Speaker Determination with Neural Networks

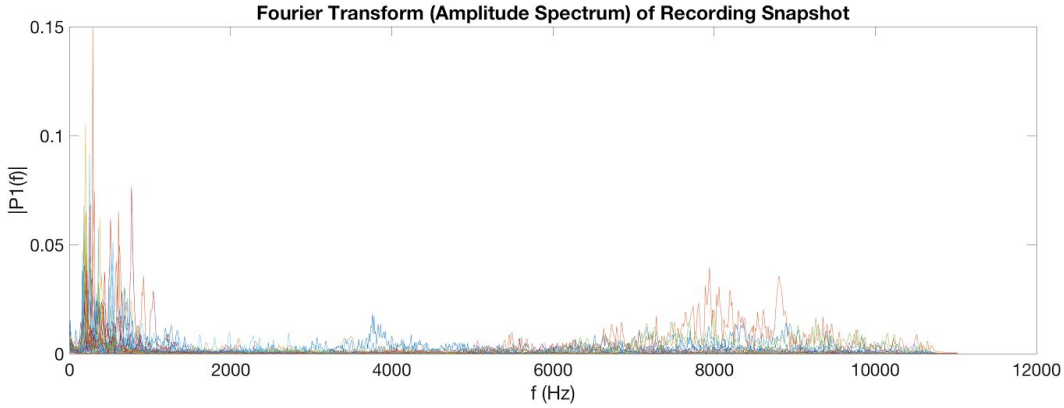
Sparsh Agarwal, Wayne Chew, Newton Wolfe, Keshav Sharma
NetID: agarwal39, mchew2, npwolfe, ksharma28

Progress Report:

1. **Data Collection**
   We are using various voice notes and recordings for our data to train the network. The voice recording are computer generated voices. We are using computer generated voices as initial training set to make sure the waves are consistent and so the training would be more accurate. Once we get success with computerised voices, we will train it on human voices.

   We are currently sampling a small portion of the data. We plan to train our classifier on a smaller sample size first before moving on to a bigger samples. We extracted 3 audiobooks from LibriVox and convert them into 10 minutes long wavfile. (“abbott_10.wav”, “aeschylus_10.wav”, “optic_10.wav”) where it is named using the last name of the author and the length of the audio file.

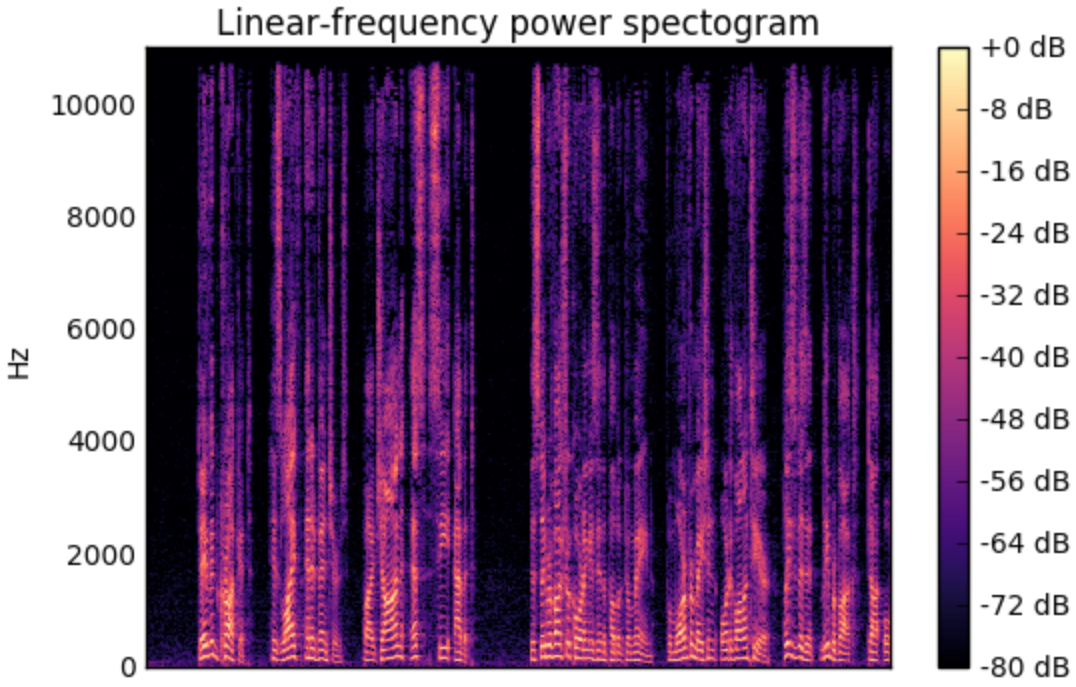
2. **Data Processing**
   We have developed a MATLAB script that will read in audio files and output Fourier transforms of 50ms sections of the recordings. These Fourier transforms indicate the energy in 10 Hz wide bands of frequencies, which are then input to our neural network. The following is a graph of this output across approximately 40 samples from a single speaker’s audio:



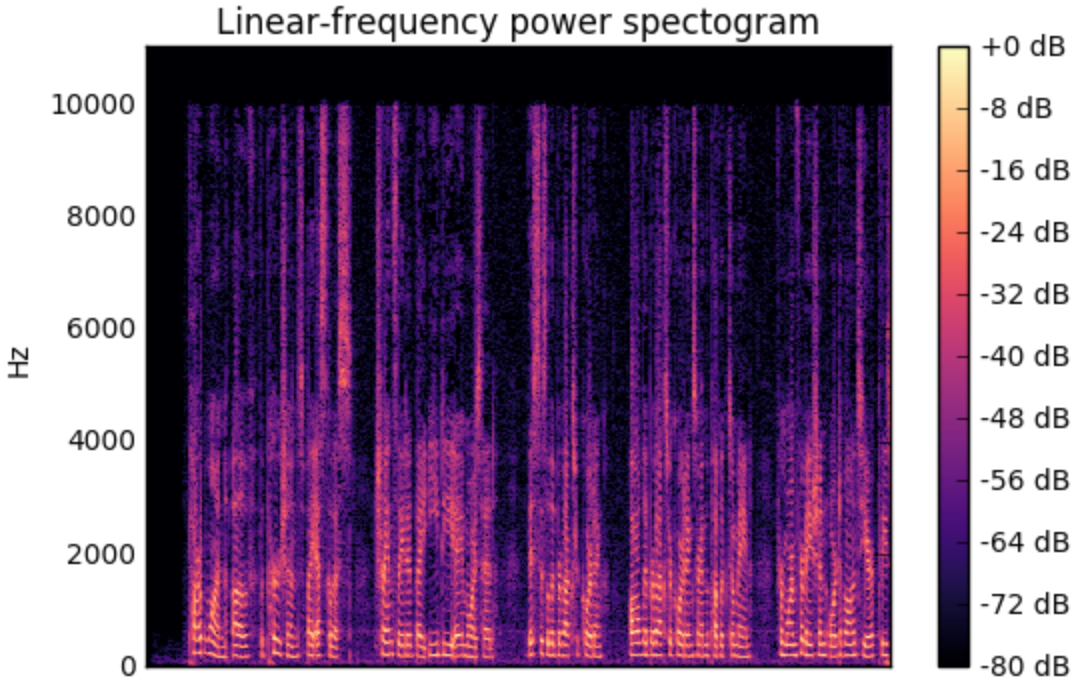
   We then export each recording as a CSV with hundreds of examples, and the CSV is labelled with the identity of the speaker.

Visual of feature vectors that we plan to use, you can see the difference in the three different authors of the audiobook (“abbott”, “aeschylus”, “optic”).

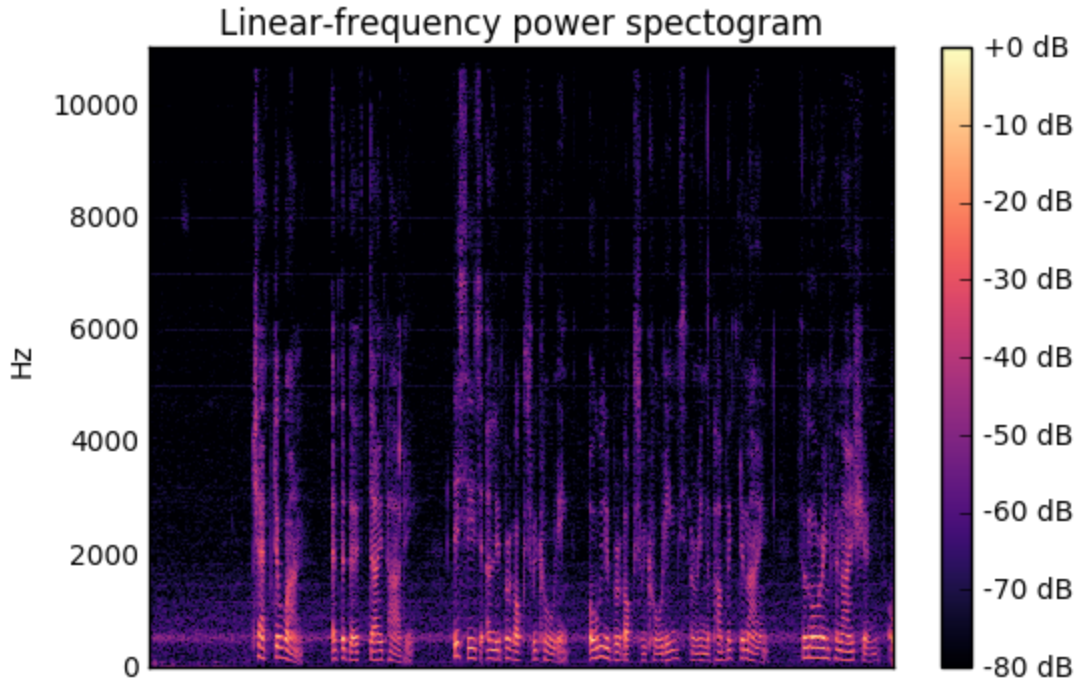
Abbott:



Aeschylus:



Optic:



As you can see there are clear patterns both in just amplitude compared with frequency, but also substantial time-variant patterns. However, this also revealed to us a complexity where the audio files' balancing doesn't correlate to normalized output from the FFT, so we're developing methods to normalize the outputs without losing information about the time-varying amplitudes. (Simple thresholding does not work because occasional non-voice elements like pops or clicks can distort the thresholds.)

3. **Neural Network**
   We're still working on getting a classifier using Keras and TensorFlow to work correctly; interfacing the audio data with Keras in a natural and efficient way has turned out to be a more complex problem than we anticipated. We would like to keep the structure of the data in both amplitude, frequencies, and time variation, but are still working on a data representation that allows us to do all 3.