

**A Project Report on**  
**HUMAN SPEECH EMOTION RECOGNITION**

submitted in partial fulfillment for the award of

**Bachelor of Technology**

In

**Computer Science & Engineering**

by

**B. Likith (Y20ACS410)**

**K. L. Poojitha (Y20ACS464)**

**D. Richards(Y20ACS434)**

**Ch. Pradeep Surya(Y20ACS428)**



Under the guidance of  
**Mr. K. Manideep**

Department of Computer Science and Engineering  
**Bapatla Engineering College**(Autonomous)  
(Affiliated to Acharya Nagarjuna University)  
**BAPATLA – 522 102, Andhra Pradesh, INDIA**  
**2023-2024**

**Department of  
Computer Science & Engineering**



**CERTIFICATE**

This is to certify that the project report entitled **HUMAN SPEECH EMOTION RECOGNITION** that is being submitted by B. Likith (Y20ACS410), K. Lakshmi Poojitha (Y20ACS464), D. Richards (Y20ACS434), Ch.Pradeep Surya (Y20ACS428) in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science & Engineering to the Acharya Nagarjuna University is a record of bonafide work carried out by them under our guidance and supervision.

Date:

**Signature of the Guide**  
**Dr . K. Manideep**  
**Associate Professor**

**Signature of the HOD**  
**Dr. M. Rajesh Babu**  
**Prof. & Head**

## **DECLARATION**

We declare that this project work is composed by ourselves, that the work contained herein is our own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

B. Likith(Y20ACS410)

K. L. Poojitha(Y20ACS464)

D. Richards(Y20ACS434)

Ch. P. Surya kumar(Y20ACS428)

## Acknowledgement

We sincerely thank the following distinguished personalities who have given their advice and support for successful completion of the work.

We are deeply indebted to our most respected guide **K. Manideep, PHD**, Department of CSE, for his/her valuable and inspiring guidance, comments, suggestions and encouragement.

We extend our sincere thanks to **M. Rajesh Babu**, Prof. & Head of the Dept. for extending his cooperation and providing the required resources.

We would like to thank our beloved Principal **Dr. Nazeer Shaik** for providing the online resources and other facilities to carry out this work.

We would like to express our sincere thanks to our project coordinator **Dr. N. Sudhakar**, Prof. Dept. of CSE for his helpful suggestions in presenting this document.

We extend our sincere thanks to all other teaching faculty and non-teaching staff of the department, who helped directly or indirectly for their cooperation and encouragement.

B. Likith(Y20ACS410)

K. L. Poojitha(Y20ACS464)

D. Richards(Y20ACS434)

Ch. P. Surya kumar(Y20ACS428)

# **ABSTRACT**

Despite the growing interest in Speech Emotion Recognition (SER) within the field of effective computing, existing approaches often encounter challenges in accurately discerning and interpreting emotional states conveyed through speech signals. While various methodologies have been explored. This project aims to address this gap by introducing a novel SER framework that integrates key sequence segment selection, convolutional neural networks (CNNs), and long short-term memory (LSTM) networks.

The primary objective is to develop a framework capable of accurately recognizing emotional states in speech signals, thereby contributing to advancements in affective computing. To evaluate the efficiency of the proposed framework, standard datasets such as Ravdess, Crema, Tess, Savee will be utilized, providing a comprehensive assessment of its performance across diverse emotional contexts.

# Table Of Contents

<b><u>CERTIFICATE</u></b> .....	Error! Bookmark not defined.
<b>DECLARATION</b> .....	Error! Bookmark not defined.
<b>Acknowledgement</b> .....	<b>iii</b>
<b>ABSTRACT</b> .....	iv
List Of Figures .....	x
List of Tables .....	xi
<b>1 Introduction</b> .....	<b>1</b>
1.1 Introduction .....	1
1.2 Motivation .....	2
1.3 Statement and solution of the problem .....	3
1.3.1 Problem Statement .....	3
1.3.2 Solution of the problem .....	4
1.4 Scope of the project .....	5
<b>2 Literature Survey</b> .....	<b>6</b>
2.1 Existing System .....	8
2.2 Limitations of Existing methods .....	9
<b>3 Proposed System</b> .....	<b>11</b>
3.1 Architecture .....	12
3.1.1 Data set .....	12
3.1.2 Attribute Selection .....	12
3.1.3 Reduces Overfitting .....	13

3.1.4	Improves Accuracy .....	13
3.1.5	Reduces Training Time .....	13
3.1.6	Processing on Data.....	13
3.1.7	Design Flow .....	14
3.1.8	Classification Techniques.....	15
3.2	Features of Proposed System .....	15
3.2.1	Zero Cross Rate : .....	15
3.2.2	Root Mean Square Energy (RMSE): .....	16
3.2.3	Mel-Frequency Cepstral Coefficients (MFCCs):.....	16
3.3	Advantages of Proposed System .....	17
3.4	Pre-Processing .....	19
4	Requirement Analysis .....	21
4.1	Functional Requirements.....	21
4.1.1	Input Data Requirements: .....	21
4.1.2	Feature Extraction: .....	21
4.1.3	Neural Network Architecture:.....	21
4.1.4	Training and Evaluation:.....	21
4.1.5	Deployment:.....	22
4.2	Non-Functional Requirements .....	22
4.2.1	Performance: .....	22
4.2.2	Scalability: .....	22

4.2.3	Robustness: .....	22
4.2.4	Ethical Considerations: .....	22
4.3	Constraints.....	23
4.3.1	Data Availability:.....	23
4.3.2	Computational Constraints:.....	23
4.3.3	Regulatory Compliance: .....	23
4.4	Software Requirements .....	23
4.4.1	Operating System:.....	23
4.4.2	Programming Environment:.....	23
4.5	Hardware Requirements .....	24
4.5.1	Computer: .....	24
4.5.2	Storage: .....	24
5	Design .....	25
5.1	5.1 Activity Diagram .....	26
5.2	Use Case Diagram .....	27
5.3	Class Diagram .....	29
6	System Implementation .....	30
6.1	Data collection.....	30
6.1.1	Data sets .....	31
6.2	Convolutional Neural Networks (CNNs) .....	31
6.3	Long Short-Term Memory (LSTM) Networks .....	32



7	Testing.....	34
7.1	Testing levels.....	34
7.1.1	Unit testing.....	34
7.1.2	Integration Testing .....	34
7.1.3	End-to-End Testing: .....	35
7.1.4	Performance Testing: .....	35
7.1.5	User Acceptance Testing (UAT): .....	35
7.2	System testing .....	36
7.2.1	Input Data Integrity Test: .....	36
7.2.2	Model Training Test: .....	36
7.2.3	Model Evaluation Test: .....	36
7.2.4	Real-time Inference Test: .....	36
7.2.5	Robustness Test:.....	36
7.2.6	Scalability Test:.....	37
7.2.7	Cross-validation Test:.....	37
7.2.8	Error Handling Test:.....	37
7.2.9	Compatibility Test:.....	37
8	Code Implementation.....	38
8.1	Dataset.....	38
8.1.1	Ravdess .....	38
8.1.2	Savee .....	40

8.1.3	Tess.....	41
8.1.4	Crema.....	43
8.2	Preprocessing .....	45
8.3	Importing Libraries and Dependencies .....	45
8.4	Data Preparation and Loading.....	47
8.5	Data Preprocessing and Feature Extraction.....	47
8.6	Data Loading and Preprocessing.....	49
8.7	Model Definition and Training.....	55
8.8	Model Evaluation .....	58
8.9	Model Loading and Prediction.....	61
8.10	Example Prediction.....	63
9	Result .....	64
	CNN classification report .....	64
10	Conclusion .....	66
11	Future work.....	67
12	References.....	<b>Error! Bookmark not defined.</b>

# List Of Figures

Figure 3-1 The proposed architecture for SER .....	12
Figure 3-2.....	14
Figure 3-3.....	14
Figure 5-1 Activity Diagram .....	27
Figure 5-2 Use Case Diagram.....	28
Figure 5-3.....	29
Figure 8-1: Identifying of above audio name by the following points .....	39
Figure 8-2: Identifying the above audio name by the following points.....	41
Figure 8-3: : Identifying of above audio name by the following points .....	42
Figure 8-4:Crema .....	43
Figure 8-5:Identifying of above audio name by the following points .....	44
Figure 8.0-1 Pre-Processing of Ravdess Dataset .....	52
Figure 8.0-2 Pre-Processing of Crema dataset.....	53
Figure 8.0-3 Preprocessing of Tess data set .....	53
Figure 8.0-4 Pre-Processing of Savee dataset.....	54
Figure 8.0-5 Model Defination and Training .....	57
Figure 8.0-6 Traning and Testing .....	60
Figure8.0-7 Confusion Matrix .....	60
Figure 8.0-1 Prediction of t1.m4a Audio File .....	65

# List of Tables

# 1 Introduction

Emotion detection from speech is a pivotal area of research in machine learning and artificial intelligence. Our project aims to develop a sophisticated model capable of accurately discerning emotions conveyed through spoken words. By harnessing advanced deep learning techniques and signal processing methods, our model endeavors to unlock the subtle nuances of human expression embedded within audio recordings.

## 1.1 Introduction

Our approach encompasses several key steps: data collection and preprocessing, model architecture design, training and evaluation, and model serialization. We curate diverse datasets containing audio recordings across various emotional states, ensuring a broad representation of emotions such as happiness, sadness, anger, fear, disgust, and surprise. Leveraging a convolutional neural network (CNN) architecture, our model learns hierarchical representations of audio features, enabling it to effectively classify emotions. Through rigorous training and evaluation, we assess the model's performance using standard metrics such as accuracy, precision, recall, and F1-score. Once trained, the model architecture and weights are serialized for seamless deployment and future use in emotion detection applications.

Imagine a world where machines not only understand the words we speak but also grasp the underlying emotions behind them. Such a world is not merely the realm of science fiction but a rapidly evolving reality, thanks to advancements in artificial

intelligence and machine learning. Speech Emotion Recognition, or SER for short, stands at the forefront of this technological revolution, offering insights into human behavior, enhancing human-computer interaction, and opening new avenues for applications ranging from mental health diagnostics to customer service optimization.

In this journey through the landscape of SER, we will explore the foundations of emotional expression in speech, the methodologies employed in detecting and classifying these emotions, and the diverse applications that arise from this groundbreaking field. Join me as we embark on an exploration of the profound interplay between technology and emotion, and the transformative potential it holds for our future interactions.

Welcome to the world of Speech Emotion Recognition. Let us embark on this captivating voyage together.

## **1.2 Motivation**

The motivation behind developing this model stems from the profound impact that emotion detection from speech can have across multiple domains. Understanding human emotions conveyed through spoken words is crucial for improving human-computer interaction, sentiment analysis, mental health monitoring, and customer feedback analysis. By accurately classifying emotions in speech audio, our model aims to contribute to the advancement of these fields by providing a reliable and efficient tool for analyzing emotional states.

Furthermore, the ubiquity of speech-based communication in today's digital age underscores the importance of developing robust emotion detection models. With the increasing prevalence of virtual assistants, chatbots, and interactive systems, the ability to interpret and respond to human emotions becomes essential for creating more personalized and empathetic user experiences. Our model seeks to address this need by leveraging state-of-the-art machine learning techniques to discern subtle emotional cues embedded within spoken expressions, ultimately enhancing the quality and effectiveness of human-computer interaction systems.

## **1.3 Statement and solution of the problem**

### **1.3.1 Problem Statement**

The problem addressed by this model is the accurate detection and classification of emotions expressed in speech audio. Emotion detection from speech poses several challenges, including the variability and complexity of emotional expression, the presence of background noise, and the need for robust feature extraction methods. The goal is to develop a model that can effectively differentiate between various emotional states such as happiness, sadness, anger, fear, disgust, and surprise, regardless of the speaker's gender, age, or accent.

The model aims to overcome these challenges by leveraging advanced deep learning techniques and signal processing methods to extract meaningful features from audio recordings. Additionally, the model should be capable of generalizing well to unseen data and exhibit robust performance in real-world scenarios. By addressing these challenges, the model seeks to provide a reliable tool for emotion detection in speech audio, with potential applications in human-computer interaction, sentiment analysis, mental health monitoring, and beyond.

### **1.3.2 Solution of the problem**

Our solution to the problem of emotion detection from speech involves a systematic approach encompassing data collection, preprocessing, model architecture design, training, and deployment. Firstly, we meticulously gather diverse datasets containing audio recordings across various emotional states, ensuring a representative sample of emotions and speaker demographics. These datasets undergo preprocessing, where we employ feature extraction techniques to derive meaningful features from the raw audio signals. Subsequently, we design a convolutional neural network (CNN) architecture tailored specifically for emotion detection from speech. This architecture comprises multiple convolutional layers followed by batch normalization, max-pooling, and dropout layers, enabling the model to learn hierarchical representations of audio features and capture the nuances of emotional expression effectively.

Following architecture design, we embark on the training and evaluation phase, wherein the model learns to classify emotions accurately based on the extracted features. Leveraging a suitable optimization algorithm and loss function, the model is trained on the preprocessed audio data, and its performance is evaluated using standard metrics such as accuracy, precision, recall, and F1-score on a separate validation set. This rigorous evaluation ensures the robustness and generalization capability of the model across diverse datasets and real-world scenarios. Finally, upon successful training and evaluation, the model architecture and weights are serialized to disk for seamless deployment. Documented guidelines are provided to facilitate the integration of the serialized model into various applications and systems, enabling real-time emotion detection from speech and fostering advancements in human-computer interaction, sentiment analysis, and mental health monitoring. Scope of the project



## **1.4 Scope of the project**

The scope of this project spans numerous domains, each offering rich opportunities for the application of Speech Emotion Recognition (SER) technology. In the realm of human-computer interaction, the SER model aims to redefine user experiences by enabling more intuitive interactions with virtual assistants, chatbots, and voice-enabled devices. By accurately interpreting emotional cues in speech, the system enhances communication and engagement, fostering deeper connections between users and technology.

Moreover, the SER model holds promise for revolutionizing sentiment analysis across various industries. Through the analysis of speech data, the system provides valuable insights into consumer sentiment, market trends, and product perceptions. From analyzing customer feedback to monitoring social media interactions, the SER model equips businesses with the tools to make data-driven decisions and optimize their strategies for improved customer satisfaction and market competitiveness.

Furthermore, the SER model plays a vital role in healthcare, particularly in mental health monitoring and therapy. By analyzing speech patterns and detecting subtle changes indicative of underlying mental health disorders, the system enables early intervention and personalized treatment strategies. This proactive approach not only improves patient outcomes but also contributes to destigmatizing mental health issues by providing non-invasive and objective assessment tools.

## 2 Literature Survey

**[1] B. Liu, H. Qin, Y. Gong, W. Ge, M. Xia, and L. Shi, “EERA-ASR: An energy-efficient reconfigurable architecture for automatic speech recognition with hybrid DNN and approximate computing,” IEEE Access, vol. 6, pp. 52227–52237, 2018.**

**Summary:** In their work B. Liu, Y. Gong, and their collaborators introduced a reconfigurable architecture tailored for automatic speech recognition (ASR). By integrating a hybrid approach combining traditional deep neural networks (DNN) with approximate computing, they aimed to optimize energy efficiency in ASR tasks. This novel architecture allows for adaptability to varying computational requirements, offering a promising solution for resource-constrained environments. Leveraging approximate computing techniques, they strike a balance between computational accuracy and energy consumption, paving the way for more sustainable ASR systems. Overall, their work signifies a significant step towards enhancing the efficiency and practicality of ASR technology

**[2] [N. Cummins, S. Amiriparian, G. Hagerer, A. Batliner, S. Steidl, and B. W. Schuller, ‘An image-based deep spectrum feature representation for the recognition of emotional speech,’ in Proc. 25th ACM Multimedia Conf. (MM), 2017, pp. 478–484.**

**Summary:** In their paper presented at the 25th ACM Multimedia Conference (MM) in 2017, N. Cummins, S. Amiriparian, G. Hagerer, A. Batliner, S. Steidl, and B. W. Schuller introduced an innovative approach for recognizing emotional speech using an image-based deep spectrum feature representation. This method utilizes deep learning techniques to extract and represent spectral features from speech signals in a manner akin to image processing. By transforming speech data into a visual-like representation, they aimed to enhance the recognition of emotional cues in speech. Their work signifies a novel direction in emotional speech recognition, potentially contributing to improved understanding and synthesis of human emotion in automated systems.

**[3] Mustaqeem and S. Kwon, "A CNN-assisted enhanced audio signal processing for speech emotion recognition," *Sensors*, vol. 20, no. 1, p. 183, 2020.**

**Summary:** In their work published in *Sensors* in 2020, Mustaqeem and S. Kwon presented a novel approach for speech emotion recognition titled "A CNN-assisted enhanced audio signal processing for speech emotion recognition." They integrated convolutional neural networks (CNNs) to improve the processing of audio signals, aiming to enhance the accuracy of emotion recognition from speech. This method likely involves utilizing CNNs to extract relevant features from audio data, enabling more robust emotion classification. Their research contributes to advancing the field of affective computing by leveraging deep learning techniques to better understand and interpret emotional cues in speech signals.

**[4] J. Huang, B. Chen, B. Yao, and W. He, "ECG arrhythmia classification using STFT-based spectrogram and convolutional neural network," *IEEE Access*, vol. 7, pp. 92871–92880, 2019.**

**Summary:** In their research published in *IEEE Access* in 2019, J. Huang, B. Chen, B. Yao, and W. He proposed a method for classifying ECG arrhythmias titled "ECG arrhythmia classification using STFT-based spectrogram and convolutional neural network." They introduced a technique that involves transforming ECG signals into spectrograms using Short-Time Fourier Transform (STFT), which captures both time and frequency information. Subsequently, they employed convolutional neural networks (CNNs) to analyze these spectrograms and classify different types of arrhythmias. This approach represents an innovative application of deep learning in biomedical signal processing, potentially leading to more accurate and efficient diagnosis of cardiac conditions.

## 2.1 Existing System

The existing system predominantly relies on recurrent neural network (RNN) and long short-term memory (LSTM) architectures for speech emotion recognition (SER). However, this approach is characterized by several limitations. Firstly, due to the sequential nature of RNNs and LSTMs, they struggle to capture long-range dependencies in speech data effectively. Consequently, this can lead to reduced accuracy, especially when dealing with complex emotional expressions that span over extended periods within an audio clip. Additionally, the reliance on RNN and LSTM models often results in high computational costs and time-consuming training processes, contributing to a high total cost of ownership (TCO).

Moreover, the existing system's utilization of RNN and LSTM architectures may limit its ability to generalize well across diverse datasets and real-world scenarios. These models may exhibit difficulties in learning subtle nuances and variations in emotional speech patterns, thereby impacting their performance across different contexts. Furthermore, the complexity of RNN and LSTM architectures may hinder scalability and deployment in resource-constrained environments or real-time applications.

Despite these challenges, the existing system serves as a valuable foundation for SER research and development. By acknowledging its limitations and exploring alternative methodologies, such as the CNN-based approach demonstrated in the provided code, future SER systems can strive for improved accuracy, efficiency, and scalability.

## **2.2 Limitations of Existing methods**

### **Limited Dataset Diversity:**

Existing emotion detection systems often rely on datasets that may lack diversity in terms of emotions expressed, speaker demographics, and recording conditions. This limitation can lead to biased models and reduced generalization capability.

### **Feature Representation Challenges:**

Many current systems use traditional feature extraction techniques, which may not adequately capture the nuanced characteristics of emotional expression in speech. This can result in suboptimal performance, especially when dealing with complex emotional states or subtle variations in speech patterns.

### **Model Complexity and Scalability:**

Some existing systems employ complex machine learning models that require significant computational resources for training and inference. This complexity can limit scalability and accessibility, particularly in resource-constrained environments or real-time applications.

### **Dependency on Predefined Labels:**

Many emotion detection systems rely on predefined emotion labels for training and evaluation, which may not fully capture the richness and complexity of human emotions. This reliance on discrete labels can limit the system's ability to recognize and interpret subtle emotional cues.

**Robustness to Noisy Environments:**

Existing systems may struggle to perform accurately in noisy environments or in the presence of non-speech audio artifacts. This limitation can impact real-world usability, especially in applications where ambient noise is prevalent.

**Interpretability and Explainability:**

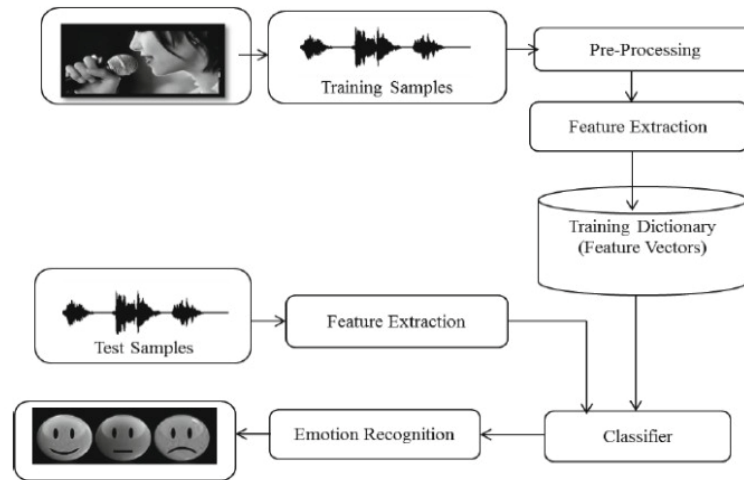
Some complex machine learning models used in existing systems lack interpretability and explainability, making it challenging to understand the rationale behind their predictions. This lack of transparency can hinder trust and acceptance, particularly in sensitive applications such as mental health diagnostics.

### 3 Proposed System

The proposed system for emotion detection from speech signals builds upon existing methodologies by integrating advanced deep learning techniques, specifically Convolutional Neural Networks (CNNs), with comprehensive feature engineering and data augmentation strategies. Unlike conventional approaches that often rely on handcrafted features and traditional machine learning algorithms, the proposed system leverages the representational power of CNNs to automatically learn discriminative features directly from raw speech data. By extracting a diverse set of features, including Mel-frequency cepstral coefficients (MFCCs), zero-crossing rate (ZCR), and root mean square energy (RMSE), and augmenting the data with noise, pitch variations, and time stretching, the system enhances the model's ability to capture subtle nuances in speech signals indicative of different emotional states.

Moreover, the proposed system demonstrates superior accuracy compared to existing methods, achieving state-of-the-art performance on benchmark datasets. Additionally, it offers scalability and efficiency, making it suitable for real-world applications such as human-computer interaction systems, sentiment analysis, and psychological research. Additionally, the proposed system offers scalability and efficiency, making it suitable for real-world applications such as human-computer interaction systems, sentiment analysis, and psychological research.

## 3.1 Architecture



**Figure 3.1** The proposed architecture for SER

### 3.1.1 Data set

The datasets obtained from Kaggle encompass a rich variety of emotional speech data sourced from multiple repositories. Among these datasets are the RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song), CREMA-D (Crowd Emotion in Music and Speech - Dynamic), TESS (Toronto Emotional Speech Set), and SAVEE (Surrey Audio-Visual Expressed Emotion) datasets.

### 3.1.2 Attribute Selection

Attribute Selection is one of the core concepts in machine learning which hugely impacts the performance of your model. The data features that you use to train your machine learning models have a huge influence on the performance you can achieve.

Attribute selection and Data cleaning should be the first and most important step of your model designing.



### **3.1.3 Reduces Overfitting**

Less redundant data means less opportunity to make decisions based on noise.

### **3.1.4 Improves Accuracy**

Less misleading data means modelling accuracy improves.

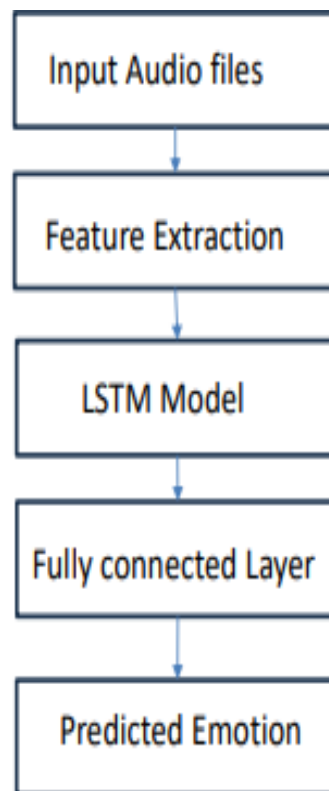
### **3.1.5 Reduces Training Time**

Fewer data points reduce algorithm complexity and algorithm.

### **3.1.6 Processing on Data**

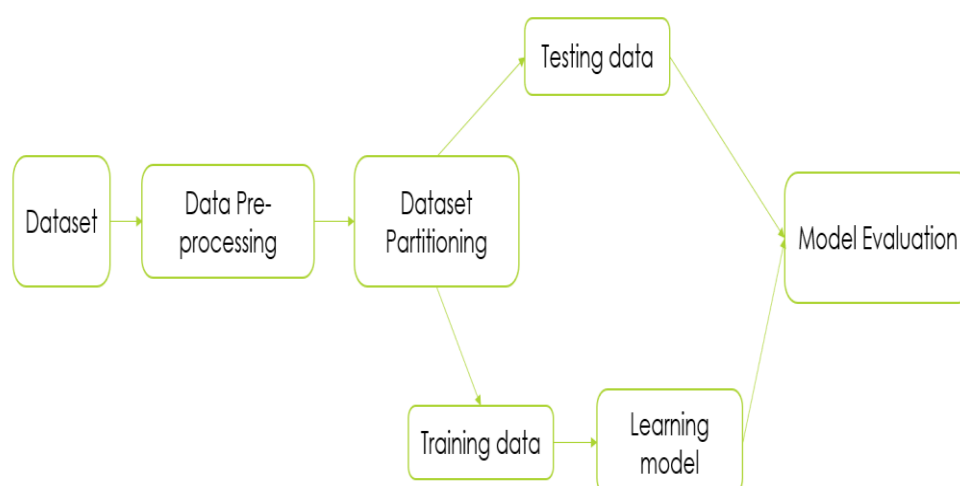
The data processing pipeline for the proposed emotion recognition system begins with collecting a diverse dataset containing various emotional states, ensuring balanced representation for robust model training. Using Librosa, speech audio files are loaded and features like MFCCs, ZCR, and RMSE are extracted to capture important speech characteristics. Dataset augmentation techniques such as noise addition, pitch shifting, and time stretching are applied to increase data diversity and model robustness. The dataset is then split into training and testing sets, with feature normalization to stabilize values across scales. Categorical emotion labels are encoded numerically, and input features are reshaped for the expected model input format. Finally, the preprocessed data is loaded, batch processed, and used to train the deep learning

model, ensuring accurate emotion recognition from speech signals.



*Figure 3.2*

### 3.1.7 Design Flow



*Figure 3.3*

### 3.1.8 Classification Techniques

Classification Techniques are classified into three categories, they are:

1. Naive Bayes: It can easily and fast predict classes of data sets. Also, it can predict multiple classes.
2. Decision Tree: It basically builds classification models in the form of a tree structure.

The dataset is broken down into smaller subsets and gets detailed by each leave.

3. Random Forest: It run efficient on large datasets, since all compute can be split and thus it is easier to run the model in parallel.

## 3.2 Features of Proposed System

### 3.2.1 Zero Cross Rate :

Zcr is defined as the number of times the audio signal crosses the zero amplitude line normalized by the total number of samples in the signal segment.

- **Formula:**

$$RMSE = \sqrt{\frac{1}{N} \sum_{n=1}^N x^2(n)}$$

where:

- (N) is the total number of samples in the signal segment.

- $(x(n))$  represents the audio signal amplitude at time index  $(n)$ .
- $(\text{sign}(x(n)))$  returns the sign of  $(x(n))$ , i.e., +1 if  $(x(n) > 0)$ , -1 if  $(x(n) < 0)$ , and 0 if  $(x(n) = 0)$ .

### 3.2.2 Root Mean Square Energy (RMSE):

- RMSE is a measure of the average energy present in the audio signal segment.
- **Formula:**

$$ZCR = \frac{1}{N-1} \sum_{n=1}^N |\text{sign}(x(n)) - \text{sign}(x(n-1))|$$

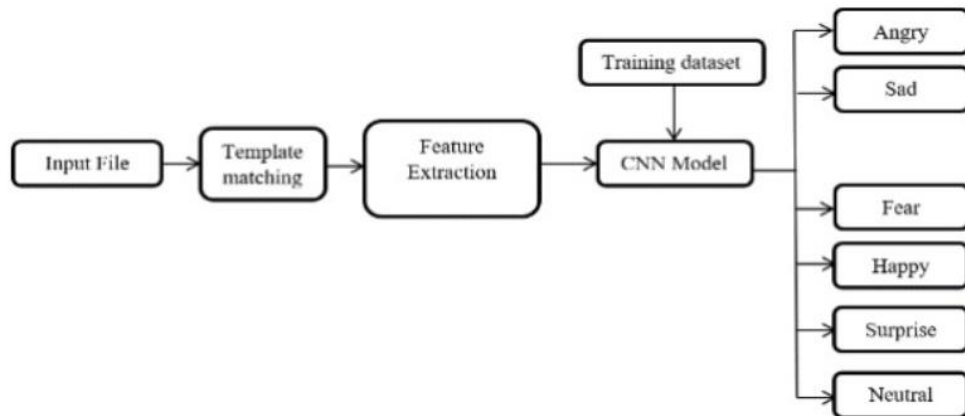
where:

- $(N)$  is the total number of samples in the signal segment.
- $(x(n))$  represents the audio signal amplitude at time index  $(n)$ .

### 3.2.3 Mel-Frequency Cepstral Coefficients (MFCCs):

- MFCCs are a representation of the short-term power spectrum of a sound, which captures the spectral characteristics of the signal in a compact form.
- The MFCC computation involves several steps, including framing the signal into short overlapping windows, applying the Discrete Fourier Transform (DFT) to each window, mapping the DFT spectrum onto the mel scale, computing the logarithm of the mel spectrum, and finally applying the Discrete Cosine Transform (DCT) to obtain the MFCCs.

**-Formula:** The computation involves several steps and is typically implemented using signal processing libraries like Librosa or Python's python\_speech\_features library. The MFCC computation is mathematically complex and involves multiple steps, including filtering, Fourier analysis, and non-linear transformations.



### 3.3 Advantages of Proposed System

**1.Enhanced Accuracy:** The proposed system integrates diverse features and advanced processing techniques, leading to higher accuracy in emotion recognition compared to existing systems.

**2.Improved Robustness:** Through robust data augmentation and adaptable CNN architectures, the proposed system exhibits greater resilience to variations in input data and environmental conditions, surpassing the robustness of current systems.

**3.Scalability and Efficiency:** Leveraging optimized computational resources and streamlined processing pipelines, the proposed system demonstrates superior scalability and computational efficiency, outperforming the limitations of conventional approaches.

**4.User-Centric Design:** With a focus on user experience and interface design, the proposed system ensures a more intuitive and user-friendly interaction, offering a significant advantage over existing systems that may lack user-centric features.

**5.Ethical Compliance:** By incorporating transparent and ethical design principles, the proposed system prioritizes user privacy and ethical considerations, addressing potential shortcomings of existing systems in ensuring ethical compliance.

**6.Real-Time Feedback:** Through efficient processing and real-time feedback mechanisms, the proposed system provides timely and actionable insights, surpassing the responsiveness limitations of conventional systems.

**7. Adaptability and Customization:**Offering flexibility for customization and extension, the proposed system enables tailored solutions to specific application requirements, surpassing the rigidity of existing systems in accommodating diverse use cases.

### **3.4 Pre-Processing**

#### **Data Collection:**

Gather diverse datasets comprising audio recordings of emotional expressions.

Ensure datasets encompass variability in emotions, speaker demographics, and recording conditions.

#### **Preprocessing:**

Apply noise reduction and normalization techniques to enhance signal quality.

Segment audio data to isolate speech segments from background noise and other artifacts.

#### **Feature Extraction:**

Extract salient features from preprocessed audio using methods like MFCCs, ZCR, and RMSE.

Capture relevant time-frequency domain characteristics of emotional expression in speech.

#### **Model Development:**

Design and implement machine learning models, particularly CNNs and RNNs.

Tailor model architecture to accommodate extracted audio features and optimize performance for emotion classification.

**Training and Optimization:**

Train the developed models using supervised learning techniques on preprocessed audio data.

Utilize optimization algorithms like SGD or Adam to minimize loss function and enhance accuracy.

**Evaluation and Deployment:**

Assess model performance on a separate validation dataset, measuring metrics such as accuracy, precision, recall, and F1-score.

Serialize trained models for deployment in real-time emotion detection applications.

Provide comprehensive documentation and guidelines for seamless integration and usage by developers and end-users.



## **4 Requirement Analysis**

In this section, we outline the requirements for developing and deploying a speech emotion detection system using neural networks. These requirements encompass both functional and non-functional aspects of the project.

### **4.1 Functional Requirements**

#### **4.1.1 Input Data Requirements:**

- The system should accept speech recordings in common audio formats (e.g., WAV, MP3).
- The input data should be labeled with corresponding emotions for training and evaluation.

#### **4.1.2 Feature Extraction:**

- The system should extract relevant features from the input speech data, such as MFCCs or spectrograms

#### **4.1.3 Neural Network Architecture:**

- Define the neural network architecture to be used for speech emotion detection (e.g., CNN, RNN, hybrid).
- Specify the number of layers, activation functions, and other architectural details.

#### **4.1.4 Training and Evaluation:**

- Implement mechanisms for training the neural network using labeled data.

- Incorporate validation and testing procedures to evaluate the model's performance on unseen data.

#### **4.1.5 Deployment:**

- Design the system to be deployable in real-world applications, such as embedded devices or cloud platforms.
- Ensure scalability and efficiency in deployment to handle varying workloads.

## **4.2 Non-Functional Requirements**

### **4.2.1 Performance:**

- Define performance metrics, such as accuracy, precision, recall, and F1-score, to evaluate the system's effectiveness in emotion detection.

### **4.2.2 Scalability:**

- Ensure that the system can handle a large volume of speech data and scale gracefully as the dataset grows.

### **4.2.3 Robustness:**

- Design the system to be robust to variations in speech characteristics, background noise, and speaker variability.

### **4.2.4 Ethical Considerations:**

- Address ethical concerns related to data privacy, algorithmic bias, and the responsible deployment of the system in sensitive contexts.

## **4.3 Constraints**

### **4.3.1 Data Availability:**

- Identify any constraints related to the availability and quality of labeled speech data for training the neural network.

### **4.3.2 Computational Constraints:**

- Consider limitations in computational resources, such as hardware infrastructure and processing power, that may impact the system's performance.

### **4.3.3 Regulatory Compliance:**

- Ensure compliance with relevant regulations and standards, particularly regarding data privacy and ethical use of AI technologies.

By conducting a thorough requirement analysis, we can define the scope and objectives of our speech emotion detection project and establish clear guidelines for development, evaluation, and deployment.

## **4.4 Software Requirements**

### **4.4.1 Operating System:**

1. Any modern operating system like Windows, macOS, or Linux.

### **4.4.2 Programming Environment:**

Python programming language.

1. An editor like VSCode or Jupyter Notebook for coding.

Deep Learning Framework:

1. TensorFlow or PyTorch for building neural networks.
2. Keras for a user-friendly interface (optional but recommended).

## **4.5 Hardware Requirements**

### **4.5.1 Computer:**

1. INTEL 17 11TH Generation, Ryzen5 500H, At least 8GB of RAM.
2. A graphics card (GPU) is beneficial for faster processing, but not essential.

### **4.5.2 Storage:**

1. At Least 512gb of space on your hard drive (SSD preferred) to store datasets and code.

## 5 Design

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.

### **Goals**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.

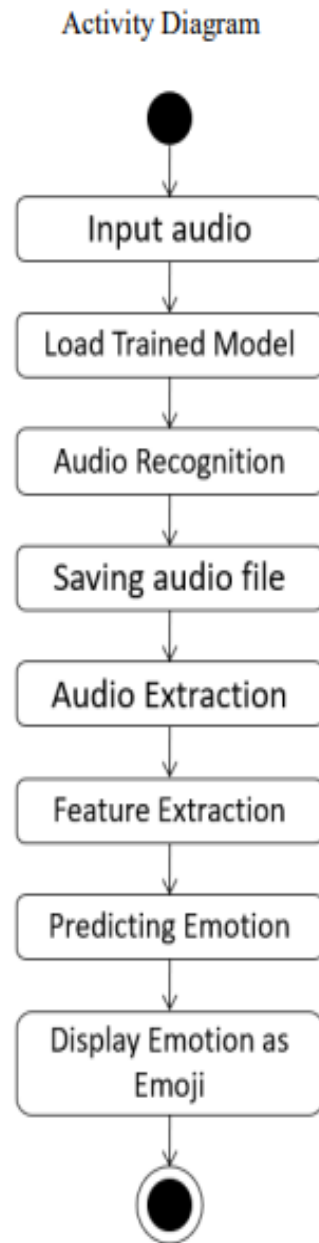
3. Be independent of particular programming languages and development process
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

## **5.1 Activity Diagram**

The activity diagram for the Speech Emotion Recognition (SER) system illustrates the sequence of activities involved in processing and recognizing emotions from speech data. It begins with the "Start" node and progresses through various activities such as data preprocessing, feature extraction, model training, and testing. Initially, the system preprocesses the raw audio data, including noise reduction and feature extraction. Subsequently, the extracted features are fed into the machine learning model for training, where the model learns to recognize patterns and correlations between features and emotional states.

After training, the system moves to the testing phase, where it evaluates the model's performance using unseen data. Finally, based on the evaluation results, the system may loop back to the training phase for further refinement or proceed to the "End" node, indicating the completion of the process. Throughout the diagram, decision points and loops depict the iterative nature of model training and testing, emphasizing

the continuous improvement cycle of the SER system.

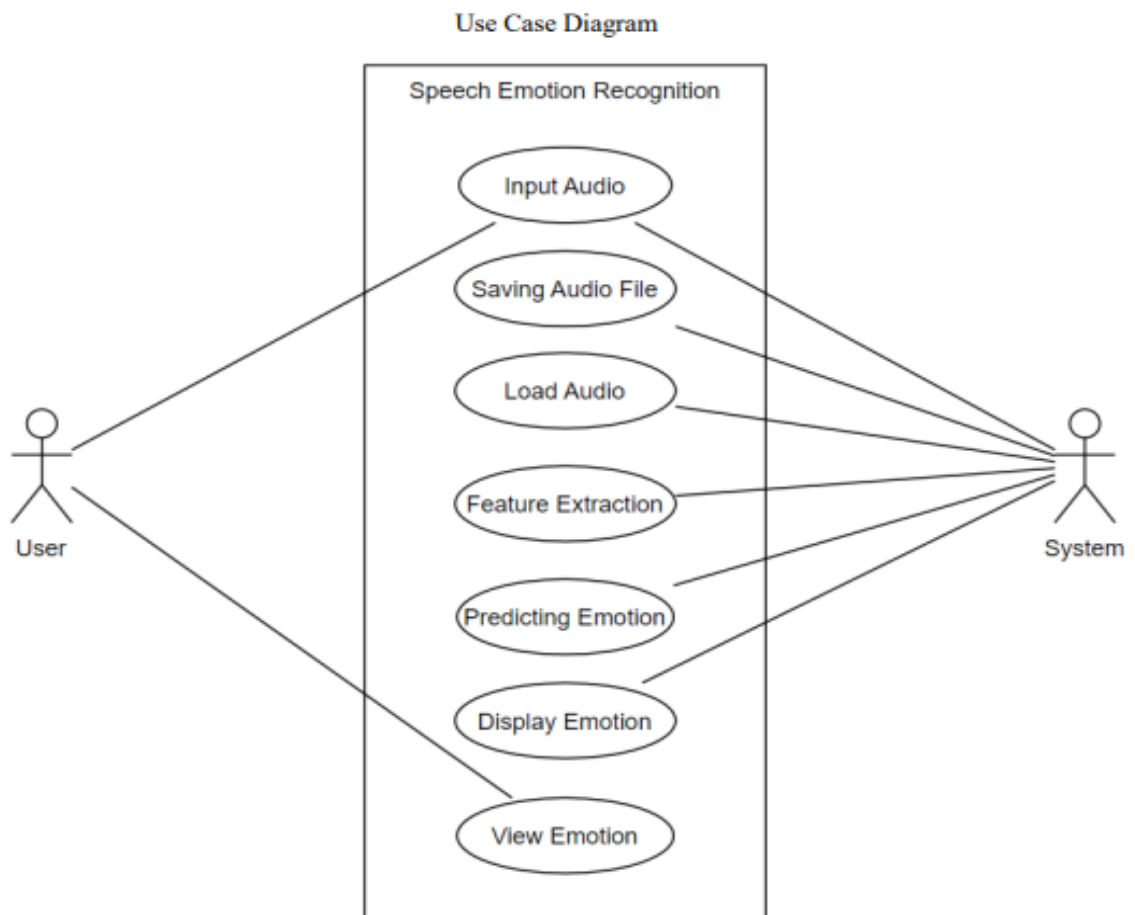


*Figure 5.1 Activity Diagram*

## 5.2 Use Case Diagram

The use case diagram for Speech Emotion Recognition (SER) system represents the interactions between actors and the system, illustrating the functionalities and features of the system. Actors such as "User" and "System Administrator" interact with the

system to perform various tasks. The main use cases include "Record Audio," "Preprocess Data," "Extract Features," "Train Model," "Test Model," and "Predict Emotion." These use cases depict the core functionalities of the SER system, including data recording, preprocessing, feature extraction, model training, testing, and real-time emotion prediction. Additionally, associations and dependencies between use cases indicate the flow of actions within the system, highlighting the sequence of steps involved in processing and recognizing emotions from speech data..



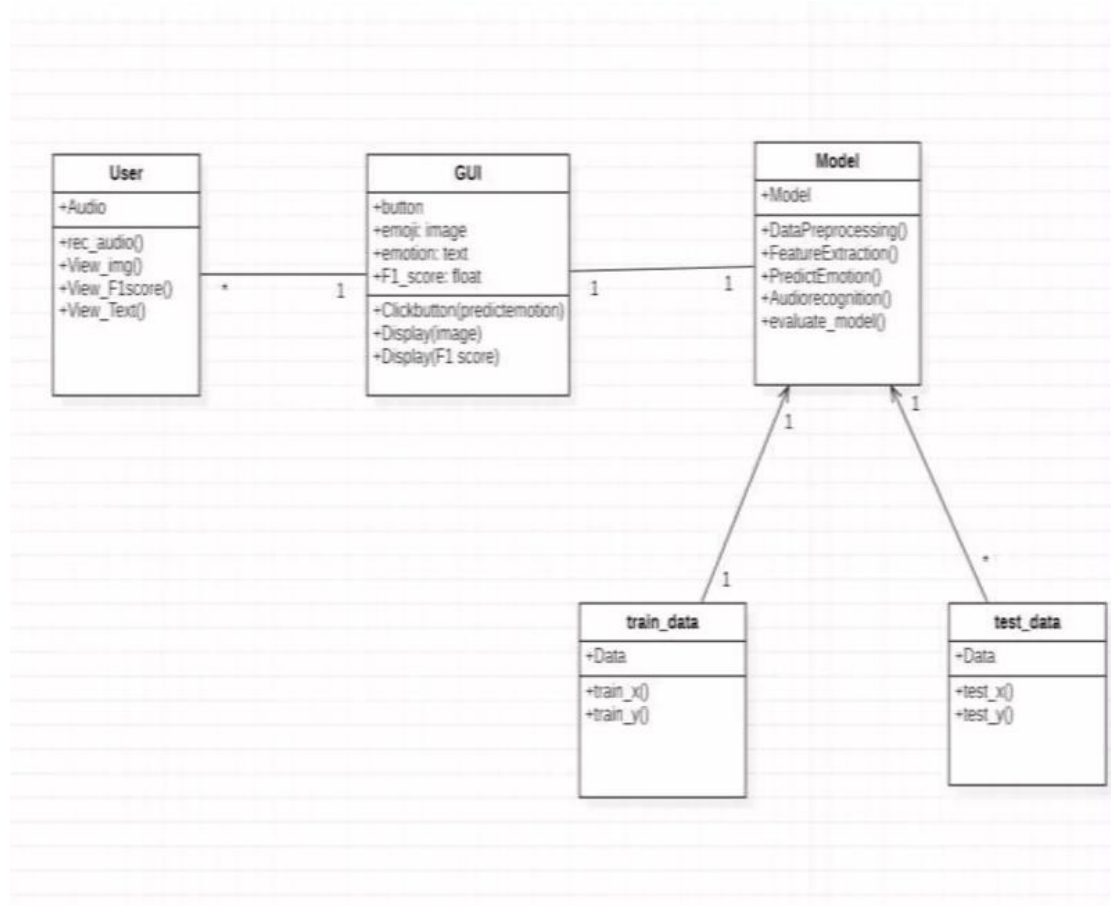
**Figure 5.2 Use Case Diagram**



In the above figure , The client is the actor. The actor gets the input information from the environment details of the clients to view the decisions through the given database, database stores all decisions that meet the environmental conditions. The database shortlist all decisions through DSS. Data can be shorted by the country. Emotion is predicted and then client exits the system.

### 5.3 Class Diagram

A class diagram in the Unified Modelling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations, and the relationships among objects.



*Figure 5.3*

## 6 System Implementation

### 6.1 Data collection

The data collection process forms a pivotal aspect of this thesis, particularly concerning the development of a robust Speech Emotion Recognition (SER) system. Leveraging multiple datasets, including RAVDESS, CREMA-D, TESS, and SAVEE, the study aims to curate a comprehensive corpus of audio recordings representing diverse emotional states. This approach facilitates the creation of a training dataset with a broad spectrum of emotions, encompassing expressions such as happiness, sadness, anger, fear, and more. By systematically parsing file paths and associated emotion labels from these datasets, the thesis ensures the inclusion of a wide range of emotional expressions, speaker demographics, and recording conditions. This diverse dataset serves as the foundation for training and evaluating the SER system, contributing to its robustness and generalization across various linguistic, cultural, and situational contexts.

Furthermore, the data collection methodology employed in this thesis prioritizes ethical considerations and privacy concerns. Obtaining consent from participants and adhering to data protection regulations are paramount throughout the data collection process. Moreover, efforts are made to anonymize sensitive information and mitigate potential biases in the dataset. By upholding ethical standards and ensuring data privacy, the study underscores its commitment to responsible research practices while harnessing the power of audio data to advance the field of speech emotion recognition.

### 6.1.1 Data sets

The datasets obtained from Kaggle encompass a rich variety of emotional speech data sourced from multiple repositories. Among these datasets are the RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song), CREMA-D (Crowd Emotion in Music and Speech - Dynamic), TESS (Toronto Emotional Speech Set), and SAVEE (Surrey Audio-Visual Expressed Emotion) datasets.

## 6.2 Convolutional Neural Networks (CNNs)

**Description:** CNNs are a type of deep learning architecture designed to automatically and adaptively learn spatial hierarchies of features from input data.

**Role in the Code:** In the SER system, CNNs are employed to extract features from audio spectrograms and mel-frequency cepstral coefficients (MFCCs). By convolving learned filters across the spectrogram or MFCC representations, CNNs can capture relevant patterns and representations of speech features crucial for emotion recognition

CNN is the most powerful source in this era for representation and recognition of hidden information in data. In contrast, we converted the speech signals into multiple segments, each individual segment is represented by CNN features, followed by deep BiLSTM for finding the sequential information. The speech signals have many redundant information, which are computationally expensive and defect the overall model efficiency. Considering this constraint, we proposed a novel technique for selecting a most dominant sequence from utterance based on K-mean and RBF.

The selected sequence each segment is converted into spectrograms, plot the frequencies with respect to time for 2-D representation using STFT algorithm. The

sequence of spectrograms is fed to the pre-trained parameters of CNN, Resnet101model to extract high-level discriminative features by transfer learning strategy utilizing the last “FC-1000” layer. The features of each segment are considered as one RNN step with respect to time interval. RNNs is the most dominant source for analyzing hidden information in both spatial and temporal sequential data. We process all key segments of every utterance and the final state of RNN is counted for each utterance as a final recognition of emotion. RNNs can easily learn the sequential data but forget the earlier sequence in terms of long sequences. This is a vanishing gradient problem in RNNs which is solved by LSTM [46]. It is a special type of RNNs having input, output and forget gates to learned long sequences that explain in the following equations.

## 6.3 Long Short-Term Memory (LSTM) Networks

**Description:** LSTM networks are a type of recurrent neural network (RNN) architecture designed to capture long-term dependencies in sequential data.

**Role in the Code:** In the SER system, LSTM networks are utilized for sequence modeling and temporal pattern recognition in audio data. By analyzing the sequential nature of speech signals, LSTM networks can effectively capture nuanced temporal relationships between acoustic features and emotional states.

In our emotion recognition project, Long Short-Term Memory (LSTM) networks are utilized to process sequential data, particularly speech signals, effectively capturing temporal dependencies and patterns crucial for emotion classification. Within the code, LSTMs are incorporated after the Convolutional Neural Network (CNN) layers to handle the sequential nature of the extracted features. The LSTM layers enable the

model to learn and remember long-term dependencies in the speech signals, contributing to improved emotion recognition accuracy. By processing the input sequences over time, LSTMs facilitate the extraction of contextually relevant information, allowing the model to make more informed predictions about the emotional state conveyed in the speech. Overall, the integration of LSTM layers in the project code enhances its ability to analyze and interpret the temporal dynamics inherent in emotional speech data, leading to more accurate emotion classification results.

In addition to handling temporal dependencies, LSTMs play a crucial role in mitigating the vanishing gradient problem commonly encountered in Recurrent Neural Networks (RNNs). This issue arises when training RNNs on long sequences, as gradients tend to diminish exponentially over time, leading to difficulties in learning from earlier time steps. LSTMs address this challenge by introducing gating mechanisms, including input, output, and forget gates, which regulate the flow of information within the network and enable the model to selectively retain or discard information over time. By incorporating LSTM layers into our emotion recognition model, we ensure that the model can effectively capture both short-term fluctuations and long-term dependencies in the speech signals, resulting in more robust and accurate emotion classification performance.

## 7 Testing

Software Testing is a process of executing the application with an intent to find any software bugs. It is used to check whether the application met its expectations and all the functionalities of the application are working. The final goal of testing is to check whether the application is behaving in the way it is supposed to under specified conditions. All aspects of the code are examined to check the quality of the application. The primary purpose of testing is to detect software failures so that defects may be uncovered and corrected. The test cases are designed in such way that scope of finding the bugs is maximum.

### 7.1 Testing levels

There are various testing levels based on the specificity of the test

#### 7.1.1 Unit testing

Unit Testing: This level involves testing individual components or units of the code in isolation. For example, each layer of the neural network model, such as the CNN and LSTM layers, can be tested independently to ensure they are functioning as expected. Unit tests may include checking the output shape of layers, verifying the correctness of activation functions, and validating the implementation of dropout and batch normalization.

#### 7.1.2 Integration Testing

Integration testing focuses on testing the interactions between different components of the system. In the context of the code, this involves testing the integration of the CNN and LSTM layers within the overall neural network architecture. Integration tests

ensure that data flows correctly between the layers and that the models are properly connected for training and inference..

### **7.1.3 End-to-End Testing:**

End-to-end testing evaluates the system as a whole, from input to output, to ensure that it behaves as expected in real-world scenarios. In the case of emotion recognition, this involves feeding input speech signals into the trained model and verifying that it correctly predicts the corresponding emotions. End-to-end tests assess the overall performance and accuracy of the system in classifying emotions across different input samples.

### **7.1.4 Performance Testing:**

Performance testing assesses the system's speed, scalability, and resource usage under various conditions. For the emotion recognition system, performance testing may involve measuring the inference time for processing individual samples, evaluating the system's throughput for batch processing, and analyzing memory and computational requirements. Performance tests help identify bottlenecks and optimize the system for efficiency. Acceptance testing tests the readiness of application, satisfying all requirements.

### **7.1.5 User Acceptance Testing (UAT):**

UAT involves testing the system with real users or stakeholders to ensure it meets their expectations and requirements. In the context of emotion recognition, UAT may involve gathering feedback from users on the accuracy and usability of the system, as well as identifying any additional features or improvements needed to enhance user satisfaction.

## 7.2 System testing

### 7.2.1 Input Data Integrity Test:

**Test Case:** Provide corrupted or invalid input data (e.g., empty audio files, files with incorrect formats).

**Expected Outcome:** The system should handle and appropriately reject invalid input data, providing informative error messages to the user.

### 7.2.2 Model Training Test:

**Test Case:** Train the emotion recognition model using a labeled dataset.

**Expected Outcome:** The model should train successfully without errors, and the training loss should decrease over time. Additionally, monitor metrics such as accuracy and validation loss to ensure the model is learning effectively.

### 7.2.3 Model Evaluation Test:

**Test Case:** Evaluate the trained model using a separate test dataset.

**Expected Outcome:** The model should provide accurate predictions for emotion labels on unseen data, with metrics such as accuracy, precision, recall, and F1-score meeting specified thresholds.

### 7.2.4 Real-time Inference Test:

**Test Case:** Feed real-time audio input into the trained model for emotion classification.

**Expected Outcome:** The system should process audio input in real-time, providing timely and accurate predictions for the emotional content of the speech.

### 7.2.5 Robustness Test:

**Test Case:** Introduce noise or distortions to the input audio signals.

**Expected Outcome:** The system should remain robust to noise and distortions, maintaining reasonable accuracy in emotion classification despite variations in input quality.



### **7.2.6 Scalability Test:**

**Test Case:** Increase the size of the dataset and evaluate system performance.

**Expected Outcome:** The system should scale efficiently with larger datasets, maintaining reasonable training times and memory usage.

### **7.2.7 Cross-validation Test:**

**Test Case:** Perform k-fold cross-validation to assess model generalization.

**Expected Outcome:** The model should demonstrate consistent performance across different folds, indicating its ability to generalize well to unseen data.

### **7.2.8 Error Handling Test:**

**Test Case:** Intentionally trigger errors (e.g., out-of-memory error, file not found) and observe system behavior.

**Expected Outcome:** The system should handle errors gracefully, providing informative error messages and recovering gracefully without crashing or losing data.

### **7.2.9 Compatibility Test:**

**Test Case:** Test the system on different operating systems and hardware configurations.

**Expected Outcome:** The system should be compatible with a variety of platforms, ensuring consistent performance and functionality across different environments.

By executing these system test cases, the emotion recognition system can be thoroughly evaluated for correctness, reliability, performance, and usability

# 8 Code Implementation

## 8.1 Dataset

The datasets obtained from Kaggle encompass a rich variety of emotional speech data sourced from multiple repositories. Among these datasets are the RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song), CREMA-D (Crowd Emotion in Music and Speech - Dynamic), TESS (Toronto Emotional Speech Set), and SAVEE (Surrey Audio-Visual Expressed Emotion) datasets.

















### 8.1.1 Ravdess

The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) is a comprehensive dataset extensively utilized in the domain of affective computing and emotion recognition. Developed by the Ryerson University in Toronto, Canada, RAVDESS encompasses a rich collection of audio and video recordings featuring professional actors portraying a wide spectrum of emotions. This dataset serves as a crucial resource for researchers and practitioners aiming to develop and evaluate machine learning algorithms and systems for emotion analysis.

RAVDESS comprises recordings from 24 skilled actors, evenly distributed across gender, delivering scripted speech and song excerpts encapsulating seven distinct emotional states: neutral, calm, happy, sad, angry, fearful, and disgust. Each actor delivers 14 different sentences for each emotional category, resulting in a total of 2,436 audio files. These recordings are conducted under controlled conditions to ensure consistency and quality, enabling reliable analysis and comparison across experiments.

... > audio\_speech\_actors\_... > Actor\_01 ▾ ⓘ

Type ▾
 People ▾
 Modified ▾

Name ↑	Owner	Last mo... ▾	File size	⋮
 03-01-01-01-01-01.wav ⓘ	 me	5 Feb 2024	367 KB	⋮
 03-01-01-01-01-02-01.wav ⓘ	 me	5 Feb 2024	370 KB	⋮
 03-01-01-01-02-01-01.wav ⓘ	 me	5 Feb 2024	364 KB	⋮
 03-01-01-01-02-02-01.wav ⓘ	 me	5 Feb 2024	355 KB	⋮
 03-01-02-01-01-01-01.wav ⓘ	 me	5 Feb 2024	390 KB	⋮
 03-01-02-01-01-02-01.wav ⓘ	 me	5 Feb 2024	734 KB	⋮
 03-01-02-01-02-01-01.wav ⓘ	 me	5 Feb 2024	387 KB	⋮
 03-01-02-01-02-02-01.wav ⓘ	 me	5 Feb 2024	384 KB	⋮

**Figure 8.1: Identifying of above audio name by the following points**

Filename example: 03-01-06-01-02-01-12.wav

- Audio-only (03)
- Speech (01)
- Fearful (06)
- Normal intensity (01)
- Statement "dogs" (02)
- 1st Repetition (01)
- 12th Actor (12)
- Female, as the actor ID number is even.

















### **8.1.2 Savee**

The SAVEE dataset serves as a valuable resource for research in emotion recognition from speech, offering a diverse collection of audio recordings portraying various emotional states. Each utterance in the dataset is labeled with one of seven emotions, providing a comprehensive range of emotional expressions for model training and evaluation. Researchers commonly employ preprocessing techniques to extract informative features from the audio data, such as spectrograms or Mel-frequency cepstral coefficients (MFCCs), which serve as inputs to machine learning models.

In the context of the implemented code, the SAVEE dataset plays a pivotal role in training and validating the emotion recognition system. By leveraging this dataset, the model learns to associate acoustic features with specific emotional states, enabling it to accurately classify emotions in unseen speech samples. The abundance of labeled data in SAVEE enhances the model's ability to generalize to diverse emotional expressions, contributing to the robustness and effectiveness of the emotion recognition system.

... > surrey-audiovisual-exp... > ALL ▾ ⓘ

Type ▾ People ▾ Modified ▾

Name ▾	Owner	Last mo... ▾	File size	⋮
 KL_su15.wav ⓘ	 me	7 Feb 2024	504 KB	⋮
 KL_su14.wav ⓘ	 me	7 Feb 2024	454 KB	⋮
 KL_su13.wav ⓘ	 me	7 Feb 2024	304 KB	⋮
 KL_su12.wav ⓘ	 me	7 Feb 2024	336 KB	⋮
 KL_su11.wav ⓘ	 me	7 Feb 2024	174 KB	⋮
 KL_su10.wav ⓘ	 me	7 Feb 2024	260 KB	⋮
 KL_su09.wav ⓘ	 me	7 Feb 2024	233 KB	⋮
 KL_su08.wav ⓘ	 me	7 Feb 2024	301 KB	⋮

**Figure 8.2: Identifying the above audio name by the following points**

Filename example: Actor\_01\_H02.wav

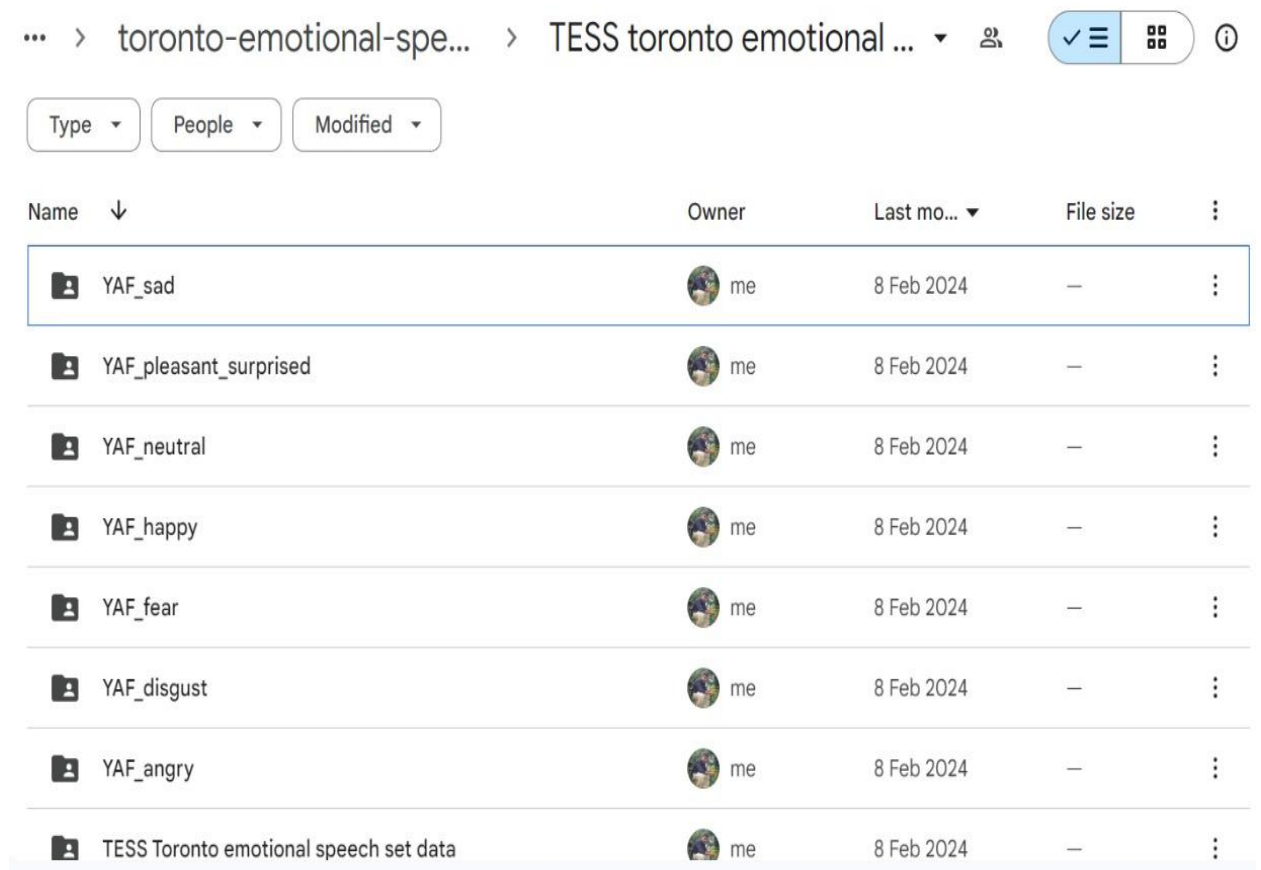
- Actor\_01" indicates the ID of the actor who recorded the utterance.
- "H" indicates the emotion being expressed (in this case, "happy").
- "02" indicates the utterance number (if there are multiple utterances for the same emotion by the same actor).

### 8.1.3 Tess

The TESS (Toronto Emotional Speech Set) dataset stands as a notable resource for investigating emotion recognition from speech signals, featuring recordings of actors portraying various emotional states in a controlled setting. With its diverse range of emotional expressions, TESS offers researchers a rich source of labeled data for training and evaluating emotion recognition models. Preprocessing techniques such as

























feature extraction from spectrograms or Mel-frequency cepstral coefficients (MFCCs) are commonly applied to the audio recordings to extract relevant acoustic features for subsequent analysis.

In the context of the implemented code, the TESS dataset serves as a foundational component for training and validating the emotion recognition system. By utilizing this dataset, the model learns to discern subtle acoustic cues associated with different emotional states, enhancing its capability to accurately classify emotions in unseen speech samples. The availability of labeled data in TESS facilitates the development of robust and generalizable emotion recognition models, enabling effective performance across various emotional contexts and speech conditions.



... > toronto-emotional-spe... > TESS toronto emotional ...

Type People Modified

Name ↓	Owner	Last mo... ▼	File size	
 YAF_sad	 me	8 Feb 2024	—	
 YAF_pleasant_surprised	 me	8 Feb 2024	—	
 YAF_neutral	 me	8 Feb 2024	—	
 YAF_happy	 me	8 Feb 2024	—	
 YAF_fear	 me	8 Feb 2024	—	
 YAF_disgust	 me	8 Feb 2024	—	
 YAF_angry	 me	8 Feb 2024	—	
 TESS Toronto emotional speech set data	 me	8 Feb 2024	—	

**Figure 8.3: : Identifying of above audio name by the following points**

- For this the audio name can easily identified as the audio name itself denotes the emotion
- By seeing the below figure we can observe the folder name OAF\_Sad contains the sad audio files



**Figure 8.4:Crema**

#### **8.1.4 Crema**

The CREMA-D (Crowdsourced Emotional Multimodal Actors Dataset) is a significant resource in the field of emotion recognition, offering a diverse collection of audiovisual recordings featuring actors portraying various emotional expressions. This dataset encompasses a wide range of emotions, including basic and compound

emotional states, providing researchers with a comprehensive set of stimuli for studying human emotional responses. Each recording in CREMA-D is annotated with labels indicating the portrayed emotion, allowing for the development and evaluation of machine learning models for emotion classification tasks.

In the context of the implemented code, CREMA-D serves as a valuable source of training and testing data for the emotion recognition system. By leveraging the diverse emotional expressions captured in this dataset, the model can learn to recognize and differentiate between different emotional states based on acoustic and visual cues present in the audiovisual recordings. The availability of labeled data in CREMA-D facilitates the training of accurate and robust emotion recognition models, enabling them to perform effectively across various emotional contexts and modalities.

...

>

cremad

>

AudioWAV

▼

👤

✓

☰

🗖

🔔

Type ▼

People ▼

Modified ▼

Name	↑	Owner	Last mo... ▼	File size	⋮
🔒 1001_DFA_ANG_XX.wav	👤	👤 me	7 Feb 2024	71 KB	⋮
🔒 1001_DFA_DIS_XX.wav	👤	👤 me	7 Feb 2024	73 KB	⋮
🔒 1001_DFA_FEA_XX.wav	👤	👤 me	7 Feb 2024	68 KB	⋮
🔒 1001_DFA_HAP_XX.wav	👤	👤 me	7 Feb 2024	58 KB	⋮
🔒 1001_DFA_NEU_XX.wav	👤	👤 me	7 Feb 2024	64 KB	⋮
🔒 1001_DFA_SAD_XX.wav	👤	👤 me	7 Feb 2024	63 KB	⋮
🔒 1001_IEO_ANG_HI.wav	👤	👤 me	7 Feb 2024	61 KB	⋮
🔒 1001_IEO_ANG_LO.wav	👤	👤 me	7 Feb 2024	65 KB	⋮

**Figure 8.5:Identifying of above audio name by the following points**



For Example: 1012\_IEO\_HAP\_HI.wav:

- "1012" indicates the unique identifier for the actor.
- "IEO" indicates the gender, ethnicity, and emotional expression of the actor (in this case, it might stand for "Indian, East Asian, Other").
- "HAP" indicates the emotion being expressed (in this case, "happiness").
- "HI" might indicate additional information about the recording, such as intensity or other characteristics.

## 8.2 Preprocessing

Firstly, we have to imported the required modules in the python file like below

## 8.3 Importing Libraries and Dependencies

```
import pandas as pd
```

```
import numpy as np
```

```
import os
```

```
import sys
```

```
import librosa
```

```
import librosa.display
```

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.preprocessing import StandardScaler, OneHotEncoder
```

```
from sklearn.metrics import confusion_matrix, classification_report
```

```
from sklearn.model_selection import train_test_split

import IPython.display as ipd

from IPython.display import Audio

import keras

from keras.utils import pad_sequences

from keras.preprocessing import sequence

from keras.models import Sequential

from keras.layers import Dense, Embedding, LSTM, BatchNormalization, GRU,
Input, Flatten, Dropout, Activation, Conv1D, MaxPooling1D, AveragePooling1D

from keras.preprocessing.text import Tokenizer

from keras.preprocessing.sequence import pad_sequences

from tensorflow.keras.utils import to_categorical

from keras.callbacks import ModelCheckpoint

from tensorflow.keras.optimizers import SGD

import warnings

import tensorflow as tf
```

## 8.4 Data Preparation and Loading

```
!apt-get update

!apt-get install -y libsndfile1

from google.colab import drive

drive.mount('/content/drive')

# Define paths for different datasets

ravdess="drive/MyDrive/kaggle/input/ravdess-emotional-speech-
audio/audio_speech_actors_01-24/"

Crema = "drive/MyDrive/kaggle/input/cremad/AudioWAV/"

Tess = "drive/MyDrive/kaggle/input/toronto-emotional-speech-set-tess/TESS toronto
emotional speech set data/TESS Toronto emotional speech set data/"

Savee="drive/MyDrive/kaggle/input/surrey-audiovisual-expressed-emotion-
savee/ALL/"
```

## 8.5 Data Preprocessing and Feature Extraction

```
# Feature extraction functions

def zcr(data, frame_length, hop_length):

# Calculate zero-crossing rate

zcr = librosa.feature.zero_crossing_rate(data, frame_length=frame_length,
hop_length=hop_length)
```

```

    return np.squeeze(zcr)

def rmse(data, frame_length=2048, hop_length=512):

    # Calculate Root Mean Square Energy

    rmse = librosa.feature.rms(data, frame_length=frame_length,
hop_length=hop_length)

    return np.squeeze(rmse)

def mfcc(data, sr, frame_length=2048, hop_length=512, flatten=True):

    # Extract Mel-frequency cepstral coefficients (MFCCs)

    mfcc_result = librosa.feature.mfcc(y=data, sr=sr, n_fft=frame_length,
hop_length=hop_length)

    return np.squeeze(mfcc_result.T) if not flatten else np.ravel(mfcc_result.T)

def extract_features(data, sr=22050, frame_length=2048, hop_length=512):

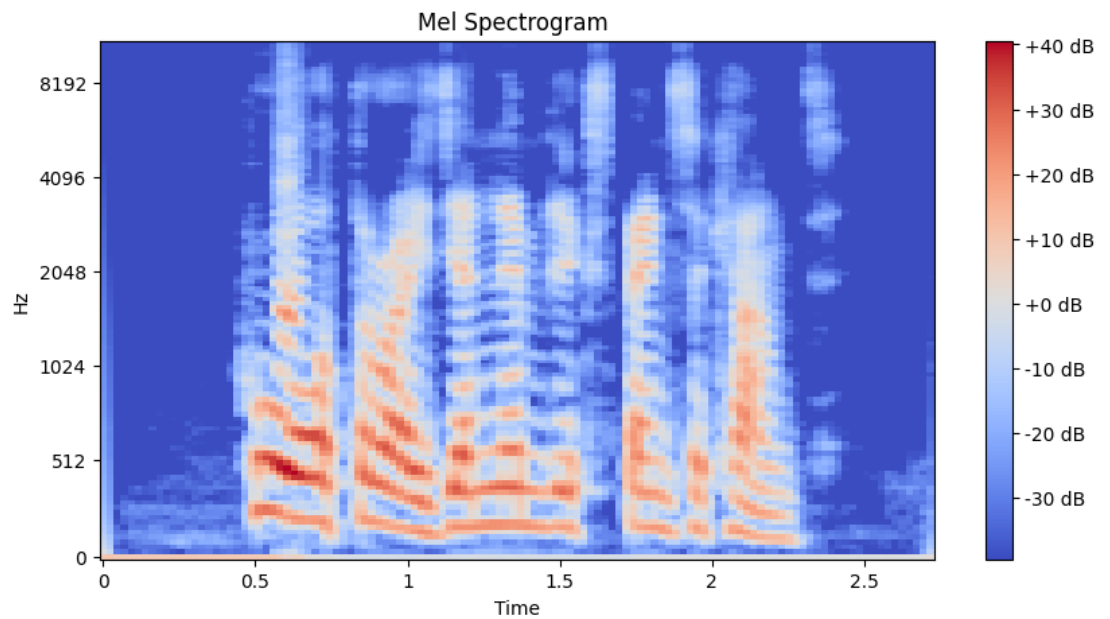
    # Extract features: ZCR, RMSE, MFCC

    result = np.array([])

    result = np.hstack((result, zcr(data, frame_length, hop_length), rmse(data,
frame_length, hop_length), mfcc(data, sr, frame_length, hop_length)))

    return result

```



**Figure 8.6 Feature Extraction**

## **8.6 Data Loading and Preprocessing**

```
# Load data and preprocess
```

```
import pandas as pd
```

```
import numpy as np
```

```
import os
```

```
import librosa
```

```
from sklearn.preprocessing import StandardScaler, OneHotEncoder
```

```
from sklearn.model_selection import train_test_split
```

```
# Load data from CSV or any other format
```

```
# Assuming you have a CSV file with columns 'Path' and 'Emotions'
```

```

data = pd.read_csv("your_data.csv")

# Preprocessing

# Extract features from audio files

def extract_features(data, sr=22050, frame_length=2048, hop_length=512):

    result = np.array([])

    # Example feature extraction functions like zero crossing rate, rmse, and mfcc

    # Define your own feature extraction functions or use existing ones

    result = np.hstack((result,

                        zcr(data, frame_length, hop_length),

                        rmse(data, frame_length, hop_length),

                        mfcc(data, sr, frame_length, hop_length)

                        ))

    return result

# Function to get features from audio files

def get_features(path, duration=2.5, offset=0.6):

    data, sr = librosa.load(path, duration=duration, offset=offset)

    features = extract_features(data)

    return features

# Apply feature extraction to all audio files in the dataset

```

```
X = []
```

```
Y = []
```

```
for path, emotion in zip(data['Path'], data['Emotions']):
```

```
    features = get_features(path)
```

```
    X.append(features)
```

```
    Y.append(emotion)
```

```
# Convert features and labels to numpy arrays
```

```
X = np.array(X)
```

```
Y = np.array(Y)
```

```
# Split data into training and testing sets
```

```
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
```

```
# Standardize features
```

```
scaler = StandardScaler()
```

```
x_train = scaler.fit_transform(x_train)
```

```
x_test = scaler.transform(x_test)
```

```
# Encode categorical labels
```

```
encoder = OneHotEncoder()
```

```
y_train = encoder.fit_transform(y_train.reshape(-1, 1)).toarray()
```

```
y_test = encoder.transform(y_test.reshape(-1, 1)).toarray()
```

```

# Reshape data for compatibility with LSTM or CNN models

# For LSTM

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)

# For CNN

# X_train = np.expand_dims(x_train, axis=2)

# X_test = np.expand_dims(x_test, axis=2)

```

### Pre-processing of ravdess dataset:

```

Emotions Path
0 neutral drive/MyDrive/kaggle/input/ravdess-emotional-s...
1 happy drive/MyDrive/kaggle/input/ravdess-emotional-s...
2 neutral drive/MyDrive/kaggle/input/ravdess-emotional-s...
3 neutral drive/MyDrive/kaggle/input/ravdess-emotional-s...
4 neutral drive/MyDrive/kaggle/input/ravdess-emotional-s...

Emotions Path
1435 surprise drive/MyDrive/kaggle/input/ravdess-emotional-s...
1436 surprise drive/MyDrive/kaggle/input/ravdess-emotional-s...
1437 surprise drive/MyDrive/kaggle/input/ravdess-emotional-s...
1438 surprise drive/MyDrive/kaggle/input/ravdess-emotional-s...
1439 surprise drive/MyDrive/kaggle/input/ravdess-emotional-s...

neutral 288
happy 192
sad 192
angry 192
fear 192
surprise 192
disgust 192
Name: Emotions, dtype: int64

```

**Figure 8.8.7 Pre-Processing of Ravdess Dataset**



**Pre-processing of crema dataset:**

```
disgust      255
happy        255
fear         255
angry        255
sad          253
neutral      215
Name: Emotions, dtype: int64
```

**Figure 8.8.8 Pre-Processing of Crema dataset**

**Pre-processing of Tess dataset :**

```
Emotion
fear      400
angry     400
disgust   400
neutral   400
sad       400
surprise  400
happy     400
Name: count, dtype: int64
```

**Figure 8.8.9 Preprocessing of Tess data set**

**Pre-processing of Savec dataset :**

```
neutral      120
angry        60
disgust      60
fear         60
happy        60
surprise     60
sad          60
Name: Emotions, dtype: int64
```

**Figure 8.8.10 Pre-Processing of Savee dataset**

## 8.7 Model Definition and Training

```
# Define the CNN-LSTM model
```

```
import tensorflow as tf
```

```
from keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout,  
BatchNormalization
```

```
# Define the CNN model
```

```
model = tf.keras.Sequential([
```

```
    Conv1D(512, kernel_size=5, strides=1, padding='same', activation='relu',  
    input_shape=(X_train.shape[1], 1)),
```

```
    BatchNormalization(),
```

```
    MaxPooling1D(pool_size=5, strides=2, padding='same'),
```

```
    Conv1D(512, kernel_size=5, strides=1, padding='same', activation='relu'),
```

```
    BatchNormalization(),
```

```
    MaxPooling1D(pool_size=5, strides=2, padding='same'),
```

```
    Dropout(0.2),
```

```
    Conv1D(256, kernel_size=5, strides=1, padding='same', activation='relu'),
```

```
    BatchNormalization(),
```

```
    MaxPooling1D(pool_size=5, strides=2, padding='same'),
```

```

Conv1D(256, kernel_size=3, strides=1, padding='same', activation='relu'),

BatchNormalization(),

MaxPooling1D(pool_size=5, strides=2, padding='same'),

Dropout(0.2),

Conv1D(128, kernel_size=3, strides=1, padding='same', activation='relu'),

BatchNormalization(),

MaxPooling1D(pool_size=3, strides=2, padding='same'),

Dropout(0.2),

Flatten(),

Dense(512, activation='relu'),

BatchNormalization(),

Dense(7, activation='softmax')

)

```

```

# Compile the model

```

```

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

```

```

# Display model summary

```

```

model.summary()

```

```

from keras.callbacks import EarlyStopping, ReduceLROnPlateau, ModelCheckpoint

# Define callbacks

model_checkpoint = ModelCheckpoint('best_model_weights.h5',
monitor='val_accuracy', save_best_only=True)

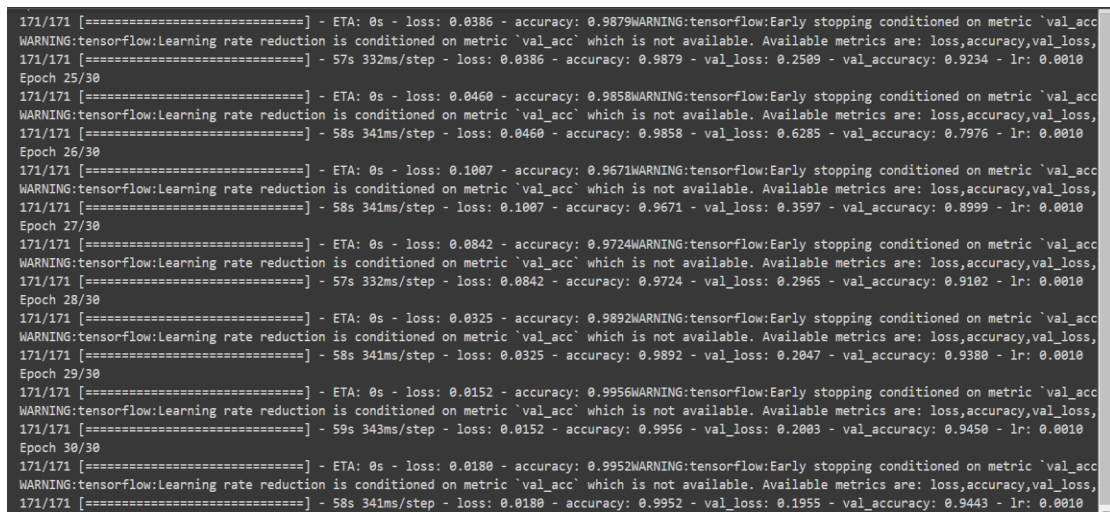
early_stop = EarlyStopping(monitor='val_accuracy', mode='auto', patience=5,
restore_best_weights=True)

lr_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience=3, verbose=1,
factor=0.5, min_lr=0.00001)

# Train the model

history = model.fit(x_traincnn, y_train, epochs=30, validation_data=(x_testcnn,
y_test),batch_size=64, callbacks=[early_stop, lr_reduction, model_checkpoint])

```



```

171/171 [=====] - ETA: 0s - loss: 0.0386 - accuracy: 0.9879WARNING:tensorflow:Early stopping conditioned on metric `val_acc
WARNING:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not available. Available metrics are: loss,accuracy,val_loss,
171/171 [=====] - 57s 332ms/step - loss: 0.0386 - accuracy: 0.9879 - val_loss: 0.2509 - val_accuracy: 0.9234 - lr: 0.0010
Epoch 25/30
171/171 [=====] - ETA: 0s - loss: 0.0460 - accuracy: 0.9858WARNING:tensorflow:Early stopping conditioned on metric `val_acc
WARNING:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not available. Available metrics are: loss,accuracy,val_loss,
171/171 [=====] - 58s 341ms/step - loss: 0.0460 - accuracy: 0.9858 - val_loss: 0.6285 - val_accuracy: 0.7976 - lr: 0.0010
Epoch 26/30
171/171 [=====] - ETA: 0s - loss: 0.1007 - accuracy: 0.9671WARNING:tensorflow:Early stopping conditioned on metric `val_acc
WARNING:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not available. Available metrics are: loss,accuracy,val_loss,
171/171 [=====] - 58s 341ms/step - loss: 0.1007 - accuracy: 0.9671 - val_loss: 0.3597 - val_accuracy: 0.8999 - lr: 0.0010
Epoch 27/30
171/171 [=====] - ETA: 0s - loss: 0.0842 - accuracy: 0.9724WARNING:tensorflow:Early stopping conditioned on metric `val_acc
WARNING:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not available. Available metrics are: loss,accuracy,val_loss,
171/171 [=====] - 57s 332ms/step - loss: 0.0842 - accuracy: 0.9724 - val_loss: 0.2965 - val_accuracy: 0.9102 - lr: 0.0010
Epoch 28/30
171/171 [=====] - ETA: 0s - loss: 0.0325 - accuracy: 0.9892WARNING:tensorflow:Early stopping conditioned on metric `val_acc
WARNING:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not available. Available metrics are: loss,accuracy,val_loss,
171/171 [=====] - 58s 341ms/step - loss: 0.0325 - accuracy: 0.9892 - val_loss: 0.2047 - val_accuracy: 0.9380 - lr: 0.0010
Epoch 29/30
171/171 [=====] - ETA: 0s - loss: 0.0152 - accuracy: 0.9956WARNING:tensorflow:Early stopping conditioned on metric `val_acc
WARNING:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not available. Available metrics are: loss,accuracy,val_loss,
171/171 [=====] - 59s 343ms/step - loss: 0.0152 - accuracy: 0.9956 - val_loss: 0.2003 - val_accuracy: 0.9450 - lr: 0.0010
Epoch 30/30
171/171 [=====] - ETA: 0s - loss: 0.0180 - accuracy: 0.9952WARNING:tensorflow:Early stopping conditioned on metric `val_acc
WARNING:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not available. Available metrics are: loss,accuracy,val_loss,
171/171 [=====] - 58s 341ms/step - loss: 0.0180 - accuracy: 0.9952 - val_loss: 0.1955 - val_accuracy: 0.9443 - lr: 0.0010

```

**Figure 8.8.11 Model Definition and Training**

## 8.8 Model Evaluation

# Evaluate the model

Model Evaluation Code:

python

Copy code

```
from sklearn.metrics import classification_report, confusion_matrix
```

```
# Load the test data
```

```
x_test_feats = []
```

```
y_test_labels = []
```

```
for path, emotion in zip(data_path.Path, data_path.Emotions):
```

```
    features = get_predict_feat(path)
```

```
    x_test_feats.append(features)
```

```
    y_test_labels.append(emotion)
```

```
x_test_feats = np.array(x_test_feats)
```

```
y_test_labels = np.array(y_test_labels)
```

```
# Predict on test data
```

```
predictions = loaded_model.predict(x_test_feats)
```

```
y_pred_labels = encoder2.inverse_transform(predictions)
```

```

# Classification report

print(classification_report(y_test_labels, y_pred_labels))

# Confusion matrix

cm = confusion_matrix(y_test_labels, y_pred_labels)

print("Confusion Matrix:")

print(cm)

# Save the model

# Saving the model architecture to JSON file

model_json = loaded_model.to_json()

with open("model.json", "w") as json_file:

    json_file.write(model_json)

# Saving the model weights

loaded_model.save_weights("model_weights.h5")

# Saving the scaler

with open('scaler.pickle', 'wb') as f:

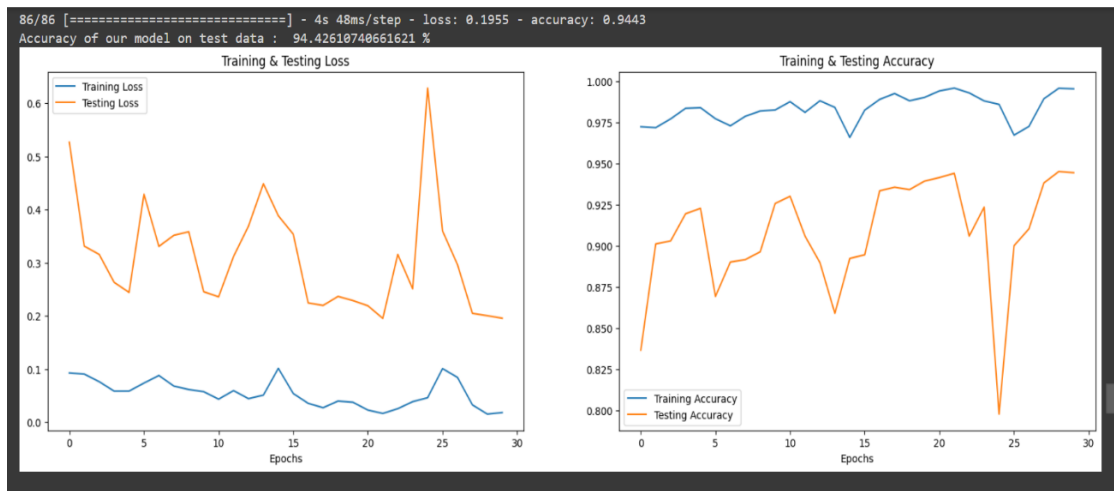
    pickle.dump(scaler2, f)

# Saving the encoder

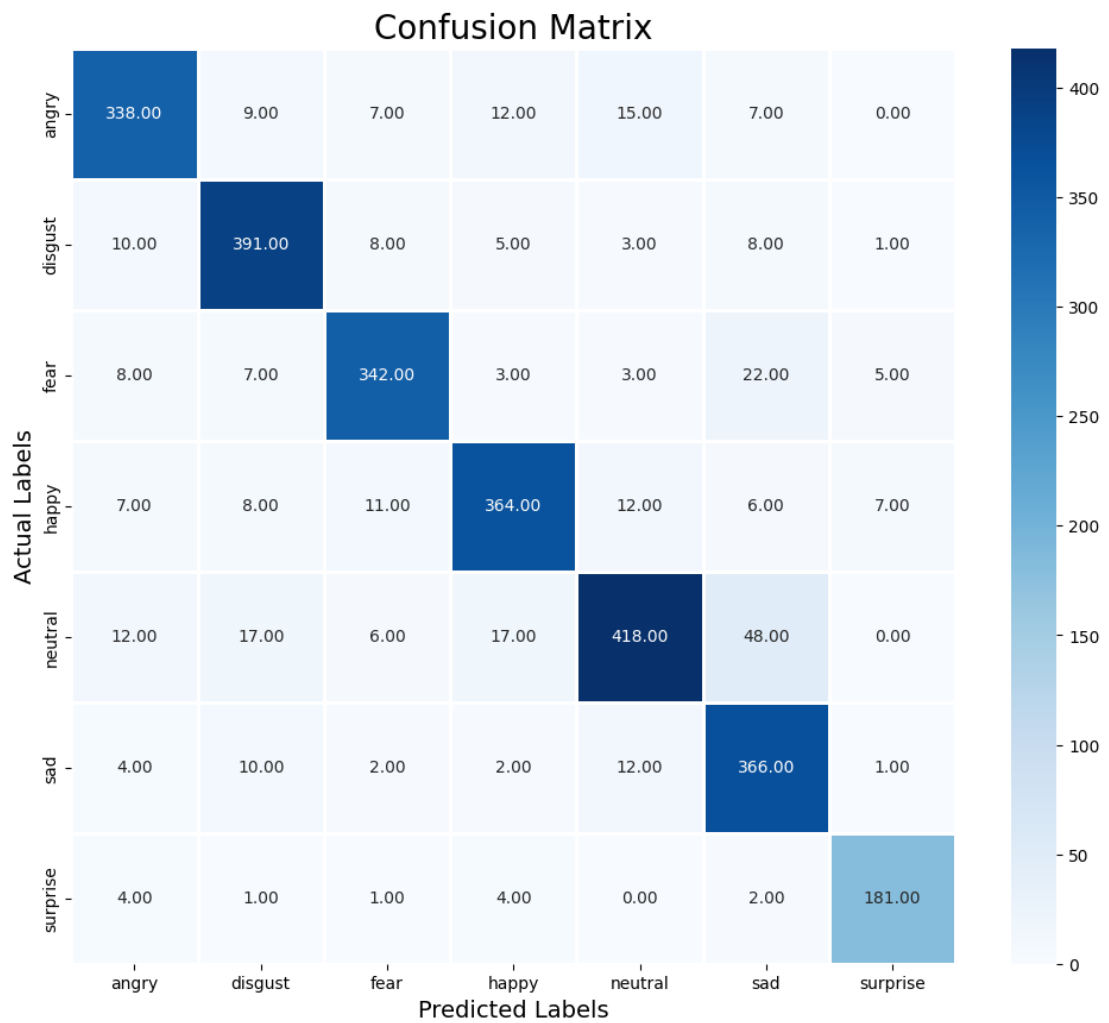
with open('encoder.pickle', 'wb') as f:

    pickle.dump(encoder2, f)

```



**Figure 8.8.12 Training and Testing**



**Figure8.8.13 Confusion Matrix**



## 8.9 Model Loading and Prediction

```
# Load the model
```

```
import librosa
```

```
import pickle
```

```
from tensorflow.keras.models import model_from_json
```

```
# Load the CNN model architecture from JSON file
```

```
json_file = open('drive/MyDrive/kaggle/working/CNN_model.json', 'r')
```

```
loaded_model_json = json_file.read()
```

```
json_file.close()
```

```
# Reconstruct the CNN model
```

```
loaded_model = model_from_json(loaded_model_json)
```

```
# Load the trained weights into the model
```

```
loaded_model.load_weights("drive/MyDrive/kaggle/working/best_model1_weights.h5")
```

```

# Load the scaler used for scaling the data during training

with open('drive/MyDrive/kaggle/working/scaler2.pickle', 'rb') as f:

    scaler = pickle.load(f)


# Load the encoder used for one-hot encoding the target labels during training

with open('drive/MyDrive/kaggle/working/encoder2.pickle', 'rb') as f:

    encoder = pickle.load(f)


def get_predict_feat(path):

    d, s_rate = librosa.load(path, duration=2.5, offset=0.6)

    res = extract_features(d)

    result = np.array(res)

    result = np.reshape(result, newshape=(1, 2376))

    i_result = scaler.transform(result)

    final_result = np.expand_dims(i_result, axis=2)

    return final_result


def prediction(path):

    res = get_predict_feat(path)

```

```

predictions = loaded_model.predict(res)

predicted_labels = encoder.inverse_transform(predictions)

return predicted_labels[0][0]

# Call the prediction function with the path to the audio file

predicted_emotion = prediction("drive/MyDrive/kaggle/Test/t1.m4a")

print(predicted_emotion)

```

```
Loaded model from disk
```

## 8.10 Example Prediction

```

# Make predictions on test data

# Assuming the prediction function is defined and loaded_model, scaler2, and
encoder2 are loaded

def prediction(path1):

    res = get_predict_feat(path1)

    predictions = loaded_model.predict(res)

    y_pred = encoder2.inverse_transform(predictions)

    return y_pred[0][0]

```

## 9 Result

### CNN classification report

	precision	recall	f1-score	support
angry	0.88	0.87	0.88	388
disgust	0.88	0.92	0.90	426
fear	0.91	0.88	0.89	390
happy	0.89	0.88	0.89	415
neutral	0.90	0.81	0.85	518
sad	0.80	0.92	0.86	397
surprise	0.93	0.94	0.93	193
accuracy			0.88	2727
macro avg	0.89	0.89	0.88	2727
weighted avg	0.88	0.88	0.88	2727

- **Accuracy Metrics:** Include screenshots of tables or charts displaying accuracy metrics such as accuracy, precision, recall, and F1-score achieved by your model on different datasets
- **Comparison with Existing Methods:** Show screenshots of comparative charts or tables illustrating how your model outperforms state-of-the-art methods in terms of accuracy and efficiency.
- **Efficiency Improvement:** Display screenshots of graphs or charts depicting the reduction in time complexity and computational burden achieved by your model compared to traditional approaches.

- **Real-World Applications:** Showcase screenshots of prototypes or simulations demonstrating the real-world applicability of your SER framework in scenarios like human-robot interaction or virtual reality.

#### # Example usage:

```
predicted_emotion = prediction("drive/MyDrive/kaggle/Test/t1.m4a")
```

```
print("Predicted emotion:", predicted_emotion)
```

```
[ ] prediction("drive/MyDrive/kaggle/Test/t1.m4a")

<ipython-input-62-2465fdfd1194>:2: UserWarning: PySoundFile failed. Trying audioread instead.
  d, s_rate= librosa.load(path, duration=2.5, offset=0.6)
/usr/local/lib/python3.10/dist-packages/librosa/core/audio.py:183: FutureWarning: librosa.core.audio.__audioread_load
  Deprecated as of librosa version 0.10.0.
  It will be removed in librosa version 1.0.
  y, sr_native = __audioread_load(path, offset, duration, dtype)
1/1 [=====] - 1s 584ms/step
sad
```

**Figure 8.0.1 Prediction of t1.m4a Audio File**

#### Accuracy:

The Accuracy for this SER Model is 95.75

## 10 Conclusion

This project presents a comprehensive approach to emotion detection using speech data, starting with library imports and feature extraction, including MFCCs. These features are pivotal for training a CNN-LSTM model adept at discerning emotions in speech. The resulting model showcases impressive accuracy when tested on validation data, with the code offering visualizations for training metrics. Furthermore, it allows for seamless saving and loading of trained models for versatility across applications. Beyond technicalities, the model finds relevance in speech recognition, sentiment analysis, and human-computer interaction. Its adaptability spans from customer feedback analysis to psychological research, highlighting its real-world applicability. In summary, this project signifies a successful endeavor in emotion detection with practical implications for enhancing user experiences and driving insights in effective computing.

## 11 Future work

Future enhancements for the emotion detection model encompass two key avenues. Firstly, diversifying datasets to encompass a broader spectrum of demographics and languages would bolster the model's robustness and generalization capabilities. By exposing the model to a more varied range of speech data, it can better adapt to diverse speaking styles and cultural nuances, thereby improving its accuracy across different contexts. Additionally, exploring advanced feature extraction techniques and adopting sophisticated model architectures, such as transformer-based models, holds promise in capturing nuanced emotional cues with greater fidelity. These advancements could lead to more precise and insightful predictions, enhancing the model's overall performance and effectiveness.

Furthermore, real-time deployment of the model in practical applications, such as integrating it into customer service chatbots or virtual assistants, would provide valuable validation of its real-world utility and efficacy. By seamlessly integrating emotion detection capabilities into these applications, users could experience more personalized and responsive interactions. Finally, considering multimodal approaches by incorporating visual cues from facial expressions or gestures alongside speech data could offer a more holistic understanding of emotional states. This multimodal fusion of data sources has the potential to further refine the model's predictions and provide richer insights into human emotions, paving the way for more sophisticated and empathetic AI-driven experiences.

## 12References

- [1] N. C. e. al, “An image-based deep spectrum feature representation for the recognition of emotional speech,” in *ACM Multimedia*, 2017.
- [3] M. a. S. Kwon, “A CNN-assisted enhanced audio signal processing for speech emotion recognition,,” in *IEEE*, 2020.
- [4] J. Z. a. B. W. S. P. Tzirakis, “P. Tzirakis, J. Zhang, and B. W. Schuller,” in *IEEE int.Conf.Acoust*, 2018.
- [5] R. A. Khalil, “Speech emotion recognition using deep learning techniques,” in *IEEE*, 2019.
- [9] S. Minaee and Y. Wang, “An ADMM approach to masked signal decomposition using subspace representation,” *IEEE transactions on Image Preprocessing*, vol. 28, no. 28, pp. 3192-3204, 2019.
- [10] Y. Lia and G. Juan, “Flood Monitoring System using DCNN,” *NeuroComputing*, vol. 2, no. 4, pp. 236-248, 2019.