Use of Machine Learning for Speech Recognition using Spectrogram Data.

***Abstract*—This report will introduce concepts of spectrogram analysis, as well as walking through the process of analysing speech data provided to me in MATLAB, and the results of said analysis using figures. I will provide a description of how machine learning techniques can be used in relation to spectrogram data in speech recognition, as well as briefly discussing other areas for machine learning in DSP.**

I. INTRODUCTION

SPECTROGRAMS are represented by a two-dimensional graph, with time on the x-axis and frequency on the y-axis, with the colour at that point indicating intensity of said frequency. They are commonly used in speech recognition software as a pre-processing step, before analysing and detecting voice activity, and separating it from the rest. Other uses include speech enhancement by filtering and removing noise, synthesizing natural sounding text to speech, or identifying emotion from speech data. However, to maximize the potential of what can be done with spectrograms requires the use of machine learning and neural networks taking speech data to learn from and apply with little human intervention.

II. SPECTROGRAM ANALYSIS OF SPEECH DATA

Spectrograms are visual representations of the intensity frequencies over a given time. They are constructed by first separating a given audio signal into L segments of equal length N, followed by calculating the Discrete Fourier Transform (DFT) of each segment l, referred to as $X_l(k)$. Typically, spectrograms will have a shorter duration for each segment, resulting in better time resolution, but a worse frequency resolution. This can be helped in some cases by overlapping segments, however we will not be doing that in this analysis. The spectrogram estimate equation is as follows:

$$\hat{S}_x(l,k) = \frac{1}{N}|X_l(k)|^2 \qquad (1)$$

A short three second clip of audio is shown in Fig. 1, which has a sampling rate f_s of 16 kHz, and contains 48000 samples. To convert these values into seconds, the equation

$$T = \frac{No.\,of\ Samples}{Sampling\ Rate} \qquad (2)$$

is used.

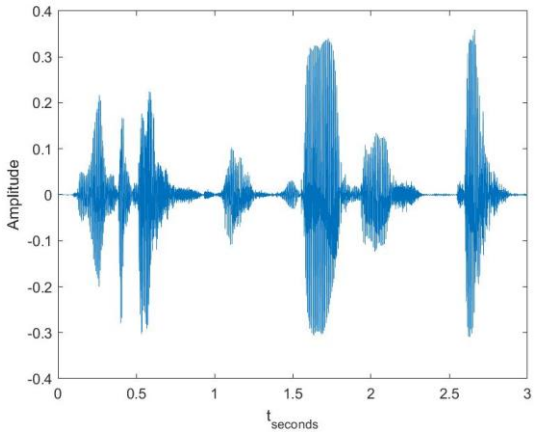

Fig. 1. Visualization of speech recording

In order to create a spectrogram visualization of this speech data, I started with a segment length of $N = 2^7$, following which I calculated L, which was simply done by dividing the total samples by N. The reason for this was to reshape the audio data into a two-dimensional array of size N x L as shown in Fig. 2, which could then be utilized to determine the DFT for each column. The final step was to substitute the values into (1), outputting an identically sized matrix Spectrogram.

```
% Reshape into columns and calculate DFT for each column
xl=reshape(x(1:nseg*seg_size),seg_size,nseg);
dxl=fft(xl);

% Spectrogram based on periodogram for each segment
Sx=(1/seg_size)*abs(dxl.*dxl);
```

Fig. 2. MATLAB code for reshaping data, calculating DFT and spectrogram estimate

After the main spectrogram calculations, scale adjustments for the X and Y axes, showing time in seconds and frequency in hertz respectively were necessary. Time adjustment was relatively simple as seen in Fig. 3, where I have inputted both the total number of samples and the sampling rate into (2). Fig. 3 also shows my method for adjusting for frequency. This was slightly

more complex, as I introduced a cut-off frequency of 3500 Hz due to frequencies in this range being the most predominant in speech. Implementing this required determining the Nyquist frequency, which is simply half the sample rate, which was then used to calculate the equivalent segment to be cut off. Finally, the axis was scaled, ready to be plot using the contourf() function. The final plot for a segment length of 128 can be seen in Fig. 4.

```
% Define nyquist freqency and cutoff frequency
nyquist_freq = fs/2;
cutoff_freq = 3500;
cutoff_segment = (cutoff_freq/nyquist_freq)*seg_size;

% Converting sample rate and total samples into time
time = linspace(0,(length(x)/fs),nseg);
freq = linspace(0,cutoff_freq,cutoff_segment);

logSx = log10(Sx(1:cutoff_segment,1:nseg));

% Plot contour map with colour bar
[M,c] = contourf(time,freq,logSx,100);
c.LineColor = 'none';
colormap(jet);
colorbar;

title(['log_{10} Spectrogram, N=' num2str(seg_size)]);
xlabel('t_{seconds}');
ylabel('f_{Hz}');
```

Fig. 3. MATLAB code for defining frequencies, translating axes, logging spectrogram, plotting heat map

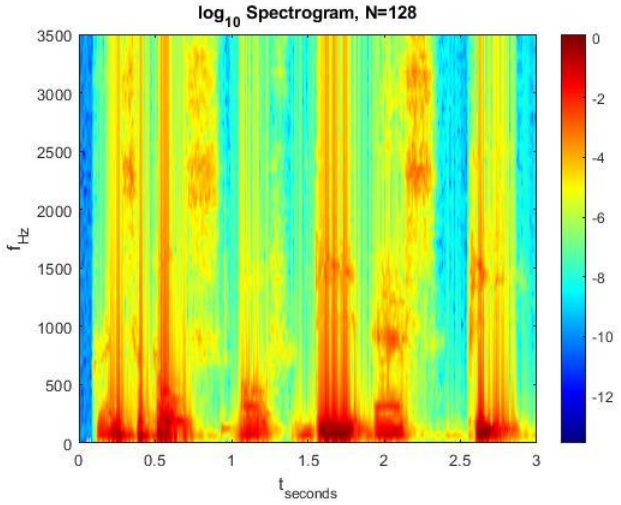

Fig. 4. Log spectrogram of speech data using segment length of $N=2^7$

Following this, I began testing the effect on varying segment length by increasing and decreasing x in the equation

$$N = 2^x \qquad (3)$$

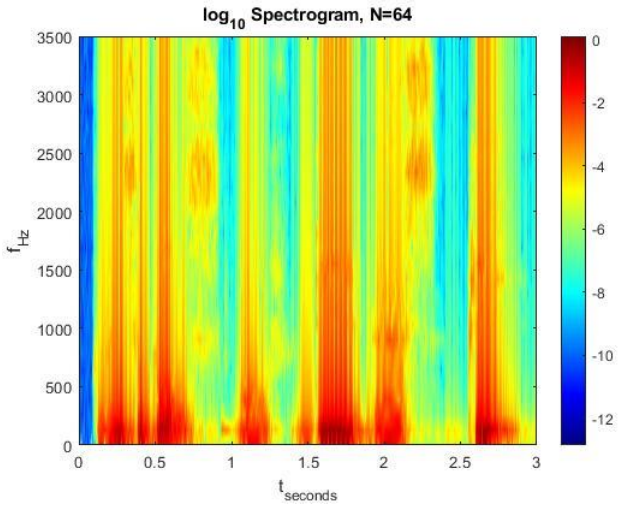

Fig. 5. Log spectrogram of speech data using segment length of $N=2^6$

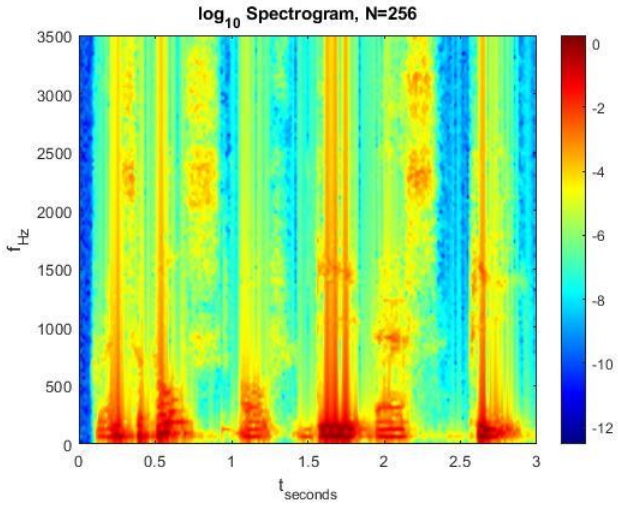

Fig. 6. Log spectrogram of speech data using segment length of $N=2^8$

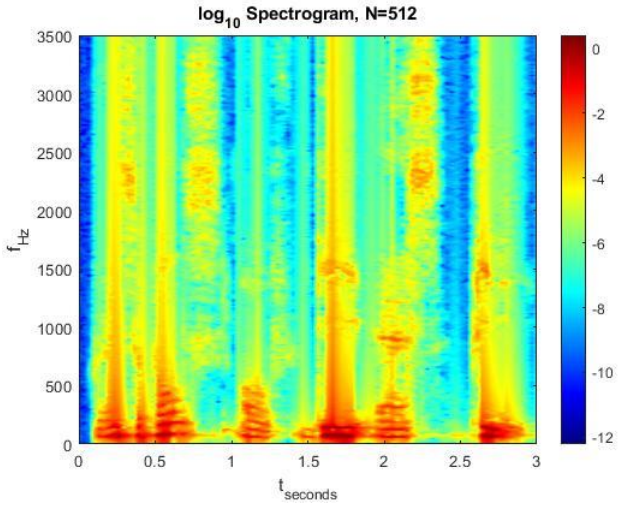

Fig. 7. Log spectrogram of speech data using segment length of $N=2^9$

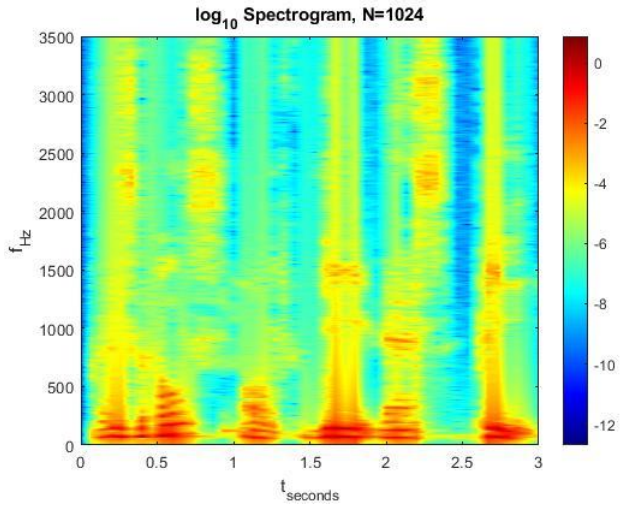

Fig. 8. Log spectrogram of speech data using segment length of $N=2^{10}$

III. Application of Machine Learning to Spectrogram Data

The area of computer science known as Machine Learning is that which has a primary focus of analysing patterns in data, and having computers interpret and learn, reason, and make decisions with no human interaction needed. Neural networks are an approach to machine learning, where supposedly with enough data, the network can analyse future inputs akin to the way a human can.

Neural networks are created to perform specific tasks and consist of varying amounts of interconnected neural computing units arranged in a particular architecture, which in the case of this report are multi-layer perceptrons. A perceptron is defined as a 'layered feed forward network', with binary or continuous inputs and/or outputs, and if there are multiple layers of perceptron units connected to the inputs, this is known a multi-layered perceptron.

These networks can be applied to spectrograms of speech data, as the visual contents of these network layers can be expressed as feature maps after convolving the input with learned filters as more data is passed through. These networks, once trained with enough inputs, would then eventually be able to be tailored to a variety of speech signal processing techniques.

IV. Discussion

As expected, the spectrogram repeats with varying N values show that lower x values in (3) result in a higher frequency resolution, whereas higher values result in much more messy frequency values, but a better time resolution for the x-axis. In terms of analysis of these spectrograms, I believe it is more suitable to have lower values for N, as in these plots it is easier to see structures appear, such as dominant frequency bands in Fig. 4 around 0.5s and 1.6s at around 100 Hz.

References

[1] A. M. Badshah et al., 'Speech Emotion Recognition from Spectrograms with Deep Convolutional Neural Network', in *2017 International Conference on Platform Technology and Service (PlatCon)*, 2017, pp. 1–5.

[2] K. V. Vijay Girish, (2019, January.3), *Beginner's guide to Speech Analysis*, Towards Data Science. [Online]. Available: https://towardsdatascience.com/beginners-guide-to-speech-analysis-4690ca7a7c05.

[3] T. Arias-Vergara et al., 'Multi-channel spectrograms for speech processing applications using deep learning methods', *Pattern Anal. Appl.*, vol. 24, no. 2, pp. 423–431.