SPEECH EMOTION RECOGNITION USING LIBROSA AND MLP CLASSIFIER

Project Report submitted in partial fulfillment

for the award of the degree of

BACHELOR OF TECNOLOGY

Submitted by

ADITYA SRIVASTAVA 19SCSE1050008

COMPUTER SCIENCE

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Mr. Arvindhan M

Professor

MONTH & YEAR

NOV-2021

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"SPEECH EMOTON RECOGNITION USING LIBROSA AND MLP CLASSIFIE R"** in partial fulfillment of the requirements for the award of the B. Tech-CSE submitted in the School of Computing Science and Engineering of Galgotias University , Greater Noida , is an original work carried out during the period of Oct, 2021 to Dec, 2021, under the supervision of Mr Arvindhan M, Department of Computer Science and Engineering / Computer Application and Information and Science , of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

Aditya Srivastava 19scse1050008

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Mr. Arvindhan M

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of Aditya Srivastava 19SCSE1050008 has been held on D D /MM/YYYY and his/her work is recommended for the award of B. Tech-CSE.

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date: November, 2021
Place: Greater Noida

SPEECH EMOTION RECOGNITION USING LIBROSA AND MLP CLASSIFIER

Abstract:

The ability to modulate vocal sounds and generate is one of the features which set humans apart from the other living beings. The human voice can be characterised by several attributes such as pitch, timbre, loudness, and vocal tone. It has often been observed that human express their emotions by varying different vocal attributes during speech generation.

Hence, deduction of human emotions through voice and speech analysis has a practical plausibility and could potentially be beneficial for improving human conversational and persuasion skills.

Developing emotion recognition systems that are based on speech has practical application benefits. However, these benefits are somewhat neglected by the real-world background noise impairing speech-based emotions recognition performance when the system is employed in practical applications. Speech Emotion Recognition (SER) is one of the most challenging tasks in the speech signal analysis domain, it is a research area problem that tries to infer emotion from speech signals.

TABLE OF CONTENTS

Introduction:

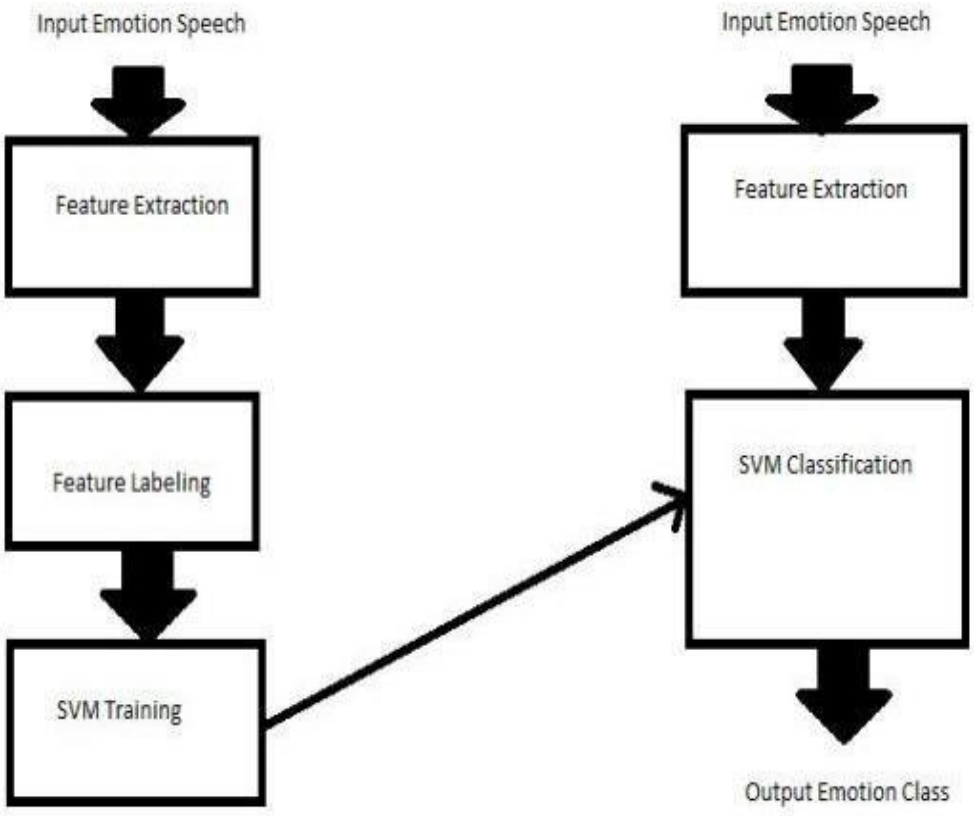
Speech Emotion Recognition (SER) is a task of recognizing the emotion from speech irrespective of the semantic contents. However, emotions are subjective and even for humans it is hard to notate them in neural speech communication regardless of the meaning. The ability to automatically conduct is very difficult task and still an ongoing subject of research. In this project we will use the audio speech files made by 24 actors (12 male and 12 female) vocalizing two lexically-matched statements in a North-American accent. Speech includes eight expressed emotions (neutral, calm, happy, angry , sad, fearful, disgust, and surprise) expressed in normal and strong intensity.

In this project, the first step is working with audio files is to turn the audio wave into numbers so that you can feed this into your machine learning algorithm. That will be like as the sound wave that move over time and which can be plotted on the graph. From that graph we can measure the height of the wave at equal intervals This is exactly what the sampling rate means.

Literature Review:

Our project deals with speech emotion recognition. In which we have use visual studio in background for coding. We have used python as a programming language for coding purpose. For database purpose we have used RAVDESS dataset. In that dataset, there is a audio file of 24 different actors (12 male and 12 female). Speech recognition can allow students with learning disabilities to become better writers. By saying the words aloud, they can increase the fluidity of their writing, and be alleviated of concerns regarding spelling, punctuation, and other mechanics of writing. For many interactions between a person and a machine, a dialogue is needed to establish a complete interaction with the machine. The Ideal dialogue allows either the user or the machine: to initiate or to choose to respond to queries initiated by the other side.

Project Design:



Modules Description:

In this the emotions in the speech are predicted using neural networks. Multi-Layer Perceptron Classifier (MLP Classifier) and RAVDESS(Rayerson Audio-Visual Database of Emotional Speech and Song dataset) are used.

a) Database Description

RAVDESS dataset has recordings of 24 actors, 12 male actors, and 12 female actors, the actors are numbered from 01 to 24. The male actors are odd in number and female actors are even in number. The emotions contained in the dataset are as sad, happy, neural, angry, disgust, surprised, fearful and calm expressions. The dataset contains all expressions in three formats, those are: Only Audio, Audio-Video and Only Video. Since our focus is on recognise emotions from speech, this model is trained on Audio-only data.

A Multi-Layer perceptron(MLP) is a network made up of perceptron. It has an input layer that receives the input signal, an output layer that makes predictions or decisions for a

given input, and the layers present in between the input and output layer is called hidden layer. In the proposed methodology for Speech Emotion Recognition, the MLP network will have one input layer, 300,40,80,40 hidden layers and one output layer. The hidden layers will be large numbers and number of hidden layers can be changed as per requirements.

b) Features

The data was acquired directly from the group of Audio files and they were transformed in 264 vectors of features. A wide range of possibilities exist for parametrically representing a speech signal and its content in a vector, with intention to extract a relevant information from it.

The learning process covers two steps , the first step is a forward processing of input data by the neurons that produces a forecasted output, the second step is the adjustment of weights within neuron layers, in order to minimize the errors of forecasted solution compared with the correct output.

Code :

To run code in the JupyterLab, you'll first need to run it with the command prompt:

C:\Users\DataFlair>jupyter lab

You'll need to install the following libraries with pip:

pip install librosa soundfile numpy sklearn pyaudio

1. Make the necessary imports:

```
import librosa
import soundfile
import os, glob, pickle
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score
```

2. Define a function extract_feature to extract the mfcc, chroma, and mel features from a sound file.

```
#DataFlair - Extract features (mfcc, chroma, mel) from a sound file
def extract_feature(file_name, mfcc, chroma, mel):
   with soundfile.SoundFile(file_name) as sound_file:
      X = sound_file.read(dtype="float32")
      sample_rate=sound_file.samplerate
      if chroma:
         stft=np.abs(librosa.stft(X))
      result=np.array([])
      if mfcc:
         mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
         result=np.hstack((result, mfccs))
      if chroma:
         chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
         result=np.hstack((result, chroma))
if mel:
         mel=np.mean(librosa.feature.melspectrogram(X, sr=sample_rate).T,axis=0)
         result=np.hstack((result, mel))
return result
```

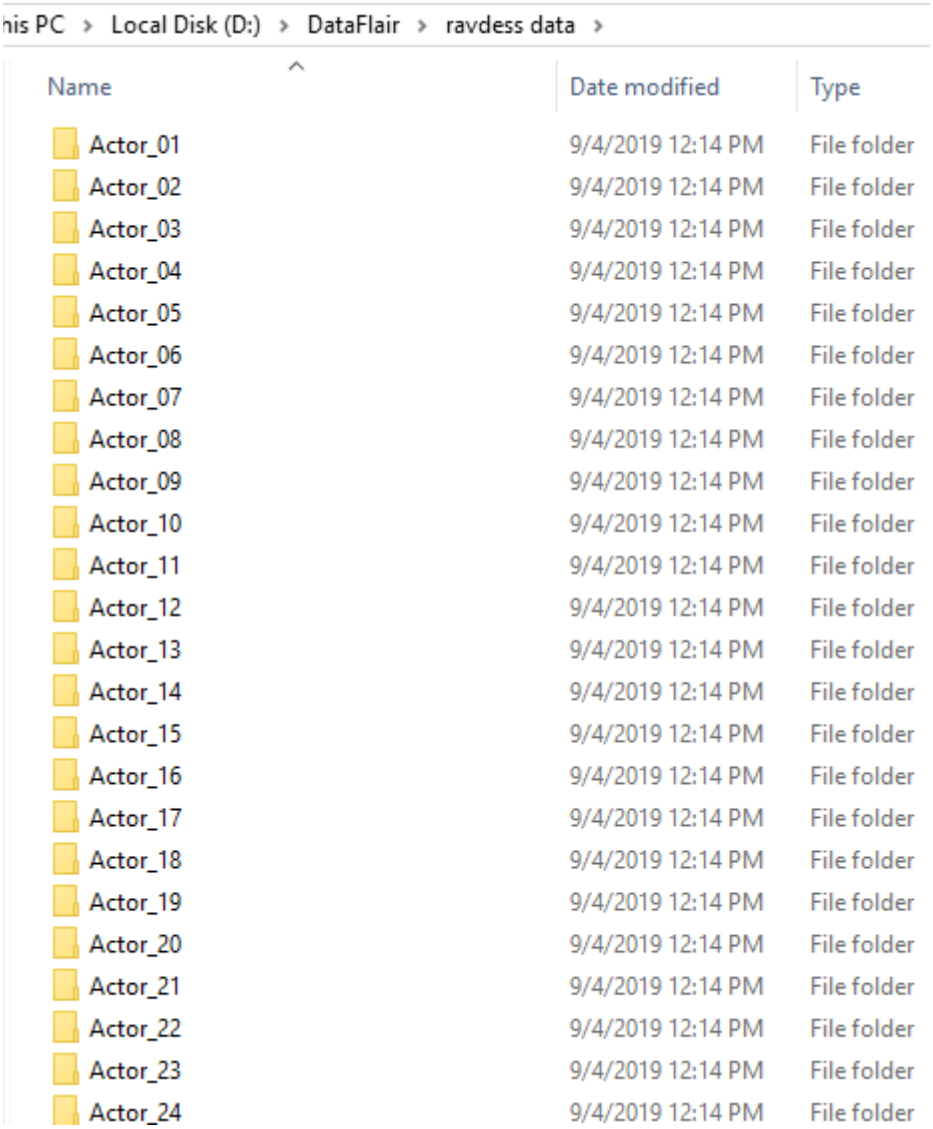
3. Now, let's define a dictionary to hold numbers and the emotions available in the RAVDESS dataset, and a list to hold those we want- calm, happy, fearful, disgust.

```
#DataFlair - Emotions in the RAVDESS dataset
emotions={
 '01':'neutral',
 '02':'calm',
 '03':'happy',
 '04':'sad',
```

```
 '05':'angry',
 '06':'fearful',
 '07':'disgust',
 '08':'surprised'
}

#DataFlair - Emotions to observe
observed_emotions=['calm', 'happy', 'fearful', 'disgust']
```

4. Now, let's load the data with a function load_data() – this takes in the relative size of the test set as parameter. x and y are empty lists; we'll use the glob() function from the glob module to get all the pathnames for the sound files in our dataset.



Using our emotions dictionary, this number is turned into an emotion, and our function checks whether this emotion is in our list of observed_emotions; if not, it continues to the

next file. It makes a call to extract_feature and stores what is returned in 'feature'. Then, it appends the feature to x and the emotion to y. So, the list x holds the features and y holds the emotions. We call the function train_test_split with these, the test size, and a random state value, and return that.

```
#DataFlair - Load the data and extract features for each sound file
def load_data(test_size=0.2):
  x,y=[],[]
  for file in glob.glob("D:\\DataFlair\\ravdess data\\Actor_*\\*.wav"):
    file_name=os.path.basename(file)
    emotion=emotions[file_name.split("-")[2]]
    if emotion not in observed_emotions:
      continue
    feature=extract_feature(file, mfcc=True, chroma=True, mel=True)
    x.append(feature)
    y.append(emotion)
  return train_test_split(np.array(x), y, test_size=test_size, random_state=9)
```

5. Time to split the dataset into training and testing sets! Let's keep the test set 25% of everything and use the load_data function for this.

```
#DataFlair - Split the dataset
x_train,x_test,y_train,y_test=load_data(test_size=0.25)
```

6. Observe the shape of the training and testing datasets:

```
#DataFlair - Get the shape of the training and testing datasets
print((x_train.shape[0], x_test.shape[0]))
```

7. And get the number of features extracted.

```
#DataFlair - Get the number of features extracted
print(f'Features extracted: {x_train.shape[1]}')
```

```
[7]: #DataFlair - Get the number of features extracted
     print(f'Features extracted: {x_train.shape[1]}')

     Features extracted: 180
```

8. Now, let's initialize an MLPClassifier. This is a Multi-layer Perceptron Classifier; it optimizes the log-loss function using LBFGS or stochastic gradient descent. Unlike SVM or Naive Bayes, the MLPClassifier has an internal neural network for the purpose of classification. This is a feedforward ANN model.

```
#DataFlair - Initialize the Multi Layer Perceptron Classifier
```

model=MLPClassifier(alpha=0.01, batch_size=256, epsilon=1e-08, hidden_layer_sizes=(300,), learning_rate='adaptive', max_iter=500)

```
[8]: #DataFlair - Initialize the Multi Layer Perceptron Classifier
     model=MLPClassifier(alpha=0.01, batch_size=256, epsilon=1e-08, hidden_layer_sizes=(300,), learning_rate='adaptive', max_iter=500)
```

9. Fit/train the model.

   #DataFlair - Train the model

   model.fit(x_train,y_train)

```
[9]: #DataFlair - Train the model
     model.fit(x_train,y_train)

[9]: MLPClassifier(activation='relu', alpha=0.01, batch_size=256, beta_1=0.9,
                  beta_2=0.999, early_stopping=False, epsilon=1e-08,
                  hidden_layer_sizes=(300,), learning_rate='adaptive',
                  learning_rate_init=0.001, max_iter=500, momentum=0.9,
                  n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5,
                  random_state=None, shuffle=True, solver='adam', tol=0.0001,
                  validation_fraction=0.1, verbose=False, warm_start=False)
```

10. Let's predict the values for the test set. This gives us y_pred (the predicted emotions for the features in the test set).

    #DataFlair - Predict for the test set

    y_pred=model.predict(x_test)

```
[10]: #DataFlair - Predict for the test set
      y_pred=model.predict(x_test)
```

11. To calculate the accuracy of our model, we'll call up the accuracy_score() function we imported from sklearn. Finally, we'll round the accuracy to 2 decimal places and print it out.

    #DataFlair - Calculate the accuracy of our model

    accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)

    #DataFlair - Print the accuracy

    print("Accuracy: {:.2f}%".format(accuracy*100))

```
[11]: #DataFlair - Calculate the accuracy of our model
      accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)

      #DataFlair - Print the accuracy
      print("Accuracy: {:.2f}%".format(accuracy*100))

      Accuracy: 72.40%
```

```
[ ]:
```

Result:

In this project we learned to recognize emotions from speech. We used an MLPClassifier for this and made use of the sound file library to read the sound file, and the librosa library to extract features from it. As you'll see, the model delivered an accuracy of 72.4%. That's good enough for us yet.

This is the final output of the project which gives us the accuracy of 72.4%.


```
[11]: #DataFlair - Calculate the accuracy of our model
      accuracy=accuracy_score(y_true=y_test, y_pred=y_pred)

      #DataFlair - Print the accuracy
      print("Accuracy: {:.2f}%".format(accuracy*100))

      Accuracy: 72.40%
```

```
[ ]:
```


Conclusion:

The overall system results how we can leverage Machine learning to obtain the ultimate emotions from audio data and some insights on the human expression of emotion through voice. This systems can be employed in a variety of setups like Call Centre for complaints or marketing, in voice-based virtual assistants or chatbots, in linguistic research, etc.

Some of the possible steps that can be implemented to make the models strongly formed and accurate are the following:

- λ An accurate implementation of the speed of the speaking can be explored to check if there is any error so it can resolve some of the deficiencies of the model.
- λ Figuring out a way to clear aimless silence from the audio clip.
- λ Exploring other acoustic properties of sound data to check their applicability in domain of speech emotion recognition.
- λ Following lexical features-based approach towards SER and using an ensemble of the lexical and acoustic models, will improve accuracy of the system.

REFERENCES

[1] Tuomas Eerola and Jonna K. Vuoskoski, "A comparison of the discrete and dimensional models of emotion in music", Psychology of Music, 1–32, The Author(s) 2010.

[2] Eddie Harmon-Jones, Cindy Harmon-Jones, and Elizabeth Summerell, "On the Importance of Both Dimensional and Discrete Models of Emotion" School of Psychology, The University of New South Wales, Sydney 2052, Australia.

[3] Roddy Cowie, Ellen Douglas-Cowie, Susie Savvidou, Edelle McMahon, Martin Sawey & Marc Schröder, 'FEELTRACE' Schools of Psychology and English, Queen's University Belfast.