CreateBase.ai

Report

Sriya Adiraju

**Data Preparation**: Script was loaded primarily using the librosa package. Librosa.load function within the package helped us load the audio file and retrieve two values, classical\_signal1 and sr. Classical\_signal1 holds the audio signal (it is a one-dimensional NumPy array) representing the audio waveform. Each element in the array depicted the amplitude of the audio signal. Sr represents the sampling rate.

**FFT Analysis:** frame of 2048 was taken and overlap of 512. These are usually the consensus. Frame size is the number of signals in each chunk of the signal. It is usually equal to the window size, or some cases more. Overlap tells us how much to the right do we hop after each window. It is important because it prevents sharp transitions which would result in discontinuity. 512, to my knowledge represents around 70% overlap to the previous window.

**Spectrogram Generation:**  I calculated the spectrogram in two ways.

Firstly, a code that calculates the STFT and directly computes the spectrogram using the square of the absolute values.

Secondly, a code that calculates the STFT and converts the amplitude spectrogram to decibels. I choose to do this because it enhanced the visibility of features in the spectrogram.

**Data Interpretation (Spectrogram):** I found the audio signal periodic and discrete from the spectrogram. There are parts of the audio with less energy/lower amplitude (bluer regions) and are parts of the audio with high energy/ high amplitude (reddish hue regions). You can see harmonics and overtones as well.

**Digital Fingerprint Development:** Peaks were found using threshold approach, which was a hyperparameter set by us. Peaks in this spectrogram often correspond to distinctive events or characteristics in the audio signal, such as musical notes, transients, or specific sound patterns. While I only implemented a simple method, it can be made more robust by considering additional features.