School of Engineering and Applied Science (SEAS), Ahmedabad University

BTech(ICT) Semester III: Signals And Systems

Project Report

Project Name: *Gender Recognition Through Audio Data*

Group Number: 12

Group Members:

Ridham Shah - (AU-1841007)
Raj Mehta - (AU-1841018)
Vidish Joshi - (AU-1841019)
Manav Patel - (AU-1841037)

1 Introduction:

This project focuses on using the tools available in *python* programming language to analyze the frequency response of data and using those tools and techniques to analyze the audio samples taken from different people to predict the gender of a person.

It aims to predict the gender of a person through their audio data with the help of signal processing tools, codes and user friendly interface to run the program. The project has a strong database of audio samples recorded and processed live at the time of building the project and also keeps on improving its accuracy and precision with each prediction.

2 Summary:

Male and Female voices can be distinguished on the basis of their *pitch or frequencies*. Female speech has higher frequency than male's. And this project uses this distinction to form an average frequency for male and average frequency for female.

To do so, first a large number of audio samples were recorded. While calculating the average frequencies for both gender, all the audio samples were recorded through recording tools available on our laptop devices. As we know, the larger the data set, the more precise and accurate our data is. We managed to calculate audio data of about 95 people which mostly consist of our colleagues and peers with around 40 female samples and 50 male samples.

The next part was to analyze these data and calculate the average frequency of these samples.In these *Python Codes*, we calculated the average frequency of each audio sample and assigned the answer to the gender of the person. After doing so with all the data, we had large number of average frequencies and calculated the threshold frequency of both gender by taking average of respective data.

After calculating the threshold frequencies, comes the testing part of the program. This part mainly has *three* section which the executed program implements.

The first one being the *User Interface(UI)*. The UI allows the user to run the program through a cascade of screens. The User Interface of this program is a JAVA program built on the JFrame platform. The UI has three functions that are connected to next section of the program.

The second section of the program involves collecting data for testing. The UI provides the option for choosing the audio data to be tested. This audio data is then run over by the backend *python* interface and the average frequency of that audio sample is calculated.

The third section gives us the prediction of the input data given for testing. Once average frequency is obtained from the second stage, it is now compared with the average frequency of male and female. From that, upon whatever range the tested average falls in, we predict the gender of the input test data. The program also outputs the confidence of the code in the prediction.

It is apparent here that as this project is based on real life data, 100% accuracy is seldom achieved, exceptions are bound to happen. And so, the code cannot be completely sure that the prediction is correct. And therefore along with Gender Prediction, there is an additional feature of showing the confidence that the program has on its prediction. The confidence is less near the edge of the range separating the two gender and the confidence is more in the range away from those edges.

3 Technical Details:

As mentioned, the first and important part is to collect audio samples. This is done by using the *Voice Recorder* tool found inbuilt in the Windows OS.

To process this audio files in python, it is necessary to convert them in .wav format from the in-built m4a format. This is done using online converter.

From these .wav files, our aim is to calculate average frequency of each data file. This is executed by python libraries. To obtain this, we need the average frequency of that file which is obtained by operation the following steps:

Step - 1: The recorded files are in *stereo* format by default. They are converted to *mono* format by the ***stereo-to-mono*** function.

Step - 2: The sampling rate of the recorded files is very high(44 to 48 KHz). Such files take too long to process. So they are down-sampled to optimal sampling rate(8 kHz). The down-sampling process is done by the *down-sample* function.

Step - 3: Now, to obtain the frequency response of the recorded file, we need to perform fast fourier transform(fft). This task is followed by *down-sample* function. The fft is done using *scipy* pack in python which uses the following formula to perform the discrete fourier transform,

$$y[k] = \sum_{n=0}^{N-1} e^{-2\pi j \frac{kn}{N}} x[n] \, ,$$

Figure 1: FFT formula-python

Step - 4: From the frequency-magnitude response of audio sample, we need to calculate the average frequency of the particular audio sample. To obtain this, we use the following formula,

$$f_{mean} = \frac{\sum_{i=0}^{n} I_i \cdot f_i}{\sum_{i=0}^{n} I_i}$$

Figure 2: Flow of the program

Step - 5: Doing this for a large number of samples, we now get threshold frequency of both gender by simply finding the mean of those calculated frequencies.

These steps are part of the training process of the program.

The other part of the program is for predicting the gender. For that, we make a virtual range of the calculated frequencies in the program. The range is limited by *maximum frequency* from the male samples and *minimum frequency* from female samples.

Suppose maximum male frequency is f_{max}

Minimum female frequency is f_{min}

And frequency of current test data is f_{audio}

So the distinction between the two genders is at the edge,

$$f_{edge} = \frac{f_{max} + f_{min}}{2} \tag{1}$$

Any frequency less than *edge* is classified as male henceforth and frequency greater than *edge* is classified as female.

$$f_{audio} > f_{edge} \implies female$$

f$_{audio} < f_{edge} \implies male(2)$

But if f_{audio} lies close to f_{edge}, the chances of wrong prediction increase and for that the program gives the *confidence level* along with gender rather than blunt prediction.

If $f_{audio} > f_{max}$, than program's output will be *FEMALE* with 100% confidence.

If $f_{audio} < f_{min}$, than program's output will be *MALE* with 100% confidence.

The formula for calculating confidence level in the rest of the frequency range will be,

$$confidence = \frac{|(f_{audio} - f_{edge})|}{f_{edge} - f_{min}} * 100\% \quad (3)$$

4 Algorithms:

Figure 3 in the adjacent side represent the flow of the program.

Suppose the threshold frequency of male is 600Hz and that of female is 650Hz. Now to explain the algorithm let us take 3 sample data.

Let us assume that the average frequency of the first test sample happens to be 580Hz. Then the algorithm will compare it with the above limits and will predict that given average frequency lies completely in the male section. So the program will output the prediction to be *MALE* with 100% confidence.

In the second example, assuming that its average frequency is 670Hz.Then the algorithm will compare it with the above limits and will predict that given average frequency lies completely in the female section. So the program will output the prediction to be *FEMALE* with 100% confidence.

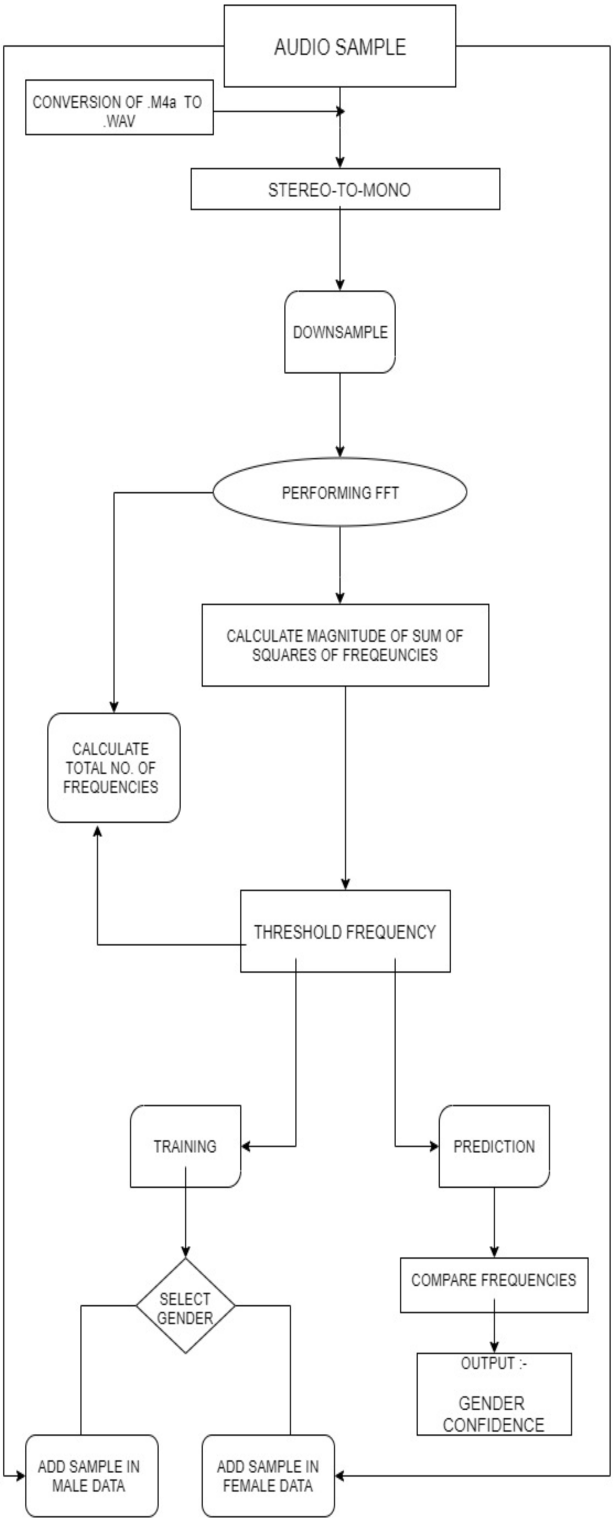


Figure 3: Flow of the program

Now the case where the average frequency lies between 600 and 650Hz (suppose 615Hz). In this case, the algorithm will compare the average frequency with 625Hz ($f_{edge} = \frac{f_{max}+f_{min}}{2}$ If the freq. of test data is less than 625 than output will be *MALE* with confidence less than 100.00% that is 90.00% from *formula (3)*.

5 Tools And Languages Used:

The Programming Languages and Tools used:

Java Interface for GUI(front-end):

- JFrame Form
- AWT Components
- Java Swing Components

Python Tools(back-end):

- Matplotlib library
- Numpy and Scipy libraries to calculate fft

6 Results:

The program shows output in terms of *Prediction* in terms of male and female and *Confidence* showing the program's confidence in its output. These results are shown during the execution of the *Testing* part of the program. Here are some images showing these results.

While training the program to calculate the average frequencies of each audio sample, the program also stores it in a separate file to keep a tab on the data. From this data, the program has a feature of showing the plotted frequencies of male and female samples that the code calculated.

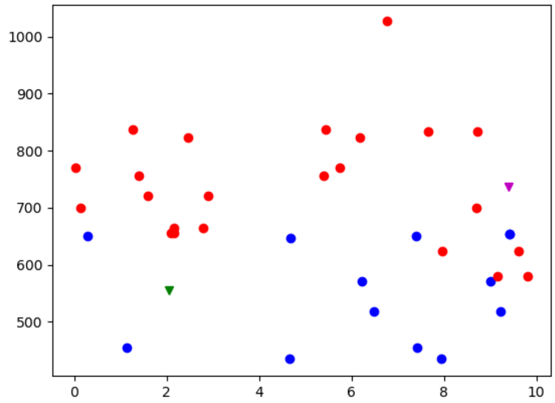


Figure 4: Average Frequency Plot

Here, the Y-axis represents the average frequency magnitude and the X-axis represents the number of data samples.

Here, the blue data points represent the average frequency of male audio samples and red data points represent those of female audio samples. As one can observe, male data points represent lower frequency than female data points.

It should also be observed here that male and female frequencies highly overlap with each other. Thus, it clearly represents that their frequency spectrum are not distinct and rather they overlap. So, a clear or sure prediction would be wrong and thus the need for *confidence* level arises in the program.

The GUI screen results are shown following:

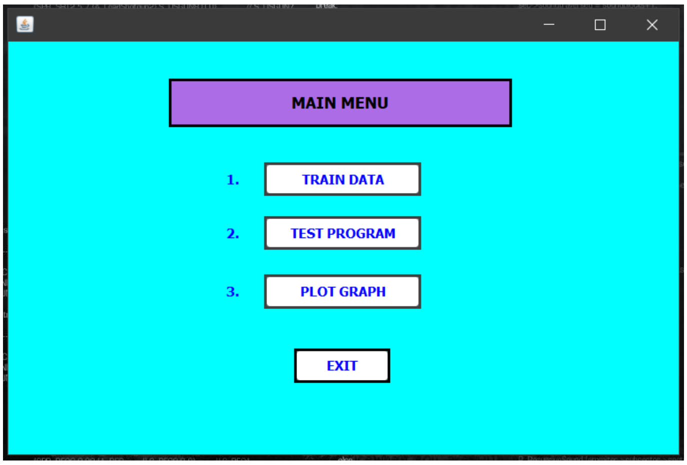
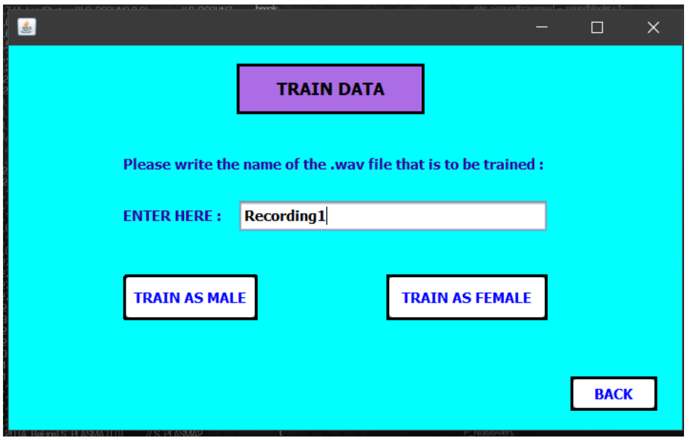


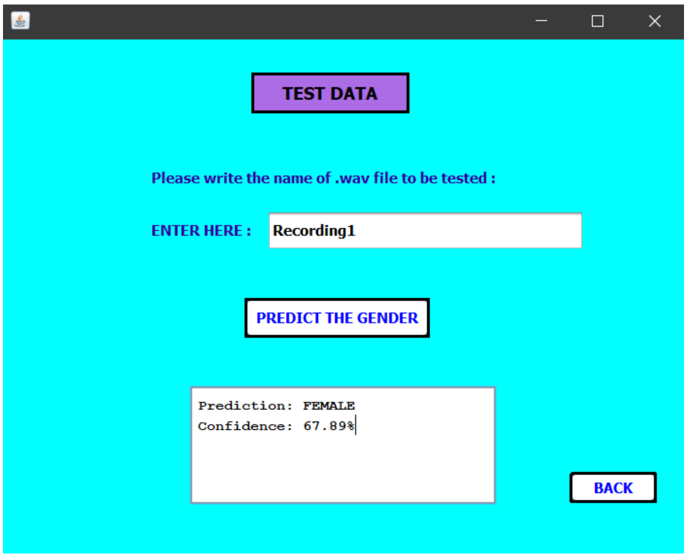
Figure 5: Main Menu



Please write the name of the .wav file that is to be trained :

ENTER HERE : Recording1

Figure 6: train data screen



Please write the name of .wav file to be tested :

ENTER HERE : Recording1

```
Prediction: FEMALE
Confidence: 67.89%
```

Figure 7: Test data screen