Speech Understanding Programming Assignment - Question 2

Subodh Kant (M23CSA531)

January 2025

1 TASK A - windowing techniques for UrbanSound8k dataset

1.1 Introduction

This project report presents an implementation of audio processing techniques applied to the UrbanSound8K dataset. The focus is on generating spectrograms using different windowing functions (Hann Window, Hamming Window, Rectangular Window), applying Mel Spectrogram transformations, and training a Convolutional Neural Network (CNN) for classification.

1.2 Data Preprocessing and Spectrogram Generation

1.2.1 Loading Audio Data

An audio file from the dataset is loaded using torchaudio.load(). The waveform and sample rate are extracted.

1.2.2 Short-Time Fourier Transform (STFT) Parameters

n-fft = 1024, hop-length = 512

1.2.3 Windowing Techniques

Three different windowing functions are applied to the audio signal: Hann Window, Hamming Window, Rectangular Window

1.2.4 Spectrogram Generation

The spectrogram is computed using torch.stft(). Magnitude is extracted and converted to decibels using AmplitudeToDB(). Spectrograms are visualized using matplotlib.

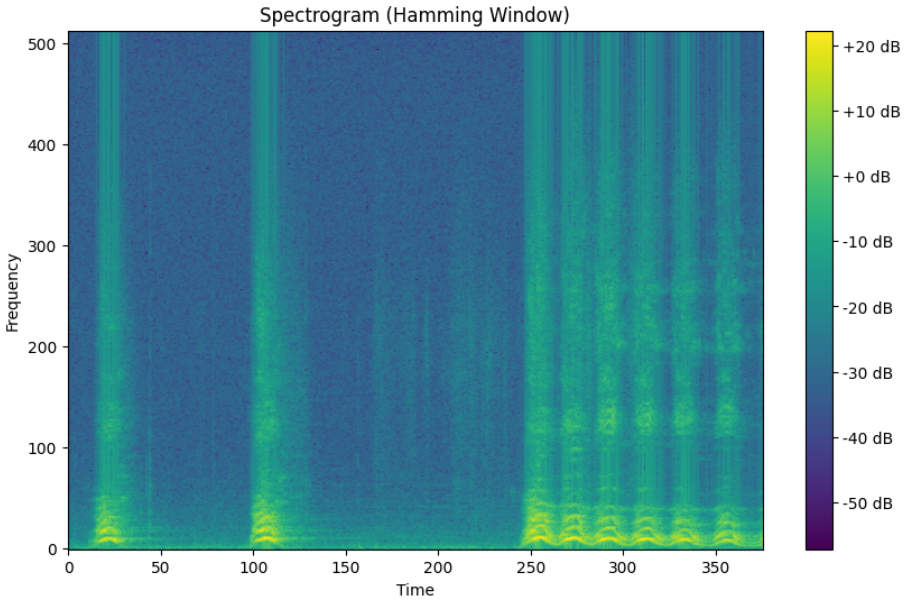

Figure 1: hamming window

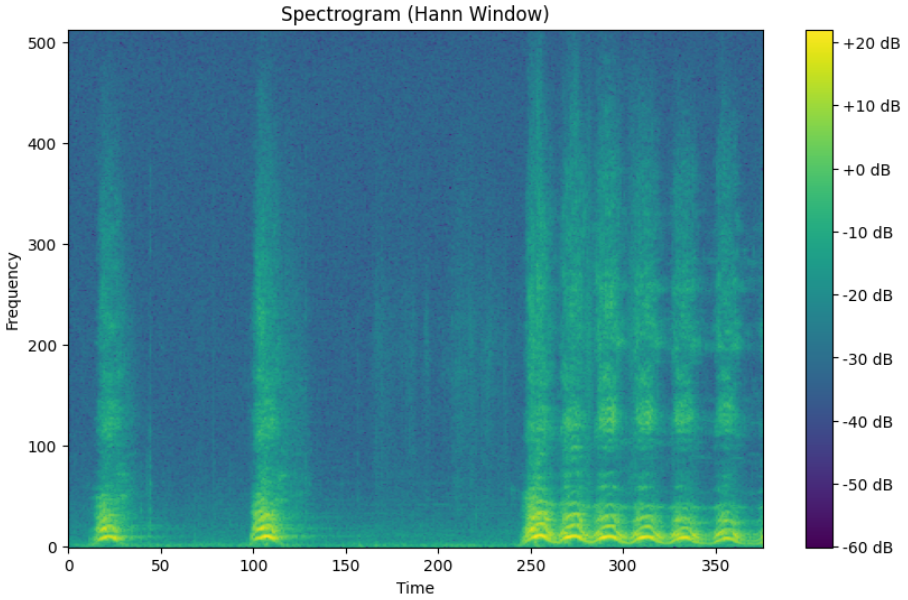

Figure 2: hann window

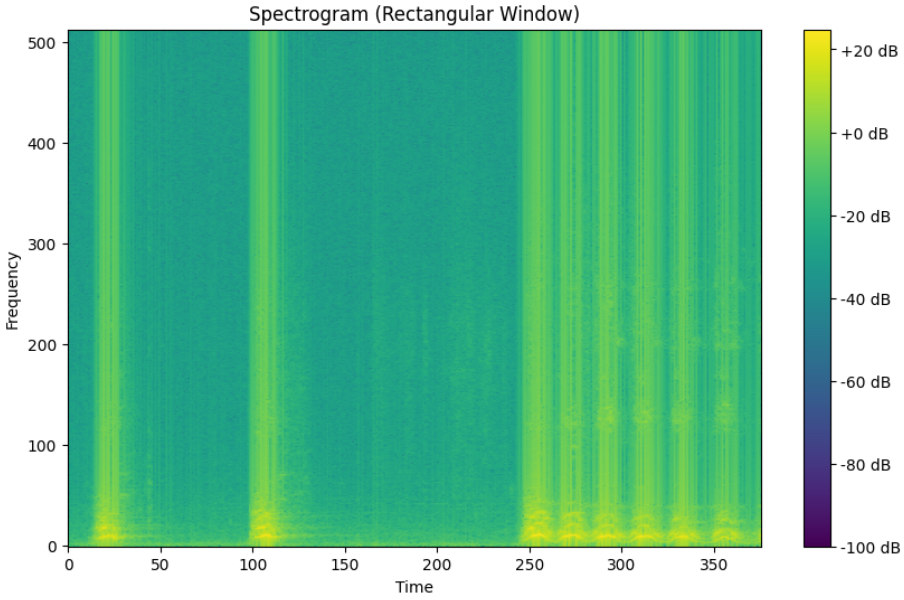

Figure 3: rectangular window

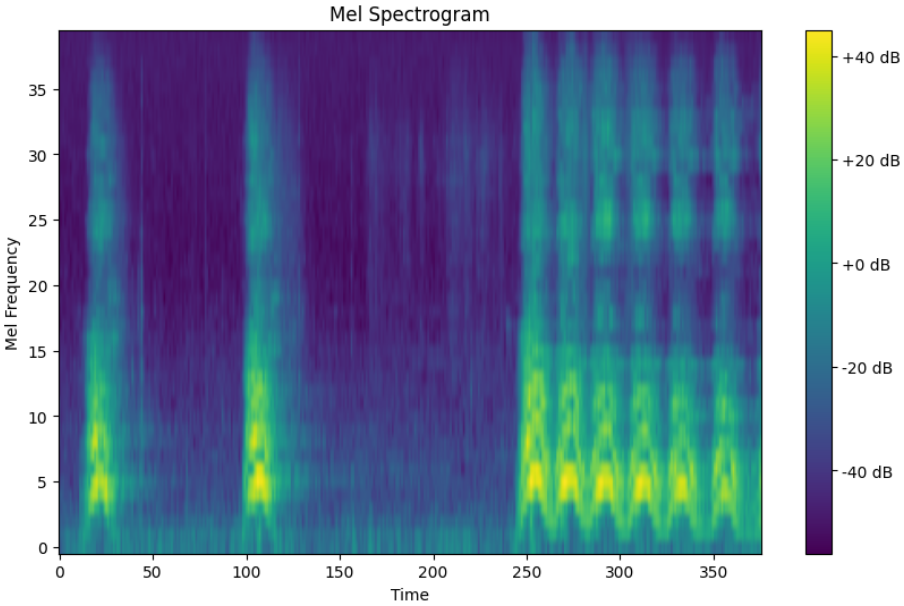

Figure 4: Mel Spectrogram

1.3 Mel Spectrogram Transformation

The Mel Spectrogram transformation [Figure 4] is applied using MelSpectrogram() with n-mels=40. The resulting spectrogram is converted to decibels for visualization.

1.3.1 Dataset Preparation

The metadata from UrbanSound8K.csv is loaded. File paths and labels are extracted.

A helper function Pads or truncates the spectrogram to the specified max length, ensures all spectrograms have a fixed length (maxlength=400).

1.3.2 UrbanSound Dataset Class Implementation

A Dataset class is defined to load audio files and apply transformations. Stereo audio is converted to mono. The Mel Spectrogram is computed for each sample. Spectrograms are padded or truncated to ensure uniform input size.

1.4 CNN-Based Audio Classification

1.4.1 Model Architecture

A simple CNN model is implemented with the following layers:

Conv2D layer (16 filters, kernel size 3x3, ReLU activation, MaxPooling)

Conv2D layer (32 filters, kernel size 3x3, ReLU activation, MaxPooling)

Fully Connected layers with:

- Flatten operation
- Linear layer with 128 units (ReLU activation)
- Output layer with 10 classes

1.4.2 Training Setup

Loss Function: CrossEntropyLoss, Optimizer: Adam with a learning rate of 0.001, Batch Size: 32, Training Epochs: 3

1.4.3 Training Loop

Features are loaded and passed through the CNN model. The loss is computed and backpropagated. The optimizer updates model weights. Debugging print statements verify data shape consistency.

2 TASK B - Spectrogram Analysis of Different Music Genres

2.1 Introduction

Spectrograms provide a visual representation of sound, displaying how frequency content evolves over time. This project analyzes spectrograms of four different music genres: Classical, Rock, Jazz, and Electronic (EDM). By using Short-Time Fourier Transform (STFT), we compare the time-frequency characteristics of each genre.

2.2 Objective

The main objectives of this analysis are:

- To generate spectrograms for different genres.
- To compare their frequency distribution and intensity over time.
- To interpret genre-specific spectral patterns.

2.3 Code Explanation

2.3.1 Dataset

Four songs, one from each genre, were selected:

- Classical: "classical-music.mp3"
- Rock: "rock-music.mp3"
- Jazz: "jazz-music.mp3"
- Electronic (EDM): "electronic-music.mp3"

2.3.2 Tools Used

Python Libraries: librosa, matplotlib, numpy

Spectrogram Generation: STFT with a Hann window function

2.4 Code Implementation

The code follows these steps:

- Load the audio files using librosa.
- Compute the STFT to transform the audio into a time-frequency domain.
- Convert amplitude values to dB scale using librosa.amplitude-to-db().
- Plot the spectrogram using matplotlib.
- Handle errors if a file is missing or incompatible.

2.5 Results and Analysis

Each genre exhibits distinct spectral characteristics:

2.5.1 Classical

Smooth, continuous frequency distribution. Strong harmonics with gradual transitions. Less high-frequency energy.

2.5.2 Rock

High energy across low, mid, and high frequencies. Guitar and percussion create sharp, transient peaks. Dense spectrogram with chaotic patterns.

2.5.3 Jazz

Moderate dynamics with varying instrument tones. Saxophone and piano exhibit clear harmonics. Less dense, highlighting improvisation.

2.5.4 Electronic (EDM)

Strong low-frequency components (bass-heavy beats). Repetitive patterns due to loops and synthesized sounds. High-energy bursts in upper frequencies.

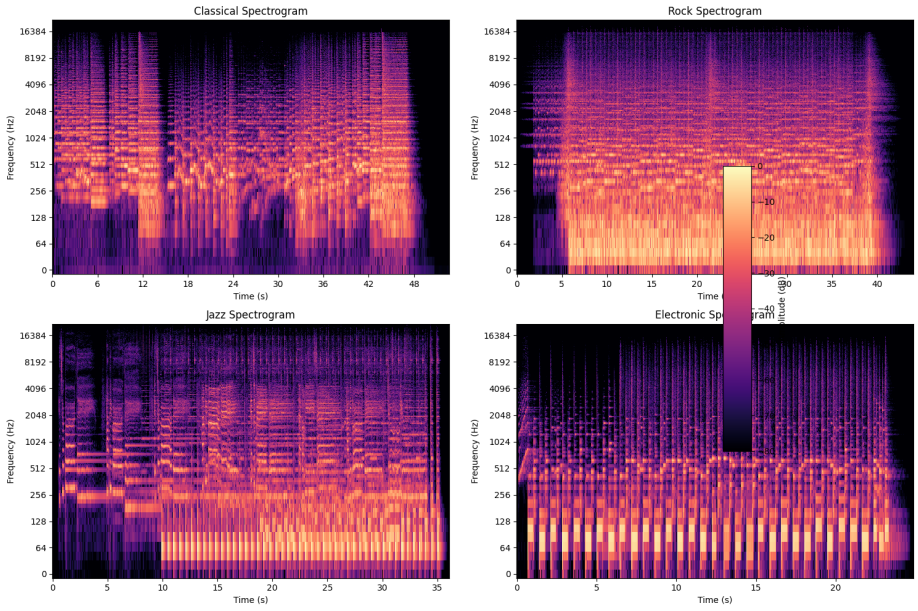

Figure 5: Spectrogram