Audio Signal Analysis and Visualization using Librosa in Google Colab

Research by Saif Ullah

September 19, 2025

Abstract

This research paper demonstrates the process of analyzing and visualizing audio signals using Python libraries such as Librosa, NumPy, and Matplotlib in the Google Colab environment. The experiment involves uploading an audio file, generating its waveform, computing and displaying the spectrogram, and plotting the frequency spectrum. This work serves as a foundation for audio signal processing, speech analysis, and music information retrieval.

1 Introduction

Audio signal processing plays a vital role in various domains such as speech recognition, music analysis, and machine learning. Visualization of audio signals allows researchers and engineers to understand the structure and frequency components of sound. In this study, we demonstrate how to perform basic audio analysis and visualization using Python libraries.

The objectives of this research are:

- To load and process an audio file in Google Colab.
- To visualize the waveform of the audio.
- To compute and display the spectrogram.
- To analyze the frequency spectrum in a mini visualizer style.

2 Methodology

The experiment was implemented in Google Colab using Python. The following steps summarize the methodology:

2.1 Installing Required Libraries

The libraries Librosa, Matplotlib, and NumPy were installed:

```
!pip install librosa matplotlib numpy
```

2.2 Uploading and Loading the Audio File

The user uploads a '.wav' or '.mp3' file. The audio is then loaded with Librosa:

```
y, sr = librosa.load(filename, sr=None)
```

Here, y is the audio time series and sr is the sampling rate.

2.3 Waveform Visualization

The waveform represents amplitude versus time:

```
librosa.display.waveshow(y, sr=sr, alpha=0.7)
```

This helps in understanding the temporal behavior of the signal.

2.4 Spectrogram Visualization

The Short-Time Fourier Transform (STFT) was applied:

```
X = librosa.stft(y)
Xdb = librosa.amplitude_to_db(abs(X))
librosa.display.specshow(Xdb, sr=sr,
                         x_axis='time', y_axis='hz', cmap='magma')
```

The spectrogram shows how frequency content changes over time.

2.5 Frequency Spectrum

The Fourier Transform was computed using NumPy:

```
fft = np.fft.fft(y)
magnitude = np.abs(fft)[:len(fft)//2]
freq = np.linspace(0, sr/2, len(magnitude))
```

The magnitude spectrum shows the strength of frequency components.

3 Results and Discussion

3.1 Waveform

The waveform clearly represents the temporal variations of the audio signal. Peaks indicate louder parts, while flat sections indicate silence or low energy.

3.2 Spectrogram

The spectrogram provides a time-frequency representation. Bright regions correspond to higher energy in specific frequency bands.

3.3 Frequency Spectrum

The frequency spectrum highlights dominant frequencies in the audio. This is particularly useful in music signal analysis and speech processing.

4 Conclusion

This research successfully demonstrated the process of analyzing and visualizing audio signals in Google Colab. The step-by-step methodology can be extended to advanced applications such as feature extraction, machine learning classification, and real-time audio processing. Future work may include mel-spectrograms, MFCCs, and deep learning-based analysis.

5 References

1. McFee, B. et al. (2015). Librosa: Audio and music signal analysis in Python. *Proceedings of the 14th Python in Science Conference*.

2. Allen, J. B. (1977). Short term spectral analysis, synthesis, and modification by discrete Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*.

3. Google Colab Documentation. Available at: `https://colab.research.google.com/`