CSCE3611: Digital Signal Processing

Project : Gender Classifier Report

Submitted by: **Mohammed Abuelwafa** **900172603**

Omer Moussa **900171920**

Ismael Elsharkawi **900173030**

Date: 12th December 2020

Fall 2020

Used tools and languages:

- Python
- R
- Tensorflow framework
- Numpy
- Pandas
- Seewave package for the analysis of the frequency spectrum

Project Structure:

The project is mainly composed of three main files:

1) Training code: this is the machine learning model trained to predict whether the speaker is a male or female.

2) Prediction code: The code use to predict whether the speaker is a male or female either using a prerecorded .wav file or taking a live recording from the user.

3) extractAcoustics: This code is written in R and used to extract the dimensions of the voice signal which are used to train the model to classify male and female voices.

Training Code:

Data Exploration:

Below are the plots of the values of the most important features of the datasets

1- Distribution of classes: Male and female number of examples is balanced in the dataset.

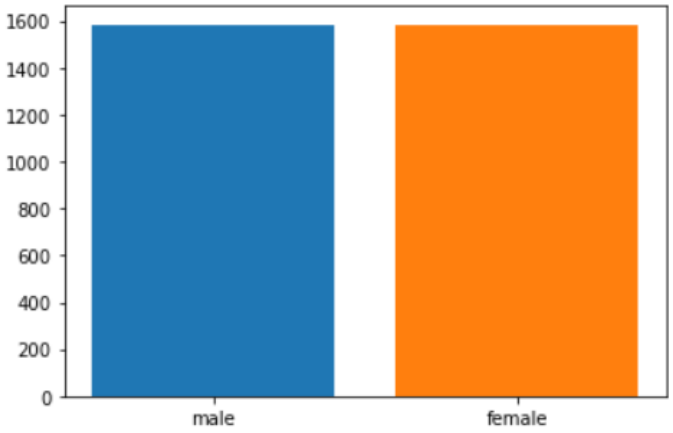


Figure 1

2- Mean frequency values :

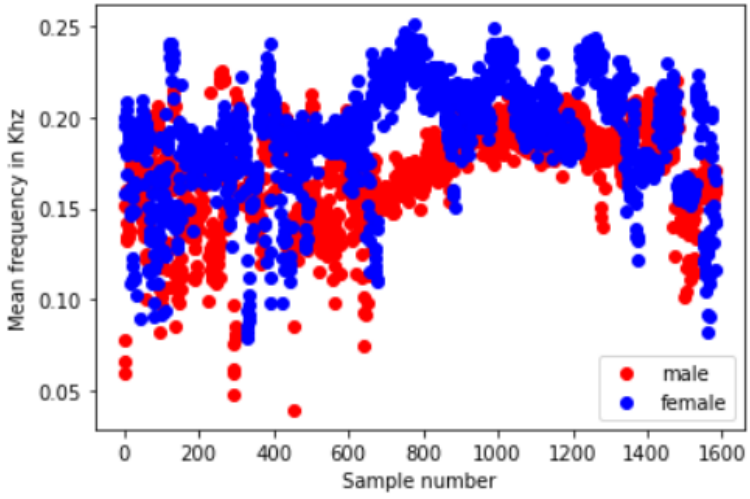


Figure 2

3- Q25 of the frequency for each sample:

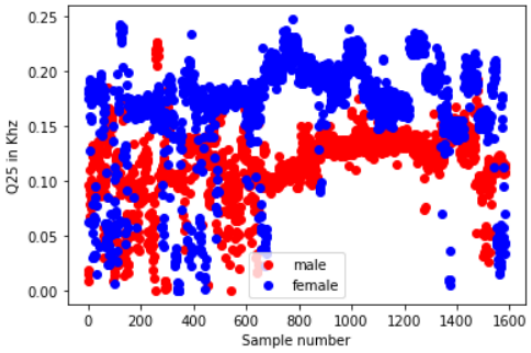


Figure 3

4- Mean Fundamental Frequency:

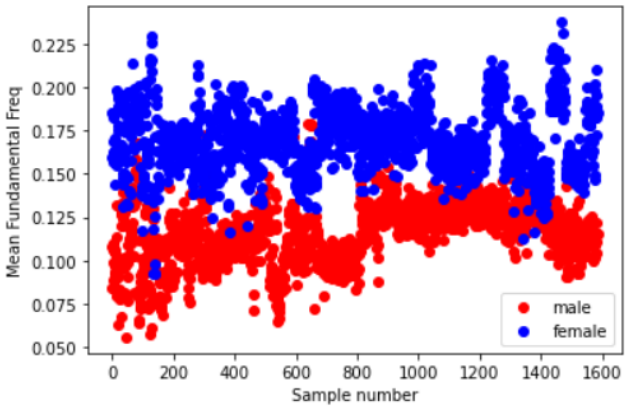


Figure 4

** More plots of the features can be found in the notebook attached.

Preprocessing:

The only preprocessing done was converting the labels from string format to numerical labels (1 for males and 0 for females.)

Training the classifier:

We used keras framework to train a machine learning model to predict whether the speaker is a male or female.

Data set: We depended on a publicly available dataset on Kaggle called in a file called voice.csv that has around 3168 training points from which around 20% of the data was used for validation and 80% was used for the training.

Model Architecture:

It is a fully connected neural network using the following architecture:

1- Input layer : takes the input dimensions (20 dimension).
2- Hidden layer 1: it has 100 nodes using a RELU activation function.
3- Hidden layer 2: It has 50 nodes using a RELU activation function.
4- Output layer: the predicted label (a value from 0 to 1) using a sigmoid activation function.

Optimizer: we used ADAM as our gradient descent optimization algorithm.

Loss : we calculated the loss using binary cross entropy loss.

The Training plots:

We reached a validation accuracy of **96.37 %**

```
[57]  H=model.fit(X_tr,y_tr,validation_data=(X_t,y_t),batch_size=32,epochs=130)

80/80 [==============================] - 0s 4ms/step - loss: 0.0989 - accuracy: 0.9684 - val_loss: 0.1203 - val_accuracy: 0.9543
Epoch 119/130
80/80 [==============================] - 0s 5ms/step - loss: 0.1106 - accuracy: 0.9653 - val_loss: 0.1236 - val_accuracy: 0.9511
Epoch 120/130
80/80 [==============================] - 0s 5ms/step - loss: 0.0955 - accuracy: 0.9672 - val_loss: 0.0995 - val_accuracy: 0.9700
Epoch 121/130
80/80 [==============================] - 0s 5ms/step - loss: 0.1177 - accuracy: 0.9594 - val_loss: 0.1311 - val_accuracy: 0.9574
Epoch 122/130
80/80 [==============================] - 0s 5ms/step - loss: 0.0951 - accuracy: 0.9669 - val_loss: 0.1793 - val_accuracy: 0.9369
Epoch 123/130
80/80 [==============================] - 0s 5ms/step - loss: 0.0999 - accuracy: 0.9629 - val_loss: 0.1016 - val_accuracy: 0.9669
Epoch 124/130
80/80 [==============================] - 0s 5ms/step - loss: 0.0867 - accuracy: 0.9724 - val_loss: 0.1215 - val_accuracy: 0.9543
Epoch 125/130
80/80 [==============================] - 0s 5ms/step - loss: 0.0976 - accuracy: 0.9653 - val_loss: 0.1192 - val_accuracy: 0.9558
Epoch 126/130
80/80 [==============================] - 0s 5ms/step - loss: 0.0920 - accuracy: 0.9672 - val_loss: 0.1248 - val_accuracy: 0.9621
Epoch 127/130
80/80 [==============================] - 0s 3ms/step - loss: 0.1007 - accuracy: 0.9617 - val_loss: 0.1737 - val_accuracy: 0.9401
Epoch 128/130
80/80 [==============================] - 0s 5ms/step - loss: 0.0949 - accuracy: 0.9629 - val_loss: 0.1127 - val_accuracy: 0.9669
Epoch 129/130
80/80 [==============================] - 0s 5ms/step - loss: 0.1018 - accuracy: 0.9649 - val_loss: 0.2351 - val_accuracy: 0.9132
Epoch 130/130
80/80 [==============================] - 0s 3ms/step - loss: 0.1001 - accuracy: 0.9641 - val_loss: 0.1087 - val_accuracy: 0.9637
```

Figure 5

The plot of training accuracy vs validation accuracy:

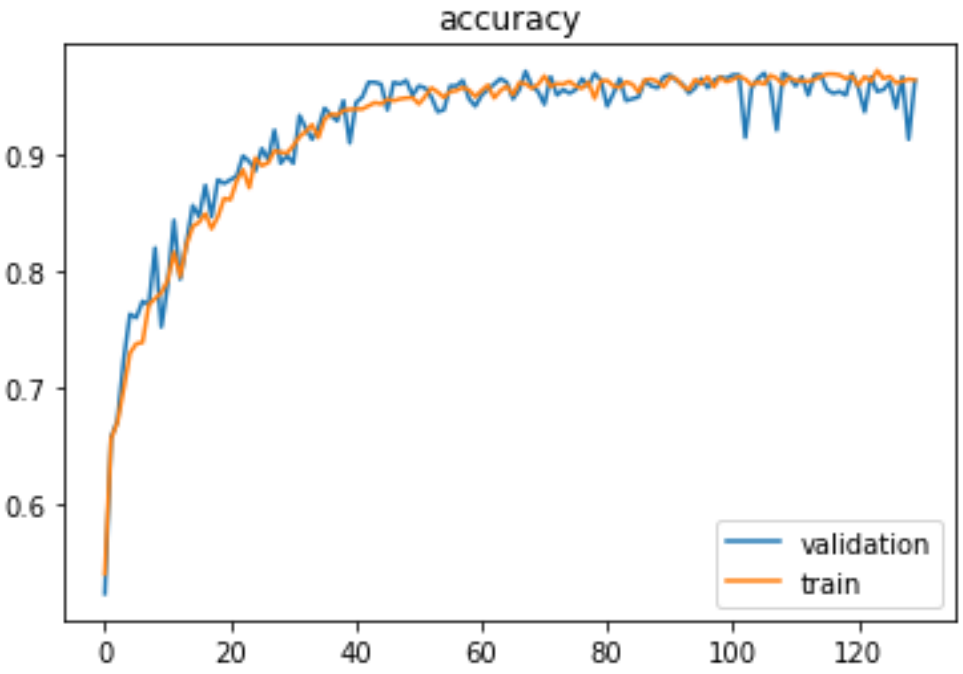


Figure 6

The plot of training loss vs validation loss:

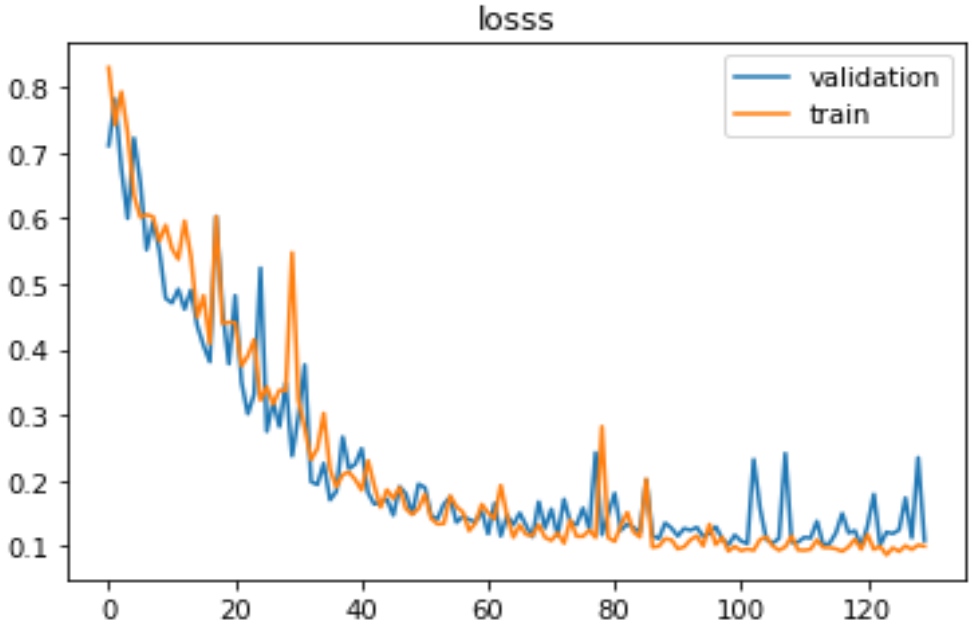


Figure 7

Prediction Code:

The prediction code basically loads the model trained and uses it to predict whether the speaker is a male or a female. We used a value of 0.5 as the threshold for the prediction. In an ideal scenario, a prediction of 1 means that the speaker is a male whereas a prediction of 0 means that the speaker is a female. However, we decided that the closer the values between 0 and 0.5 would be interpreted as "female" and the values between 0.5 and 1 would be interpreted as "male". The predicted value serves as a confidence measurement of the prediction rather than a mere prediction as well.

We added two options for the user to enhance the user experience of the classifier where the user gets to choose where to use a pre-recorded .wav file or to record one on the spot. For recording a file on the spot pyaudio library was used.

The flow goes as following:

1) We get the .wav file either in a prerecorded form or using live recording.

```
print("if you want to record to get the result enter: 1")
print("if you want to use a prerecorded wav file enter: 2")
n = input()
n = int(n)
```

```
if you want to record to get the result enter: 1
if you want to use a prerecorded wav file enter: 2
1
```

Figure 8

2) The code .wav file is fed into the extractAcoustics file written in R in order to analyze the spectrum of the file and extract the dimensions of the file that were used in the training.

```
#setwd("/home/prathamsolanki/github/gender-recognition-by-vo

data = data.frame("output.wav", 0, 0, 20)
names(data) <- c('sound.files', 'selec', 'start', 'end')

acoustics <- specan3(data, parallel=1)
write.csv(acoustics, file = "output.csv")
```

Figure 9

3) using the model, we predict whether the speaker is a male or a female using the predicted value.

Female Example:

```
data = pd.read_csv('voice.csv')
print("please enter the file name:")
file_path = input()
data_test = pd.read_csv(file_path)
data_test.drop(['sound.files', 'Unnamed: 0'], inplace=True, axis=1)
X = data_test[data.columns[:-1]].values
res = model.predict(X)[0][0]
print(res)
if (res > 0.5):
    print('You are a male !!')
else:
    print("You are a female !!")
```

```
please enter the file name:
Ammy.csv
0.07959977
You are a female !!
```

Figure 10

Male Example:

```
In [7]: data = pd.read_csv('voice.csv')
        print("please enter the file name:")
        file_path = input()
        data_test = pd.read_csv(file_path)
        data_test.drop(['sound.files', 'Unnamed: 0'], inplace=True, axis=1)
        X = data_test[data.columns[:-1]].values
        res = model.predict(X)[0][0]
        print(res)
        if (res > 0.5):
            print('You are a male !!')
        else:
            print("You are a female !!")

        please enter the file name:
        output.csv
        0.9995147
        You are a male !!
```

Figure 11

Funny fact: we used the model to predict the voice of a male pretending to be a female and the model managed to detect it and vice versa.

ExtractAucostics code:

R was used to analyze the .wav file and produce a .csv file that is required for the classification problem. In the R script, the wav file was checked at first to see if there is any error with loading the file, then the wav file was analyzed in the frequency domain and based upon that analysis, there were many characteristics that were determined. Those characteristics were used by the model to determine whether the voice was a male or a female.

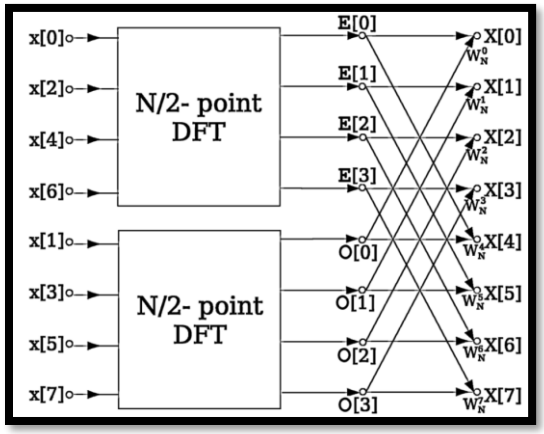
At first, the wav file was loaded into a structure called a data frame. The data frame was then entered into a function called specan3. Specan3 function performs checks that include the validity of the input(that the input is a data frame), checking that the start value is not smaller than the end value and it also checks that the wav file is in the same directory that contains the R script.

The package calculates the FFT for the wav file, then calculates the quantities shown in the 2 tables later in this section. There is a package called seewave that is used. This package is the reason why R was used in the first place. Spec and Spec prop were used.

It is important at first to explain the FFT (Fast Fourier transform). The DFT(discrete Fourier transform) is the method of moving a sampled signal from the time domain to the frequency domain. The DFT is calculate according to the following equation where X_k is the kth frequency domain sample. X_n is the time domain sample. N is the total number of samples.

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N} kn}$$

However, the FFT is a faster way to produce the same output as the DFT. The way to calculate the DFT is illustrated in the following diagram, where W_N is the N-th root of unity and it is calculated using this formula $e^{-j\,2\pi/N}$.



After calculating the FFT, the following are some statistical quantities that need to be calculated on the calculated frequencies.

Mean frequency	Mean of magnitude of frequencies from FFT
Standard Deviation	Standard deviation of mean of frequencies from FFT
Median Frequency	Median frequency from output frequency spectrum of FFT
Q25	First Quantile of frequencies Q25 = $(\frac{n+1}{4})^{th}$ term
Q75	Fourth Quantile of frequencies Q75 = $(\frac{3(n+1)}{4})^{th}$ term
IQR	Q75- Q25
Skewness	Measure of asymmetry skewness =$\sum_{i=1}^{N} \frac{(f-\bar{f})^3}{s^3 N}$,where s is the standard deviation and f bar is the mean frequency.
Kurtosis	Measure of peakedness Kurtosis = =$\sum_{i=1}^{N} \frac{(f-\bar{f})^4}{s^4 N}$, where s is the standard deviation and f bar is the mean frequency.
Spectral Entropy	Spectral Entropy describes the complexity of a system $S = -\frac{\sum_{i=1}^{N} y_i log_2(y_i)}{log_2(N)}$ Where y_i is a particular frequency from the DFT and N is the total number of frequencies.
spectral flatness measure	Spectral flatness is the ratio of the geometric mean to the arithmetic mean of the magnitude spectrum of the signal, as obtained from the DFT/FFT.

	$F = N \times \frac{\sqrt[N]{\prod_{i=1}^{N} y_i}}{\sum_{i=1}^{N} y_i}$
Mode	Most frequent frequency
Centroid	$C = \sum_{i=1}^{N} x_i \times y_i$

Now, we need to explain what the fundamental frequency is and what is the dominant frequency. The amplitude spectrum of a periodic function exhibits equally-spaced frequency components of varying height (magnitude). The first of these amplitude components is called the fundamental frequency and the amplitude component with the highest magnitude is called the dominant frequency.

Mean fundamental frequency	Average of fundamental frequency measured across acoustic signal
Minimum fundamental frequency	Minimum of fundamental frequency measured across acoustic signal
Maximum fundamental frequency	Maximum of fundamental frequency measured across acoustic signal
Mean dominant frequency	Average of dominant frequency measured across acoustic signal
Minimum dominant frequency	Minimum of dominant frequency measured across acoustic signal

Maximum dominant frequency	Maximum of dominant frequency measured across acoustic signal
Dominant Frequency Range	Maximum dominant frequency - Minimum dominant frequency

The range of a male fundamental frequency is from 85 Hz to 180 Hz and the range of female fundamental frequencies of a 155 Hz to 255 Hz. As you can see, there is an overlap in the range of frequencies between male and female fundamental frequencies, that is why we need machine learning models to produce more accurate results based on all the above quantities not only the fundamental frequency.

The following 2 graphs are the frequency spectrums of 2 wav files, one for a male and one for a female:

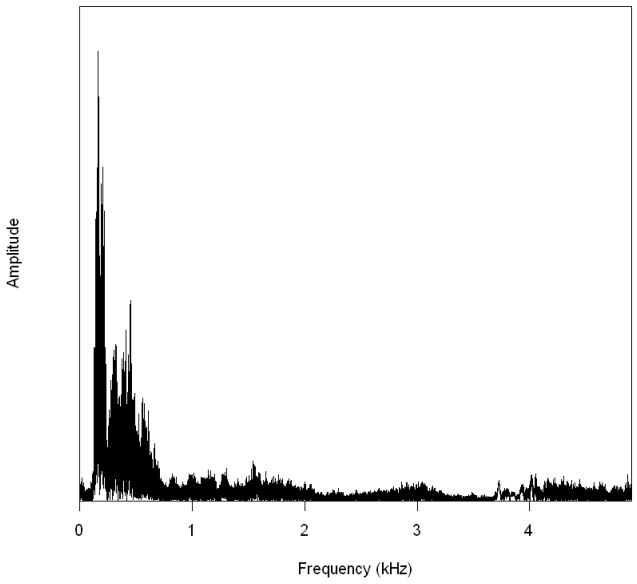


Figure 12: Female frequency spectrum

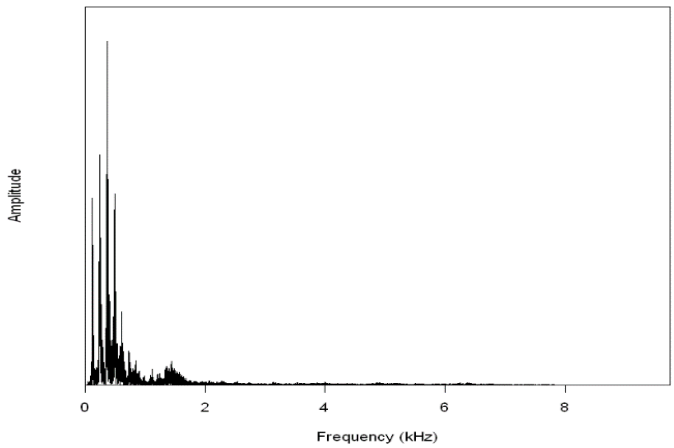


Figure 13: Male Frequency spectrum

The following 2 graphs are the graphs of the probabilities of occurrence of frequencies of male and female voices. The Q25, Q75, median and mode are shown on the graphs:

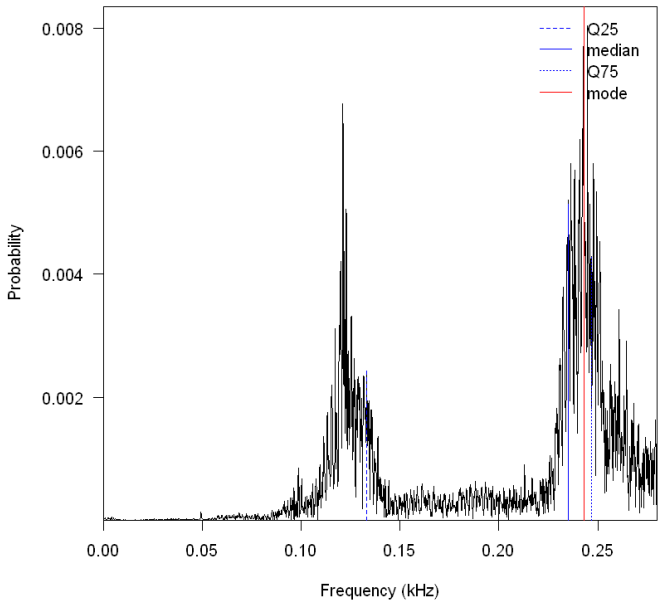


Figure 14: Male

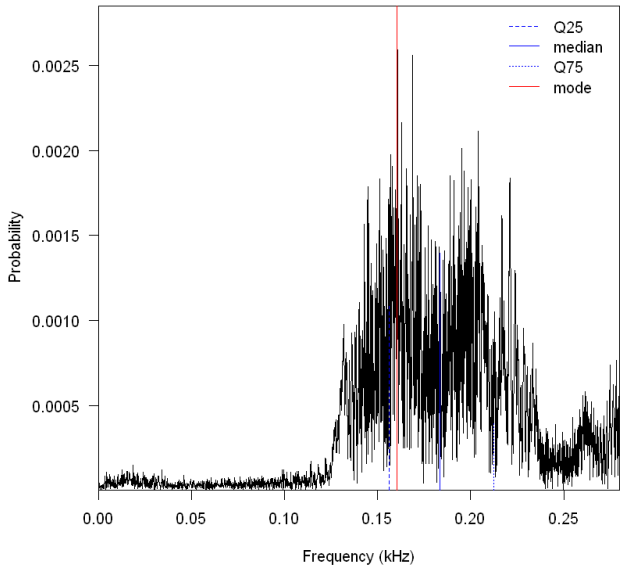


Figure 15: Female