**Project Requirement and Specification**

***ON***

***Speech Emotion Recognition***



*Submitted by****: Dhruv Singhal***

*Roll. No.:* ***2014388***

*Semester-****6th*** *Session:* ***2021-2022***

*Guided by:* ***Mr. Yuvraj Joshi***

# DECLARATION

I, Dhruv Singhal, student of Bachelor’s in Technology VI Semester, Department of Computer Science and Engineering, Graphic Era Deemed to be University, Dehradun, declare that the technical project work entitled “**Speech** **Emotion Recognition**” has been carried out by me and submitted in partial fulfilment of the course requirements for the award of degree in Bachelor’s in Technology of Graphic Era Deemed to be University, Dehradun during the academic year 2021-2022. The matter embodied in this synopsis has not been submitted to any other university or institution for the award of any otherdegree

***About Project***

1. **Project Highlights**

This project that is designed to detect human emotions from the linked dataset. Things learned by completing this project are:

* How to use librosa
* How to extract features(frequency) from the voice
* How to split the data into training and testing sets
* How the model is able to predict the accuracy

1. **Description->**

Speech Emotion Recognition, abbreviated as SER, is the act of attempting to recognize human emotion and affective states from speech. This is capitalizing on the fact that voice often reflects underlying emotion through tone and pitch. This is also the phenomenon that animals like dogs and horses employ to be able to understand human emotion.

The human voice can be characterized by several attributes such as **pitch, loudness, and vocal tone and mainly frequency**. It has often been observed that human express their emotions by varying different vocal attributes during speech generation.

1. **Software and Libraries**

This project uses the following software and Python libraries:

* + 1. Python 3.8
    2. Kaggle (To use these datasets, you can use Kaggle notebooks within your browser )
    3. Librosa (It is a python package for music and audio analysis, used when we work with audio data and for emotion extraction (using LSTM’s). It isrequired to plot graphs.)
    4. TESS Dataset ( Toronto emotional speech set)
    5. NumPy ( It is a Python package for scientific calculation, for example performing different operations on matrix..)
    6. Seaborn (Seaborn is one of an amazing libraries for visualization of the graphical statistical plotting in Python. Seaborn provides many color palettes and defaults beautiful styles to make the creation of many statistical plots in Python more attractive.)

* + 1. Matplotlib (Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.)
    2. Os(: It is a Python package for using an operating system, for example, obtain the base name of a file, open the file in different modes like reading, write, append.)

NOTE: You will also need to python installed in your system and ide to code like vs code and then we also need to install all these packages in your system. But with predefined high-quality dataset it is highly recommended to use **KAGGLE** (as it provides us with all the necessary audio files).

**SOME TECHNICAL TERMS USED:**

* **Pitch**— how high or low a sound is. It depends on frequency, higher pitch is high frequency
* **Frequency** — speed of vibration of sound, measures wave cycles per second
* **Chroma**— Representation for audio where spectrum is projected onto 12 bins representing the 12 distinct semitones (or chroma). Computed by summing the log frequency magnitude spectrum across octaves.
* **MFCC** — Mel Frequency Cepstral Coefficients: Voice is dependent on the shape of vocal tract including tongue, teeth, etc. Representation of short-time power spectrum of sound, essentially a representation of the vocal tract.

## Requirement of Project

**Hardware Requirement**

* + - 1. Memory and disk space required per user: 512MB RAM + 1GB of disk

+ .5 CPU core.

2. Server overhead: 2-4GB or 10% system overhead (whatever is larger),

.5 CPU cores, 10GB disk space

**WORKING OF PROJECT**

**STEPS:**

**1.**Ensuring whether we have a proper dataset to test or not. (so we are using a predefined large collection of dataset, nearly 50 Gb from the Kaggle platform i.e. Toronto emotion speech set). This is having sound files.

**2**.Now we are using this Kaggle cloud service for coding. We have used python as a programming language for coding purpose.

3. We will load the data from ADD DATA in KAGGLE.

4. We will import all the necessary files.

5.We will the create data frames from dataset, to examine individual audio file.

6.We will then create different graphs and charts to examine the varying nature of frequency in different audios.

7. We will now extract features from it, then split the dataset into training and testing sets.

8.Finally, we’ll calculate the accuracy of our model

**CODE used to develop this project**

## Import

### **import** pandas **as** pd

### **import** numpy **as** np

### **import** os

### **import** seaborn **as** sns

### **import** matplotlib.pyplot **as** plt

### **import** librosa

### **import** librosa.display

### **from** IPython.display **import** Audio

### **import** warnings

### warnings**.**filterwarnings('ignore')

## Load the Dataset

paths **=** []

labels **=** []

**for** dirname, \_, filenames **in** os**.**walk('/kaggle/input'):

**for** filename **in** filenames:

paths**.**append(os**.**path**.**join(dirname, filename))

label **=** filename**.**split('\_')[**-**1]

label **=** label**.**split('.')[0]

labels**.**append(label**.**lower())

**if** len(paths) **==** 2800:

**break**

print('Dataset is Loaded')

len(paths)

paths[:5]

labels[:5]

*## Create a dataframe*

df **=** pd**.**DataFrame()

df['speech'] **=** paths

df['label'] **=** labels

df**.**head()

df['label']**.**value\_counts()

sns**.**countplot(df['label'])

**def** waveplot(data, sr, emotion):

plt**.**figure(figsize**=**(10,4))

plt**.**title(emotion, size**=**20)

librosa**.**display**.**waveplot(data, sr**=**sr)

plt**.**show()

**def** spectogram(data, sr, emotion):

x **=** librosa**.**stft(data)

xdb **=** librosa**.**amplitude\_to\_db(abs(x))

plt**.**figure(figsize**=**(11,4))

plt**.**title(emotion, size**=**20)

librosa**.**display**.**specshow(xdb, sr**=**sr, x\_axis**=**'time', y\_axis**=**'hz')

plt**.**colorbar()

emotion **=** 'fear'

path **=** np**.**array(df['speech'][df['label']**==**emotion])[0]

data, sampling\_rate **=** librosa**.**load(path)

waveplot(data, sampling\_rate, emotion)

spectogram(data, sampling\_rate, emotion)

Audio(path)

emotion **=** 'angry'

path **=** np**.**array(df['speech'][df['label']**==**emotion])[1]

data, sampling\_rate **=** librosa**.**load(path)

waveplot(data, sampling\_rate, emotion)

spectogram(data, sampling\_rate, emotion)

Audio(path)

emotion **=** 'disgust'

path **=** np**.**array(df['speech'][df['label']**==**emotion])[0]

data, sampling\_rate **=** librosa**.**load(path)

waveplot(data, sampling\_rate, emotion)

spectogram(data, sampling\_rate, emotion)

Audio(path)

emotion **=** 'neutral'

path **=** np**.**array(df['speech'][df['label']**==**emotion])[0]

data, sampling\_rate **=** librosa**.**load(path)

waveplot(data, sampling\_rate, emotion)

spectogram(data, sampling\_rate, emotion)

Audio(path)

emotion **=** 'sad'

path **=** np**.**array(df['speech'][df['label']**==**emotion])[0]

data, sampling\_rate **=** librosa**.**load(path)

waveplot(data, sampling\_rate, emotion)

spectogram(data, sampling\_rate, emotion)

Audio(path)

## Feature Extraction

**def** extract\_mfcc(filename):

y, sr **=** librosa**.**load(filename, duration**=**3, offset**=**0.5)

mfcc **=** np**.**mean(librosa**.**feature**.**mfcc(y**=**y, sr**=**sr, n\_mfcc**=**40)**.**T, axis**=**0)

**return** mfcc

extract\_mfcc(df['speech'][0])

X\_mfcc **=** df['speech']**.**apply(**lambda** x: extract\_mfcc(x))

X\_mfcc

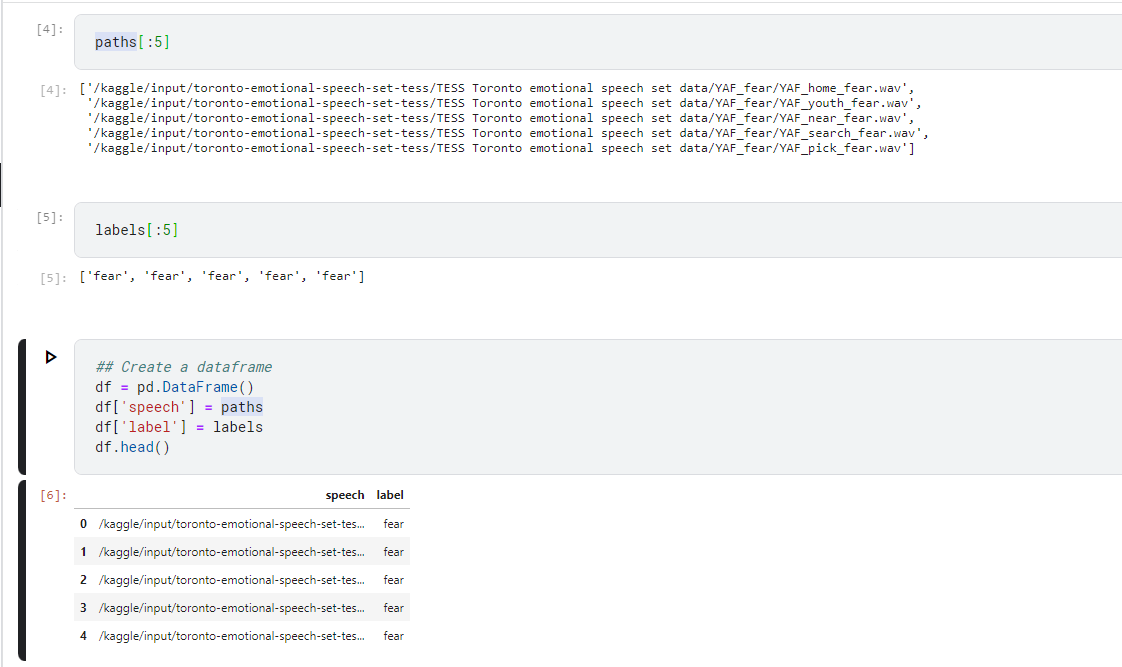
**REFERENCE-**

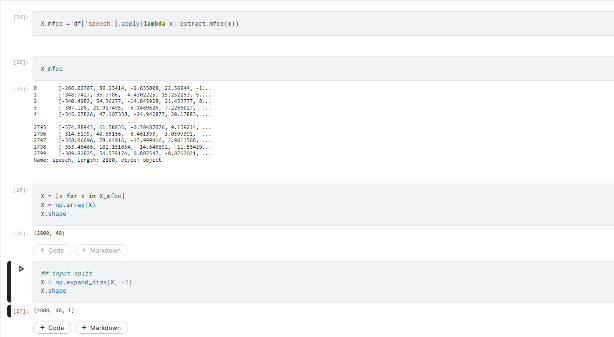
GOOGLE

YOUTUBE

UDEMY

FEW SCREENSHOTS OF PROJECT IMPLEMENTATION



Graphical user interface

Description automatically generated