COURSE PROJECT

Name: Gaddameedi Suvardhan Dileep

Roll Number: B210876EE

Semester: V

Subject: Artificial Neural Networks

Time Spent On Project: 3 Weeks

Number Of Project Members: 1

Audio Classification
with Artificial Neural Networks

ABSTRACT

This report's objective is creating a model using Artificial Neural Networks (ANN) capable of accurately predicting various sound types, encompassing categories such as dog barking, car horns, drilling, siren, engine idling, air conditioners, street music, gunshot, jackhammer, and children playing. Leveraging the UrbanSound8k dataset, which comprises 8732 labelled audio samples distributed across 10 distinct classes, our methodology involves meticulous data preprocessing. We extract essential features, specifically Mel-Frequency Cepstral Coefficients (MFCCs), from each audio file, treating them as independent features, while the class labels serve as dependent features. To develop and evaluate our model, we divide the dataset into training and testing sets. By implementing this approach, our trained model demonstrates an impressive 80% accuracy in predicting sound categories. The metadata associated with the Urban Sound 8k dataset, providing information about file locations and class IDs, further enhances the transparency and interpretability of our model. This project not only serves as an effective tool for sound classification but also highlights the significance of employing advanced machine-learning techniques in audio signal processing.

Audio Signal Exploration

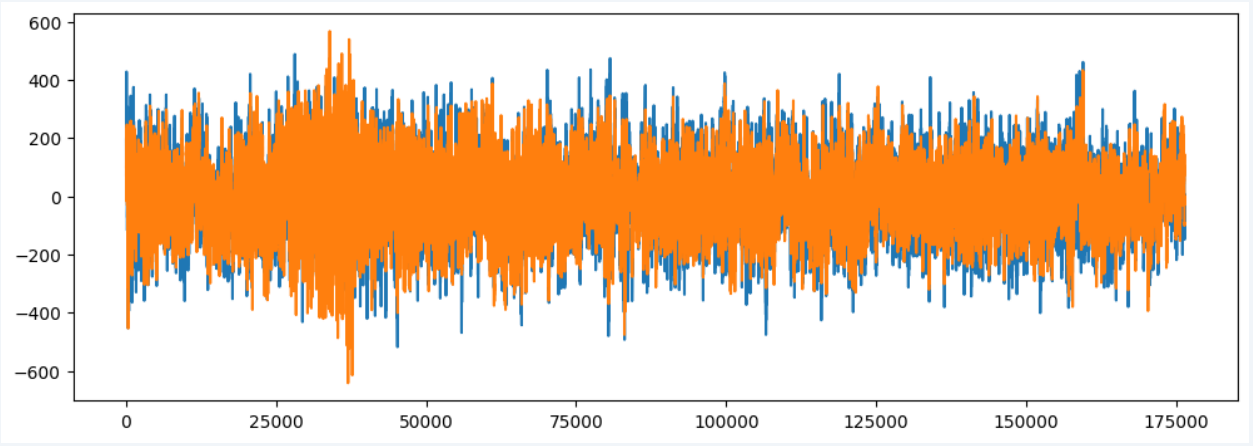
Understanding basic domain knowledge of audio signals is required in audio classification. One crucial aspect is the sample rate, defined as the number of samples captured per second of audio. Moreover, audio signals can exist in different channel configurations, typically categorized as mono or stereo. Mono signals consist of a single audio channel, while stereo signals comprise two channels, allowing for a spatial representation of sound. The presence of multiple channels introduces an additional layer of complexity, as it necessitates handling and processing each channel appropriately during classification tasks. The difference between two sounds is determined by various characteristics present in their respective audio signals. Machine learning models, especially those designed for audio classification, leverage these characteristics for feature extraction(like using MFCCs).

Data Preprocessing

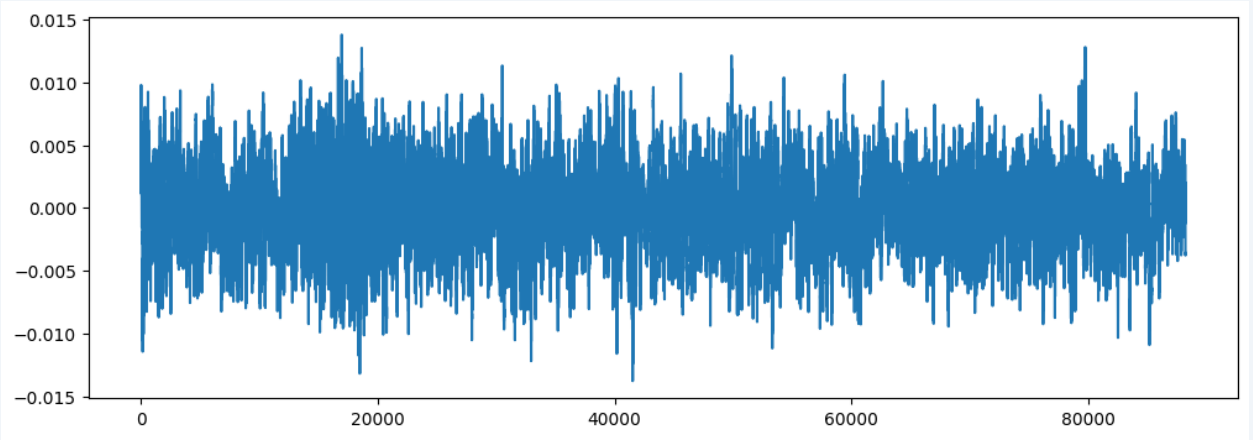
Librosa stands out as a widely adopted and robust library for handling various aspects of audio signal processing. Typically, the SciPy library is employed for fetching essential details such as audio data and sample rates. SciPy effectively sustains the integrity of audio signals, including all channels, providing a solid foundation for further processing.

Librosa, on the other hand, excels in reading audio data and ensuring a consistent sample rate across diverse audio files. Notably, it contributes significantly to the normalization of audio data, standardizing wave signal values between minus one and plus one. The default sample rate assigned by Librosa is 22,050 Hz. One distinctive feature of Librosa is its capability to automatically convert stereo audio signals into mono channels, facilitating a unified representation for subsequent analysis. This mono-channel conversion is particularly valuable, as audio signals are commonly recorded in either mono or stereo configurations, and a consistent format streamlines processing.

The integration of the SciPy library further enhances the preprocessing pipeline by sustaining audio signals with all channels intact, enabling the visualization of two-dimensional features as shown below.



One-dimensional feature visualization, achieved through Librosa as shown below.

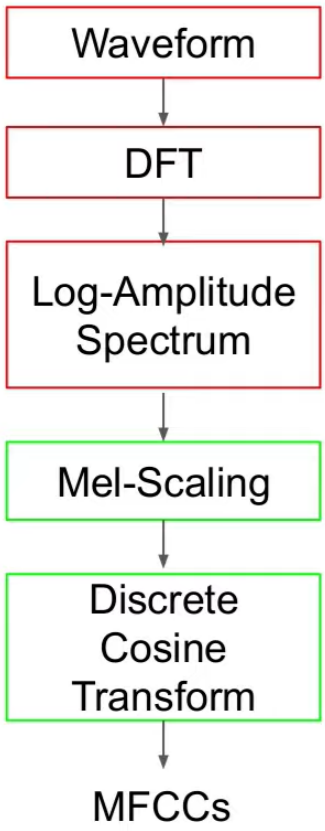


This approach lays the groundwork for effective feature extraction and model development in the realm of audio classification.

Extracting Features

Here, we will be utilizing Mel-Frequency Cepstral Coefficients (MFCCs) extracted from audio samples. In the realm of machine learning, the concept of mel spectrograms has been developed by incorporating MFCCs as an important audio feature. By summarizing the distribution of frequency over the size of the window, enables the analysis of frequency and time properties of sound. These audio representations help identify distinctive features for sound classification.

The process of computing Mel-Frequency Cepstral Coefficients (MFCCs) involves several steps, each contributing to the extraction of features that capture the relevant characteristics of an audio signal.

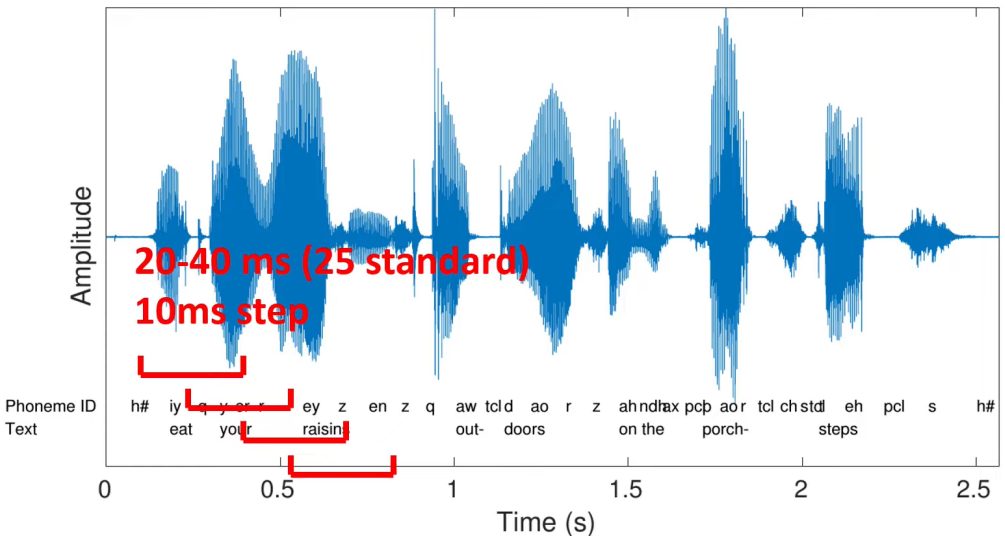


The audio signal is initially represented in the time domain as a waveform. This waveform is a plot of the amplitude of the signal over time.

Discrete Fourier Transform(DFT):

To avoid spectral leakage, we apply a window function to each frame of the waveform, which is divided into small, overlapping segments.

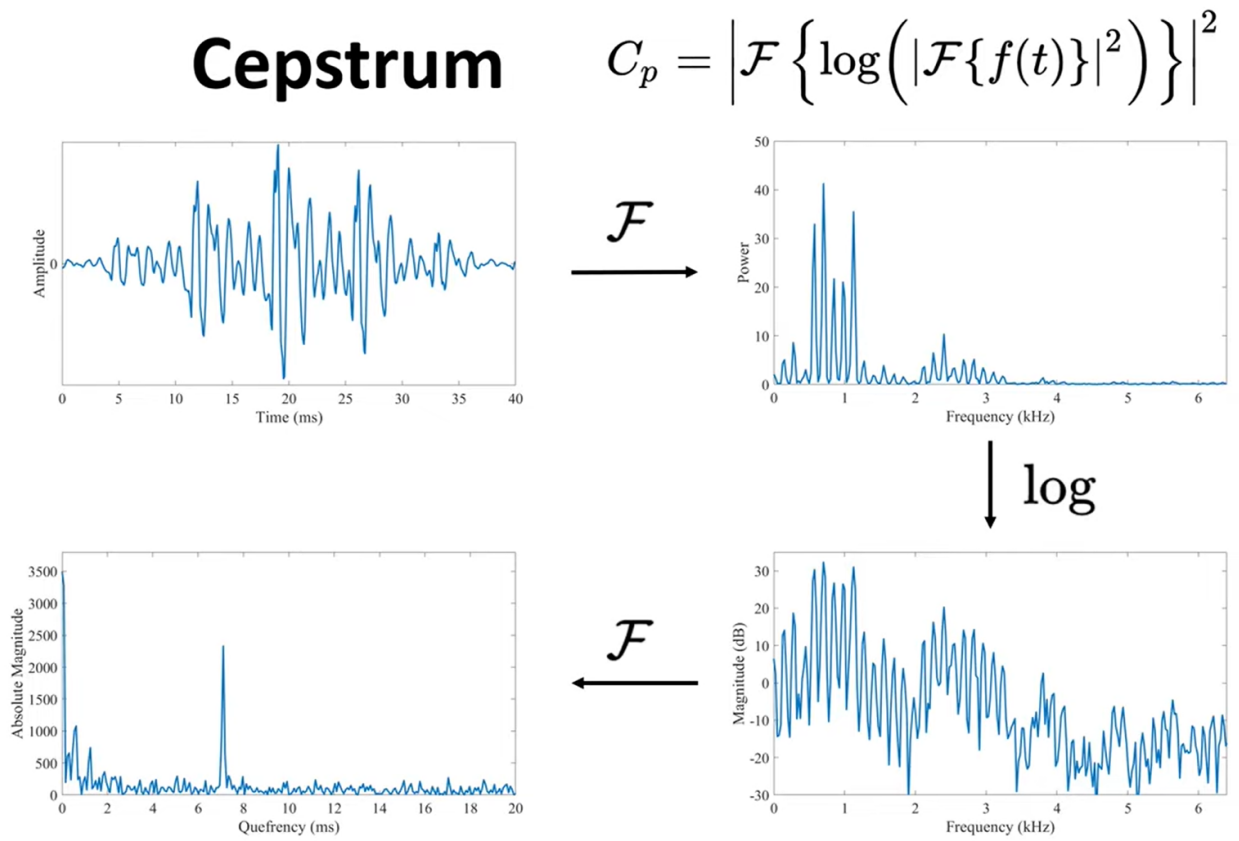
The frequency spectrum of each frame is obtained by performing the Fast Fourier Transform (FFT) on each windowed frame.



Windowing is done in 20 to 40 milliseconds, with 25 milliseconds being the standard.

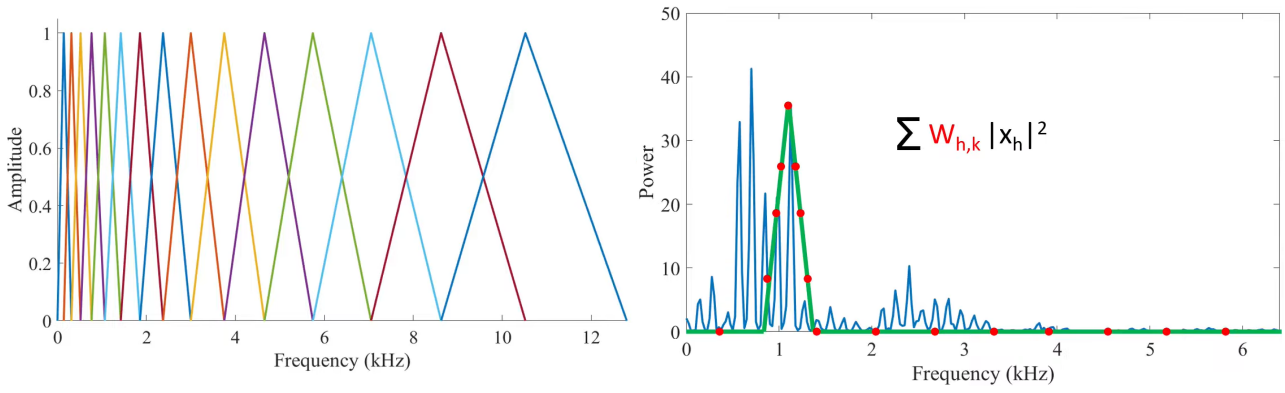
Log-Amplitude Spectrum:

After obtaining the frequency spectrum, the amplitude spectrum is computed by taking the magnitude of the complex values obtained from the DFT. The log of the amplitude spectrum is taken to better align with the logarithmic nature of human perception of loudness. Here we get the topic cepstrum(a spectrum of the spectrum).



Mel-Scaling:

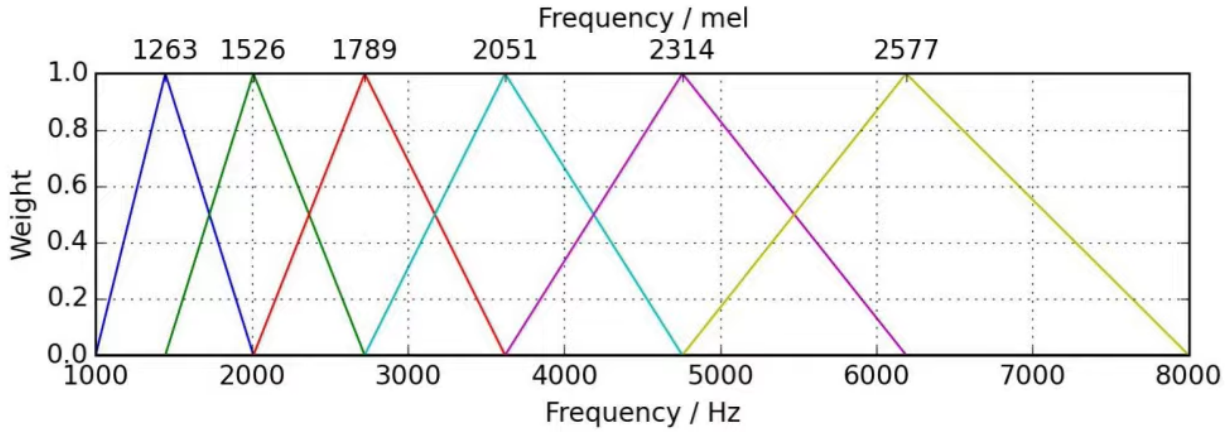
The subsequent step involves the creation of a Mel filterbank, which comprises a collection of triangular filters that are allocated uniformly across the Mel frequency scale. The log-amplitude spectrum is then subjected to each and every filter in the Mel filterbank, and energy under each filter is computed.



The filter bank coefficients are computed by multiplying the filter values with corresponding energy values in the frequency domain and summing them up.
The operation of computing filter bank coefficients involves taking a weighted average of frequencies in the triangle and disregarding other frequencies.

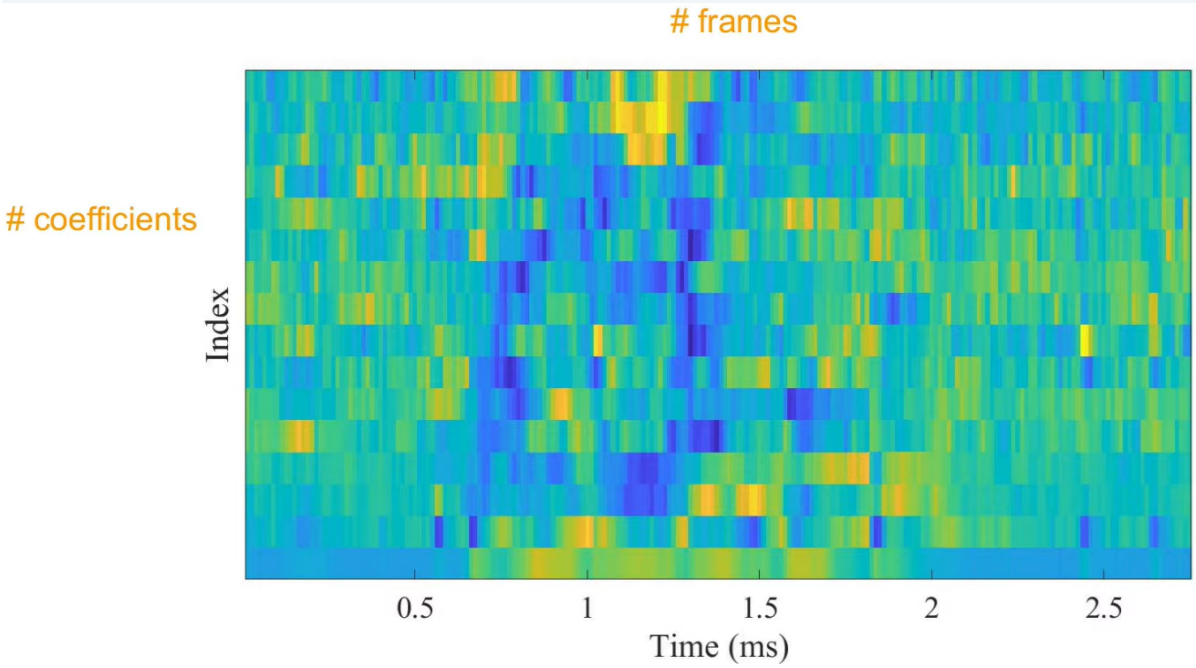
Discrete Cosine Transform (DCT):

Energies from Mel filters are subjected to the DCT, to decorrelate filterbank energies and extract coefficients that capture the most relevant information.



MFCCs:

Applying a triangular filter bank on each window helps obtain the Mel spectrogram, which is smoother than the simple spectrogram.



This sequence of steps, from the raw waveform to the computation of MFCCs, is a standard process in audio signal processing and feature extraction.

Model Creation

The code snippet is a TensorFlow implementation of an artificial neural network using the Keras API for audio classification. The model architecture consists of four layers within a Sequential model. The input layer has 40 nodes, reflecting the count of features derived from audio data. Then subsequent layers include fully connected (Dense) layers with 100, 200, and 100 nodes, each accompanied by an activation function called as Rectified Linear Unit (ReLU). Dropout layers having dropout rate of 0.5 are incorporated after each activation function to prevent overfitting by randomly deactivating a portion of nodes during training. The final layer of consists of nodes equivalent to 10(classes count). It uses the softmax activation function, which is ideal for multi-class classification tasks. Overall, the model is designed to learn and generalize patterns from the input audio features to predict the class labels, making it suitable for audio classification tasks.

Label Encoding

The LabelEncoder from the scikit-learn library is employed to transform categorical class labels into numerical representations, assigning each unique class a whole number (e.g., 0, 1, 2, 3, etc.). This process is known as label encoding and is particularly useful for preparing target variables in machine learning models. Additionally, the to_categorical function from Keras is applied to further encode these numerical labels into dummy variables, creating a binary matrix representation of the original class labels. This matrix is structured such that each column corresponds to a unique class, and the presence of a "1" in a particular column indicates the assigned class for a given sample. This encoding is beneficial, especially in the context of neural network models, where categorical variables need to be represented in a format compatible with training algorithms. The combination of label encoding and the creation of dummy variables enhances the model's ability to effectively learn and classify the different classes in the target variable.

Training The Model

In the training phase, a neural network model is created for audio classification and trained on the UrbanSound8K dataset. Essential features are extracted from the dataset, with MFCCs being computed from each audio file, capturing crucial characteristics for the model. These features are organized into a Pandas data frame, and dividing the dataset into training and testing sets as 20% allocated for testing. The neural network model, comprising three hidden layers with 100, 200, and 100 nodes, is constructed using the Sequential API from TensorFlow's Keras module. Dropout layers are strategically placed to alleviate overfitting. The final layer, configured with nodes equal to the unique classes in the dataset and a softmax activation function, is suited for multi-class classification. Training is executed over a specified number of epochs, with each epoch representing a full iteration through the entire training dataset. In this case, the model undergoes training for 100 epochs. To preserve the best model based on validation accuracy, we utilize a ModelCheckpoint callback. This ensures that the model can be retrieved and used later. Upon

completion of training, the model's accuracy is evaluated on the reserved test set, and it gets 80 percent, revealing its ability to generalize to new, unseen data. Additionally, a testing section demonstrates the application of the trained model to classify new audio data, where the model predicts class labels based on extracted MFCCs. The predicted labels are then inversely transformed using the labelencoder, providing the final predictions.

Conclusion

This audio classification project demonstrates a systematic approach to effectively recognize and categorize urban sounds using deep learning techniques. Mel-Frequency Cepstral Coefficients (MFCCs) were extracted from audio samples, providing essential features for the model. The neural network, constructed with three hidden layers and dropout regularization, showcased the capability to capture intricate patterns and relationships within the data. The model was trained over 100 epochs, achieving a commendable accuracy of 80 percent on the test set. Testing on new, unseen audio data demonstrated the model's adaptability and successful prediction of class labels. Despite the complexity of urban sound classification, this project underscores the efficacy of deep learning approaches, specifically highlighting the significance of MFCCs as crucial features. Overall, the developed model serves as a valuable tool for accurately classifying urban sounds, holding potential applications in various domains such as environmental monitoring and surveillance.

References:

1. UrbanSound8K Dataset Download Link(need to fill google form):
   https://urbansounddataset.weebly.com/download-urbansound8k.html
   Or use my Drive link to download:
   https://drive.google.com/drive/folders/1Yle_LGdrpvrbZ_QN4WWb9sNXxEg3aA8K?usp=sharing

2. Understanding Audio Signal Processing:
   https://www.codewithc.com/using-ann-in-audio-signal-processing-a-case-study/

3. MFCC Technique for Speech Recognition:
   https://www.analyticsvidhya.com/blog/2021/06/mfcc-technique-for-speech-recognition/#:~:text=MFCC%20is%20a%20feature%20extraction,speech%20recognition%20and%20music%20analysis.

4. How To Use Google Colab:
   https://www.geeksforgeeks.org/how-to-use-google-colab/