CS 7180 Project-3: Time Sequences

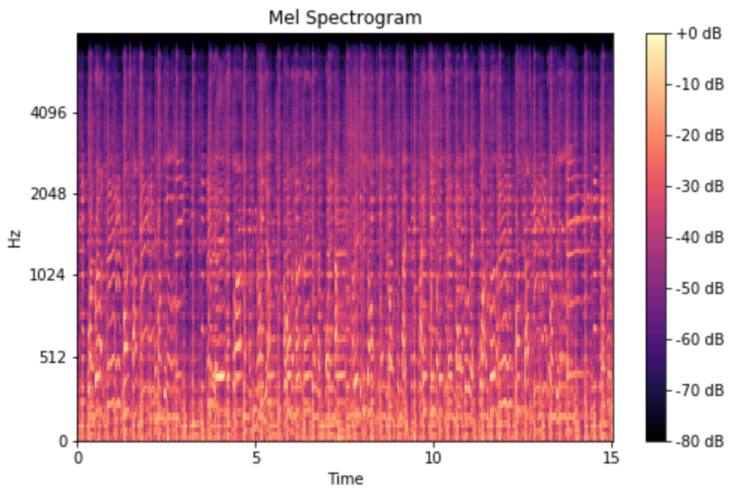
Abstract

The aim of this project was to implement an analysis based on a time sequence of data. For this project, I focused on a speech recognition application, more specifically identifying Capuchin bird calls in a rainforest in order to estimate the density of birds calls in the rainforest. A high level view of the process looks like converting the audio data into waveforms, which are then transformed into Mel spectrograms which are then used the classify bird calls using a deep learning model.

Introduction

Analyzing time sequences for various applications is an essential step to getting an accurate and flexible system. It gives us not only more details about features we are trying to extract but also helps us understand the variance the features might go through. For this project, I chose to focus on analyzing audio samples and implement a simple audio classification model. Audio classification has many applications such as speaker recognition, voice modulation, understanding background noises better in an effort to segment them and plays into the bigger aim of making our machines more intelligent. This project focuses on identifying Capuchin bird calls in a rainforest by classifying clips using Mel spectrograms. There is a lot of interesting research on the role of Mel spectrograms in audio classification.

To give a quick overview of spectrograms, they are generated from sound signals using Fourier transform. A Fourier Transform decomposes the signal into its constituent frequencies and displays the amplitude of each frequency in the signal. Plots of Frequency (y-axis) vs Time (x-axis) are then plotted in a spectrogram. Different colors are used to indicate the Amplitude of each frequency. Simply put, the brighter the color the higher the energy of the signal.



Converting audio signals into spectrograms allows you to tap into the image classification technology and networks. Some interesting papers I read on the same are AST: Audio Spectrogram Transformer and Spectrogram based multi-task audio classification.

Methods

The simplified overview of the method looks like converting the audio data into waveforms, which is further converted in spectrograms which are then used for audio classification using a deep learning model. The final output is a binary output indicating if a bird call was detected in a particular splice of the audio. Continuous detections are contacted as one call and the results are compiled.

Data - The data consists of Capuchin bird call clips which are approximately 3 seconds in length, forest recordings without Capuchin bird calls and Forest recordings which require estimation of bird calls which are about 3 minutes in length.

Converting to Waveforms - The audio signals are converted to single channel 16 Hz waveforms using TensorFlow audio library. What this gives us is the representation of the amplitude of the audio signal/waveform which we can then visualize in a spectrogram. Additionally all the sample bird calls are marked as 1 in the dataset and non-capuchin samples are marked as 0.

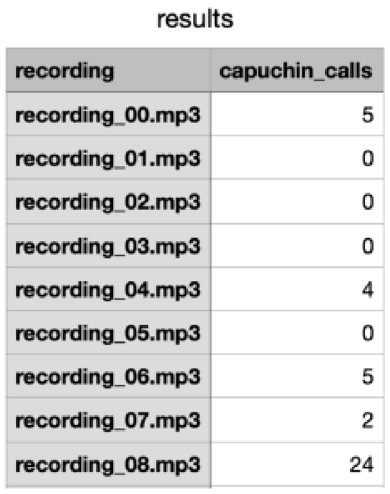
Converting Waveforms to Spectrograms - A short time Fourier Transform is used to convert the waveforms into spectrograms. We find the average length of a Capuchin bird call and pad/clip the audio signals to that length and then convert them to spectrograms with absolute values.

Model - We build a sequential model with convolutional, dense and flatten layers. The model contains 2 conv2d layers with 16 3x3 kernels and a relu activation and takes in the spectrogram as the input. The output from these layers is then flattened into a single dimension using a flatten layer. The next layer is a dense layer with 128 units and relu activation. The final layer is a dense layer with 1 units and a sigmoid activation to give us binary output. The model is then compiles with an Adam optimizer and a BinaryCrossEntropy loss function.

After training the model, we move our model across 3 second splices of the testing data(also converted to spectrograms) of forest recordings and calculate predictions. If the obtained prediction > 0.9 it is considered a positive for bird calls, else a negative. Consecutive results are then combined to a single bird call and the results are calculated and stored in a text file.

Results

Some sample results are shown below. All results can be found here: https://drive.google.com/file/d/1gRbWVZytiAZhPJXDc1J1cFizMs_wgyy1/view?usp=sharing



results

recording	capuchin_calls
recording_00.mp3	5
recording_01.mp3	0
recording_02.mp3	0
recording_03.mp3	0
recording_04.mp3	4
recording_05.mp3	0
recording_06.mp3	5
recording_07.mp3	2
recording_08.mp3	24

The network was 100% accurate after playing around with the threshold value, for the given dataset.

Reflections

It is always interesting to see various methods we learn in papers in action via applications. This project gave me a good overview of audio classification and its differences and similarities from image based methods. It was also interesting to see spectrograms being used for audio applications to access various image based methods. This was a relatively simple project but there is a lot of potential for expanding on this, by extracting various other audio features and expanding the training and classification range from binary to labeled.

Acknowledgements

Papers read in class for Audio Classification. AST: Audio Spectrogram Transformer and Spectrogram based multi-task audio classification for a deeper understanding of spectrograms. https://towardsdatascience.com/audio-deep-learning-made-simple-part-2-why-mel-spectrograms-perform-better-aad889a93505 for application based ideation. https://www.kaggle.com/datasets/kenjee/z-by-hp-unlocked-challenge-3-signal-processing for the challenge and the dataset.

Edit - Unfortunately could not get images of mel spectrograms and network architecture. I did not save them during the first run and did not have access to the system with Tensorflow support after that.