier Transform, we may simply ignore the higher frequencies.

Aliasing

- We need to filter the higher frequencies in order to avoid aliasing.
- Aliasing: Distortion that results when a signal reconstructed from samples is different from the original continuous signal.
- We achieve the same by filtering the signal before downsampling (using a low pass filter)



Discrete Fourier Transform

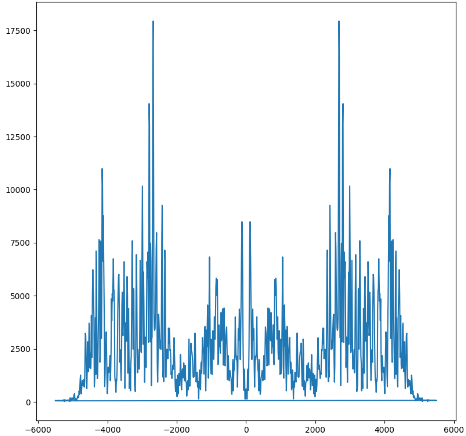
- Gives us the frequency spectrum.
- Formula:

$$X(n) = \sum_{k=0}^{N-1} x[k] e^{-j(2\pi kn/N)}$$

- To obtain frequencies of each small part of the song for spectral analysis, we have to apply DFT on each small part of the song.
- This small part of the song can be seen as a window of N samples on which the DFT is performed.
- Such windows are extracted using a window function.

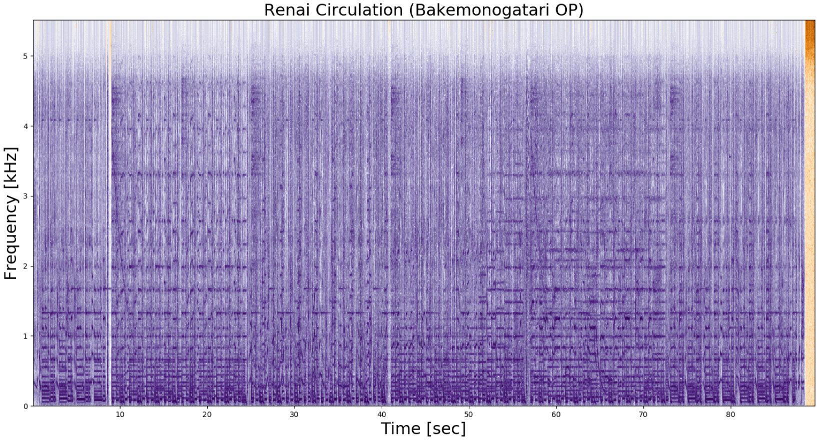
Fast Fourier Transform

- Discrete Fourier Transform requires $\mathcal{O}(N^2)$ computations where N is the number of samples.
- Today's Fast Fourier Transform implementations are $\mathcal{O}(NlogN)$, which is a huge improvement.



Spectrogram

- A three dimensional graph where:
  - X-axis indicates time
  - Y-axis indicates frequency
  - Color indicates amplitude of a frequency at a certain time



Spectogram Filtering/Fingerprinting

- We only have to keep the loudest notes
- Simple solution:
    - For 512 bins of frequencies, we create six logarithmic bands to segregate the bins.
        - very low sound band (0-10)
        - low sound band (10-20)
        - low-mid sound band (20-40)
        - mid sound band (40-80)
        - mid-high sound band (80-160)
        - high sound band (160-511)
    - Keep the strongest bin of frequencies in each band.
    - Do the same procedure to the recorded data from the user.

Music Indexing and Matching

- We store these frequencies as a hashed value in our database.
- We compare the user data with every song's data. We can compute the offset (time delay) by subtracting their positions.
- If we have a lot of hashes with matching offsets, we've found our song.

The End