Speech Emotion Recognition In Neurological Disorder using Deep Learning

A Project Report submitted in partial fulfillment of the requirements for the award of the degree of,

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Thummalapenta Venkata Sai Runvitha	**322010326002**
Kolla Supreeth Choudary	**322010326001**
Sompallem Sindhu	**322010326015**
Rajula Asritha	**322010326050**

Under the esteemed guidance of

Mrs Swasthika Jain T J
Assistant Professor

Department of Computer Science & Engineering,

GITAM SCHOOL OF TECHNOLOGY

GANDHI INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(Deemed to be University)

Bengaluru Campus.

April 2024

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled **"Speech Emotion Recognition In Neurological Disorder using Deep Learning"** is a bonafide record of work carried out by **T.V.S. Runvitha (322010326001), K. Supreeth Choudary (322010326001), S. Sindhu (322010326015), R. Asritha (322010326050)** submitted in partial fulfillment of requirement for the award of degree of **Bachelors of Technology in Computer Science and Engineering**.

Project Guide.	**Head of the Department.**
SIGNATURE OF THE GUIDE	**SIGNATURE OF THE HOD**
Mrs Swasthika Jain T J	**Dr. Vamsidhar Yendapalli,**
Assistant Professor	Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

DECLARATION

We, hereby declare that the project report entitled **“Speech Emotion Recognition In Neurological Disorder Using Deep Learning”** is an original work done in the **Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University)** submitted in partial fulfillment of the requirements for the award of the degree of **B.Tech.** in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree.

Date:

Registration No(s).	Name(s)	Signature(s)
1. 322010326002	**Thummalapenta Venkata Sai Runvitha**	
2. 322010326001	**Kolla Supreeth Choudary**	
3. 322010326015	**Sompallem Sindhu**	
4. 322010326050	**Rajula Asritha**	

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose consistent guidance and encouragement crowned our efforts with success.

We consider it our privilege to express our gratitude to all those who guided us in the completion of the project.

We express our gratitude to Director Prof. **Basavaraj Gundappa Katageri** for having provided us with the golden opportunity to undertake this project work in their esteemed organization.

We sincerely thank **Dr. Y. Vamshidhar,** HOD, Department of Computer Science and Engineering, Gandhi Institute of Technology and Management, Bengaluru for the immense support given to us.

We express our gratitude to our project guide **(MRS SWASTHIKA JAIN T J), (ASSISTANT PROFESSER)**, Department of Computer Science and Engineering, Gandhi Institute of Technology and Management, Bengaluru, for their support, guidance, and suggestions throughout the project work.

Student Name's	**Registration No.**
Thummalapenta Venkata Sai Runvitha	322010326002
Kolla Supreeth Choudary	322010326001
Sompallem Sindhu	322010326015
Rajula Asritha	322010326050

ABSTRACT

This project endeavors to address the critical challenge of early detection and monitoring of Alzheimer's neurological disorder by developing a sophisticated speech emotional recognition system. The system is designed to utilize machine learning technology in order to analyze emotional patterns in speech, ultimately aiding in the early detection of disorders. By using this technology, diagnosis accuracy and efficiency could be significantly improved and disease progression could be tracked.

In this process of recognition, Multilayer perceptron classifier is used to predict the emotion state from input data. It underscores their adaptability, robustness, and wide applicability, making them a valuable tool for solving complex machine learning problems across various domains.

In human machine interface application, emotion recognition from the speech signal has been research topic since many years. Emotions play an extremely important role in human mental life. It is a medium of expression of one's perspective or one's mental state to others. Speech Emotion Recognition (SER) in Neurological Disorder can be defined as extraction of the emotional state of the speaker from his or her speech signal. There are few universal emotions- including Neutral, Anger, Happiness, Sadness etc in which any intelligent system with finite computational resources can be trained to identify or synthesize as required. In this work, we are extracting Mel-frequency cepstral coefficients (MFCC). Deep Neural Network is used to classify the emotion in this work.

TABLE OF CONTENTS

Title	Page No.
Declaration	I
Acknowledgement	II
Abstract	III
Table of Contents	IV
List of Figures	V
List of Tables	V
1. INTRODUCTION	1
2. LITERATURE SURVEY	2
3. SOFTWARE AND HARDWARE SPECIFICATIONS	6
3.1 Introduction	6
3.2 Specific Requirements	6
3.2.1 Functional Requirement	6
3.2.2 Non-Functional Requirement	7
3.3 Hardware and Software Requirement	7
3.3.1 Hardware Requirement	7
3.3.2 Software Requirement	7
4. PROBLEM STATEMENT	8
4.1 Objectives	8
5. DESIGNING	9
5.1 System Architecture	9
5.2 Methodology	14
5.2.1 Feature Extraction	14
5.2.2 CNN Model	14
5.2.3 Model Training and Testing	14
5.3 Modules	15
6. IMPLEMENTATION	16
7. TESTING	28
8. EXPERIMENTAL RESULTS	29
8.1 Output	29
8.2 Result	31
9. CONCLUSION	32
10. FUTURE WORK	32
REFERENCES	33

LIST OF FIGURES

Fig	Title	Page No.
5.1	Speech Emotion Recognition in Neurological Disorders System Architecture	9
5.1.1.1	Class Diagram	10
5.1.1.2	Object Diagram	11
5.1.1.3	Use Case Diagram	11
5.1.1.4	Sequence Diagram	12
5.1.1.5	Activity Diagram	12
5.1.1.6	Data Flow Diagram (DFD)	13

LIST OF TABLES

Table	Table Title	PageNo.
8.2	Result	31

1. INTRODUCTION

What is Alzheimer’s Disease:

Alzheimer's disease is a progressive neurodegenerative disorder that is the most common cause of dementia. There is no cure for Alzheimer's disease, but there are treatments that can help manage the symptoms. It will destroy the cell themselves degrade and die, it eventually destroying the memory and other important mental functions. Till now no treatment is exists to cure the disorder.

The disease is characterized by the accumulation of abnormal protein deposits in the brain, including beta-amyloid plaques and tau tangles, which disrupt communication between brain cells and lead to their deterioration. As Alzheimer's advances, the brain experiences atrophy, and symptoms become more pronounced. Common manifestations include memory loss, cognitive decline, confusion, and changes in behaviour and personality. Though the exact cause remains elusive, a combination of genetic, environmental, and lifestyle factors is believed to contribute. While there is currently no cure, medications and interventions can help manage symptoms and improve the quality of life for those affected. Research is ongoing to explore potential treatments and preventive strategies. Early diagnosis is crucial for effective intervention, making consultation with healthcare professionals important for individuals experiencing memory loss or cognitive decline.

Neurological disorders often change in speech patterns and emotional expressions. So now based on traditional diagnosis it is hard to find the problem and it is taking lot of time. Using early detection capabilities requires a robust system that can effectively recognize speech patterns and analyse emotions. The disorder will be analyzed using machine learning algorithms.

Neurological disorders often change in speech patterns and emotional expressions. So now based on traditional diagnosis it is hard to find the problem and it is taking lot of time. Using early detection capabilities requires a robust system that can effectively recognize speech patterns and analyze emotions. The disorder will be analyzed using machine learning algorithms.

2 LITERATURE SURVEY:

In their 2019 study, (Setiyo Budiyanto), Harry Candra Sihombing, and Fajar Rahayu I. investigated the detection of depression and anxiety utilizing the Closed-Loop method with the DASS-21 instrument. Their research focused on analyzing data obtained from Facebook users, culminating in the successful identification of depression and anxiety through various techniques. The study aimed to address a gap in the existing literature by exploring the feasibility and accuracy of closed-loop approaches in mental health monitoring and support. By evaluating the effectiveness of these methods, the researchers sought to contribute to a more comprehensive understanding of their potential benefits and limitations in aiding individuals experiencing depression and anxiety.[1]

In their 2017 study, (Nurulhuda Zainuddin), Ali Selamat, and Roliana Ibrahim explored hybrid sentiment classification within aspect-based sentiment analysis on Twitter. They highlighted the efficacy of supervised sentiment analysis techniques, particularly support vector machine and naïve Bayesian classifications. The researchers identified a research gap in the selection of features, employing approaches such as information gain and the chi-square test to enhance accuracy in sentiment analysis. Their investigation aimed to bridge this gap by evaluating the effectiveness of various feature selection methods, ultimately contributing to advancements in sentiment analysis accuracy within the context of Twitter data.[2]

In their 2019 study, (Mrs. Dhanamma Jagli and Ms. Pooja Shetty) delved into the realm of human emotion recognition using machine learning techniques. They focused on the classification of human emotions based on facial expressions, aiming to develop a system capable of accurately identifying emotions depicted in images. The researchers identified a significant research gap in the potential of artificial intelligence (AI) systems to comprehend users' feelings and emotions, and subsequently respond accordingly. They proposed the development of an application that could seamlessly detect users' emotions, allowing them to choose their preferred emotional response. This research seeks to bridge the divide between human emotion recognition technology and user interaction, with the ultimate goal of enhancing user experience and emotional engagement.[3]

In their 2016 research, (Soujanya Poria), Erik Cambria, Newton Howard, Guang-Bin Huang, and Amir Hussain explored the integration of audio, visual, and textual cues for sentiment analysis from multimodal content. They employed MFCC calculation and spectral centroid analysis to extract audio features, while enriching textual sentiment analysis with sentic computing-based features, leading to substantial enhancements in performance. The researchers identified key gaps in their work, focusing on improving the decision-level fusion process through a cognitively-inspired fusion engine. Additionally, they aimed to reduce the time complexities of their developed methods to move closer to the ambitious goal of creating a real-time system. Their ultimate objective was to establish an efficient and reliable framework for multimodal sentiment analysis.[4]

In 2023, (Jiaxuan He, Sijie Mai, and Haifeng Hu) introduced a novel model, the Unimodal Reinforced Transformer with Time Squeeze Fusion, for multimodal sentiment analysis. Their proposed approach outperforms baseline models in sentiment analysis, demonstrating superior accuracy and F1 scores while requiring fewer parameters. The researchers specifically target the challenge of unaligned multimodal sentiment analysis by introducing innovative techniques such as time squeeze fusion and unimodal reinforced Transformer methods, thereby addressing a significant research gap in the field. Their work contributes to advancing the capabilities of sentiment analysis across multiple modalities, paving the way for more effective and efficient analysis of complex emotional content.[5]

In 2021, (Li Yang, Ying Li, and Jin Wang) conducted an investigation into Arabic language sentiment analysis within both unimodal and multimodal contexts. They underscored the significance of sentiment analysis in comprehending online opinions, particularly noting the underexplored nature of Arabic sentiment analysis despite its widespread presence online. The researchers highlighted the potential for improvement through the application of multimodal sentiment analysis and deep learning techniques. They identified a research gap stemming from insufficient attention to Arabic sentiment analysis, exacerbated by a scarcity of Arabic language resources, which presents obstacles to thorough analysis. This gap signals an opportunity for further research, particularly in exploring multimodal approaches and leveraging deep learning methodologies to enhance sentiment analysis in the Arabic language.[6]

In 2020, (Guorong Xiao, Geng Tu, Lin Zheng, and Teng Zhou) introduced a novel approach for multimodality sentiment analysis in the social Internet of Things (IoT) context, employing hierarchical attentions and a CSAT-TCN with an MBM network. Their method is particularly tailored for summarizing sentiment in product reviews, with each hidden neuron representing a single word. Utilizing deep learning techniques, they employed an RNN model to handle bigrams and trigrams extraction, enhancing the analysis process. Addressing a research gap, they introduced the H-SATF (Hierarchical Self-Attention Fusion) mechanism to fuse multimodality features, thereby improving the performance of the CSAT-TCN model, especially in handling long memory issues. The integrated components collectively contribute to effectively enhancing sentiment recognition accuracy, thus advancing the field of multimodality sentiment analysis in the context of the social Internet of Things.[7]

In 2022, (Stankevich Maxim), Nikolay Ignatiev, and Ivan Smirnov explored the prediction of depression using social media images. Their study employed various machine learning models such as Logistic Regression, Support Vector Machine, and Multi-layer Perceptron to forecast depression based on these images. A notable research gap was identified concerning the scarcity of studies utilizing Vkontakte data for depression prediction, which differs from the typical Instagram-style data often used in similar research. This divergence in data sources poses challenges when comparing data collection methods and model performance with other related works. Despite these challenges, their investigation provides valuable insights into the potential of social media image analysis for predicting depression, contributing to advancements in mental health research and technology.[8]
In 2018, (Iti Chaturvedi, Ranjan Satapathy, Sandro Cavallari, and Erik Cambria) explored fuzzy commonsense reasoning for multimodal sentiment analysis. Their study aimed to categorize text and video content into positive, negative, or neutral sentiments, with applicability across various languages such as English and Spanish. They identified a research gap concerning the conventional use of Long Short-Term Memory (LSTM), which typically models each word with a single neuron. In contrast, their approach involved employing a low-dimensional Recurrent Neural Network (RNN) trained through deep learning techniques for classification purposes. This innovative methodology presents a departure from traditional LSTM approaches and holds promise for advancing the field of sentiment analysis across multiple modalities.[9]
In 2021, (Lukas Stappen and Alice Baird) from the University of Augsburg delved into sentiment analysis and topic recognition within video transcriptions. Their approach involved utilizing SenticNet to extract natural language concepts, with a focus on exploring the significance of high-contextual features for transcription analysis. The researchers identified a notable research gap pertaining to the absence of prior studies examining the application of high-contextual features in the analysis of video transcriptions, especially concerning sentiment and topic recognition. Their investigation aimed to address this gap by shedding light on the potential benefits of incorporating such features into the analysis process, thereby contributing to advancements in sentiment analysis and topic recognition within the realm of video transcription studies.[10]
In 2023, (Li Yang, Ying Li, and Jin Wang) introduced a novel approach named the Sentiment Lexicon-based Convolutional Neural Networks and Bidirectional Gated Recurrent Units (SLCABG) model for sentiment analysis of e-commerce product reviews in Chinese. This model is designed as a fusion of sentiment lexicon-based methods with convolutional neural networks (CNN) and bidirectional gated recurrent units (BiGRU). The research paper briefly outlines the objective of the SLCABG model, aiming to address the limitations of existing sentiment analysis techniques for product reviews. However, it does not explicitly specify these shortcomings nor provide an extensive

comparison with established methodologies. Through the SLCABG model, the researchers aim to contribute to the advancement of sentiment analysis in e-commerce contexts, particularly in the Chinese language domain, by leveraging the combined strengths of lexicon-based approaches and deep learning methodologies.[11]

In 2021, (Andreea-Maria Copaceanu) from The Bucharest University of Economic Studies in Romania conducted a study on sentiment analysis utilizing a machine learning approach. The research explored various machine learning algorithms and feature extraction methods, ultimately finding that Support Vector Machine (SVM) yielded the best performance. Specifically, the analysis focused on sentiment within Amazon mobile phone reviews, utilizing a dataset comprising 67,986 instances across different brands. While the text does not explicitly delineate the research gap, potential avenues for further investigation include addressing challenges related to imbalanced data, exploring advanced deep learning techniques, delving into multimodal analysis, and developing real-time sentiment analysis solutions tailored for e-commerce platforms. These areas represent promising directions for enhancing the effectiveness and applicability of sentiment analysis methods in the context of online reviews and consumer feedback.[12]

3 SOFTWARE AND HARDWARE SPECIFICATIONS

3.1 Introduction

Software specifications encompass a detailed outline of the functionalities, requirements, and limitations of a particular software application or system. These specifications define parameters such as supported operating systems, minimum hardware requirements, programming languages used, database compatibility, and user interface design. By providing a comprehensive overview of what a software product can deliver and how it operates, software specifications guide developers, stakeholders, and end-users in understanding the software's scope and capabilities.

3.2 Specific Requirements

3.2.1 Functional Requirement

Processor - Intel Core i5 or equivalent:

The choice of an Intel Core i5 processor or its equivalent ensures sufficient processing power to handle complex algorithms and computations involved in real-time speech emotion recognition. The i5 processor strikes a balance between performance and power efficiency, making it suitable for applications requiring moderate computational resources.

RAM - 4GB (minimum): Explanation: A minimum of 4GB of RAM is necessary to facilitate smooth and efficient operation of the speech emotion recognition software. Neurological disorders may impact cognitive functions, and allocating adequate RAM helps in minimizing latency and ensuring seamless processing of speech data, enhancing user experience and accuracy.

Hard Disk - 128GB: Explanation: A hard disk with a minimum capacity of 128GB provides ample storage space for storing speech data, application files, and system resources. This ensures that the software can operate without storage constraints and allows for the retention of a substantial amount of data for analysis and future reference.

Keyboard - Standard Windows Keyboard: Explanation: Using a standard Windows keyboard ensures familiarity and ease of use for individuals with neurological disorders, many of whom may be accustomed to standard input devices. Consistency in input methods helps in reducing cognitive load and facilitating user interaction with the software.

Mouse - Two or Three Button Mouse: Explanation: A two or three-button mouse offers simple and intuitive navigation control, which is essential for individuals with neurological disorders who may have motor impairments or difficulty in precise movements. The ergonomic design of such mice enhances accessibility and usability for users with varying levels of motor function.

Monitor - Any: Explanation: The choice of monitor is flexible, allowing users to utilize any available display device that meets their preferences and requirements. This accommodates individual user preferences regarding screen size, resolution, and visual comfort, ensuring a personalized and comfortable viewing experience.

3.2.2 Non-Functional Requirement

Why Other Functional Requirements may not be Suitable:

While alternative specifications may exist, they may not be suitable for speech emotion recognition in neurological disorders due to several reasons:

Lower-grade processors or insufficient RAM may result in performance bottlenecks, leading to degraded accuracy and slower response times, which are critical for real-time emotion recognition.

Limited storage capacity could constrain the amount of data that can be processed and stored, potentially hindering comprehensive analysis and impacting the effectiveness of the software.

Non-standard keyboards or input devices may introduce unfamiliarity and difficulty in interaction for users with neurological disorders, impeding usability and accessibility.

Monitors with specific features or unconventional configurations may not align with user preferences or accessibility requirements, leading to discomfort or reduced usability for individuals with neurological disorders.

3.3 Hardware and Software Requirement

3.3.1 Hardware Requirement

• Processor	- I5/Intel Processor
• RAM	- 4GB (min)
• Hard Disk	- 128 GB
• KeyBoard	- Standard Windows Keyboard
• Mouse	- Two or Three Button Mouse
• Monitor	- Any

3.3.2 Software Requirement:

• Operating System	: Windows 7+
• Server Side Script	: Python 3.6+
• IDE	: Google Colab
• Libraries Used	: Pandas, Keras, Tensorflow, Librosa, Pickle, Matplotlib.
• Dataset	: RAVDESS speech dataset

4 PROBLEM STATEMENT

Neurological disorders often change in speech patterns and emotional expressions. So now based on traditional diagnosis it is hard to find the problem and it is taking lot of time. Using early detection capabilities requires a robust system that can effectively recognize speech patterns and analyze emotions. The disorder will be analyzed using machine learning algorithms.

Though numerous studies quantify people's emotions and sentiments (anger, fear, disgust and panic) during disasters, they are inefficient in recognizing people's attitudes toward the philanthropic aid they receive. the importance of addressing the communication difficulties faced by individuals with neurological disorders, emphasizes the limitations of current speech emotion recognition systems in this context, and underscores the need for tailored solutions to improve the quality of life for affected individuals.

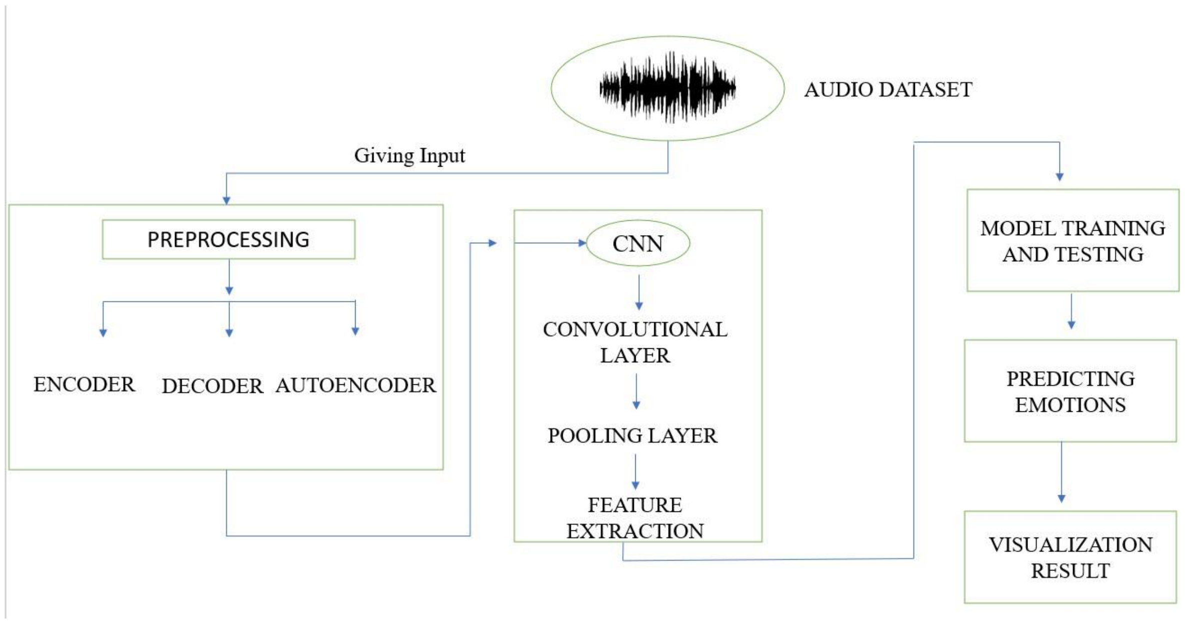
4.1 OBJECTIVES

To recognize the emotions, who are facing with Alzheimer’s Neurological disorder in patients.

To provide a efficient method for identifying a disease progression. This analysis can help to identify the emotional state of the patients. This project can be defined as a collection of methodologies that process and classify speech signals to detect emotions in them. The objective is to detect the emotions of a person or speaker and to implement a Deep Neural Network (DNN) model to create the application.

5 DESIGNING

5.1 System Architecture



The above Fig describing a process for emotion recognition from an audio dataset using a machine learning model, specifically a Convolutional Neural Network (CNN). Here's a step-by-step explanation of the process depicted: First the audio is taken and inserted this is the collection of audio samples that will be used to train and test the machine learning model. Each audio sample likely contains vocal expressions of emotions. This indicates that the audio dataset is being input into the preprocessing stage. In this stage, the audio data is prepared for analysis. This could involve normalizing the audio, removing noise, extracting features like Mel-frequency cepstral coefficients (MFCCs), or converting the audio into a spectrogram. After preprocessing, the data is passed to a CNN. A CNN is a deep learning algorithm which can take in an input (like an image or, in this case, a processed audio signal), assign importance (learnable weights and biases) to various aspects/objects in the input, and be able to differentiate one from the other. After that the extracted features are used to train the CNN model. The model learns to associate the features with different emotions. After training, the model is tested with new data to evaluate its performance. Then the trained model is now capable of predicting emotions from new audio data by analyzing the features extracted by the CNN. Finally, the results of the emotion prediction are visualized in some form, which could be a graph, chart, or any other visual representation to interpret the outcomes of the model's predictions.

5.1.1 UML Diagrams

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

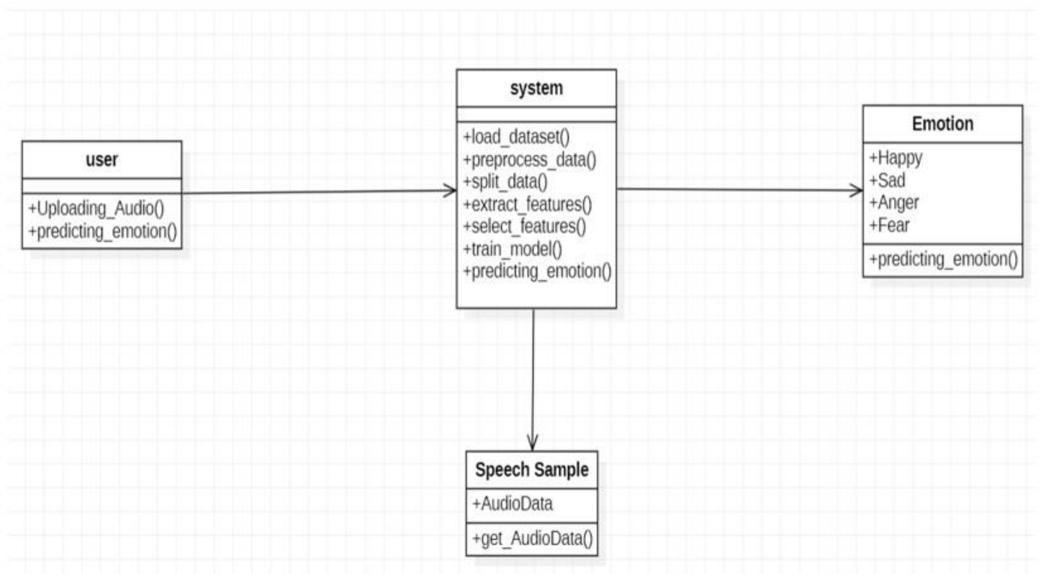
The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

5.1.1.1 CLASS DIAGRAM:

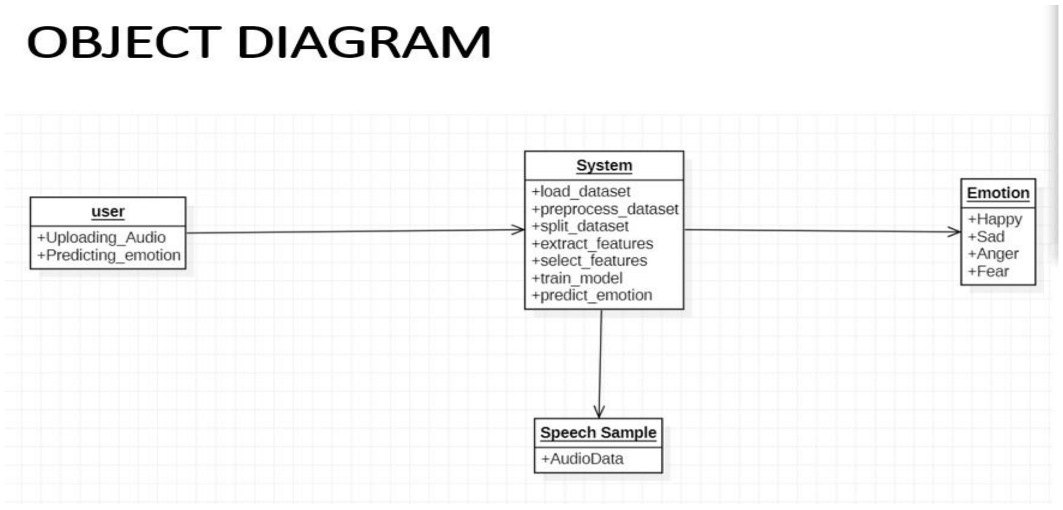
In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

• CLASS DIAGRAM



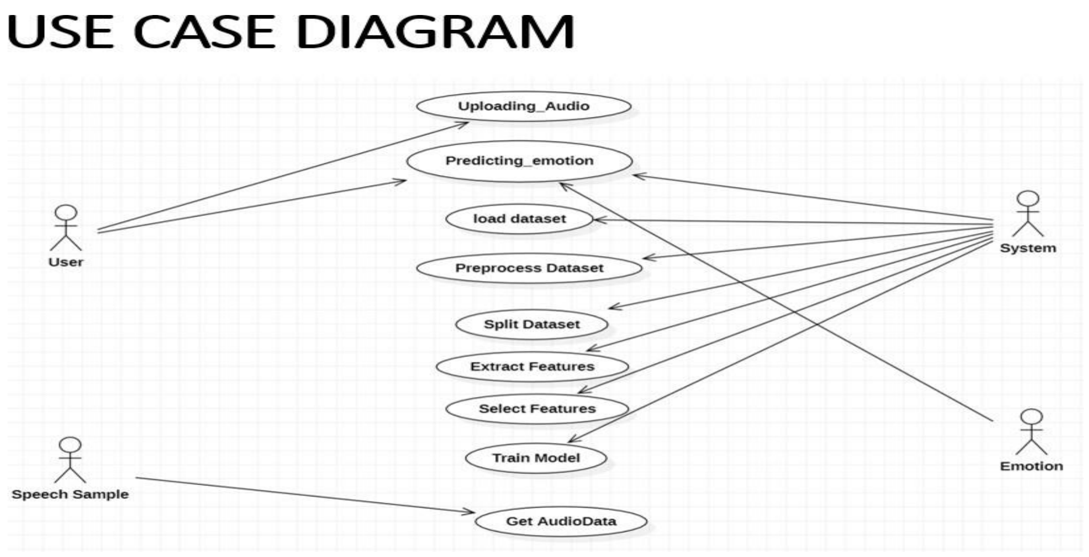
5.1.1.2 OBJECT DIAGRAM:

An object diagram in UML (Unified Modeling Language) represents a specific instance of a class diagram at a particular moment in time, showing objects and their relationships. Object diagrams are used to illustrate examples of objects and how they interact in a system.



5.1.1.3 USECASE DIAGRAM:

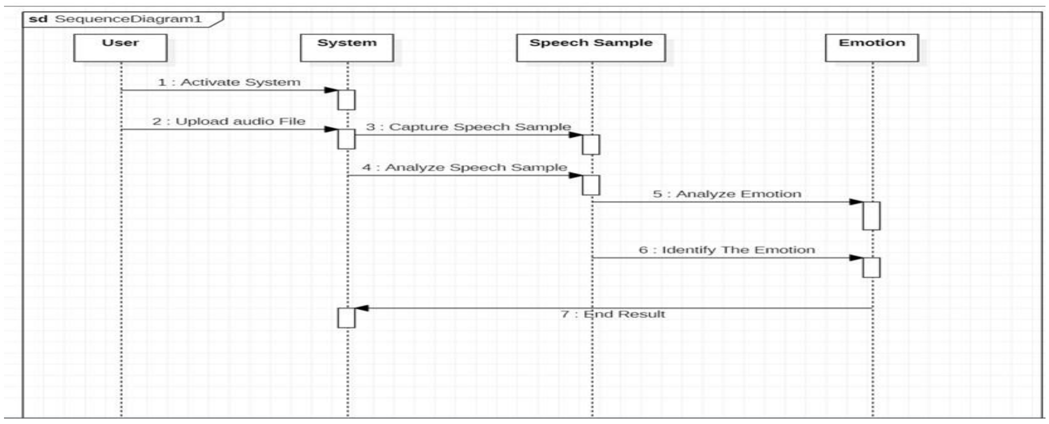
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



5.1.1.4 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

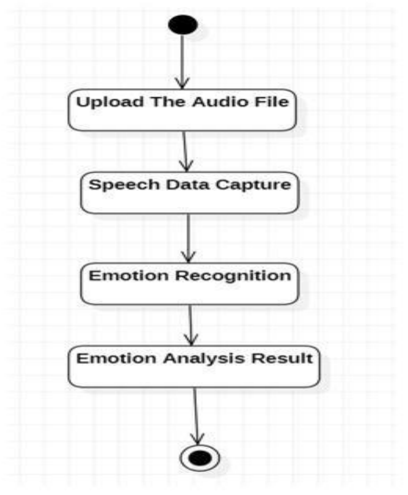
SEQUENCE DIAGRAM



5.1.1.5 ACTIVITY DIAGRAM:

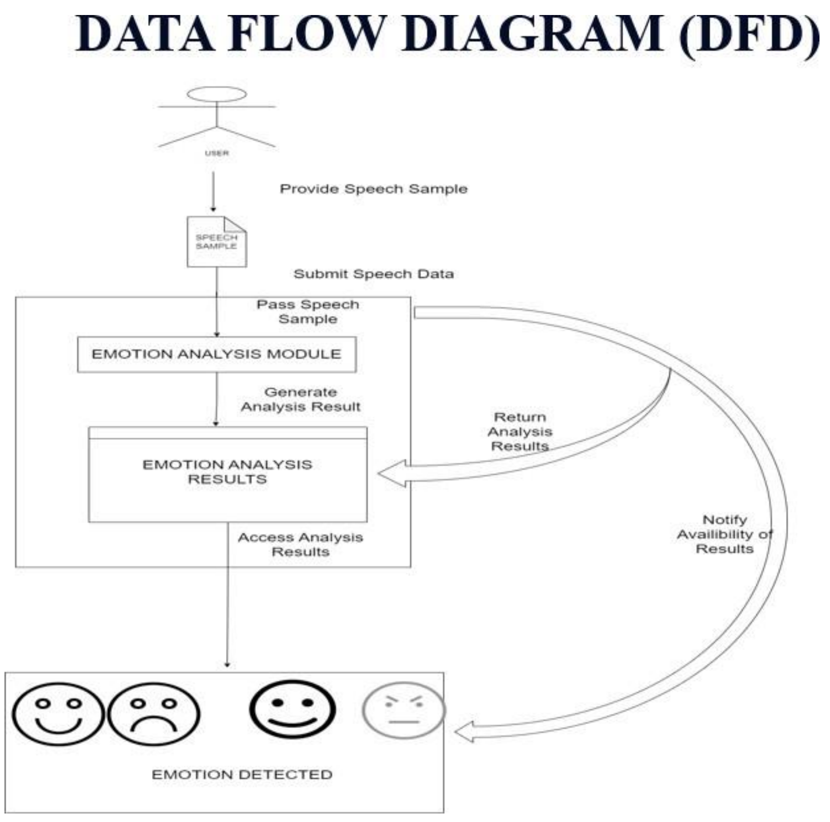
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

ACTIVITY DIAGRAM



5.1.1.6 DFD DIAGRAM:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.



5.2 Methodology

5.2.1 Feature Extraction

This work analyses sentiment from multimodal data using effective deep learning approaches. Preprocessing methodology involves three main components: an encoder network, a decoder network, and the autoencoder architecture. These terms suggest that an autoencoder might be used in the preprocessing stage. An autoencoder is a type of neural network that is trained to encode the input into a lower-dimensional representation and then decode that representation back to the original input. It can be used for feature reduction or denoising. The encoder network learns to map high-dimensional input speech data to a lower-dimensional latent space, capturing essential information while reducing noise and irrelevant features. The decoder network reconstructs the original input from the encoded representation, guiding the autoencoder to learn meaningful features that preserve the underlying structure of the data. During training, the autoencoder minimizes the reconstruction error between the input and reconstructed signals, facilitating the learning of discriminative features for emotion recognition tasks.

5.2.2 CNN Model

The proposed methodology consists of multiple layers of convolution, pooling layer and Feature extraction operations within a CNN architecture. The convolutional layers serve to extract relevant features from the input speech data, capturing both local and global patterns indicative of different emotional states. Subsequent pooling layers help to reduce the dimensionality of the feature maps while preserving important information, facilitating efficient feature extraction.

5.2.3 Model Training And Testing

The extracted features are then fed into fully connected layers for classification, where the model learns to associate the extracted features with specific emotional labels. The extracted features are used to train the CNN model. The model learns to associate the features with different emotions. After training, the model is tested with new data to evaluate its performance. The trained model is now capable of predicting emotions from new audio data by analyzing the features extracted by the CNN. Finally, the results of the emotion prediction are visualized in some form, which could be a graph, chart, or any other visual representation to interpret the outcomes of the model's predictions.

5.3 Modules

1) Upload:

Upload the dataset of audio (.wav files) to be read using librosa library.

2) View:

Uploaded dataset can be viewed.

3) Pre-processing:

Data Pre-processing is a technique that is used to convert the raw data into a clean data set. Cleaning the data refers to removing the null values, filling the null values with meaningful value, removing duplicate values, removing outliers, removing unwanted attributes. If dataset contains any categorical records means convert those categorical variables to numerical values.

4) Identifying Features:

The extracted features are Mel-frequency cepstral coefficients (MFCC), Chromogram.

5) Train and Test Split:

We split our dataset of audio files, training data with audio files and testing data with audio files.

6) Building the model:

To understand the audio and predict emotions, we are proposing a Deep learning-based method Deep learning can provide increased accuracy and decrease in computational power. We will use Deep Neural Networks (DNN) to create the model. Deep Neural Network (DNN) is widely used in deep learning to train models for tasks which traditional machine learning algorithms cannot do or is hard to do the model is created using layers of neural networks.

7) Prediction:

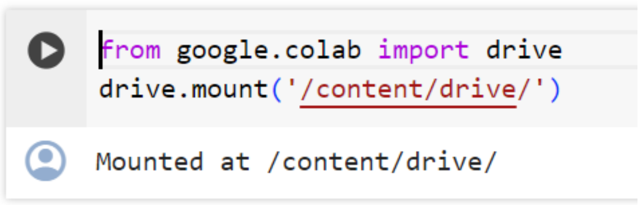
An audio is uploaded by the user (which includes speech of a person), and the model is used to predict the emotion of the speaker in the audio.

STEPS FOR EXECUTING THE PROJECTS

1. Import all the Libraries/packages.
2. Load the RAVDESS speech dataset.
3. Extract the features from the audio files.
4. Build the CNN layer .
5. The labels to the features indicating the emotions must be separated and label encoder.
6. Split the dataset in train and test dataset with test size.
7. A Sequential() model with layers is created.
8. Train dataset is used for training the dataset using epochs.
9. The best one with respect to test accuracy is the model which we will use for prediction.
10. The model is loaded.

6 IMPLEMENTATION

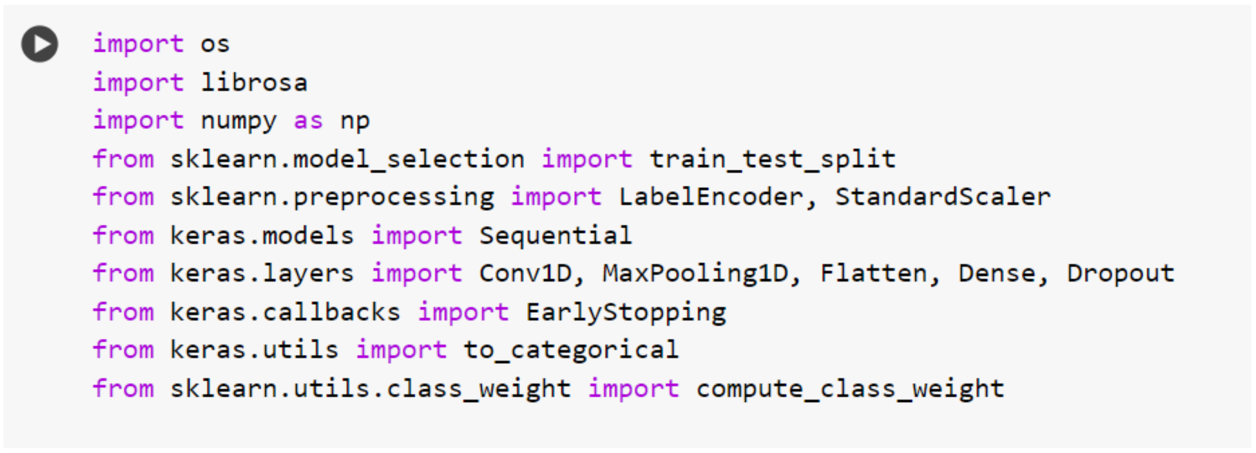
Connect to the Google Drive in Google Colab.



```
from google.colab import drive
drive.mount('/content/drive/')
```

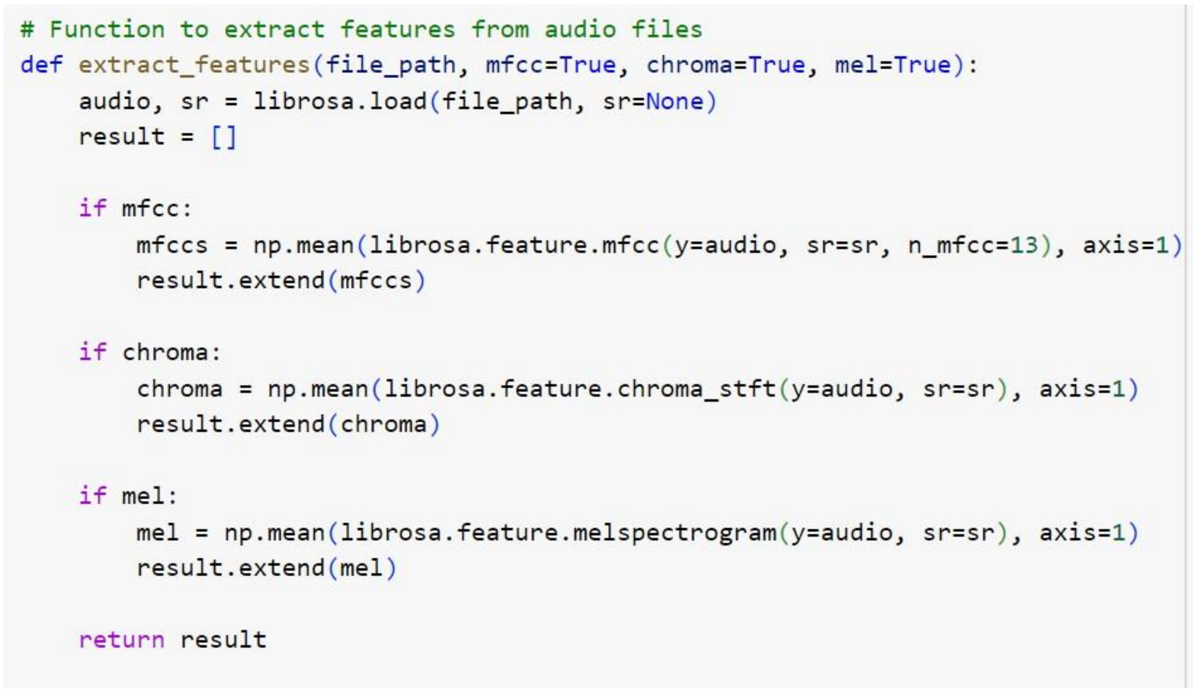
```
Mounted at /content/drive/
```

Import the Libraries.



```
import os
import librosa
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from keras.models import Sequential
from keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout
from keras.callbacks import EarlyStopping
from keras.utils import to_categorical
from sklearn.utils.class_weight import compute_class_weight
```

Feature extraction with MFCC, Chroma.



```
# Function to extract features from audio files
def extract_features(file_path, mfcc=True, chroma=True, mel=True):
    audio, sr = librosa.load(file_path, sr=None)
    result = []

    if mfcc:
        mfccs = np.mean(librosa.feature.mfcc(y=audio, sr=sr, n_mfcc=13), axis=1)
        result.extend(mfccs)

    if chroma:
        chroma = np.mean(librosa.feature.chroma_stft(y=audio, sr=sr), axis=1)
        result.extend(chroma)

    if mel:
        mel = np.mean(librosa.feature.melspectrogram(y=audio, sr=sr), axis=1)
        result.extend(mel)

    return result
```

Function is designed to load audio data and extract features from .wav files

```python
# Load audio data and extract features
def load_data(folder_path, test_size=0.2):
    X, y = [], []

    for filename in os.listdir(folder_path):
        file_path = os.path.join(folder_path, filename)
        if os.path.isfile(file_path) and filename.endswith('.wav'):
            emotion = filename.split('-')[2]
            feature = extract_features(file_path)
            X.append(feature)
            y.append(emotion)

    X = np.array(X)
    y = np.array(y)

    label_encoder = LabelEncoder()
    y = label_encoder.fit_transform(y)
    y = to_categorical(y)

    return train_test_split(X, y, test_size=test_size, random_state=42)
```

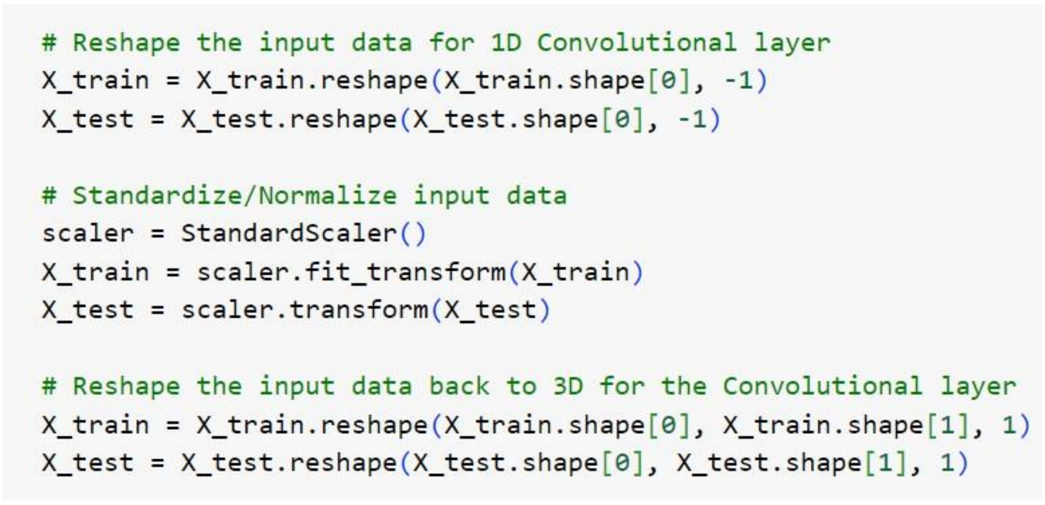
Function creates a Convolutional Neural Network (CNN) model for audio processing.

```python
# Build CNN model
def build_model(input_shape, num_classes):
    model = Sequential()
    model.add(Conv1D(128, 5, padding='same', input_shape=(input_shape, 1), activation='relu'))
    model.add(MaxPooling1D(3))
    model.add(Dropout(0.25))

    model.add(Conv1D(256, 5, padding='same', activation='relu'))
    model.add(MaxPooling1D(3))
    model.add(Dropout(0.25))

    model.add(Flatten())
    model.add(Dense(512, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(num_classes, activation='softmax'))

    return model
```

Uploading Dataset.

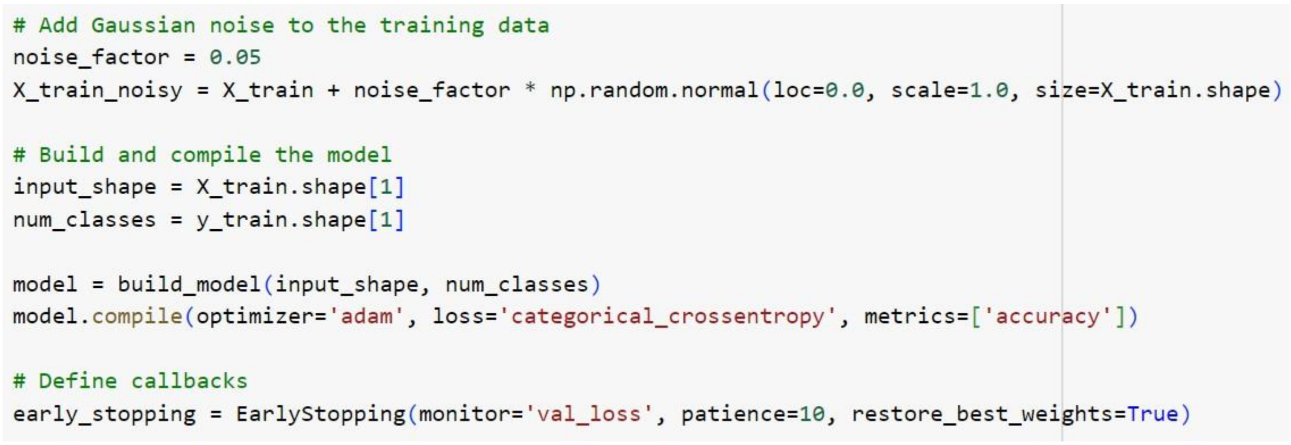
```python
# Load data and split into training and testing sets
folder_path = '/content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01'
X_train, X_test, y_train, y_test = load_data(folder_path)
```

Reshaping the given data.



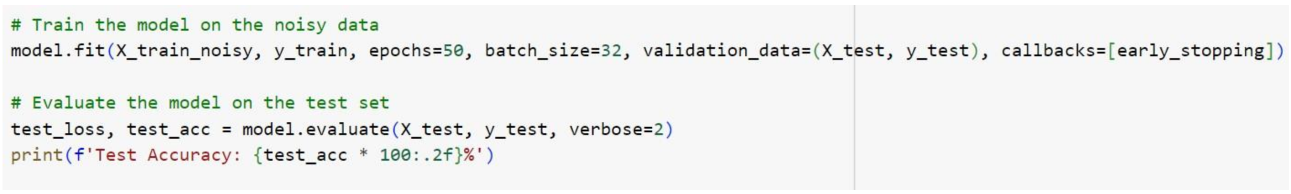
```
# Reshape the input data for 1D Convolutional layer
X_train = X_train.reshape(X_train.shape[0], -1)
X_test = X_test.reshape(X_test.shape[0], -1)

# Standardize/Normalize input data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Reshape the input data back to 3D for the Convolutional layer
X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)
```

Adding Gaussian noise Method



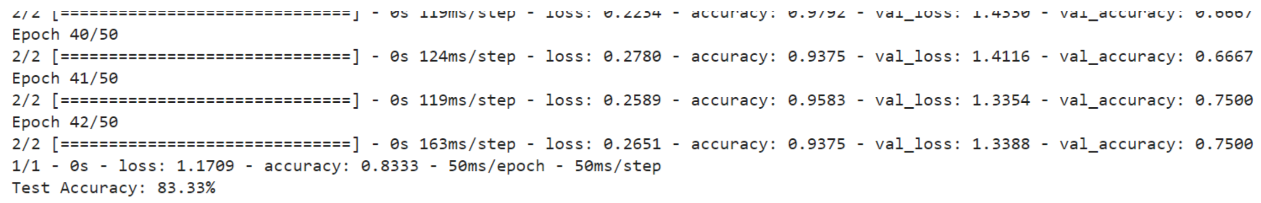
```
# Add Gaussian noise to the training data
noise_factor = 0.05
X_train_noisy = X_train + noise_factor * np.random.normal(loc=0.0, scale=1.0, size=X_train.shape)

# Build and compile the model
input_shape = X_train.shape[1]
num_classes = y_train.shape[1]

model = build_model(input_shape, num_classes)
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Define callbacks
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)
```

Training and Evaluating the model.



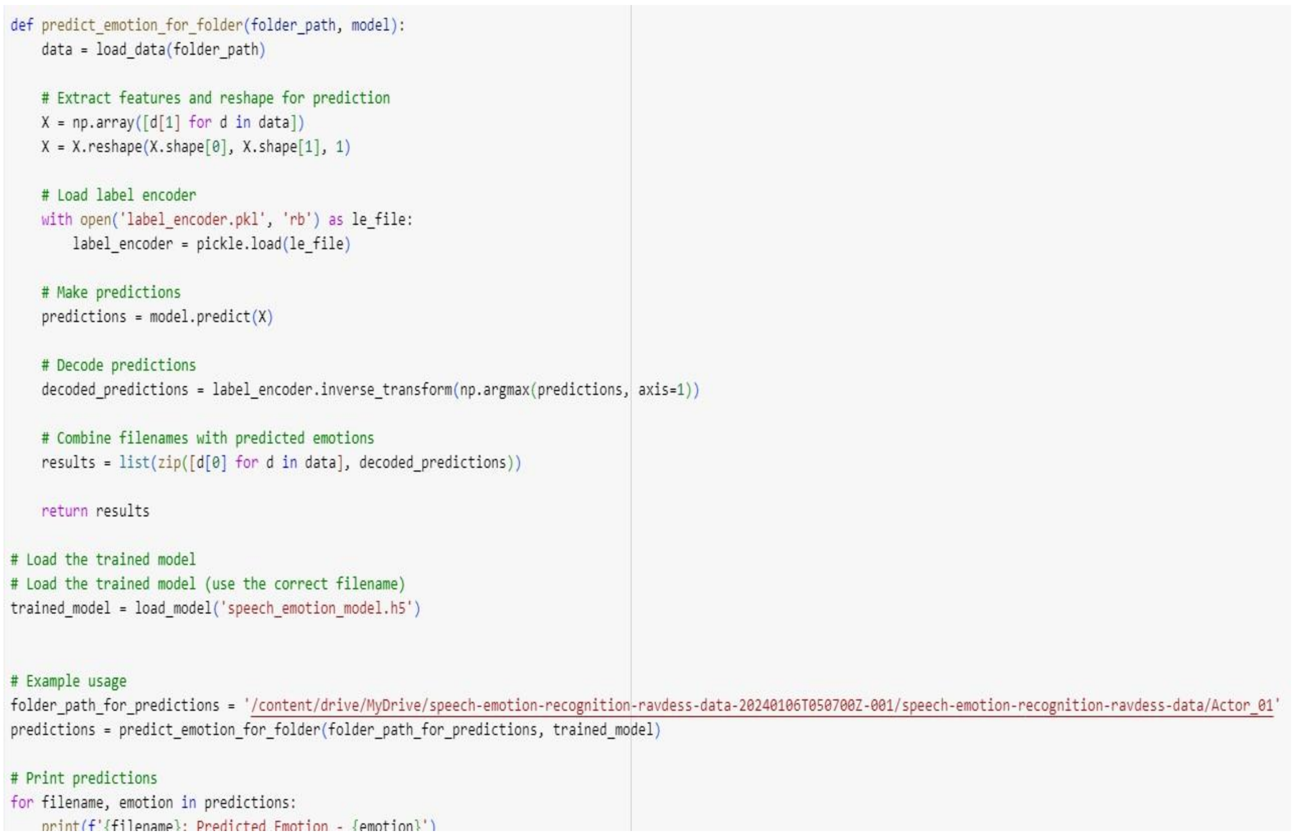
```
# Train the model on the noisy data
model.fit(X_train_noisy, y_train, epochs=50, batch_size=32, validation_data=(X_test, y_test), callbacks=[early_stopping])

# Evaluate the model on the test set
test_loss, test_acc = model.evaluate(X_test, y_test, verbose=2)
print(f'Test Accuracy: {test_acc * 100:.2f}%')
```

Output of the model:

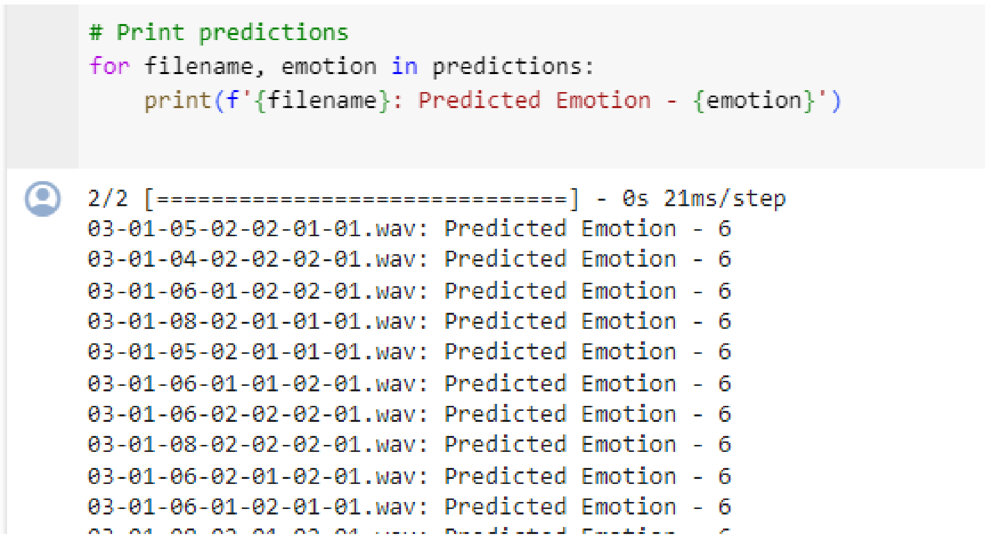


```
2/2 [==============================] - 0s 119ms/step - loss: 0.2254 - accuracy: 0.9792 - val_loss: 1.4530 - val_accuracy: 0.6667
Epoch 40/50
2/2 [==============================] - 0s 124ms/step - loss: 0.2780 - accuracy: 0.9375 - val_loss: 1.4116 - val_accuracy: 0.6667
Epoch 41/50
2/2 [==============================] - 0s 119ms/step - loss: 0.2589 - accuracy: 0.9583 - val_loss: 1.3354 - val_accuracy: 0.7500
Epoch 42/50
2/2 [==============================] - 0s 163ms/step - loss: 0.2651 - accuracy: 0.9375 - val_loss: 1.3388 - val_accuracy: 0.7500
1/1 - 0s - loss: 1.1709 - accuracy: 0.8333 - 50ms/epoch - 50ms/step
Test Accuracy: 83.33%
```

To predict the Emotion of an Actor.



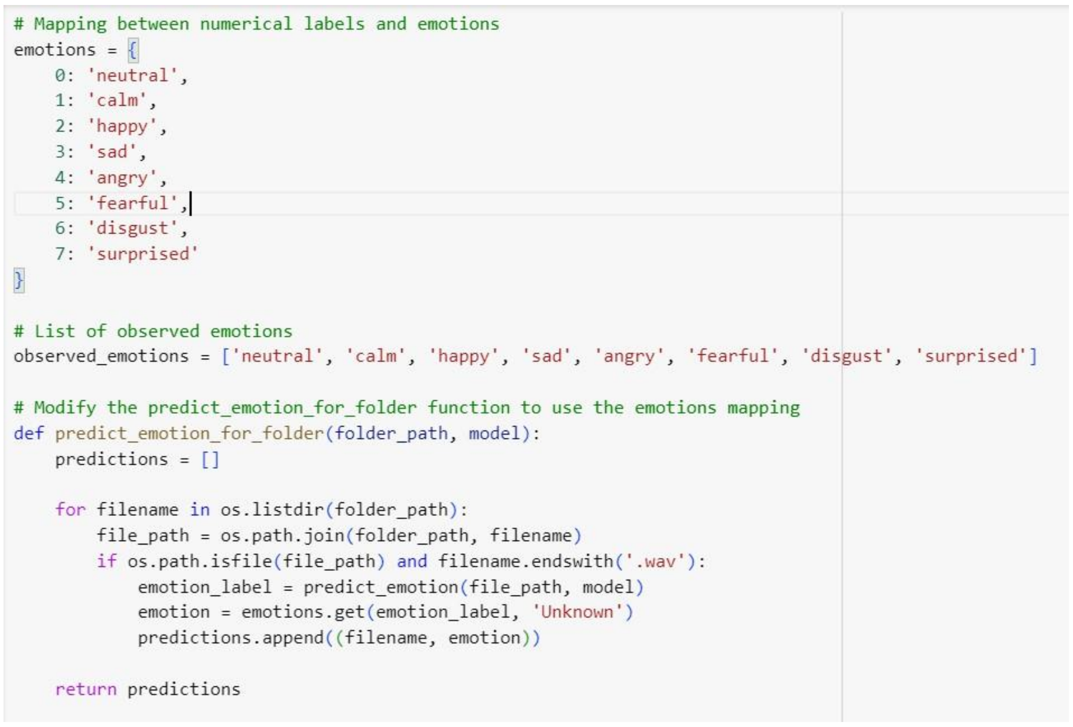
```python
def predict_emotion_for_folder(folder_path, model):
    data = load_data(folder_path)

    # Extract features and reshape for prediction
    X = np.array([d[1] for d in data])
    X = X.reshape(X.shape[0], X.shape[1], 1)

    # Load label encoder
    with open('label_encoder.pkl', 'rb') as le_file:
        label_encoder = pickle.load(le_file)

    # Make predictions
    predictions = model.predict(X)

    # Decode predictions
    decoded_predictions = label_encoder.inverse_transform(np.argmax(predictions, axis=1))

    # Combine filenames with predicted emotions
    results = list(zip([d[0] for d in data], decoded_predictions))

    return results

# Load the trained model
# Load the trained model (use the correct filename)
trained_model = load_model('speech_emotion_model.h5')


# Example usage
folder_path_for_predictions = '/content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01'
predictions = predict_emotion_for_folder(folder_path_for_predictions, trained_model)

# Print predictions
for filename, emotion in predictions:
    print(f'{filename}: Predicted Emotion - {emotion}')
```



```python
# Print predictions
for filename, emotion in predictions:
    print(f'{filename}: Predicted Emotion - {emotion}')
```

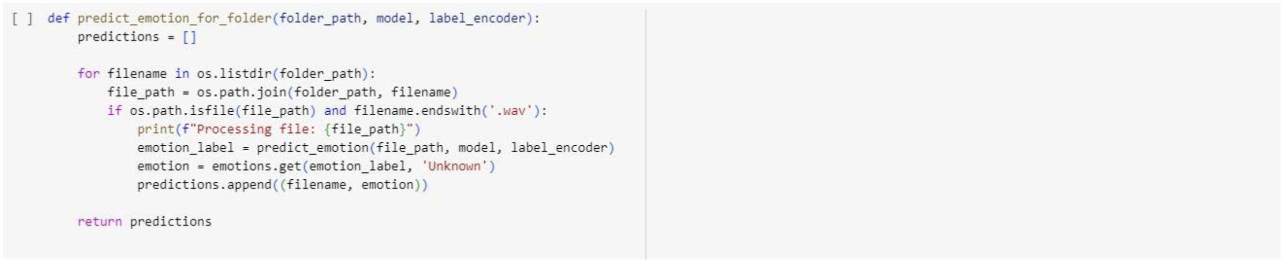
```
2/2 [==============================] - 0s 21ms/step
03-01-05-02-02-01-01.wav: Predicted Emotion - 6
03-01-04-02-02-02-01.wav: Predicted Emotion - 6
03-01-06-01-02-02-01.wav: Predicted Emotion - 6
03-01-08-02-01-01-01.wav: Predicted Emotion - 6
03-01-05-02-01-01-01.wav: Predicted Emotion - 6
03-01-06-01-01-02-01.wav: Predicted Emotion - 6
03-01-06-02-02-02-01.wav: Predicted Emotion - 6
03-01-08-02-02-02-01.wav: Predicted Emotion - 6
03-01-06-02-01-02-01.wav: Predicted Emotion - 6
03-01-06-01-02-01-01.wav: Predicted Emotion - 6
[illegible]
```

Labelling the Emotions.

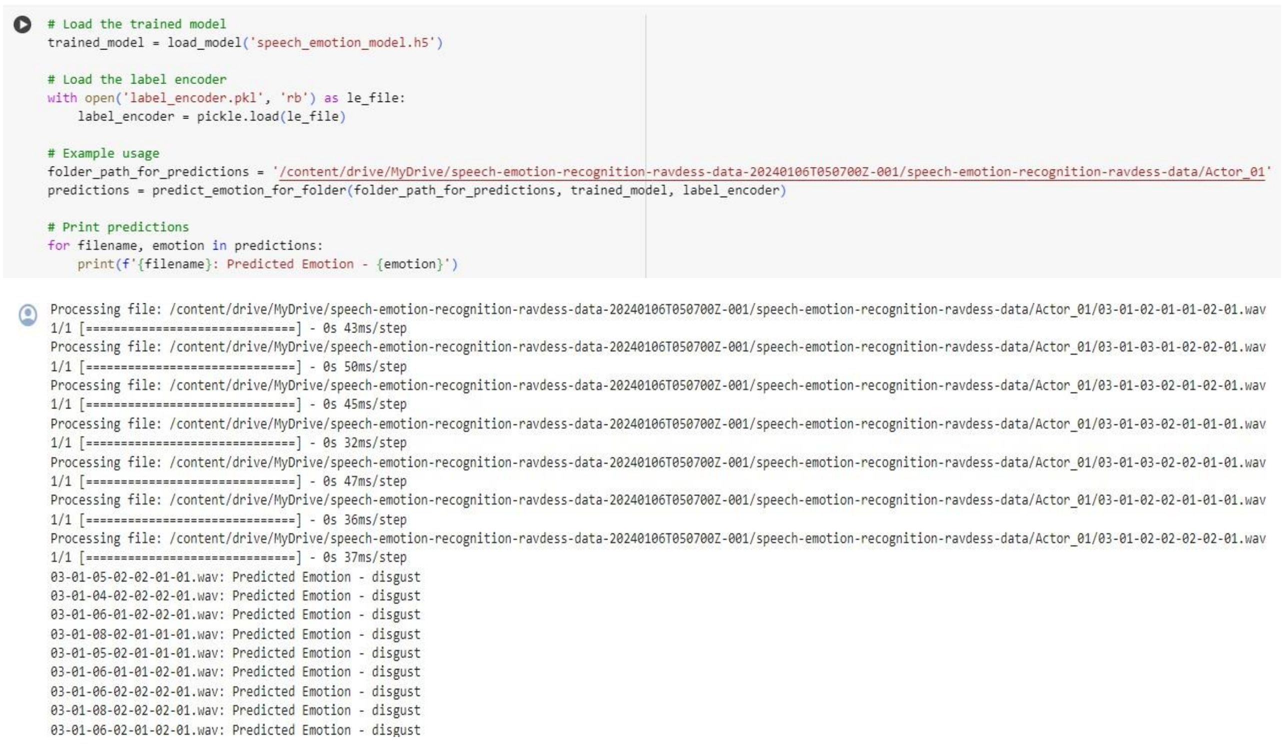


```python
# Mapping between numerical labels and emotions
emotions = {
    0: 'neutral',
    1: 'calm',
    2: 'happy',
    3: 'sad',
    4: 'angry',
    5: 'fearful',
    6: 'disgust',
    7: 'surprised'
}

# List of observed emotions
observed_emotions = ['neutral', 'calm', 'happy', 'sad', 'angry', 'fearful', 'disgust', 'surprised']

# Modify the predict_emotion_for_folder function to use the emotions mapping
def predict_emotion_for_folder(folder_path, model):
    predictions = []

    for filename in os.listdir(folder_path):
        file_path = os.path.join(folder_path, filename)
        if os.path.isfile(file_path) and filename.endswith('.wav'):
            emotion_label = predict_emotion(file_path, model)
            emotion = emotions.get(emotion_label, 'Unknown')
            predictions.append((filename, emotion))

    return predictions
```

After labelling the emotions, need to predict the stage of emotion.



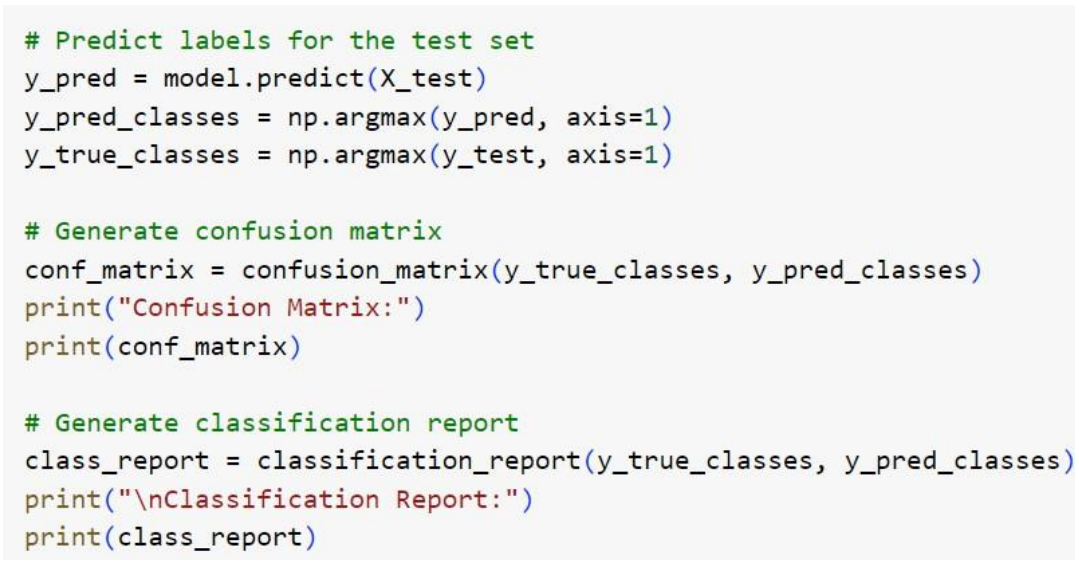
```python
[ ]  def predict_emotion_for_folder(folder_path, model, label_encoder):
         predictions = []

         for filename in os.listdir(folder_path):
             file_path = os.path.join(folder_path, filename)
             if os.path.isfile(file_path) and filename.endswith('.wav'):
                 print(f"Processing file: {file_path}")
                 emotion_label = predict_emotion(file_path, model, label_encoder)
                 emotion = emotions.get(emotion_label, 'Unknown')
                 predictions.append((filename, emotion))

         return predictions
```



```python
# Load the trained model
trained_model = load_model('speech_emotion_model.h5')

# Load the label encoder
with open('label_encoder.pkl', 'rb') as le_file:
    label_encoder = pickle.load(le_file)

# Example usage
folder_path_for_predictions = '/content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01'
predictions = predict_emotion_for_folder(folder_path_for_predictions, trained_model, label_encoder)

# Print predictions
for filename, emotion in predictions:
    print(f'{filename}: Predicted Emotion - {emotion}')
```

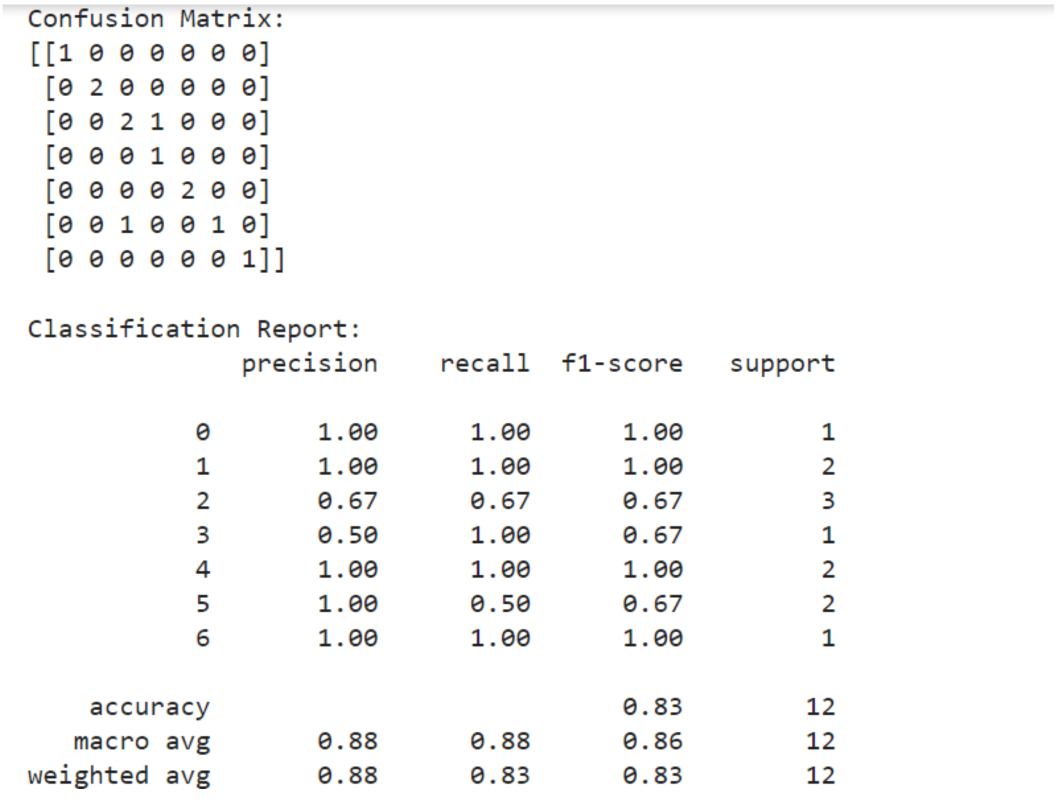
```
Processing file: /content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01/03-01-02-01-01-02-01.wav
1/1 [==============================] - 0s 43ms/step
Processing file: /content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01/03-01-03-01-02-02-01.wav
1/1 [==============================] - 0s 50ms/step
Processing file: /content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01/03-01-03-02-01-02-01.wav
1/1 [==============================] - 0s 45ms/step
Processing file: /content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01/03-01-03-02-01-01-01.wav
1/1 [==============================] - 0s 32ms/step
Processing file: /content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01/03-01-03-02-02-01-01.wav
1/1 [==============================] - 0s 47ms/step
Processing file: /content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01/03-01-02-02-01-01-01.wav
1/1 [==============================] - 0s 36ms/step
Processing file: /content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01/03-01-02-02-02-02-01.wav
1/1 [==============================] - 0s 37ms/step
03-01-05-02-02-01-01.wav: Predicted Emotion - disgust
03-01-04-02-02-02-01.wav: Predicted Emotion - disgust
03-01-06-01-02-02-01.wav: Predicted Emotion - disgust
03-01-08-02-01-01-01.wav: Predicted Emotion - disgust
03-01-05-02-01-01-01.wav: Predicted Emotion - disgust
03-01-06-01-01-02-01.wav: Predicted Emotion - disgust
03-01-06-02-02-02-01.wav: Predicted Emotion - disgust
03-01-08-02-02-02-01.wav: Predicted Emotion - disgust
03-01-06-02-01-02-01.wav: Predicted Emotion - disgust
```

Generating Confusion Matrix and Classification Report.



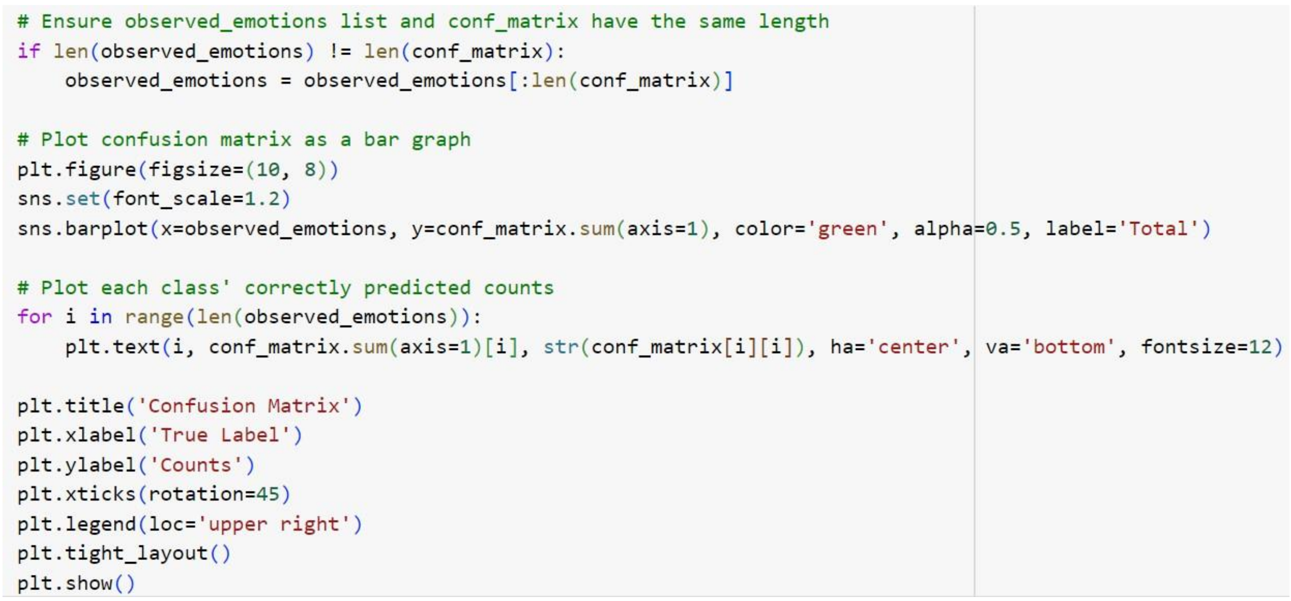
```
# Predict labels for the test set
y_pred = model.predict(X_test)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true_classes = np.argmax(y_test, axis=1)

# Generate confusion matrix
conf_matrix = confusion_matrix(y_true_classes, y_pred_classes)
print("Confusion Matrix:")
print(conf_matrix)

# Generate classification report
class_report = classification_report(y_true_classes, y_pred_classes)
print("\nClassification Report:")
print(class_report)
```

Output of Confusion Matrix and Classification Report:



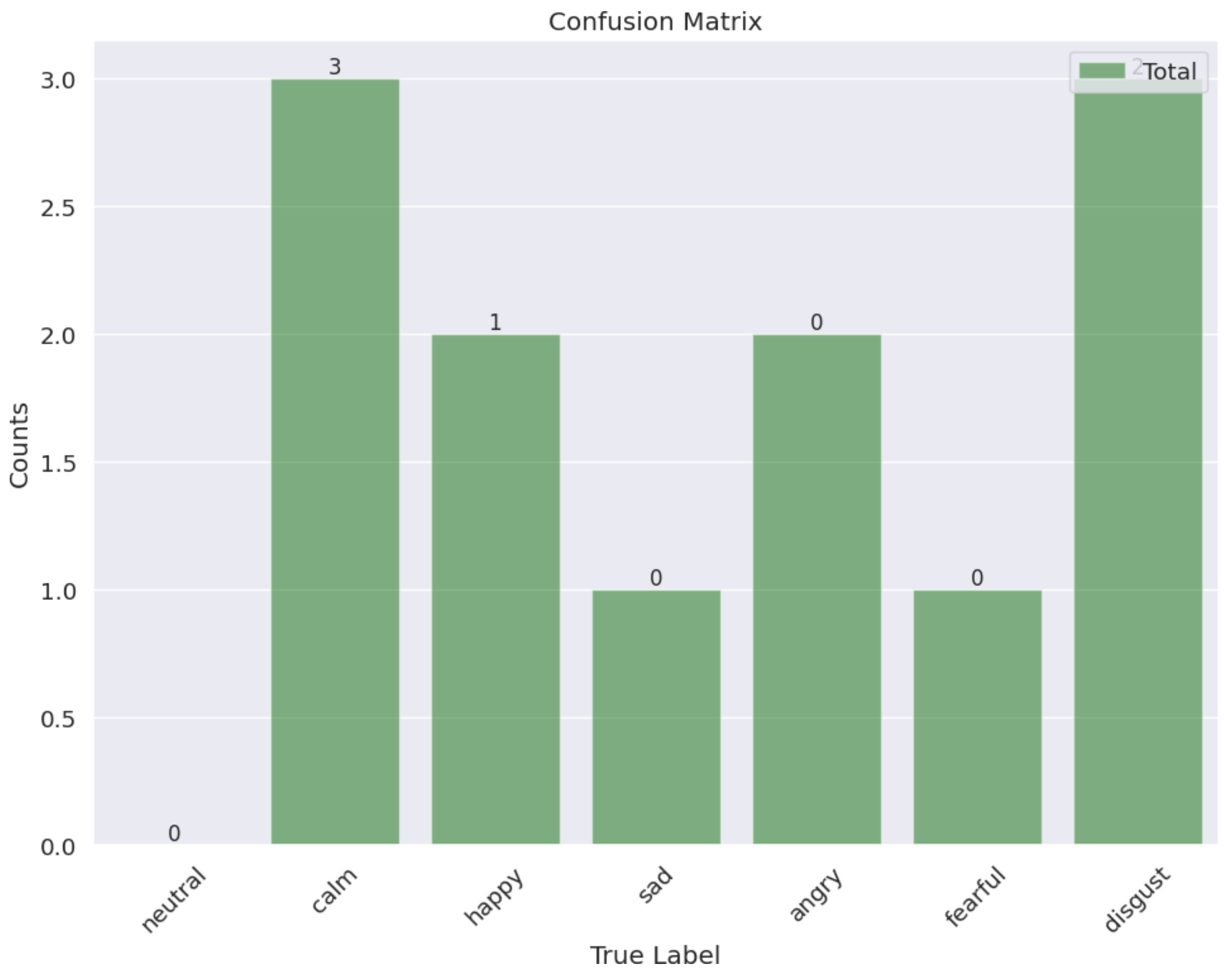
```
Confusion Matrix:
[[1 0 0 0 0 0 0]
 [0 2 0 0 0 0 0]
 [0 0 2 1 0 0 0]
 [0 0 0 1 0 0 0]
 [0 0 0 0 2 0 0]
 [0 0 1 0 0 1 0]
 [0 0 0 0 0 0 1]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         2
           2       0.67      0.67      0.67         3
           3       0.50      1.00      0.67         1
           4       1.00      1.00      1.00         2
           5       1.00      0.50      0.67         2
           6       1.00      1.00      1.00         1

    accuracy                           0.83        12
   macro avg       0.88      0.88      0.86        12
weighted avg       0.88      0.83      0.83        12
```

Plotting a BarGraph.

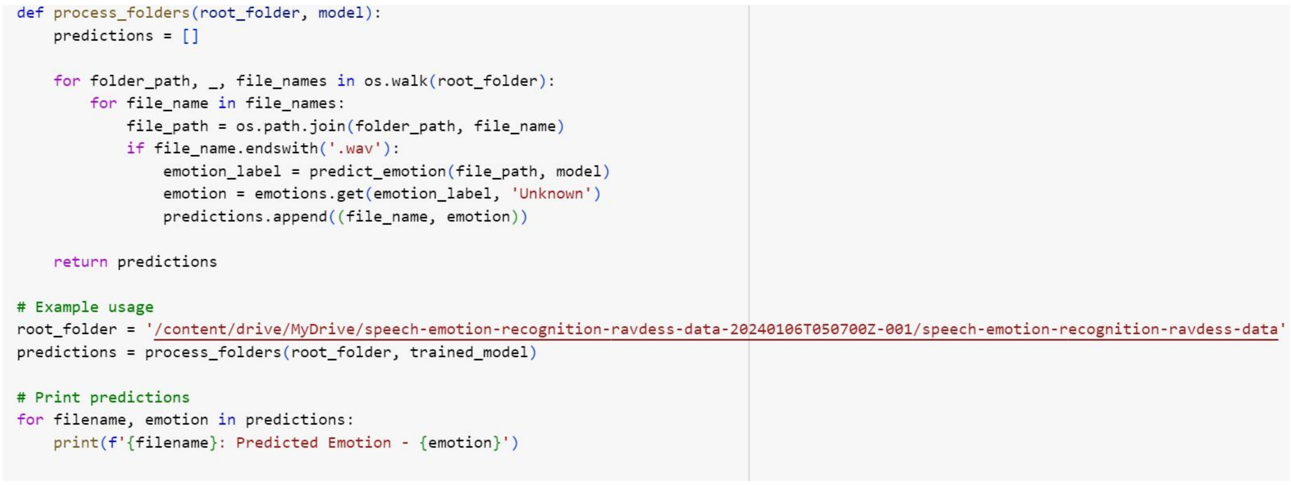


```
# Ensure observed_emotions list and conf_matrix have the same length
if len(observed_emotions) != len(conf_matrix):
    observed_emotions = observed_emotions[:len(conf_matrix)]

# Plot confusion matrix as a bar graph
plt.figure(figsize=(10, 8))
sns.set(font_scale=1.2)
sns.barplot(x=observed_emotions, y=conf_matrix.sum(axis=1), color='green', alpha=0.5, label='Total')

# Plot each class' correctly predicted counts
for i in range(len(observed_emotions)):
    plt.text(i, conf_matrix.sum(axis=1)[i], str(conf_matrix[i][i]), ha='center', va='bottom', fontsize=12)

plt.title('Confusion Matrix')
plt.xlabel('True Label')
plt.ylabel('Counts')
plt.xticks(rotation=45)
plt.legend(loc='upper right')
plt.tight_layout()
plt.show()
```

Output of Bar Graph:

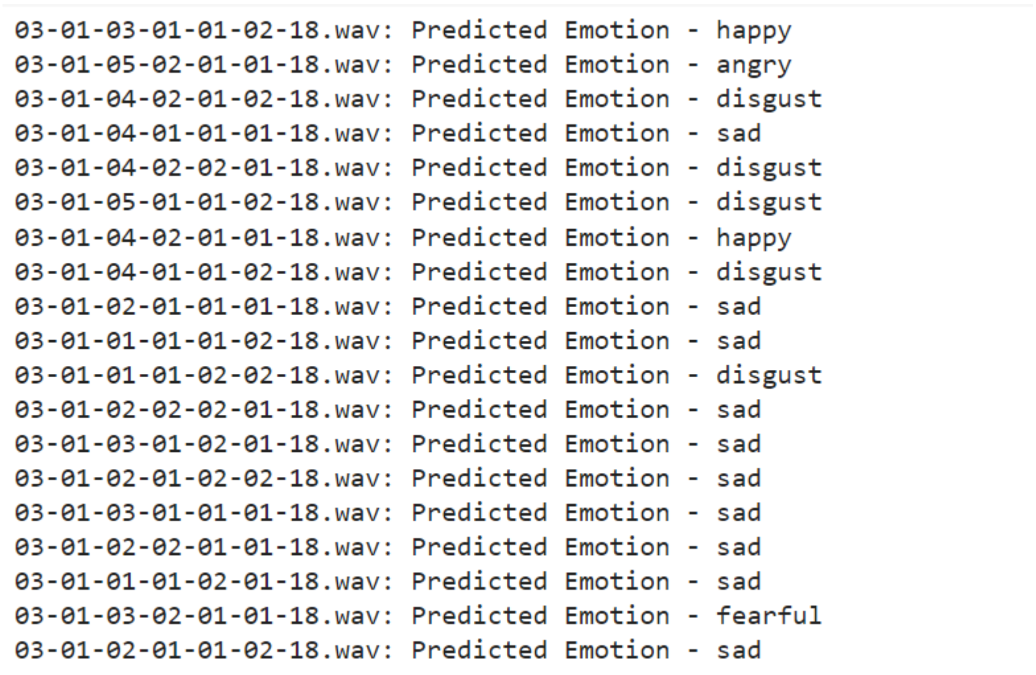


Predicting the state of Emotion from the Dataset.



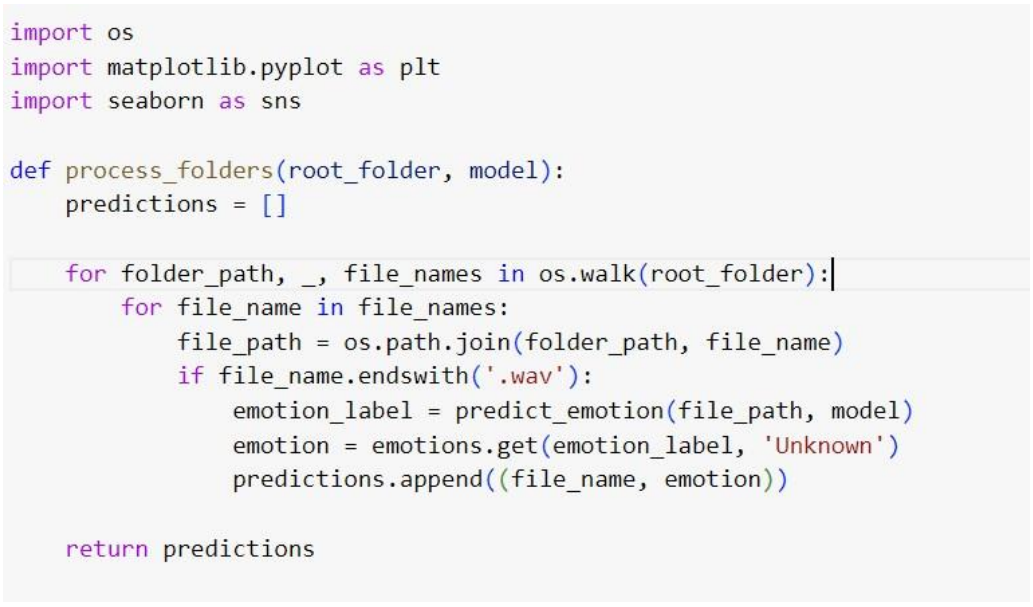
```python
def process_folders(root_folder, model):
    predictions = []

    for folder_path, _, file_names in os.walk(root_folder):
        for file_name in file_names:
            file_path = os.path.join(folder_path, file_name)
            if file_name.endswith('.wav'):
                emotion_label = predict_emotion(file_path, model)
                emotion = emotions.get(emotion_label, 'Unknown')
                predictions.append((file_name, emotion))

    return predictions

# Example usage
root_folder = '/content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data'
predictions = process_folders(root_folder, trained_model)

# Print predictions
for filename, emotion in predictions:
    print(f'{filename}: Predicted Emotion - {emotion}')
```

Output of Emotions:



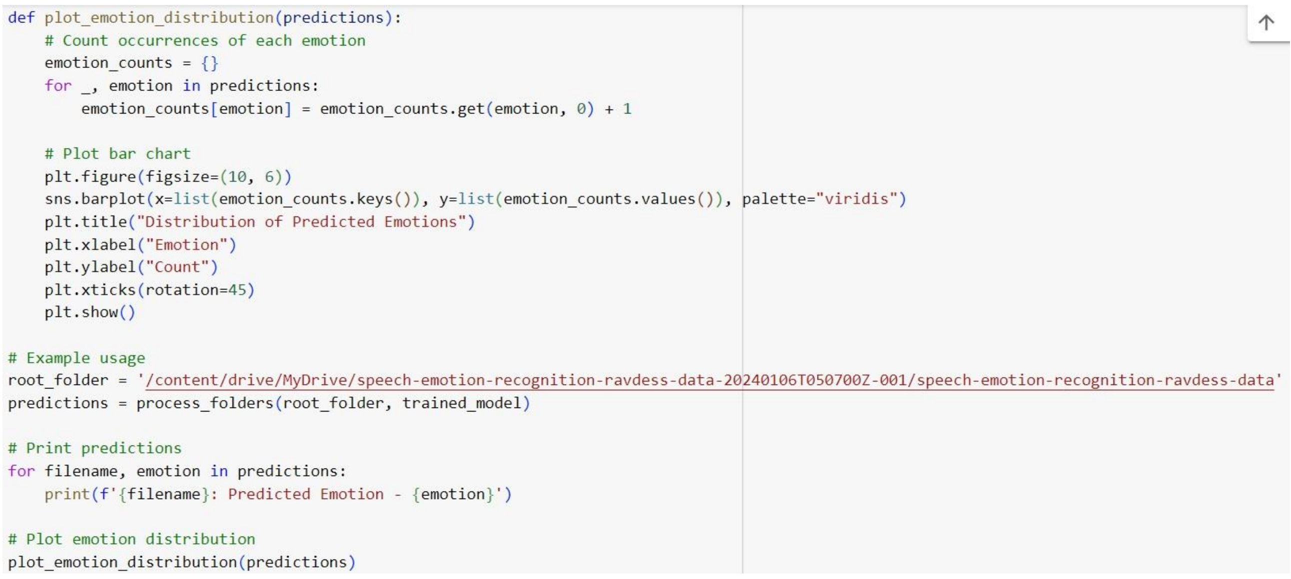
```
03-01-03-01-01-02-18.wav: Predicted Emotion - happy
03-01-05-02-01-01-18.wav: Predicted Emotion - angry
03-01-04-02-01-02-18.wav: Predicted Emotion - disgust
03-01-04-01-01-01-18.wav: Predicted Emotion - sad
03-01-04-02-02-01-18.wav: Predicted Emotion - disgust
03-01-05-01-01-02-18.wav: Predicted Emotion - disgust
03-01-04-02-01-01-18.wav: Predicted Emotion - happy
03-01-04-01-01-02-18.wav: Predicted Emotion - disgust
03-01-02-01-01-01-18.wav: Predicted Emotion - sad
03-01-01-01-01-02-18.wav: Predicted Emotion - sad
03-01-01-01-02-02-18.wav: Predicted Emotion - disgust
03-01-02-02-02-01-18.wav: Predicted Emotion - sad
03-01-03-01-02-01-18.wav: Predicted Emotion - sad
03-01-02-01-02-02-18.wav: Predicted Emotion - sad
03-01-03-01-01-01-18.wav: Predicted Emotion - sad
03-01-02-02-01-01-18.wav: Predicted Emotion - sad
03-01-01-01-02-01-18.wav: Predicted Emotion - sad
03-01-03-02-01-01-18.wav: Predicted Emotion - fearful
03-01-02-01-01-02-18.wav: Predicted Emotion - sad
```

Processing the folder into the root folders.

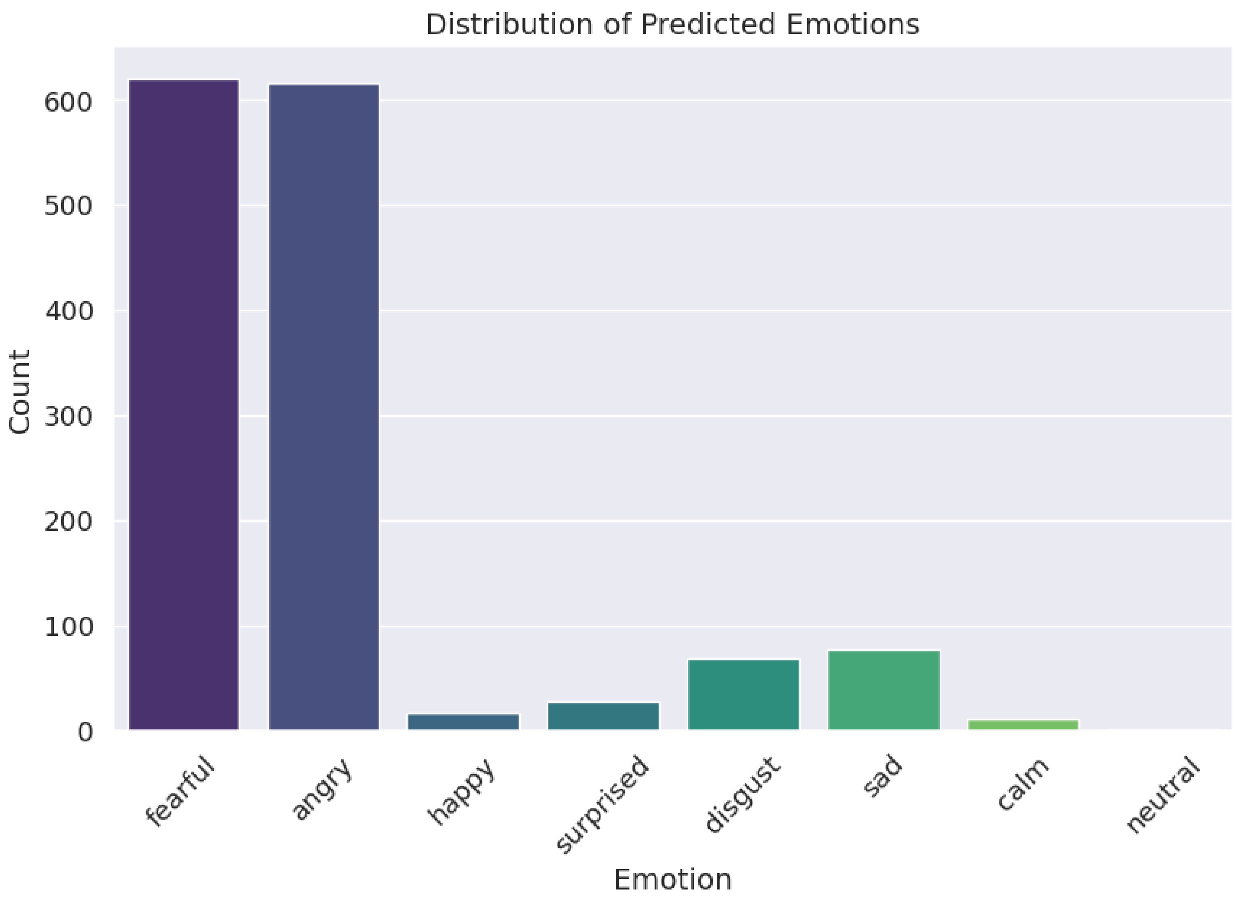


```python
import os
import matplotlib.pyplot as plt
import seaborn as sns

def process_folders(root_folder, model):
    predictions = []

    for folder_path, _, file_names in os.walk(root_folder):
        for file_name in file_names:
            file_path = os.path.join(folder_path, file_name)
            if file_name.endswith('.wav'):
                emotion_label = predict_emotion(file_path, model)
                emotion = emotions.get(emotion_label, 'Unknown')
                predictions.append((file_name, emotion))

    return predictions
```

Plotting the emotions into the Bar Graph.



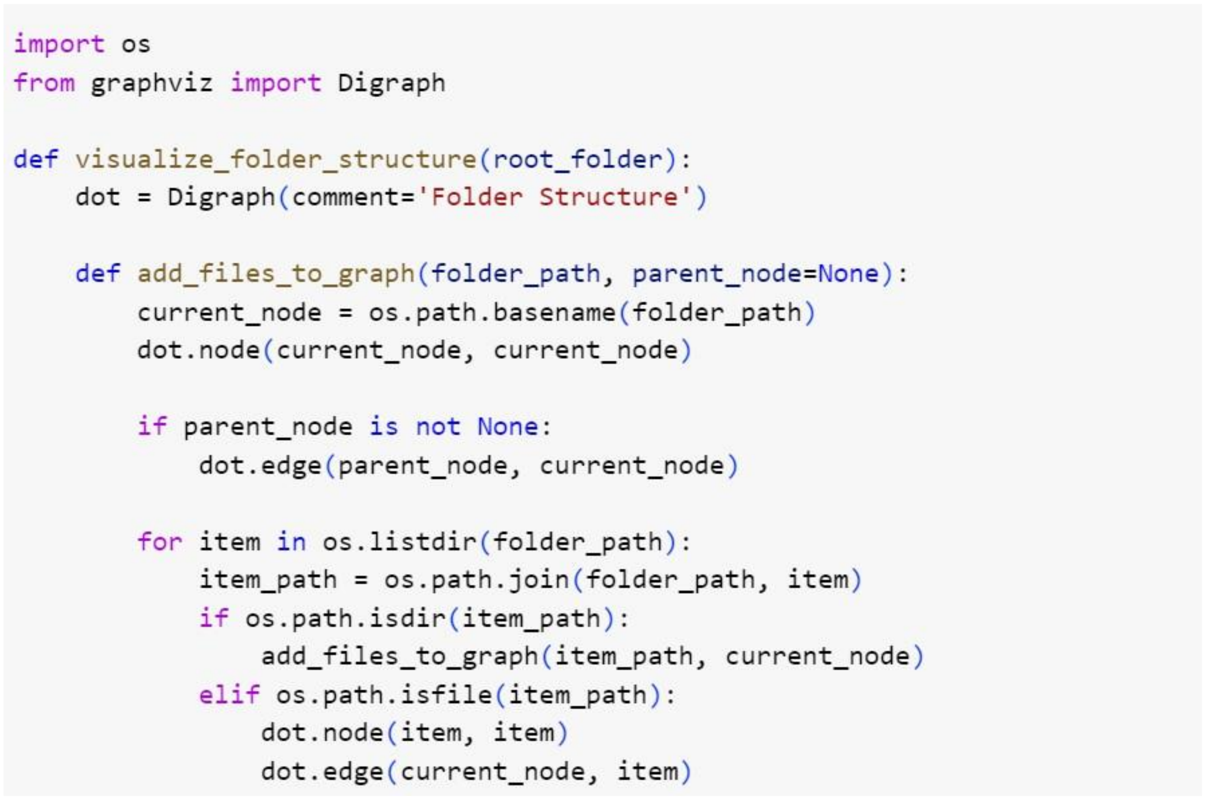
```python
def plot_emotion_distribution(predictions):
    # Count occurrences of each emotion
    emotion_counts = {}
    for _, emotion in predictions:
        emotion_counts[emotion] = emotion_counts.get(emotion, 0) + 1

    # Plot bar chart
    plt.figure(figsize=(10, 6))
    sns.barplot(x=list(emotion_counts.keys()), y=list(emotion_counts.values()), palette="viridis")
    plt.title("Distribution of Predicted Emotions")
    plt.xlabel("Emotion")
    plt.ylabel("Count")
    plt.xticks(rotation=45)
    plt.show()

# Example usage
root_folder = '/content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data'
predictions = process_folders(root_folder, trained_model)

# Print predictions
for filename, emotion in predictions:
    print(f'{filename}: Predicted Emotion - {emotion}')

# Plot emotion distribution
plot_emotion_distribution(predictions)
```



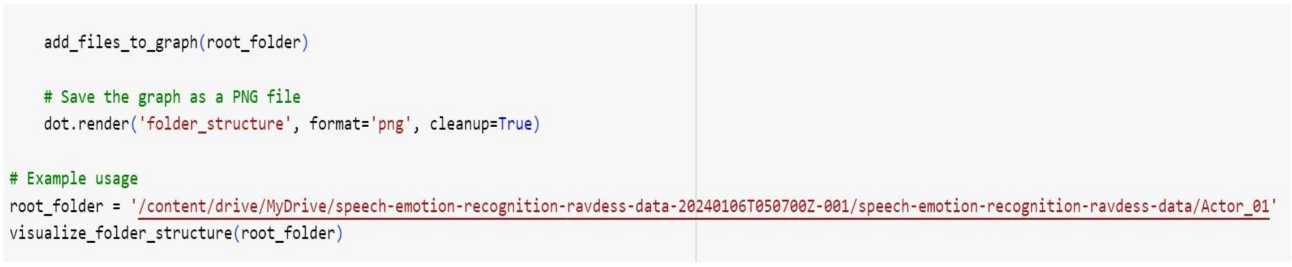
Install graphviz for visualization.

```
[ ]  pip install graphviz

     Requirement already satisfied: graphviz in /usr/local/lib/python3.10/dist-packages (0.20.3)
```

Implementing Root folder into folder path.

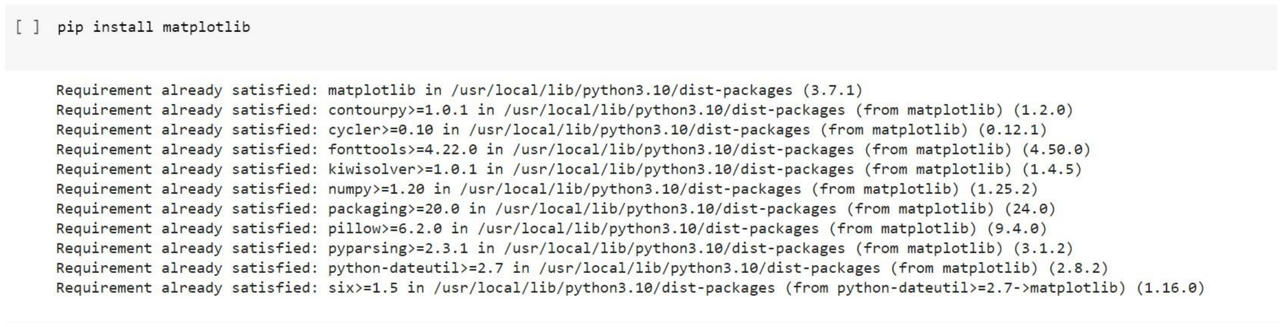


```python
import os
from graphviz import Digraph

def visualize_folder_structure(root_folder):
    dot = Digraph(comment='Folder Structure')

    def add_files_to_graph(folder_path, parent_node=None):
        current_node = os.path.basename(folder_path)
        dot.node(current_node, current_node)

        if parent_node is not None:
            dot.edge(parent_node, current_node)

        for item in os.listdir(folder_path):
            item_path = os.path.join(folder_path, item)
            if os.path.isdir(item_path):
                add_files_to_graph(item_path, current_node)
            elif os.path.isfile(item_path):
                dot.node(item, item)
                dot.edge(current_node, item)
```



```python
    add_files_to_graph(root_folder)

    # Save the graph as a PNG file
    dot.render('folder_structure', format='png', cleanup=True)

# Example usage
root_folder = '/content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data/Actor_01'
visualize_folder_structure(root_folder)
```

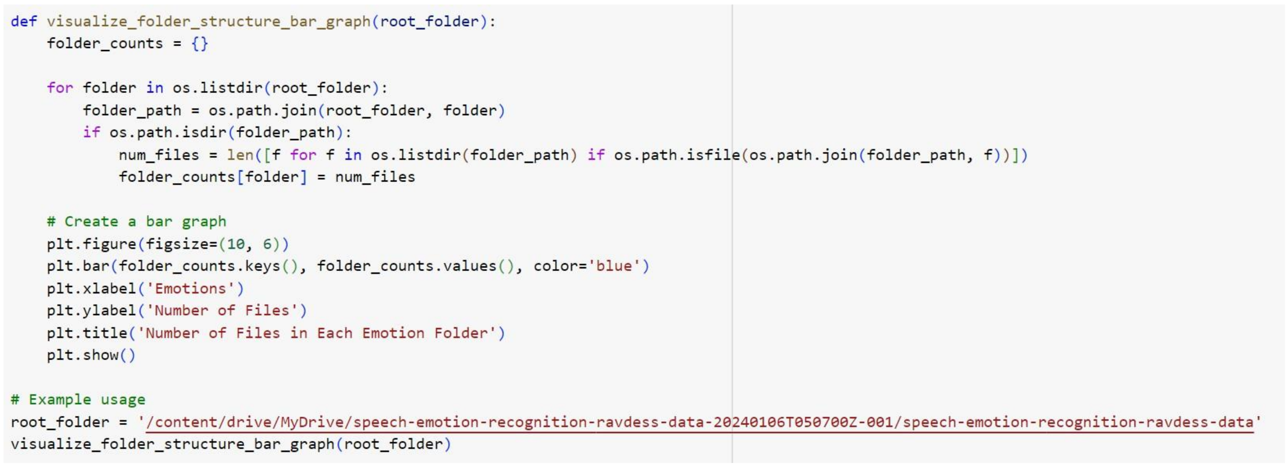
Install Matplotlib



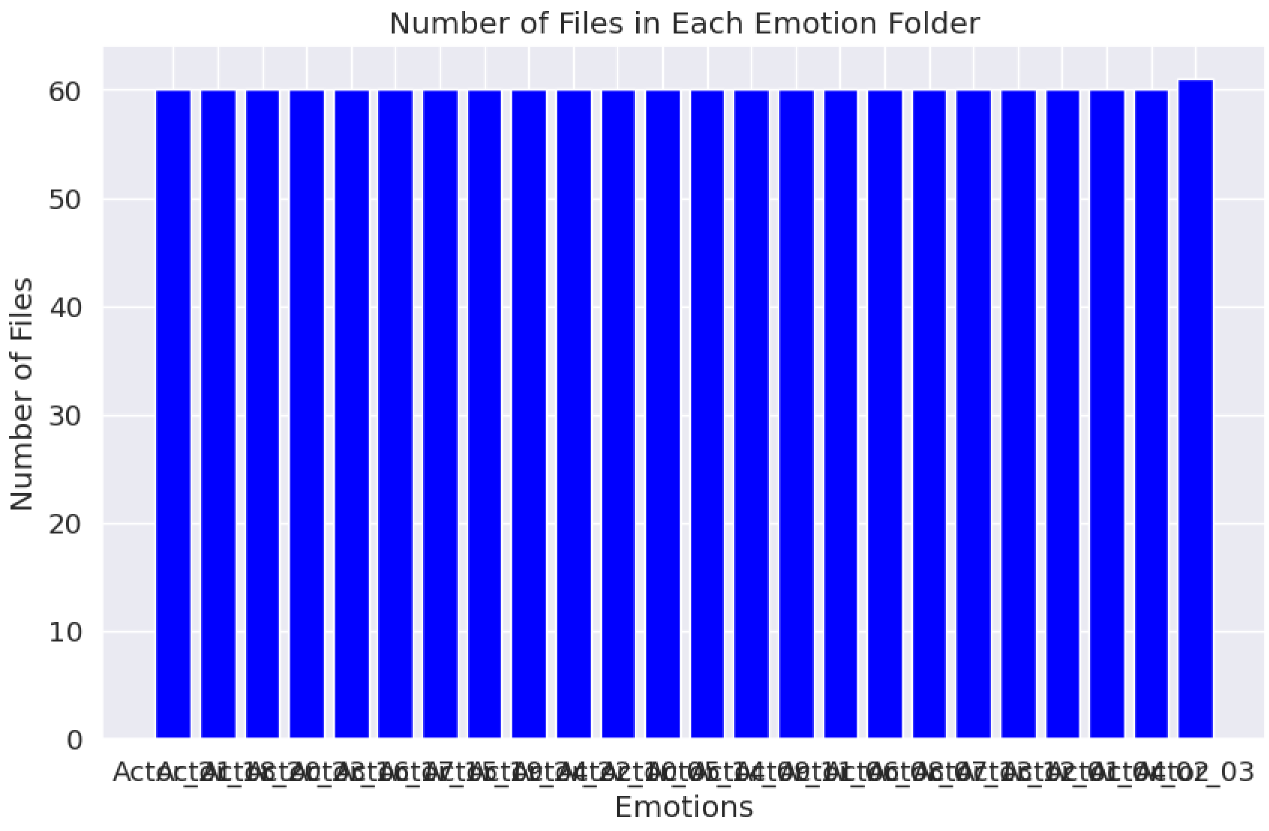
```
[ ]  pip install matplotlib
```

```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.50.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.25.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

Overall Dataset Visualization in Bargraph.



```python
def visualize_folder_structure_bar_graph(root_folder):
    folder_counts = {}

    for folder in os.listdir(root_folder):
        folder_path = os.path.join(root_folder, folder)
        if os.path.isdir(folder_path):
            num_files = len([f for f in os.listdir(folder_path) if os.path.isfile(os.path.join(folder_path, f))])
            folder_counts[folder] = num_files

    # Create a bar graph
    plt.figure(figsize=(10, 6))
    plt.bar(folder_counts.keys(), folder_counts.values(), color='blue')
    plt.xlabel('Emotions')
    plt.ylabel('Number of Files')
    plt.title('Number of Files in Each Emotion Folder')
    plt.show()

# Example usage
root_folder = '/content/drive/MyDrive/speech-emotion-recognition-ravdess-data-20240106T050700Z-001/speech-emotion-recognition-ravdess-data'
visualize_folder_structure_bar_graph(root_folder)
```



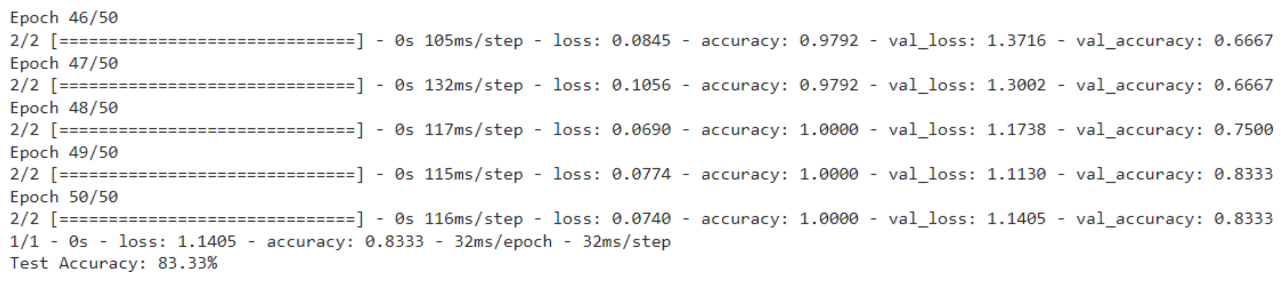
7 TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. We employed a rigorous evaluation approach to assess the performance of the Convolutional Neural Network (CNN) model. We partitioned the dataset into training and testing sets, ensuring that each contained a representative distribution of speech samples across different emotions. After training the model on the training set, we evaluated its performance on the testing set using various metrics such as accuracy, confusion matrix, and classification report. Additionally, we visualized the model's performance using a bar graph representation of the confusion matrix, providing a clear depiction of the model's ability to classify emotions accurately. Through extensive testing and evaluation, we confirmed the effectiveness of the CNN model in accurately recognizing emotions from speech audio data, demonstrating its potential for practical applications in diverse fields such as human-computer interaction and mental health monitoring.

8 EXPERIMENTAL RESULTS

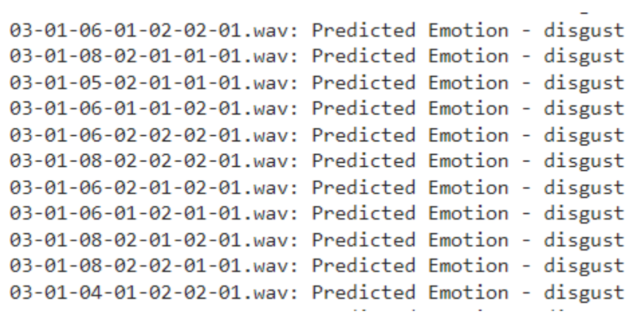
8.1 Output

Accuracy output:



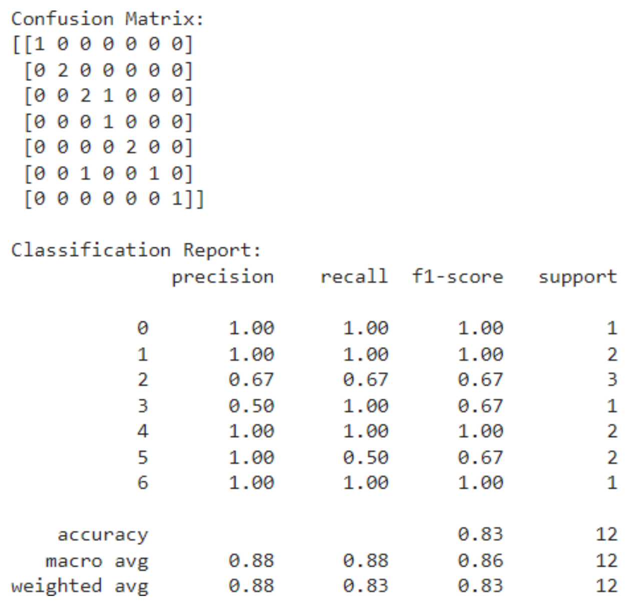
```
Epoch 46/50
2/2 [==============================] - 0s 105ms/step - loss: 0.0845 - accuracy: 0.9792 - val_loss: 1.3716 - val_accuracy: 0.6667
Epoch 47/50
2/2 [==============================] - 0s 132ms/step - loss: 0.1056 - accuracy: 0.9792 - val_loss: 1.3002 - val_accuracy: 0.6667
Epoch 48/50
2/2 [==============================] - 0s 117ms/step - loss: 0.0690 - accuracy: 1.0000 - val_loss: 1.1738 - val_accuracy: 0.7500
Epoch 49/50
2/2 [==============================] - 0s 115ms/step - loss: 0.0774 - accuracy: 1.0000 - val_loss: 1.1130 - val_accuracy: 0.8333
Epoch 50/50
2/2 [==============================] - 0s 116ms/step - loss: 0.0740 - accuracy: 1.0000 - val_loss: 1.1405 - val_accuracy: 0.8333
1/1 - 0s - loss: 1.1405 - accuracy: 0.8333 - 32ms/epoch - 32ms/step
Test Accuracy: 83.33%
```

The output for the single actor in the dataset



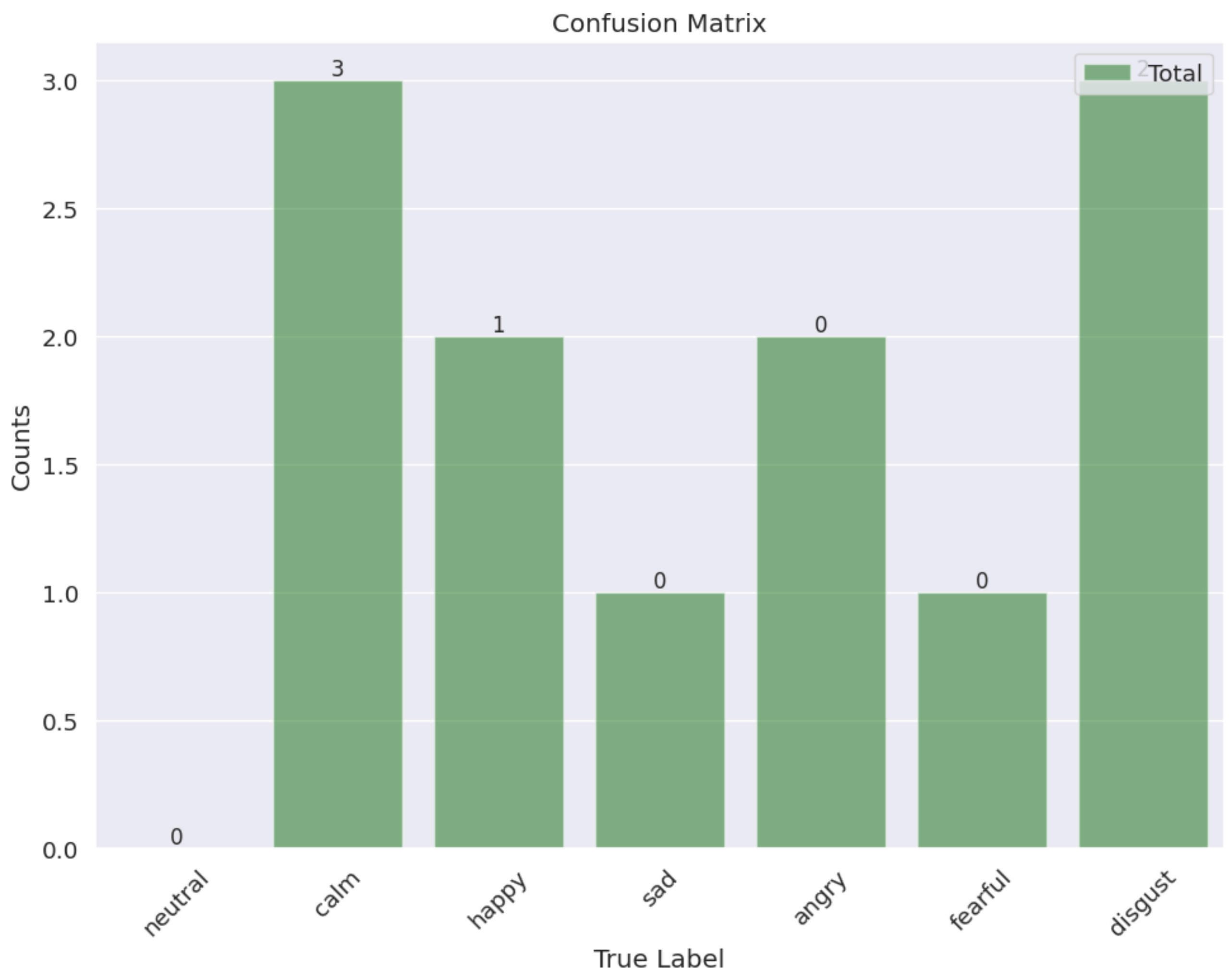
```
03-01-06-01-02-02-01.wav: Predicted Emotion - disgust
03-01-08-02-01-01-01.wav: Predicted Emotion - disgust
03-01-05-02-01-01-01.wav: Predicted Emotion - disgust
03-01-06-01-01-02-01.wav: Predicted Emotion - disgust
03-01-06-02-02-02-01.wav: Predicted Emotion - disgust
03-01-08-02-02-02-01.wav: Predicted Emotion - disgust
03-01-06-02-01-02-01.wav: Predicted Emotion - disgust
03-01-06-01-02-01-01.wav: Predicted Emotion - disgust
03-01-08-02-01-02-01.wav: Predicted Emotion - disgust
03-01-08-02-02-01-01.wav: Predicted Emotion - disgust
03-01-04-01-02-02-01.wav: Predicted Emotion - disgust
```

Confusion matrix and classification report for the predicted emotions

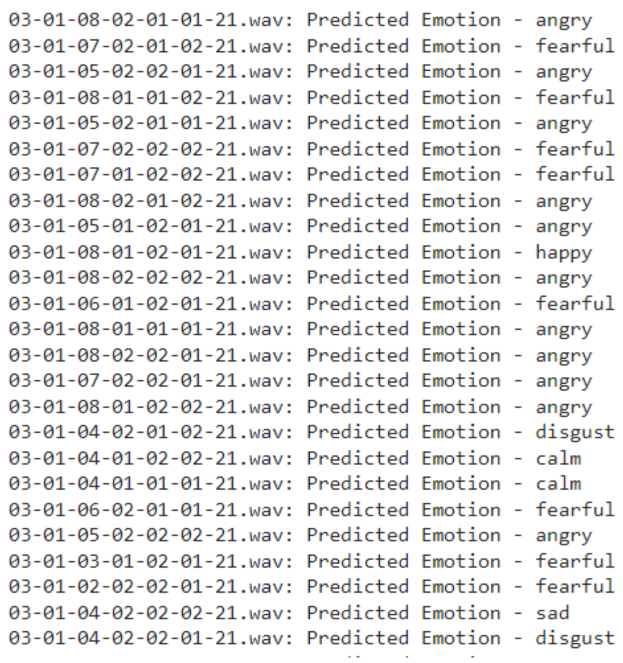


```
Confusion Matrix:
[[1 0 0 0 0 0 0]
 [0 2 0 0 0 0 0]
 [0 0 2 1 0 0 0]
 [0 0 0 1 0 0 0]
 [0 0 0 0 2 0 0]
 [0 0 1 0 0 1 0]
 [0 0 0 0 0 0 1]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       1.00      1.00      1.00         2
           2       0.67      0.67      0.67         3
           3       0.50      1.00      0.67         1
           4       1.00      1.00      1.00         2
           5       1.00      0.50      0.67         2
           6       1.00      1.00      1.00         1

    accuracy                           0.83        12
   macro avg       0.88      0.88      0.86        12
weighted avg       0.88      0.83      0.83        12
```

Bar chart based on emotions predicted.

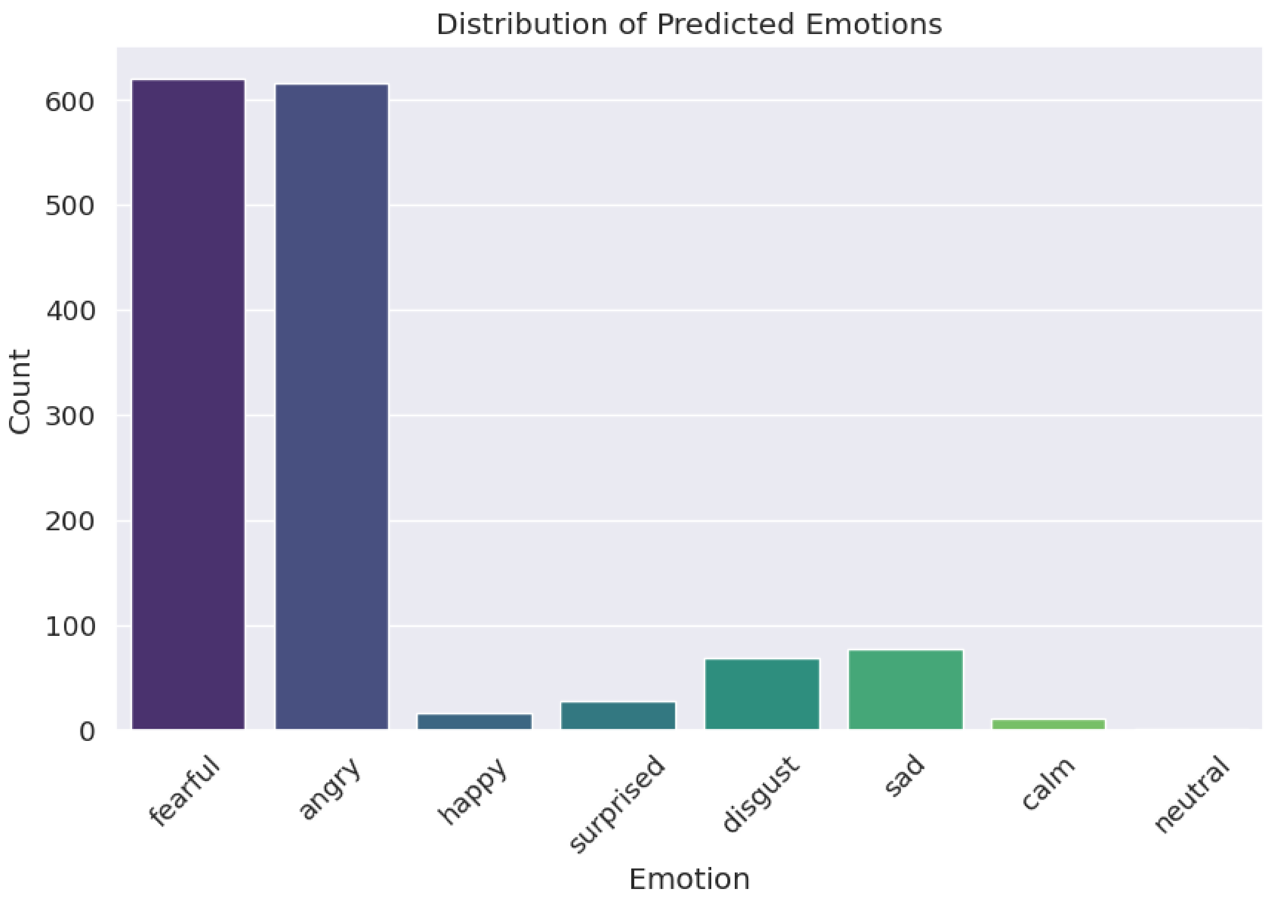


Emotions predicted for the overall dataset.



```
03-01-08-02-01-01-21.wav: Predicted Emotion - angry
03-01-07-02-01-02-21.wav: Predicted Emotion - fearful
03-01-05-02-02-01-21.wav: Predicted Emotion - angry
03-01-08-01-01-02-21.wav: Predicted Emotion - fearful
03-01-05-02-01-01-21.wav: Predicted Emotion - angry
03-01-07-02-02-02-21.wav: Predicted Emotion - fearful
03-01-07-01-02-02-21.wav: Predicted Emotion - fearful
03-01-08-02-01-02-21.wav: Predicted Emotion - angry
03-01-05-01-02-01-21.wav: Predicted Emotion - angry
03-01-08-01-02-01-21.wav: Predicted Emotion - happy
03-01-08-02-02-02-21.wav: Predicted Emotion - angry
03-01-06-01-02-01-21.wav: Predicted Emotion - fearful
03-01-08-01-01-01-21.wav: Predicted Emotion - angry
03-01-08-02-02-01-21.wav: Predicted Emotion - angry
03-01-07-02-02-01-21.wav: Predicted Emotion - angry
03-01-08-01-02-02-21.wav: Predicted Emotion - angry
03-01-04-02-01-02-21.wav: Predicted Emotion - disgust
03-01-04-01-02-02-21.wav: Predicted Emotion - calm
03-01-04-01-01-01-21.wav: Predicted Emotion - calm
03-01-06-02-01-01-21.wav: Predicted Emotion - fearful
03-01-05-02-02-02-21.wav: Predicted Emotion - angry
03-01-03-01-02-01-21.wav: Predicted Emotion - fearful
03-01-02-02-02-01-21.wav: Predicted Emotion - fearful
03-01-04-02-02-02-21.wav: Predicted Emotion - sad
03-01-04-02-02-01-21.wav: Predicted Emotion - disgust
```

Bar chart for the overall dataset based on the emotions predicted.



8.2 RESULT

Title	Author	Accuracy
An Experimental Study of Speech Emotion Recognition Based on Deep Convolutional Neural Networks	W. Q. Zheng, J. S. Yu, Y. X. Zou 2015	40%
Effective emotion recognition in movie audio tracks	Margarita Kotti, Yannis Stylianou	65.63%
Information Fusion in Attention Networks Using Adaptive and Multi-Level Factorized Bilinear Pooling for Audio-Visual Emotion Recognition	Hengshun Zhou, Jun Du, Yuanyuan Zhang, Qing Wang, Qing-Feng Liu, Chin-Hui 14 July 2021	75.49%
Speech Emotion Recognition in Neurological Disorders		83.33

9 CONCLUSION

The proposed scheme presented an approach to recognize the emotion from the human speech. The implementation of a Convolutional Neural Network (CNN) model for accurately classifying emotions from speech audio data. With an achieved accuracy of 83.3% on the testing set, the model demonstrates its efficacy in discerning various emotional states expressed in speech. Through meticulous preprocessing of the RAVDESS dataset and the design of an appropriate CNN architecture, we were able to capture meaningful features from the audio signals, enabling the model to make accurate predictions. The evaluation process, including the analysis of the confusion matrix and classification report, further confirms the model's robust performance across different emotions. By visualizing the model's classification results through a bar graph representation, stakeholders gain insights into its strengths and areas for improvement. Overall, the project underscores the potential of deep learning techniques in speech emotion recognition, paving the way for enhanced human-computer interaction and emotional intelligence applications.

10 FUTURE WORK

The ability to record a voice live and to predict the emotions in real time as the speaker is speaking. Several avenues for improvement and expansion present themselves. Firstly, exploring more sophisticated deep learning architectures beyond the CNN could yield further enhancements in accuracy and robustness. Architectures such as recurrent neural networks (RNNs) or their variants, such as long short-term memory (LSTM) networks, are well-suited for sequential data like speech, and integrating them into the model could capture temporal dependencies more effectively.

10 REFERENCES

[1] “Fusing audio, visual and textual clues for sentiment analysis from multimodal content”, 2016. (Elsevier)

[2] “Human Emotion Recognition using Machine Learning”, 2019. (Annual Research Journal)

[3] “Depression and anxiety detection through the Closed-Loop method using DASS-21”, 2019. (ResearchGate)

[4] Zainuddin, Nurulhuda, Ali Selamat, and Roliana Ibrahim. "Hybrid sentiment classification on twitter aspect-based sentiment analysis." Applied Intelligence 48, no. 5 (2018): 1218-1232.

[5] Chaturvedi, Iti, Ranjan Satapathy, Sandro Cavallari, and Erik Cambria. "Fuzzy commonsense reasoning for multimodal sentiment analysis." Pattern Recognition Letters 125 (2019): 264-270.

[6] Xiao, Guorong, Geng Tu, Lin Zheng, Teng Zhou, Xin Li, Syed Hassan Ahmed, and Dazhi Jiang. "Multimodality Sentiment Analysis in Social Internet of Things Based on Hierarchical Attentions and CSAT-TCN With MBM Network." IEEE Internet of Things Journal 8, no. 16 (2020): 12748-12757.

[7] He, Jiaxuan, Sijie Mai, and Haifeng Hu. "A unimodal reinforced transformer with time squeeze fusion for multimodal sentiment analysis." IEEE Signal Processing Letters 28 (2021): 992-996.

[8] Yang, Li, Ying Li, Jin Wang, and R. Simon Sherratt. "Sentiment analysis for E-commerce product reviews in Chinese based on sentiment lexicon and deep learning." IEEE access 8 (2020): 23522-23530.

[9] Youcef, Fatima Zohra, and Fatiha Barigou. "Arabic language investigation in the context of unimodal and multimodal sentiment analysis." In 2021 22nd International Arab Conference on Information Technology (ACIT), pp. 1-7. IEEE, 2021.

[10] Stappen, Lukas, Alice Baird, Erik Cambria, and Björn W. Schuller. "Sentiment analysis and topic recognition in video transcriptions." IEEE Intelligent Systems 36, no. 2 (2021): 88-95.

[11] Copaceanu, A.M., 2021. Sentiment Analysis Using Machine Learning Approach. Ovidius University Annals, Economic Sciences Series, 21(1), pp.261-270.

[12] Maxim, S., Ignatiev, N. and Smirnov, I., 2020. Predicting depression with social media images. In Proc. 9th Int. Conf. Pattern Recognit. Appl. Methods (Vol. 2, pp. 128-138