

RECOGNITION OF EMOTIONAL SPEECH USING MFCC AND MACHINE LEARNING TECHNIQUE

A Project Report

Submitted to the Faculty of Engineering of
**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA,
KAKINADA**

In partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY
In
COMPUTER SCIENCE AND ENGINEERING

By

T.DIVYA
(20481A05N3)

SK.SUBHANI
(20481A05L6)

S.MOUNIKA
(20481A05M2)

T.VAMSI NANDAN
(20481A05N5)

Under the guidance of
Mrs.K.NANDINI, M.Tech(Ph.D.)
Assistant Professor of CSE Department



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE
(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)
SESHADRI RAO KNOWLEDGE VILLAGE
GUDLAVALLERU – 521356
ANDHRA PRADESH
2023-2024

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)

SESHADRI RAO KNOWLEDGE VILLAGE, GUDLAVALLERU

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project report entitled **“RECOGNITION OF EMOTIONAL SPEECH USING MFCC AND MACHINE LEARNING TECHNIQUE”** is a bonafide record of work carried out by **T. DIVYA (20481A05N3), SK. SUBHANI (20481A05L6), S. MOUNIKA (20481A05M2), T. VAMSI NANDHAN (20481A05N5)** under the guidance and supervision in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Jawaharlal Nehru Technological University Kakinada, Kakinada during the academic year 2023-24.

Project Guide
(Mrs.K.NANDINI)

Head of the Department
(Dr. M. BABU RAO)

External Examiner

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We would like to express our deep sense of gratitude and sincere thanks to **Mrs. K. Nandini, Assistant Professor**, Department of Computer Science and Engineering for her constant guidance, supervision and motivation in completing the project work.

We feel elated to express our floral gratitude and sincere thanks to **Dr. M. Babu Rao**, Head of the Department, Computer Science and Engineering for his encouragement all the way during analysis of the project. His annotations, insinuations and criticisms are the key behind the successful completion of the project work.

We would like to take this opportunity to thank our beloved principal **Dr. B. Karuna Kumar** for providing a great support for us in completing our project and giving us the opportunity for doing the project.

Our Special thanks to the faculty of our department and programmers of our computer lab. Finally, we thank our family members, non-teaching staff and our friends, who had directly or indirectly helped and supported us in completing our project in time.

Team members

T. Divya	(20481A05N3)
Sk. Subhani	(20481A05L6)
S. Mounika	(20481A05M2)
T. Vamsi Nandhan	(20481A05N5)

INDEX

Title	Page No
LIST OF ABBREVIATIONS	I
LIST OF FIGURES	II
ABSTRACT	IV
CHAPTER 1: INTRODUCTION	1-3
1.1 INTRODUCTION	1
1.2 OBJECTIVES OF THE PROJECT	2
1.2.1 Enhancing Human-Computer Interaction(HCI)	2
1.2.2 Developing Assistive Technologies	2
1.2.3 Improving Psychological Analysis	2
1.2.4 Advancing Human-Robot Interaction	2
1.2.5 Enabling Sentiment Analysis in Social Media	3
1.2.6 Facilitating Emotional Intelligence Applications	3
1.3 PROBLEM STATEMENT	3
CHAPTER 2: LITERATURE REVIEW	4
CHAPTER 3: PROPOSED METHOD	5-36
3.1 METHODOLOGY	6-14
3.1.1 Data Set & Data Visualization	6
3.1.2 Pre-processing	7
3.1.3 Feature Extraction	9
3.1.4 Feature Selection	10
3.1.5 Training and Testing Samples	11
3.1.6 Training Dictionary	11
3.1.7 Deep Learning Technique(CNN)	12
3.1.7.1 Convolutional Neural Networks	12
3.1.8 Machine Learning Techniques(SVM, Random Forest)	13
3.1.9 Classification	14
3.1.10 Emotion Recognition	14
3.2 IMPLEMENTATION	15-33
3.2.1 Setup google collab for code execution	15

3.2.2 Data Collection	16
3.2.3 Loading Data	18
3.2.4 Data Visualization	20
3.2.5 Feature Extraction	20
3.2.6 Displaying Frequencies of Different types of datasets	21
3.2.7 Exposing Dataset	25
3.2.8 Training and Testing	26
3.2.9 MLP Classifier	26
3.2.10 Data Pre-processing	26
3.2.11 Model Selection	27
3.2.12 Model Training	29
3.2.13 Model Evaluation	30
3.2.14 Convolutional Neural Network	31
3.2.15 Support Vector Machine	32
3.2.16 Random Forest Classifier	33
3.3 DATA PREPARATION	35-36
CHAPTER 4: RESULTS AND DISCUSSION	37-41
4.1 Convolutional Neural Network(CNN)	37
4.2 Decision Tree Classifier	38
4.3 MLP Classifier	39
4.4 Support Vector Machine(SVM)	39
4.5 Random Forest	40
4.5.1 Robustness to overfitting	40
4.5.2 Feature Importance	40
4.5.3 Out-of-bag (OOB) error estimation	40
4.6 The Heat Map For Labeled MFCC Coefficients	41
CHAPTER 5: CONCLUSION AND FUTURE SCOPE	42-43
5.1 CONCLUSION	42
5.2 FUTURE SCOPE	43
BIBLIOGRAPHY	44
Program Outcomes and Program Specific Outcomes	45-47
Published paper	48

LIST OF ABBREVIATIONS

Abbreviation	Explanation
MFCC	Mel Frequency Cepstral Coefficients
SER	Speech Emotion Recognition
CNN	Convolutional Neural Networks
SVM	Support Vector Machine
MLP	Multi Layer Perceptron
ASR	Automatic Speech Recognition
GMM	Gaussian mixture model
AR	Auto Regressive model
RMS	Root Mean Square
RNN	Recurrent Neural Networks
RBF	Radial Bias Function
RAVDESS	Ryerson Audio-Visual Database of Emotional Speech and Song
TESS	Toronto Emotional Speech Set
CREMA-D	Crowd Sourced Emotional Multimodel Actors Dataset

LIST OF FIGURES

Figure No.	Description	Page No.
1	Speech Emotion Recognition	6
2	Waveform for the fear emotion	6
3	3D Waveform for fear emotion	7
4	MFCC Coefficients of each audio file	10
5	Distribution of Emotions	11
6	UML Diagram of Speech Emotion Recognition	13
7	Count of Female and Male Emotions	14
8	Create a new notebook in google colab	15
9	Colab Interface	15
10	Import drive from google colab	16
11	Import all the packages	16
12	RAVDESS Dataset	17
13	TESS Dataset	17
14	CREMA-D Dataset	17
15	Loading Data	18
16	Create a list for emotions of CREMA-D data	18
17	Create a list for emotions of RAVDESS data	19
18	Create a list for emotions of TESS data	19
19	Combine all data frames	20
20	Save the data and visualize using barplot	20
21	Barplot representation of 8 Emotions	20
22	Feature Extraction	21
23	Feature Normalization	21
24	Normal Audio	22
25	Audio with Noise	22

26	Shifted Audio	23
27	Stretched Audio	23
28	Audio with Pitch	24
29	Audio with High Speed	24
30	Audio with Lower Speed	25
31	Exposing dataset	25
32	One hot encoding	25
33	Training and Testing	26
34	MLP Classifier	26
35	Applying CNN to train and test data	26
36	Model Selection	28
37	Model Training	29
38	Model Evaluation	31
39	Confusion Matrix for CNN Model	31
40	Classification report(CNN)	32
41	Confusion Matrix for SVM Model	33
42	Classification report(SVM)	33
43	Random Forest Classifier Model	34
44	CNN Model Using Deep Frequency Features	37
45	Confusion Matrix (CNN)	38
46	CNNs Classification report	38
47	MLP Classifier Model	39
48	SVMs Classification report	39
49	Random Forest Model	41
50	Heat map for labeled MFCC coefficients	41

ABSTRACT

Speech to text conversion and interpreting a person's emotions from their speech are essential tasks in human-computer interaction. Applications such as affective computing necessitate comprehension and reaction to human emotions. We are employing a spectral feature in this project. Technology for determining the various characteristics, such as loudness, tone, and intensity, etc through Mel Frequency Cepstral Coefficients (MFCC) . For voice processing jobs, MFCCs are frequently utilized because of their efficiency in recording pertinent acoustic properties, such as compact depiction of the spectrum of a signal in audio. Our method includes taking spoken utterances and extracting vectors. putting them through a machine learning model's input. Our primary goal is to categorize eight emotions into a collection of pre-established categories, including neutral, peaceful, pleased, sad, angry, scared, disgusted, and surprised. The retrieved vectors consist of data to record spectral characteristics that are then trained into the classification model. The suggested technique not only helps to increase voice recognition systems' accuracy but also emphasizes how crucial it is to use coefficients as potent features in this situation. This method has the potential to enhance applications in recommender systems, affective computing, and interaction between humans and computers that require accurate identification of emotions.

Keywords: Spectral features, MFCC, Acoustic properties, Voice processing.

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The recognition of emotional speech has numerous applications in domains like customer service, mental health monitoring, and interaction between humans and computers, acknowledging emotional speech has drawn increased attention recently.

Emotion is a vital component of communication because it shapes our verbal expression of ourselves. Emotional cues in speech must be recognized by sensitive and intuitive systems in order to interpret speech and interact with users in a more effective manner. Making use of machine learning methods in conjunction with Mel Frequency Cepstral Coefficients (MFCCs) is one well-known approach in this field. Because MFCCs can reduce dimensionality while capturing pertinent features in the frequency domain, they have a broad application in speech signal processing. These coefficients, when applied to emotional speech, can offer important insights into the distinctive patterns and traits connected to various emotions. Using MFCCs to identify and categorize emotional states has shown to be a successful application of machine learning techniques, such as classification algorithms. These algorithms are able to distinguish between various feelings according to the features that are extracted by training models on labeled datasets that contain examples of different emotional speech patterns.

The integration of MFCCs and machine learning methods for the identification of emotive speech is examined in this paper. It explores the theoretical underpinnings of MFCCs and explains why they are useful for capturing speech signals' spectral characteristics. The paper also addresses the choice and application of machine learning algorithms for efficient emotional state classification. The performance of classifiers can be said to be primarily dependent on methods for extracting features from voice and characteristics that are important for a particular feeling. The classifiers can be strengthened if more features such as linguistic and visual elements can be combined from various modalities. But this depends on how important and accessible it is. The classification system is then given permission to use a wide range of classifiers after receiving these features.

1.2 OBJECTIVES OF THE PROJECT

The primary objective of this study is to design and implement a speech emotion recognition system that accurately classifies the emotional states conveyed in spoken utterances. Specifically, the goals are:

To extract discriminative features from speech signals using Mel-Frequency Cepstral Coefficients (MFCCs).

To explore different machine learning algorithms for classifying emotional states based on the extracted features.

To evaluate the performance of the proposed SER system in terms of classification accuracy, robustness, and computational efficiency.

Here are the key objectives of SER using machine learning:

1.2.1. Enhancing Human-Computer Interaction (HCI):

Recognizing emotions in speech enables machines to respond appropriately to users' emotional states, leading to more natural and personalized interactions.

Improving user experience by adapting system responses based on detected emotions, such as adjusting music playlists or tone of voice in virtual assistants.

1.2.2. Developing Assistive Technologies:

Creating tools to aid individuals with disabilities, such as those with autism spectrum disorder, in recognizing and responding to emotional cues in spoken communication.

Supporting mental health applications by providing insights into individuals' emotional states through their speech patterns, aiding in early detection and intervention for conditions like depression or anxiety.

1.2.3. Improving Psychological Analysis:

Assisting psychologists and therapists in analyzing emotional states expressed during therapy sessions, facilitating more objective assessments and personalized treatment plans.

Enabling researchers to study emotional dynamics in various contexts, such as analyzing political speeches, customer service interactions, or educational settings.

1.2.4. Advancing Human-Robot Interaction:

Integrating emotion recognition into robotic systems to enhance social interactions and collaboration between humans and robots, particularly in settings like eldercare or education.

Enabling robots to understand and respond to human emotions, fostering empathy and rapport in human-robot interactions.

1.2.5. Enabling Sentiment Analysis in Social Media:

Analyzing sentiment in audio content shared on social media platforms, providing insights into public opinion, consumer preferences, and trends.

Enhancing marketing strategies by understanding the emotional impact of advertisements, product reviews, and user-generated content.

1.2.6. Facilitating Emotional Intelligence Applications:

Supporting the development of emotionally intelligent systems that can perceive, understand, and respond to human emotions effectively.

Empowering individuals to track and manage their own emotional well-being through applications that analyze their speech patterns and provide feedback or interventions.

In summary, speech emotion recognition using machine learning encompasses a broad spectrum of objectives, ranging from improving human-computer interaction and assistive technologies to enhancing psychological analysis and advancing human-robot interaction. These objectives collectively contribute to the development of more empathetic, intuitive, and socially aware systems that can better understand and respond to human emotions.

1.3. PROBLEM STATEMENT

Developing a system that can accurately identify and classify emotions expressed in spoken language. This involves training a CNN model to analyze audio input and classify it into different emotion categories, such as neutral, peaceful, pleased, sad, angry, scared, disgusted, and surprised etc. The goal is to create a robust and real-time system that can be applied in various applications, including customer service, mental health support, and human-computer interaction, to enhance the understanding of emotional cues in spoken communication.

CHAPTER 2

LITERATURE REVIEW

Conversation is required as input for most natural language processing systems, including voice-activated systems. The standard protocol involves using This speech input to be converted to text using Automatic Speech Recognition (ASR) systems, and then using the text output from ASR for classification or other learning operations. ASR corrects speaker-independent differences in speech transcriptions. Using probabilistic acoustic and language models, ASR systems generate highly accurate outputs but lose a considerable portion of speech from various users, resulting in information that suggests emotion from speech. Due to this gap, speech-based emotion recognition (SER) systems have attracted attention from researchers in recent years. Three essential problems for a SER system to succeed.

- Selecting an effective emotional speech database.
- Extract functional elements.
- Create trustworthy classifiers by utilizing machine learning algorithms.

The primary problem with the extraction of emotional features is what the SER system does. A standard speech recognition system (SERS) extracts features from speech, including spectral, pitch frequency, and energy related features. It then performs a classification task to predict different emotion classes. Emotionally recognized speech data can be either speaker-dependent or speaker-independent.

The goal is to improve emotion detection accuracy by utilizing a Machine learning-based emotion classifier to identify emotional characteristics in speech features and semantic information from text. We present various Convolutional neural architectures for text and speech feature-based emotion classification.

Every emotion contains different vocal parameters that exhibit diverse speech characteristics. An MFCC-based vocal emotional recognition performed using ANN in which MFCC features were used as speech parameters and five different emotional states were considered for analysis. Back-Propagation algorithm applied for interpretation of speaker emotion. Also the proposed system for recognition is independent of linguistic background and achieved 60.55% of average accuracy of recognition.

CHAPTER 3

PROPOSED METHOD

Our study introduces a new method for identifying emotional speech using sophisticated machine learning techniques and Mel Frequency Cepstral Coefficients (MFCCs). Our method consists of a carefully constructed pipeline that is designed to identify distinct emotional states accurately by extracting features from speech signals. We start by collecting a large dataset of emotional speech recordings that spans a variety of situations and emotions. Preprocessing is the process of carefully cleaning and dividing the audio data in order to enhance the input features' quality. Our method's central component is the extraction of MFCCs, which record crucial speech signal spectral properties. These coefficients serve as the foundation for feature vectors that capture the subtle acoustic characteristics connected to various emotions.

We then use a cutting-edge machine learning algorithm to classify emotions, selecting Support Vector Machines (SVMs) based on their track record of success in comparable tasks. A specific section of the dataset is used for model training, with the goal of optimizing parameters to achieve high accuracy and generalization. We perform extensive evaluations on a validation set for parameter tuning as well as a different testing set for objective performance assessment to guarantee robust performance. Through metrics like recall, accuracy, precision, and F1 score, Our approach enables a more thorough analysis of the model's performance and provides insight into how well it can distinguish between different emotional expressions. In addition, we highlight the improvements and contributions of our suggested method by contrasting its results with those of previous approaches in the literature. We outline possible drawbacks, such as dataset biases and difficulties managing particular emotional states, opening the door for further study avenues. Overall, our suggested approach demonstrates the potential for practical uses, such as mental health monitoring and human-computer interaction, in addition to showcasing the effectiveness of MFCCs in capturing emotional cues.

3.1 METHODOLOGY

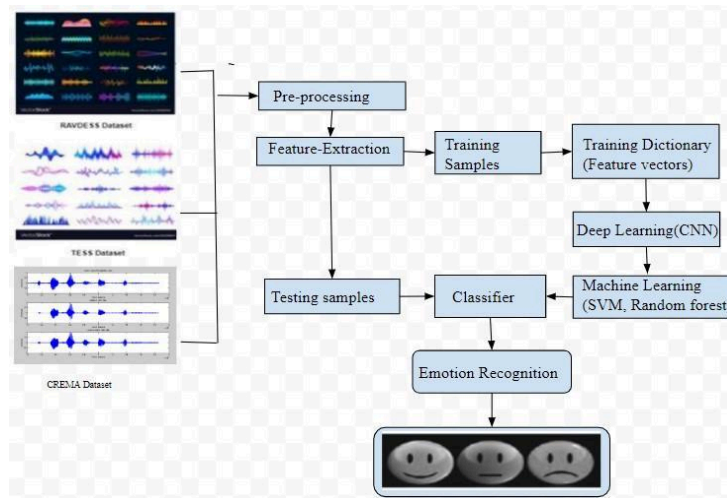


Fig: 1: Speech Emotion Recognition

3.1.1. Data Set & Data Visualization:

For this project, we will be using the RAVDESS, TESS and CREMA-D datasets which is the abbreviated form of Ryerson Audio-Visual Database of Emotional Speech and Song dataset, Toronto emotional speech set, and Crowd Sourced Emotional Multimodal Actors Dataset. This dataset has 7356 files rated by 247 individuals 10 times on emotional validity, intensity, and genuineness.

Using an audio file from the RAVDESS Dataset, the Speech Emotion Recognition Project will upload the file in .wav format before the file upload process is validated. This process pertains to the file format and empty file input, and the file will be connected directly to Python files where the resultant product is produced as Emotional Labels. Information about the provided audio data in the document is provided by data visualization, pictorial and graphic formats.

The waveform for the fear emotion is as follows:

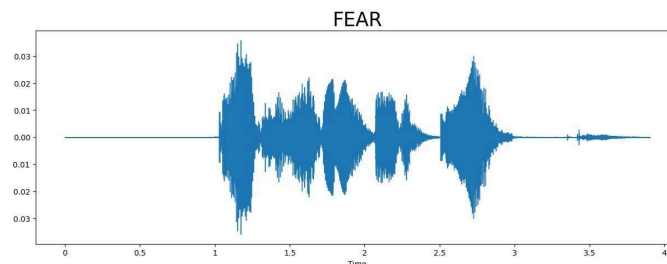


Fig: 2: waveform for the fear emotion

The 3D waveform for fear emotion is as follows:

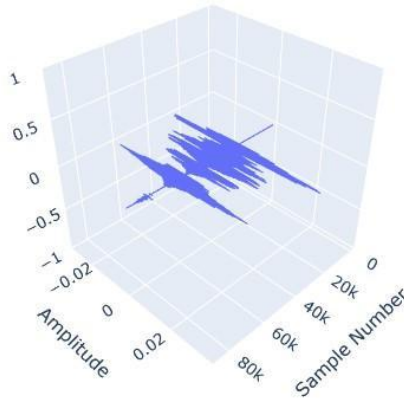


Fig: 3: 3D waveform for fear emotion

3.1.2. Pre-processing:

The first step after collecting the data is preprocessing. The collected data would be utilized to prepare the classifier in an SER system. While few of these preprocessing procedures are utilized for feature extraction, others take care of the normalization of the features so that the variations in the recordings of the speakers do not affect the recognition process.

Signals for speech are examined within the domain of speech emotion recognition (SER) in order to ascertain the emotional condition of the speaker. To ensure the quality of the data input, extract pertinent features, and raise the general efficacy of emotion recognition models, pre-processing is an essential step in the SER process. The following pre-processing actions are frequently used in speech emotion recognition:

3.1.2.1. Gathering of Data:

Gather a representative and varied collection of speech recordings that demonstrate a range of emotional states. Make sure the dataset includes a variety of speakers, recording settings, and relevant emotions.

3.1.2.2. uniformity of the sampling rate:

Verify that each audio recording has the same sampling rate. In order to guarantee compatibility with particular models and algorithms, standardization is required. While there isn't a predetermined formula, it is necessary to ensure that each audio recording has a consistent sampling rate. For this, resampling techniques are typically employed.

3.1.2.3. Removal of Noise:

Remove any unnecessary noises and other disruptions from the speech signal. This can be achieved by employing methods such as spectral subtraction and noise reduction algorithms. Use noise removal techniques to reduce background noise and enhance the clarity of speech signals. Typical methods include wavelet denoising, adaptive subtraction, and adaptive filtering. $Y(f) = X(f) - \alpha \cdot N(f)$ is the formula for spectral subtraction, where $Y(f)$ is the cleaned signal, $X(f)$ stands for the observed signal, $N(f)$ for the estimated noise, and α for a scaling factor.

3.1.2.4. Segmentation:

Split the speech signal up into frames, or shorter segments. This makes it possible to analyze brief characteristics and contributes to capturing the dynamic quality of speech. Split the speech signals up into frames, or smaller sections. This stage is essential for capturing speech's dynamic quality and enables the examination of its fleeting characteristics. To maintain continuity, overlapping frames are frequently employed. There isn't a set formula, but segmenting is breaking the speech signal up into smaller frames. The requirements of the analysis determine the frame length and overlap that are used, which is common in overlapping frames.

3.1.2.5. Normalization:

To guarantee that the amplitude levels of the speech signals are constant, normalize them. By taking this step, the variability brought on by various recording conditions is reduced. To improve speech signal clarity and minimize background noise, apply noise removal techniques. Adaptive filtering, adaptive subtraction, and wavelet denoising are common techniques. There isn't a set formula, but normalization entails scaling the speech signal's amplitude to guarantee constant levels. RMS normalization and peak normalization are two popular techniques.

3.1.2.6. Pre-emphasis:

By boosting the higher frequencies with a pre-emphasis filter, you can increase the ratio of signal to noise and boost the efficiency with which feature extraction techniques. To enhance the speech signal's higher frequencies, apply a pre-emphasis filter. Pre-emphasis can boost the performance of later feature extraction techniques and aid in increasing the signal-to-noise ratio. Pre-emphasis Equation: $Y(n) = X(n) - \alpha \cdot X(n-1)$, in which $Y(n)$ is the pre-emphasized signal, $X(n)$ represents the original signal, α denotes the coefficient of pre-emphasis (usually set at approximately 0.95).

3.1.2.7. Windowing:

When performing the Fourier transform, minimize spectral leakage by applying window functions to the speech signal. Gaussian, Hanning, and Hamming windows are examples of common window functions. The speech signal frames should be subjected to window functions (such as Hanning and Hamming) to minimize spectral leakage during the Fourier transform. In order to get the signal ready for feature extraction, windowing is necessary.

3.1.2.8. Feature Normalization:

To ensure consistency between feature vectors from various speakers and recording environments, normalize them. Z-score normalization is one popular normalization method. To ensure consistency between feature vectors from various speakers and recording environments, normalize them. Min-max scaling and z-score normalization are two popular normalization methods. Formula for Z-Score Normalization:

3.1.2.9. Aggregation of Features:

To get a representation for the full speech segment, aggregate the features over time. Statistics like mean, standard deviation, and other statistical moments may be used in this. To get a representation for the full speech segment, aggregate the features over time. Statistics like mean, standard deviation, and other statistical moments may be used in this. Aggregation is the process of summarizing feature vectors over time; there is no set formula for this.

3.1.3. Feature Extraction:

A survey of the MFCC's feature extraction examination has been conducted. The programs mandate the integration of a voice print function withdrawal method, anticipating improved price attractiveness, a decrease in reduction over latency. Pitch fee, the elocution approach, and emotion unprejudiced popularity version design are all excellent ideas that should be taken into consideration as essential parameters. By considering these factors, the speaker discusses the high rate of acknowledgment regardless of feeling and during gastrointestinal illness.

3.1.3.1. Mel Frequency Cepstral Coefficients

The most widely used spectral feature in automatic speech recognition is Mel Frequency Cepstral Coefficient (MFCC). MFCCs represent the envelope of the short-time power spectrum, which represents the shape of the vocal tract. The utterances are split into various segments before converting into the frequency domain using short-time discrete Fourier transform to obtain MFCC. Mel filter bank is utilized to calculate several sub-band energies. After that, the

logarithm of respective sub-bands is computed. Lastly, MFCC is determined by applying the inverse Fourier transform.

3.1.3.2. A Study of Feature Extraction Metrics

I merely summarize the analysis of several extractions carried out using MFCC in this section, which is based on the numerous contributions made by numerous academics. A voice recognition model was proposed by Nidhi Srivastava and Harsh Dev (2018) using the MFCC approach. The author used the MFCC technique in this work to improve word recognition rate. The MATLAB platform has been used by the author.

3.1.3.3. Related work

The power spectrum of a fast-term valid signal in audio processing, represented by the Mel Frequency Cepstrum (MFC), which is based only on the linear cosine alter in a nonlinear frequency Mel scale. Mel Frequency Cepstral Coefficients (MFCC) are a set of coefficients combining to form an MFC.

Since MFCC values aren't very resilient to additive noise, standardizing their values in speech popularity structures is a reasonable way to reduce the noise's impact. A few Researchers recommend modifying the basic MFCC algorithm in order to improve robustness by increasing the log-mel-amplitudes before taking the DCT to an appropriate ener (roughly two or three). The results MFCC for the dataset:

	0	1	2	3	4	5	6	7	8	9	...	31	
0	-429.301880	138.093689	12.359602	66.641853	-21.215841	26.068295	-20.639458	13.854490	-12.364154	5.450428	...	0.731685	4.7944
1	-368.219734	79.594505	42.511378	33.682749	9.000123	3.918399	-2.629232	0.246298	-2.844988	-0.322330	...	1.043888	2.5855
2	-494.193176	130.992905	17.355747	57.633629	-20.700792	24.379467	-20.369572	14.412844	-13.881453	7.908208	...	7.168769	1.1996
3	-464.820190	139.803864	9.313372	67.316628	-22.780102	27.141413	-22.381947	14.725032	-13.665572	4.997799	...	1.463877	5.3994
4	-464.820190	139.803864	9.313372	67.316628	-22.780102	27.141413	-22.381947	14.725032	-13.665572	4.997799	...	1.463877	5.3994

5 rows x 41 columns

Fig: 4: MFCC Coefficients of each audio file

3.1.4. Feature Selection:

This project primarily consists of features that extract the essence of the provided audio, such as the Mel Spectrogram and MFCC (Mel information retrieval in the music category). These are a few commonly used audio features for acoustic recognition, negative or from negative to 0 to positive, and emotional audio content. Spectrograms that visualize Mel Spectrograms are sounds that are recorded on the Mel scale as opposed to the frequency domain. The frequency of

a signal must be transformed logarithmically to create the Mel Scale. Through these feature selection metrics the emotions are distributed in the following manner.

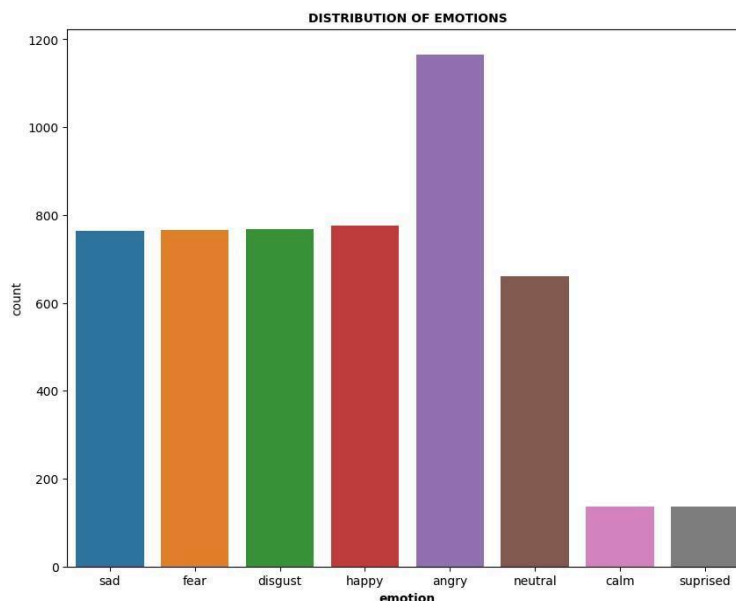


Fig: 5: Distribution of Emotions

3.1.5. Training and Testing Samples:

Popular resources in order to instruct and evaluate Speech Emotion Recognition (SER) systems include the TESS (Toronto Emotional Speech Set), RAVDESS and CREMA-D datasets. By these datasets, you can create a robust SER system using the TESS, RAVDESS and CREMA-D datasets and evaluate its performance using MFCC features. Remember to adapt these steps based on the specific characteristics and requirements of your SER task.

3.1.6. Training Dictionary:

A training dictionary is used for representation that holds feature vectors with MFCC and the associated emotion labels. Each entry in the dictionary, which is arranged as a list or array, is made up of a tuple (X, Y), where X is the MFCC feature vector that was taken from a speech segment and Y is the associated emotion label, like "happy" or "sad." These feature vectors are fed training algorithms, enabling them to discover patterns and correlations between acoustic features and emotions. The training loop refines the model's comprehension by repeatedly iterating through the dictionary over several epochs. Accuracy is one of the metrics used in model evaluation along with standardization techniques to improve and evaluate the SER model's performance after training on the given emotion-labeled dataset.

3.1.7. Deep Learning Technique(CNN):

First, the raw speech waveforms are transformed into spectrograms, which show the frequency content over time visually. CNN uses these spectrograms as input. The CNN's convolutional layers identify acoustic features that correspond to different emotions by identifying local patterns in the spectrogram.

Next come the pooling layers, which emphasize salient features, lower computational load, and downsample spatial dimensions. High-level relationships across the spectrogram are further abstracted through flattening and fully connected layers. The output layer generates probability distributions across various emotion classes and is commonly fitted with a softmax activation function. Using labeled datasets, CNNs are trained to map spectrograms to corresponding emotion labels via optimization and backpropagation. Careful hyperparameter tuning is essential for optimal performance, and methods such as data augmentation and transfer learning from pre-trained models can improve generalization. The CNN is tested on a different testing set after training to see how well it can generalize and identify emotions in speech data that hasn't been seen before. CNNs are essential for expanding the capabilities of SER systems due to their adaptability and efficiency.

3.1.7.1. Convolutional Neural Networks:

The convolution operation represents the hierarchical extraction of speech features, while the maximum pooling operation removes redundant information in the previous layer features, and the operation is simplified. An activation layer is set after each convolution layer, and the best activation function is determined by experiments. Two fully connected neurons are set in the output layer to divide the speech signals into two categories. In addition, considering the complexity of the network structure, random Dropout is set after each hidden layer to prevent the network from over-fitting in the training process. After adding Dropout, the neurons in each hidden layer will have a certain probability of not updating the weights in the training process, and the probability of each neuron not updating the weights is equal.

Algorithm:

Step 1: The sample audio is provided as input.

Step 2: The Spectrogram and Waveform is plotted from the audio file.

Step 3: Using the LIBROSA, a python library, we extract the MFCC (Mel Frequency Cepstral Coefficient) usually about 10–20.

Step 4: Remixing the data, dividing it in train and test and thereafter constructing a CNN model and its following layers to train the dataset.

Step 5: Predicting the human voice emotion from that trained data (sample no.- predicted value - actual value)

UML Diagram:

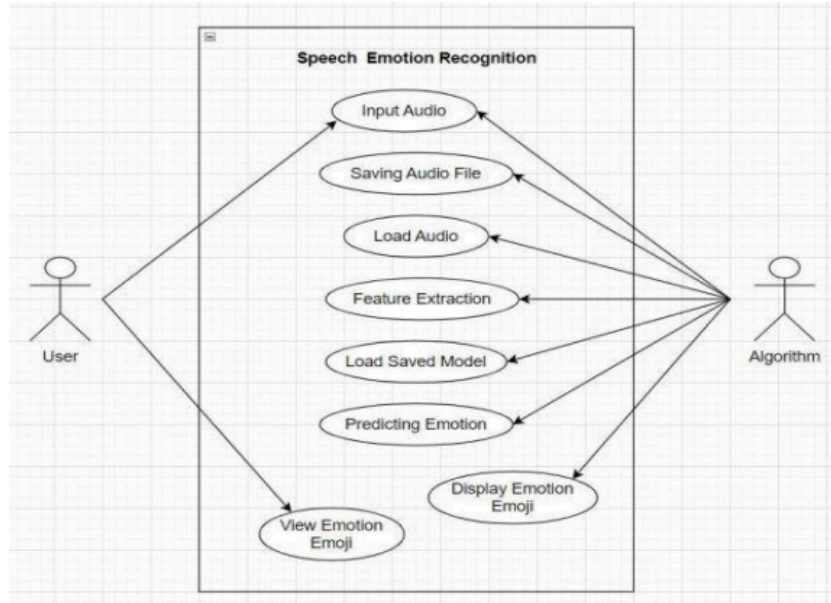


Fig: 6: UML Diagram of Speech Emotion Recognition

3.1.8. Machine Learning Techniques(SVM, random forest):

An SVM classifier is supervised and preferential. The classifier is generally described for linearly separable patterns by splitting hyperplanes. SVM makes use of the kernel trick to model nonlinear decision boundaries. The SVM classifier aims to detect that hyperplane having a maximum margin between two classes' data points. The original data points are mapped to a new space if the given patterns are not linearly separable by utilizing a kernel function.

The SVM algorithm finds the best hyperplane to divide the various emotion classes in the feature space. SVMs are capable of capturing intricate decision boundaries and are useful for binary and multiclass classification tasks. In contrast, Random Forests use a group of decision trees to generate predictions. A voting mechanism determines the final prediction after a portion of the data has been used to train each tree in the forest. Because Random Forests are strong and adaptable to a wide range of feature sets, they can capture a wide range of emotional expressions with flexibility.

After labeled feature vectors are fed into the corresponding algorithms during the training phase, the models' ability to identify emotions in fresh speech samples can be tested on a different testing set. To maximize the performance of these machine learning models in SER systems, hyperparameter tuning and feature selection are essential.

3.1.9. Classification:

Throughout the training process, the trained model picks up on the patterns and connections between MFCCs and emotions. The method for CNNs is converting unprocessed audio waveforms into representations resembling spectrograms and then using convolutional layers to record frequency-domain temporal patterns. These layers are followed by fully connected and pooling layers, which result in an output layer with probability for each emotion class. Supervised learning is a technique used to train CNNs and machine learning classifiers. Using the training data, the model generalizes to classify emotions in previously unseen speech samples. The training dataset's equality and diversity, the algorithm of choice, and proper hyperparameter tuning all affect how effective the system is. The models are tested on a testing set after training to see how well they can identify emotions in speech and how well they can generalize. A comprehensive and efficient speech recognition system that can identify subtle patterns in emotional speech expressions can be achieved by combining both conventional machine learning methods and deep learning methods, like CNNs.

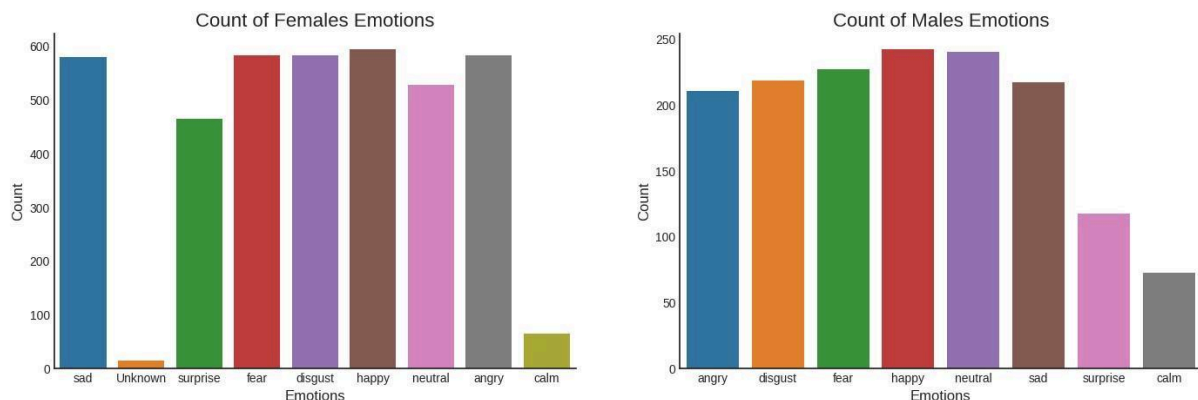


Fig: 7: Count of Female and Male Emotions

3.1.10. Emotion Recognition: Conventional machine learning involves training algorithms, like Support Vector Machines (SVMs) or Random Forests, on labeled datasets to associate particular emotional labels with acoustic patterns. Conversely, CNNs are very good at extracting hierarchical features from speech representations that resemble spectrograms. The idea is to make it possible for systems to recognize minute changes in tone, rhythm, pitch, and other

acoustic cues that represent various emotional conditions. The effectiveness of these systems depends on the selected features, the variety and quality of the training data, as well as the intricacy of the model architecture. Applications for emotion recognition in SER range widely, from mental health monitoring to human-computer interaction. It provides information about speakers' emotional states and enhances the functionality of different tech applications.

3.2. IMPLEMENTATION

3.2.1 Setup google colab for code execution

You'll need the following things to easily and safely installing Linux alongside Windows:

A computer that comes preinstalled with Windows 10.

Install Google on your pc/Laptop.

Search for google colab and open the foremost link on the page.

1. In colab create a new notebook as shown in the figure

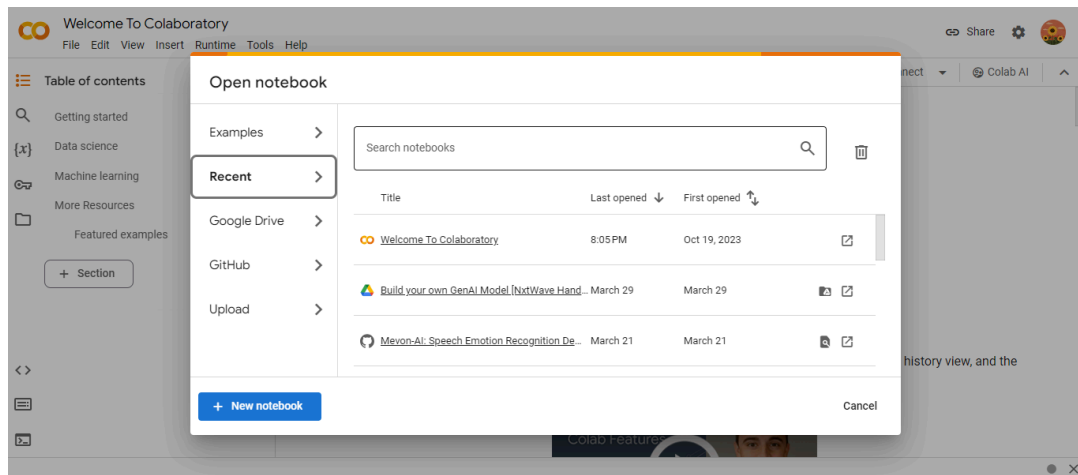


Fig: 8: Create a new notebook in Google Colab

2. You will get an interface in which jupyter service will be started to develop codes.

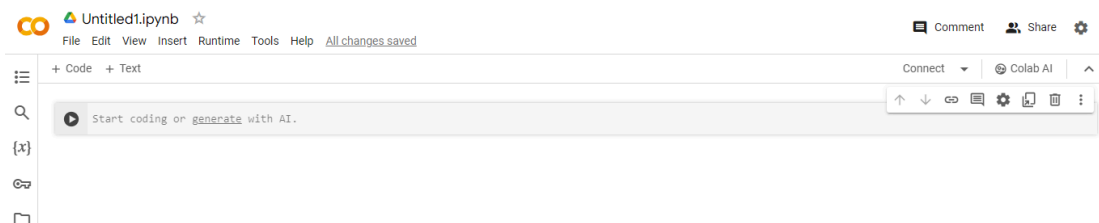
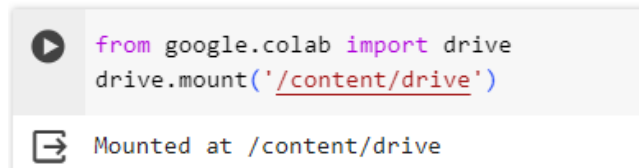


Fig: 9: Colab Interface

- import drive from google.colab and call mount method and run the cell.

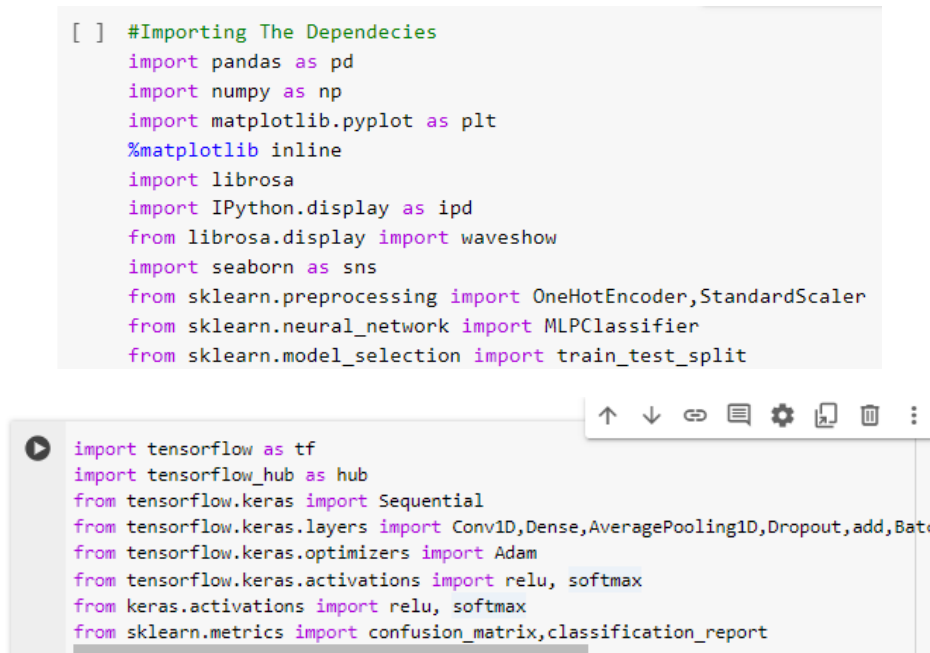


```
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

Fig: 10: Import drive from google colab

- Import all the dependencies required for the execution.



```
[ ] #Importing The Dependencies
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import librosa
import IPython.display as ipd
from librosa.display import waveshow
import seaborn as sns
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split

import tensorflow as tf
import tensorflow_hub as hub
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Conv1D, Dense, AveragePooling1D, Dropout, add, BatchNormalization
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.activations import relu, softmax
from keras.activations import relu, softmax
from sklearn.metrics import confusion_matrix, classification_report
```

Fig: 11: Import all the packages

3.2.2. Data Collection:

- Go to Kaggle and search for 'RAVDESS', 'TESS' and 'CREMA_D' and download the data sets shown.

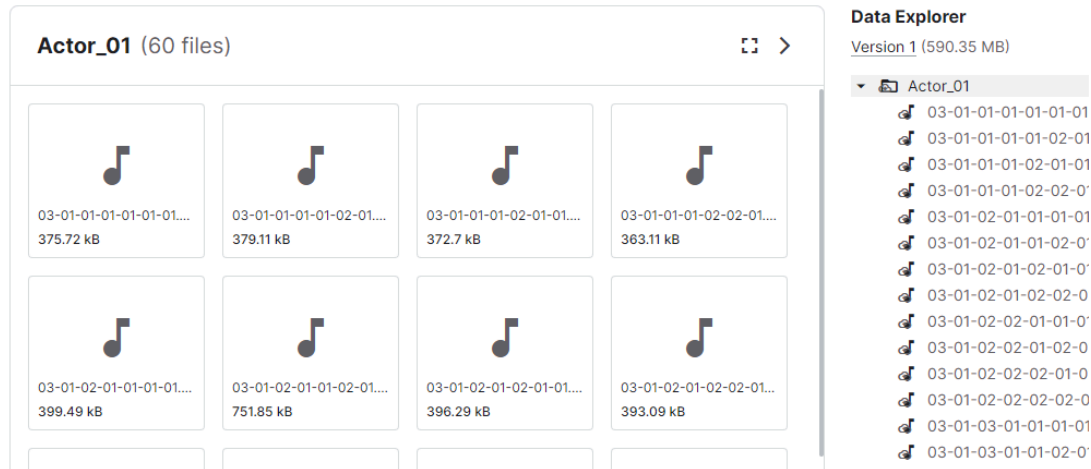


Fig: 12: RAVDESS dataset

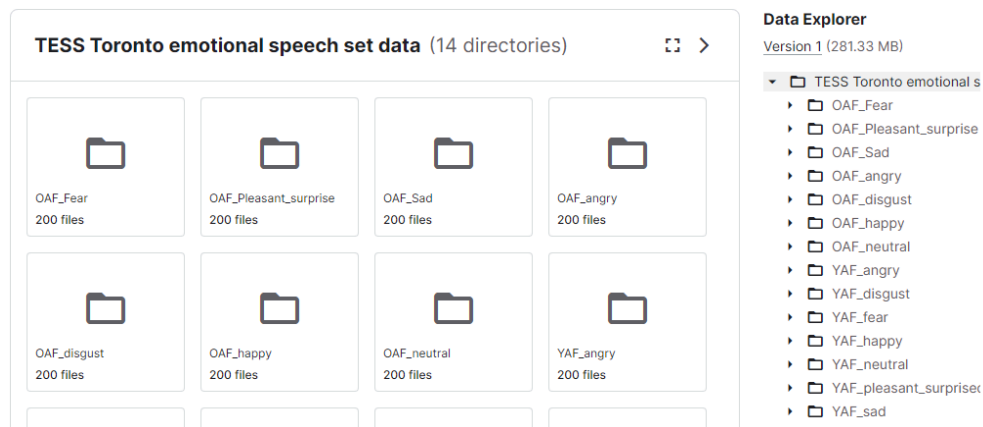


Fig: 13: TESS dataset

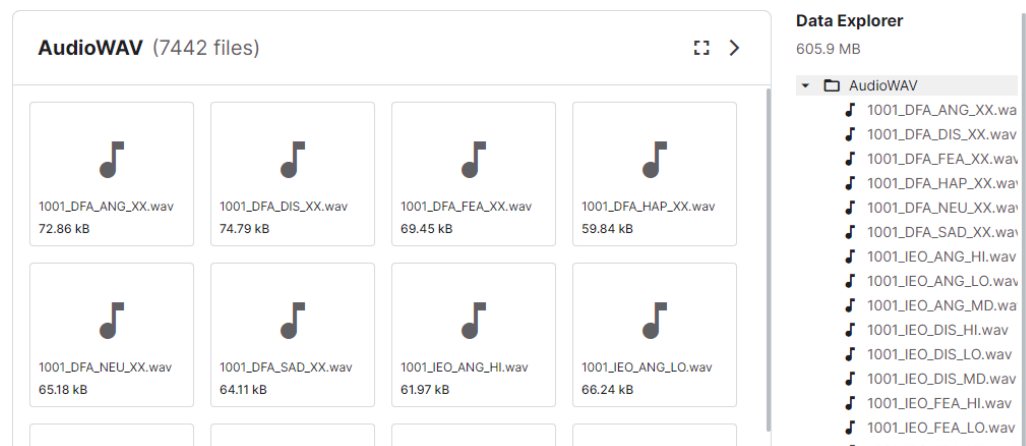
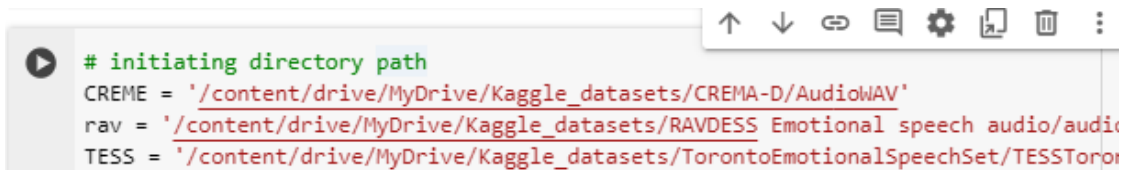


Fig: 14: CREMA-D dataset

2. Explore metadata associated with the dataset, such as speaker information, recording conditions, and emotion labels.

3. Check for any ethical considerations, such as obtaining proper permissions for dataset usage.
4. Convert audio files to a consistent format for processing (e.g., WAV or MP3).
5. Consider augmenting the dataset through techniques like pitch shifting or time stretching to increase diversity.
6. Split the dataset into training and testing sets, maintaining a balanced distribution of emotions.
7. Explore any potential biases in the dataset and address them appropriately.
8. Document the details of the dataset for future reference in your project documentation.
9. Initiate all the directory paths into the code.

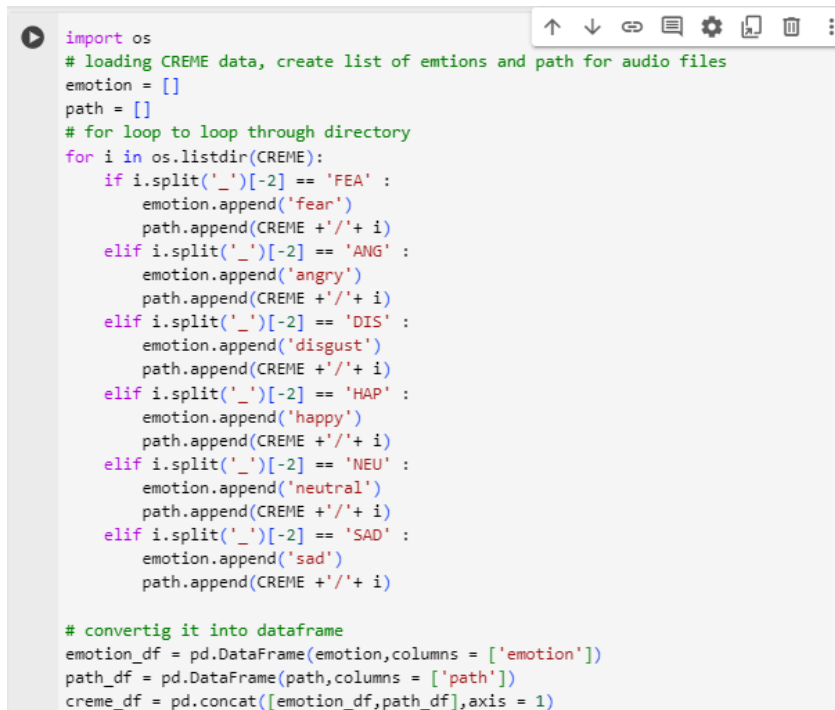
3.2.3.Loading Data:



```
# initiating directory path
CREME = '/content/drive/MyDrive/Kaggle_datasets/CREMA-D/AudioWAV'
rav = '/content/drive/MyDrive/Kaggle_datasets/RAVDESS Emotional speech audio/audio'
TESS = '/content/drive/MyDrive/Kaggle_datasets/TorontoEmotionalSpeechSet/TESSTorontoEmotionalSpeechSet'
```

Fig: 15: Loading Data

Step 1: loading CREME data, create list of emotions and path for audio files.



```
import os
# loading CREME data, create list of emotions and path for audio files
emotion = []
path = []
# for loop to loop through directory
for i in os.listdir(CREME):
    if i.split('_')[-2] == 'FEA' :
        emotion.append('fear')
        path.append(CREME + '/' + i)
    elif i.split('_')[-2] == 'ANG' :
        emotion.append('angry')
        path.append(CREME + '/' + i)
    elif i.split('_')[-2] == 'DIS' :
        emotion.append('disgust')
        path.append(CREME + '/' + i)
    elif i.split('_')[-2] == 'HAP' :
        emotion.append('happy')
        path.append(CREME + '/' + i)
    elif i.split('_')[-2] == 'NEU' :
        emotion.append('neutral')
        path.append(CREME + '/' + i)
    elif i.split('_')[-2] == 'SAD' :
        emotion.append('sad')
        path.append(CREME + '/' + i)

# convertig it into dataframe
emotion_df = pd.DataFrame(emotion,columns = ['emotion'])
path_df = pd.DataFrame(path,columns = ['path'])
creme_df = pd.concat([emotion_df,path_df],axis = 1)
```

Fig: 16: Create a list for emotions of CREMA-D data

Step 2: Load RAVDESS data

```
# loading ravdess data
emotion = []
path = []
for i in os.listdir(rav) :
    # 20 actors in directory
    for aud in os.listdir(rav + '/' + i) :
        split = aud.split('.')[0].split('--')
        temp = int(split[2])

        if split[2] == '01' :
            emotion.append('neutral')
            path.append(rav + '/' + i + '/' + aud)
        if split[2] == '02' :
            emotion.append('calm')
            path.append(rav + '/' + i + '/' + aud)
        if split[2] == '03' :
            emotion.append('happy')
            path.append(rav + '/' + i + '/' + aud)
        if split[2] == '04' :
            emotion.append('sad')
            path.append(rav + '/' + i + '/' + aud)
        if split[2] == '05' :
            emotion.append('angry')
            path.append(rav + '/' + i + '/' + aud)
        if split[2] == '06' :
            emotion.append('fear')
            path.append(rav + '/' + i + '/' + aud)
        if split[2] == '07' :
            emotion.append('disgust')

            path.append(rav + '/' + i + '/' + aud)
        if split[2] == '08' :
            emotion.append('surprised')
            path.append(rav + '/' + i + '/' + aud)

# creating dataframe
emotion_df = pd.DataFrame(emotion,columns = ['emotion'])
path_df = pd.DataFrame(path,columns = ['path'])
rav_df = pd.concat([emotion_df,path_df],axis = 1)
```

Fig: 17: Create a list for emotions of RAVDESS data

Step3 : Load TESS data

```
# loading tess data
emotion = []
path = []
for i in os.listdir(TESS) :
    fname = os.listdir(TESS + '/' + i)
    for f in fname:
        if i == 'OAF_angry' or i == 'YAF_angry' :
            emotion.append('angry')
            path.append(TESS + '/' + i + '/' + f)
        if i == 'OAF_disgust' or i == 'YAF_disgust' :
            emotion.append('disgust')
            path.append(TESS + '/' + i + '/' + f)
        if i == 'OAF_Fear' or i == 'YAF_fear' :
            emotion.append('fear')
            path.append(TESS + '/' + i + '/' + f)
        if i == 'OAF_happy' or i == 'YAF_happy' :
            emotion.append('happy')
            path.append(TESS + '/' + i + '/' + f)
        if i == 'OAF_neutral' or i == 'YAF_neutral' :
            emotion.append('neutral')
            path.append(TESS + '/' + i + '/' + f)
        if i == 'OAF_pleasant_surprise' or i == 'YAF_surprise' :
            emotion.append('surprise')
            path.append(TESS + '/' + i + '/' + f)
        if i == 'OAF_angry' or i == 'YAF_angry' :
            emotion.append('angry')
            path.append(TESS + '/' + i + '/' + f)
        if i == 'OAF_Sad' or i == 'YAF_sad' :
            emotion.append('sad')
            path.append(TESS + '/' + i + '/' + f)
```

Fig: 18: Create a list for emotions of TESS data

Step 4: Combine all data frames

```
# combining all dataframe
audio_df = pd.concat([
    creme_df, rav_df, tess_df
], axis = 0)
audio_df.reset_index(drop = 'index', inplace = True)
```

Fig: 19: Combine all data frames

3.2.4. Data visualization

1. Save the data and visualize using barplot.

```
# saving audio_csv
audio_df.to_csv('audio.csv')

[ ] plt.figure(figsize = (10,8))
    sns.countplot(data = audio_df,x= 'emotion')
    plt.title('DISTRIBUTION OF EMOTIONS',fontweight = 'bold',fontsize = 10)
    plt.xlabel('emotion',fontweight = 'bold');
```

Fig: 20: Save the data and visualize using barplot

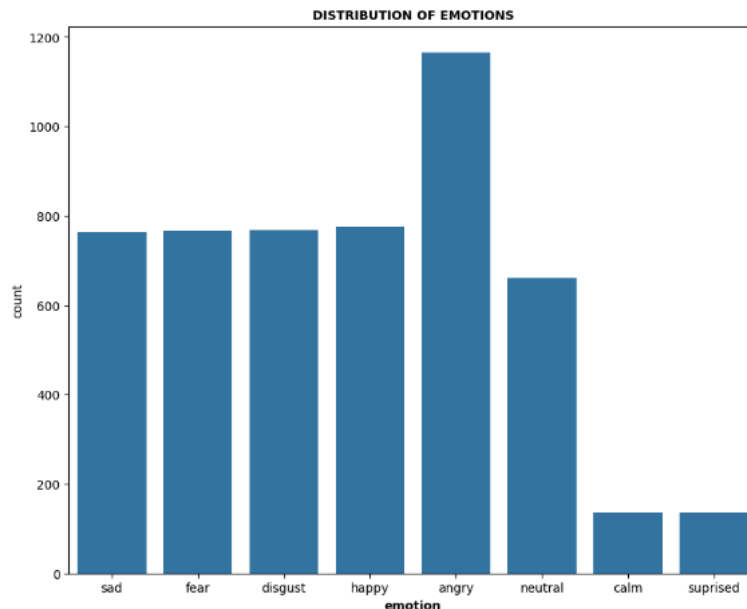


Fig: 21: Barplot representation of 8 Emotions

3.2.5. Feature Extraction:

1. Familiarize with the concept of Mel-Frequency Cepstral Coefficients (MFCCs) and their relevance in speech processing.
2. Use a library called 'librosa' in Python to compute MFCCs from audio signals efficiently.
3. Experiment with different parameters for the MFCC extraction process, such as the number of coefficients and frame size.

```
def audio_visual(path,emotion) :
    data , sample_rate = librosa.load(path)
    plt.figure(figsize = (10,5))
    plt.title(f'Waveplot for audion with {emotion} emotions')
    waveshow(data,sample_rate)
    plt.show()
    return ipd.Audio(path)

[ ] def feature_extraction(file) :
    mfcc_features = librosa.feature.mfcc(y=file,sr = sample_rate,n_mfcc = 40)
    mfcc_scaled_feature = np.mean(mfcc_features.T,axis = 0)
    return mfcc_scaled_feature
    # NOISE
    def noise(data):
        noise_amp = 0.035*np.random.uniform()*np.amax(data)
        data = data + noise_amp*np.random.normal(size=data.shape[0])
        return data
    # STRETCH
    def stretch(data, rate=0.8):
        return librosa.effects.time_stretch(y=data, rate=rate)
    # SHIFT
    def shift(data):
        shift_range = int(np.random.uniform(low=-5, high = 5)*1000)
        return np.roll(a=data, shift=shift_range)
    # PITCH
    def pitch(data, sampling_rate, pitch_factor=0.7):
        return librosa.effects.pitch_shift(y=data, sr=sampling_rate, n_steps=pitch_factor)
    #higher speed
    def higher_speed(data, speed_factor = 1.25):
        return librosa.effects.time_stretch(y=data, rate=speed_factor)
    # lower speed
    def lower_speed(data, speed_factor = 0.75):
        return librosa.effects.time_stretch(y=data, rate=speed_factor)
```

Fig: 22: Feature Extraction

4. Visualize the extracted MFCCs to gain insights into the characteristics of the speech signals.
5. Explore other feature extraction techniques, such as chroma features or spectrogram analysis, and compare their performance with MFCCs.
6. Implement a feature normalization step to bring the extracted features to a consistent scale.

```
from tqdm import tqdm
x = []
y = []

for path,emotion in tqdm(zip(audio_df['path'],audio_df['emotion'])) :
    feature = get_feat(path)
    for ele in feature :
        x.append(ele)
        y.append(emotion)
```

5170it [36:40, 2.35it/s]

Fig: 23: Feature Normalization

7. Handle edge cases or anomalies in the data during the feature extraction process.
8. Document the parameters and settings used for feature extraction for reproducibility.
9. Consider dimensionality reduction techniques if the feature space is large.
10. Ensure that the feature extraction process is compatible with the chosen machine learning model.

3.2.6. Displaying Frequencies of Different Types of Datasets

3.2.6.1. Normal Audio

The model's capacity to differentiate between the baseline and different emotional states is made possible by the analysis of normal audio, which is essential for accurate emotion recognition and enhances the system's overall efficacy in real-world applications like affective computing and human-computer interaction.

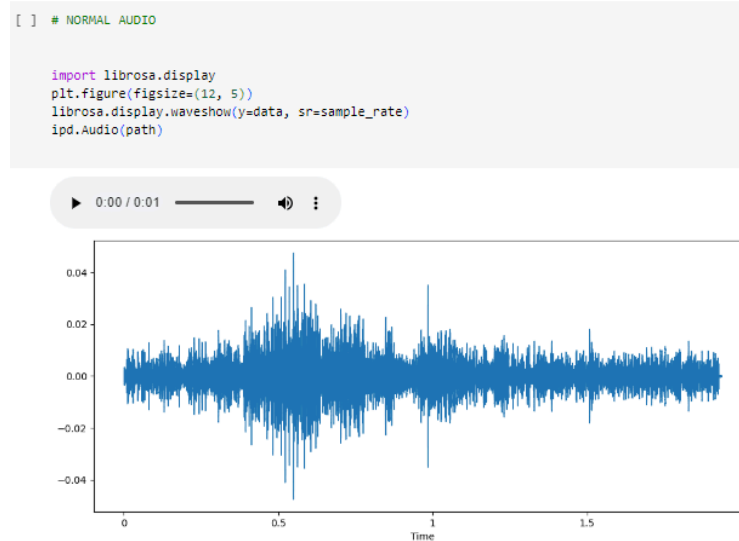


Fig: 24: Normal Audio

3.2.6.2. Audio with Noise

Noise makes it more difficult to distinguish minute differences in tone, pitch, and other acoustic characteristics that are essential for classifying emotions. To lessen the negative effects of noise, researchers and practitioners in SER frequently use a variety of signal processing and noise reduction techniques, such as wavelet denoising or spectral subtraction.

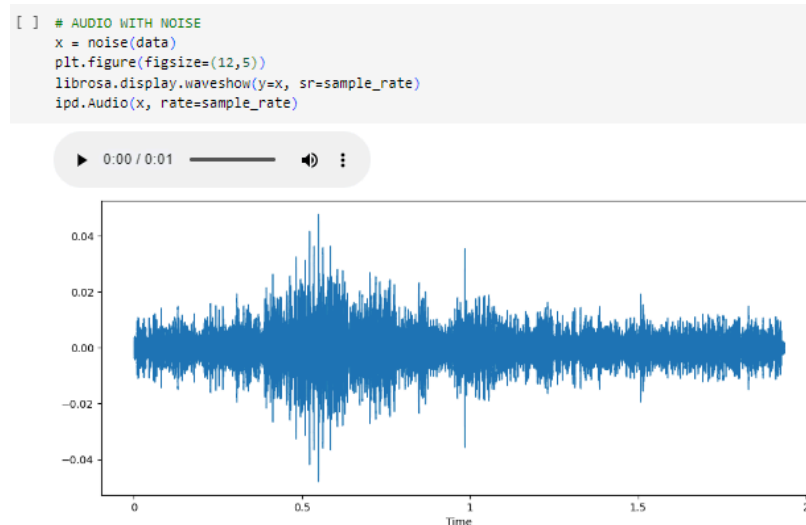


Fig: 25: Audio with Noise

3.2.6.3. Shifted Audio

Audio signals can be intentionally or unintentionally shifted, which can affect how relevant acoustic features that are essential for identifying emotions align. In order to guarantee temporal consistency throughout the dataset, extra care must be taken during the preprocessing stage. To improve the resilience of SER models, temporal shifts can be lessened by using strategies like dynamic time warping or alignment algorithms.

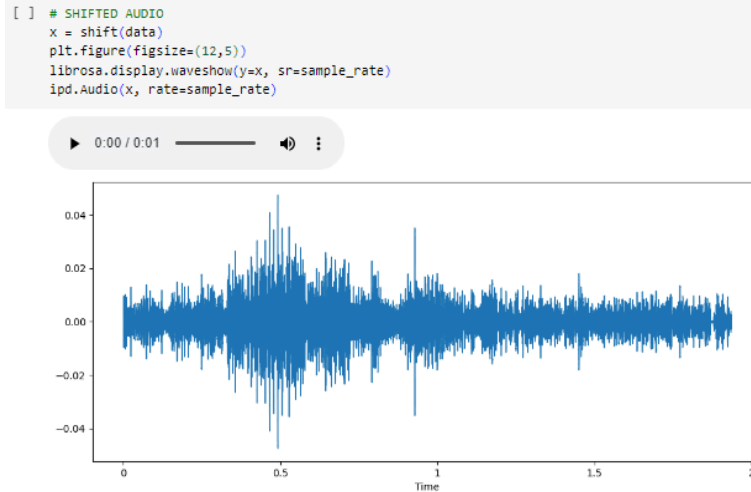


Fig: 26: Shifted Audio

3.2.6.4. Stretched Audio

The alignment of pertinent acoustic features that are essential for emotion recognition can be affected by intentionally or unintentionally shifting audio signals. Maintaining temporal consistency throughout the dataset necessitates extra care during the preprocessing stage.

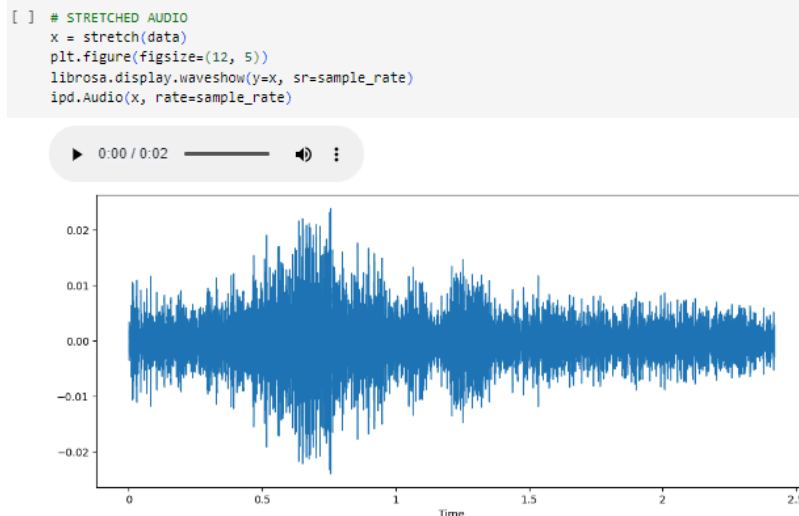


Fig: 27: Stretched Audio

3.2.6.5. Audio with Pitch

SER systems help improve the accuracy of emotion classification by identifying pitch patterns linked to various emotional states through the extraction and analysis of pitch-related features. This is especially important in situations where it is possible to deduce the emotional condition of the speaker from minute fluctuations in pitch that correspond to shifts in emphasis, stress, or intonation.

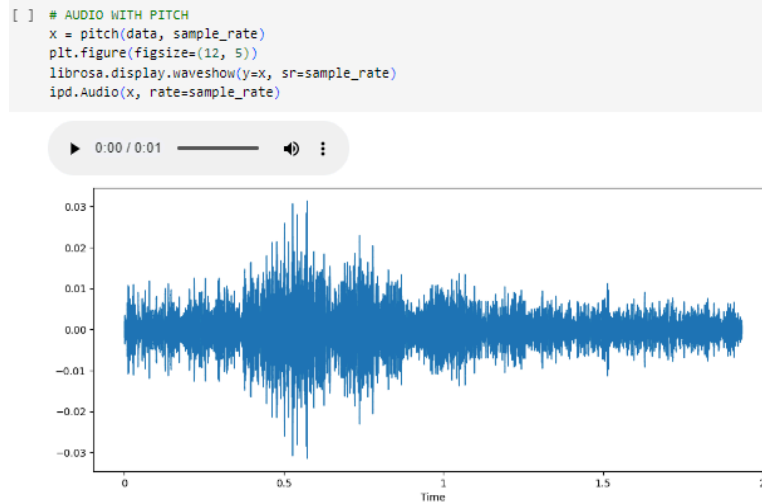


Fig: 28: Audio with Pitch

3.2.6.6. Audio with High speed

In the context of audio processing, modifying the speed of an audio file refers to changing the playback rate of the audio. Increasing the speed of an audio file results in a higher pitch and faster playback, while decreasing the speed leads to a lower pitch and slower playback.

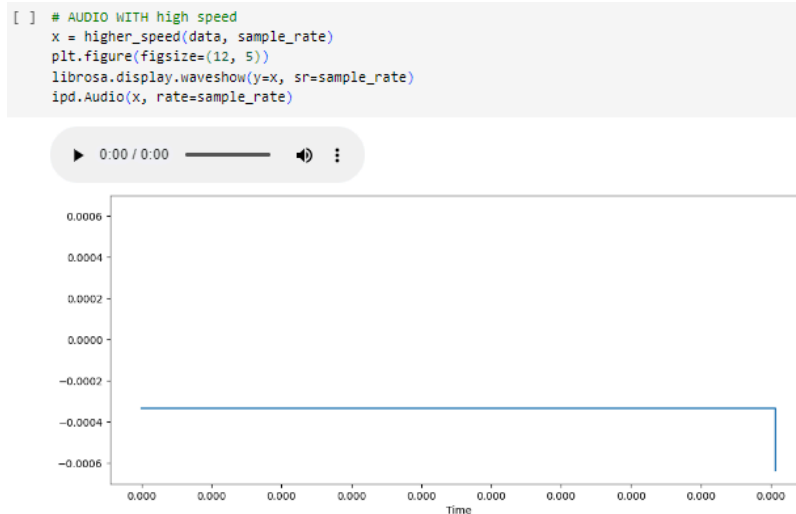


Fig: 29: Audio with High speed

3.2.6.7. Audio with Lower Speed

Lowering the speed of audio can be useful in various applications, such as creating slow-motion effects in audiovisual content, enhancing speech comprehension for language learners, or analyzing audio data at a reduced pace for research or educational purposes.

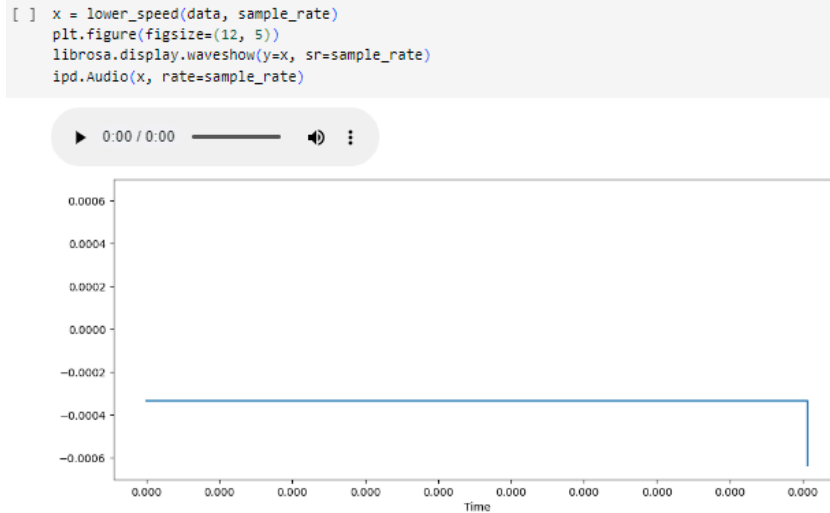


Fig: 30: Audio with Lower Speed

3.2.7. Exposing Dataset

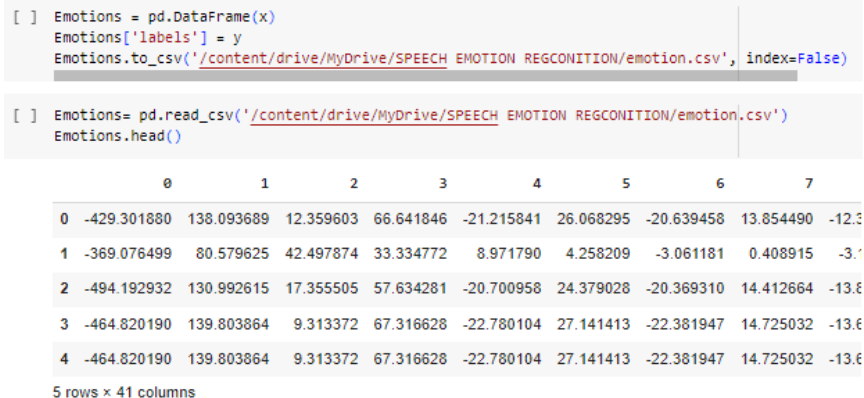


Fig: 31: Exposing dataset

Due to the multiclass classifier, we performed one hot encoding in which normalization of data is done.

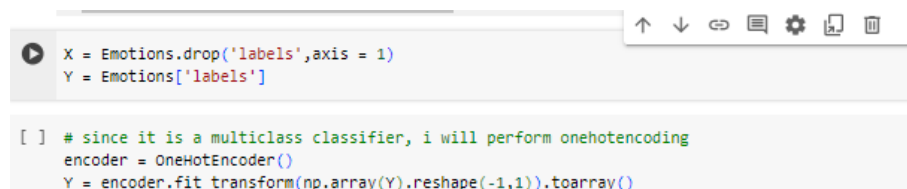


Fig: 32: One hot encoding

3.2.8. Training and Testing

By importing `train_to_split` from decision tree regression we separated data in two parts test data and training data

```
[ ] # scaling our data with sklearn's Standard scaler
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
x_train.shape, y_train.shape, x_test.shape, y_test.shape

((20680, 40), (20680, 8), (5170, 40), (5170, 8))

[ ] # decision tree
from sklearn.tree import DecisionTreeClassifier
clf = DecisionTreeClassifier()
clf.fit(x_train,y_train)

DecisionTreeClassifier()

[ ] y_pred = clf.predict(x_test)

[ ] print(f'the training score for decision tree is {clf.score(x_train,y_train)}')

the training score for decision tree is 0.9996131528046421

[ ] print(f'the test score for decision tree is {clf.score(x_test,y_test)}')

the test score for decision tree is 0.8071566731141199
```

Fig: 33: Training and Testing

3.2.9. MLP Classifier

```
[ ] from sklearn.neural_network import MLPClassifier
clf2=MLPClassifier(alpha=0.01, batch_size=270, epsilon=1e-08, hidden_layer_sizes=(400,), learr
clf2.fit(x_train,y_train)

/usr/local/lib/python3.10/dist-packages/sklearn/neural_network/_multilayer_perceptron.py:686: (
warnings.warn(

MLPClassifier
MLPClassifier(alpha=0.01, batch_size=270, hidden_layer_sizes=(400,),
learning_rate='adaptive', max_iter=400)
```

Fig: 34: MLP Classifier

Applying CNN to the train and test data:

```
[ ] #CNN
x_traincnn = np.expand_dims(x_train, axis=2)
x_testcnn = np.expand_dims(x_test, axis=2)
x_traincnn.shape, y_train.shape, x_testcnn.shape, y_test.shape

((20680, 40, 1), (20680, 8), (5170, 40, 1), (5170, 8))
```

Fig: 35: Applying CNN to train and test data

3.2.10. Data Preprocessing

1. Normalize the extracted features to ensure they are centered around zero with a standard deviation of one.

2. Explore the distribution of emotion labels in the dataset and handle class imbalances if present.
3. Encode categorical emotion labels into numerical representations (e.g., one-hot encoding).
4. Handle missing or noisy data through techniques like data imputation or removal of problematic samples.
5. Explore techniques for data augmentation to increase the diversity of the training set.
6. Consider scaling features to a specific range if required by the chosen machine learning algorithm.
7. Implement a data preprocessing pipeline to automate these steps for future use.
8. Document any data preprocessing decisions made, including the reasoning behind them.
9. Address any potential biases introduced during data preprocessing.
10. Verify that the preprocessed data maintains its interpretability for later analysis.

3.2.11. Model Selection

1. Research and understand the characteristics of different machine learning models suitable for speech emotion recognition.
2. Choose models based on the size of your dataset, computational resources, and the complexity of the problem.
3. Experiment with traditional machine learning algorithms like Support Vector Machines (SVM) or Random Forests.
4. Explore the potential of deep learning models such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) for improved performance.
5. Consider ensemble methods to combine predictions from multiple models.
6. Investigate pre-trained models or transfer learning approaches for leveraging existing knowledge in related tasks.
7. Evaluate the trade-offs between model complexity and interpretability.
8. Implement a baseline model to establish a performance benchmark.
9. Experiment with different hyperparameters to fine-tune model performance.
10. Document the rationale behind the model selection, including comparisons between different models.

```
[ ] early_stopping = tf.keras.callbacks.EarlyStopping(
    monitor='val_loss',
    patience=5,
    verbose=2,)

[ ] my_model=Sequential()
my_model.add(Conv1D(256, kernel_size=7, strides=1, padding='same', activation='relu', input_shape=(x_train.shape[1], 1)))
my_model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))
my_model.add(BatchNormalization())

my_model.add(Conv1D(128, kernel_size=7, strides=1, padding='same', activation='relu'))
my_model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))
my_model.add(BatchNormalization())

my_model.add(Conv1D(128, kernel_size=7, strides=1, padding='same', activation='relu'))
my_model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))
my_model.add(BatchNormalization())

my_model.add(Conv1D(64, kernel_size=7, strides=1, padding='same', activation='relu'))
my_model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))
my_model.add(BatchNormalization())
my_model.add(Dropout(0.5))

my_model.add(Flatten())
my_model.add(Dense(units=32, activation='relu'))
my_model.add(Dropout(0.5))

my_model.add(Dense(units=8, activation='softmax'))
my_model.compile(optimizer = 'adam' , loss = 'categorical_crossentropy' , metrics = ['accuracy'])

my_model.summary()
```

[] Model: "sequential"

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 40, 256)	2048
max_pooling1d (MaxPooling1D)	(None, 20, 256)	0
batch_normalization (Batch Normalization)	(None, 20, 256)	1024
conv1d_1 (Conv1D)	(None, 20, 128)	229504
max_pooling1d_1 (MaxPooling1D)	(None, 10, 128)	0
batch_normalization_1 (Batch Normalization)	(None, 10, 128)	512
conv1d_2 (Conv1D)	(None, 10, 128)	114816
max_pooling1d_2 (MaxPooling1D)	(None, 5, 128)	0
batch_normalization_2 (Batch Normalization)	(None, 5, 128)	512
conv1d_3 (Conv1D)	(None, 5, 64)	57408
max_pooling1d_3 (MaxPooling1D)	(None, 3, 64)	0
batch_normalization_3 (Batch Normalization)	(None, 3, 64)	256
dropout (Dropout)	(None, 3, 64)	0

flatten (Flatten)	(None, 192)	0
dense (Dense)	(None, 32)	6176
dropout_1 (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 8)	264
=====		
Total params: 412520 (1.57 MB)		
Trainable params: 411368 (1.57 MB)		
Non-trainable params: 1152 (4.50 KB)		

Fig: 36: Model Selection

3.2.12. Model Training

1. Split the preprocessed dataset into training and testing sets, maintaining a consistent distribution of emotions in both sets.
2. Implement cross-validation if the dataset size allows for it to obtain more robust performance estimates.
3. Monitor the training process, observing metrics such as loss and accuracy over epochs.
4. Implement early stopping to prevent overfitting if necessary.
5. Experiment with different optimization algorithms and learning rates.
6. Regularize the model to improve generalization on unseen data.
7. Visualize training and validation curves to identify potential issues or areas for improvement.
8. Save the trained model and its weights for later use.
9. Evaluate the impact of different batch sizes on training efficiency.
10. Document the training process, including hyperparameters and any challenges encountered.

```
history = my_model.fit(x_traincnn, y_train, batch_size=32, epochs=50, validation_data=(x_testcnn, y_test), callbacks = early_stopping)

Epoch 1/50
647/647 [=====] - 31s 41ms/step - loss: 1.1721 - accuracy: 0.5786 - val_loss: 0.8588 - val_accuracy: 0.6627
Epoch 2/50
647/647 [=====] - 26s 41ms/step - loss: 0.8851 - accuracy: 0.6636 - val_loss: 0.7483 - val_accuracy: 0.7091
Epoch 3/50
647/647 [=====] - 27s 42ms/step - loss: 0.8110 - accuracy: 0.6884 - val_loss: 0.6904 - val_accuracy: 0.7217
Epoch 4/50
647/647 [=====] - 26s 41ms/step - loss: 0.7434 - accuracy: 0.7162 - val_loss: 0.6664 - val_accuracy: 0.7509
Epoch 5/50
647/647 [=====] - 26s 40ms/step - loss: 0.7004 - accuracy: 0.7319 - val_loss: 0.6011 - val_accuracy: 0.7679
Epoch 6/50
647/647 [=====] - 26s 40ms/step - loss: 0.6475 - accuracy: 0.7502 - val_loss: 0.5774 - val_accuracy: 0.7826
Epoch 7/50
647/647 [=====] - 26s 40ms/step - loss: 0.5985 - accuracy: 0.7750 - val_loss: 0.5345 - val_accuracy: 0.7865
Epoch 8/50
647/647 [=====] - 26s 40ms/step - loss: 0.5525 - accuracy: 0.7987 - val_loss: 0.4749 - val_accuracy: 0.8232
Epoch 9/50
647/647 [=====] - 26s 40ms/step - loss: 0.4897 - accuracy: 0.8211 - val_loss: 0.4312 - val_accuracy: 0.8364
Epoch 10/50
647/647 [=====] - 26s 40ms/step - loss: 0.4548 - accuracy: 0.8349 - val_loss: 0.4105 - val_accuracy: 0.8468
Epoch 11/50
647/647 [=====] - 26s 40ms/step - loss: 0.4162 - accuracy: 0.8490 - val_loss: 0.3579 - val_accuracy: 0.8739
Epoch 12/50
647/647 [=====] - 26s 40ms/step - loss: 0.3721 - accuracy: 0.8662 - val_loss: 0.3442 - val_accuracy: 0.8720
Epoch 13/50
647/647 [=====] - 26s 40ms/step - loss: 0.3352 - accuracy: 0.8851 - val_loss: 0.3257 - val_accuracy: 0.8878
Epoch 14/50
647/647 [=====] - 26s 40ms/step - loss: 0.3066 - accuracy: 0.8916 - val_loss: 0.3378 - val_accuracy: 0.8874
Epoch 15/50
647/647 [=====] - 26s 40ms/step - loss: 0.2825 - accuracy: 0.9029 - val_loss: 0.3042 - val_accuracy: 0.8985
Epoch 16/50
647/647 [=====] - 26s 40ms/step - loss: 0.2479 - accuracy: 0.9159 - val_loss: 0.3049 - val_accuracy: 0.8996
Epoch 17/50
647/647 [=====] - 26s 40ms/step - loss: 0.2318 - accuracy: 0.9232 - val_loss: 0.3112 - val_accuracy: 0.9002
Epoch 18/50
```

```

647/647 [=====] - 26s 41ms/step - loss: 0.2217 - accuracy: 0.9289 - val_loss: 0.2859 - val_accuracy: 0.9110
Epoch 19/50
647/647 [=====] - 26s 40ms/step - loss: 0.2048 - accuracy: 0.9344 - val_loss: 0.2852 - val_accuracy: 0.9116
Epoch 20/50
647/647 [=====] - 26s 41ms/step - loss: 0.1811 - accuracy: 0.9406 - val_loss: 0.2735 - val_accuracy: 0.9133
Epoch 21/50
647/647 [=====] - 26s 40ms/step - loss: 0.1649 - accuracy: 0.9460 - val_loss: 0.2906 - val_accuracy: 0.9159
Epoch 22/50
647/647 [=====] - 26s 40ms/step - loss: 0.1721 - accuracy: 0.9456 - val_loss: 0.2907 - val_accuracy: 0.9122
Epoch 23/50
647/647 [=====] - 26s 40ms/step - loss: 0.1522 - accuracy: 0.9508 - val_loss: 0.2461 - val_accuracy: 0.9261
Epoch 24/50
647/647 [=====] - 26s 40ms/step - loss: 0.1436 - accuracy: 0.9549 - val_loss: 0.3270 - val_accuracy: 0.9122
Epoch 25/50
647/647 [=====] - 26s 40ms/step - loss: 0.1396 - accuracy: 0.9570 - val_loss: 0.2653 - val_accuracy: 0.9259
Epoch 26/50
647/647 [=====] - 26s 40ms/step - loss: 0.1319 - accuracy: 0.9580 - val_loss: 0.2522 - val_accuracy: 0.9292
Epoch 27/50
647/647 [=====] - 26s 40ms/step - loss: 0.1174 - accuracy: 0.9638 - val_loss: 0.2668 - val_accuracy: 0.9288
Epoch 28/50
647/647 [=====] - 26s 40ms/step - loss: 0.1194 - accuracy: 0.9633 - val_loss: 0.2701 - val_accuracy: 0.9263
Epoch 28: early stopping

```

Fig: 37: Model Training

3.2.13. Model Evaluation

1. Assess the model's performance using evaluation metrics such as accuracy, precision, recall, and F1-score.
2. Examine the confusion matrix to understand how well the model is classifying each emotion.
3. Investigate any misclassifications and analyze common patterns or challenges.
4. Explore metrics specific to the emotion recognition task, such as weighted accuracy or emotion-specific metrics.
5. Consider using receiver operating characteristic (ROC) curves and area under the curve (AUC) for binary emotion classification.
6. Evaluate the model's performance across different demographic groups to identify potential biases.
7. Implement cross-validation to obtain more reliable performance estimates.
8. Perform statistical significance tests to compare the performance of different models or configurations.
9. Explore the impact of feature choices on model performance.
10. Document the evaluation results and any insights gained for future reference.

```
[ ] df = pd.DataFrame(columns=['Predicted Labels', 'Actual Labels'])
df['Predicted Labels'] = y_pred.flatten()
df['Actual Labels'] = y_test.flatten()

[ ] df
```

	Predicted Labels	Actual Labels
0	fear	fear
1	angry	angry
2	happy	happy
3	fear	fear
4	fear	fear
...
5165	neutral	neutral
5166	neutral	neutral
5167	neutral	neutral
5168	disgust	disgust
5169	sad	sad

5170 rows x 2 columns

Fig: 38: Model Evaluation

3.2.14. Convolutional Neural Network

CNN output will be visualized with the help of confusion_matrix from sklearn.metrics

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize = (12, 10))
cm = pd.DataFrame(cm, index = [i for i in encoder.categories_], columns = [i for i in encoder.categories_])
sns.heatmap(cm, linewidth=1, cmap='Blues', linewidth=1, annot=True, fmt='')
plt.title('Confusion Matrix', size=20)
plt.xlabel('Predicted Labels', size=14)
plt.ylabel('Actual Labels', size=14)
plt.show()
```

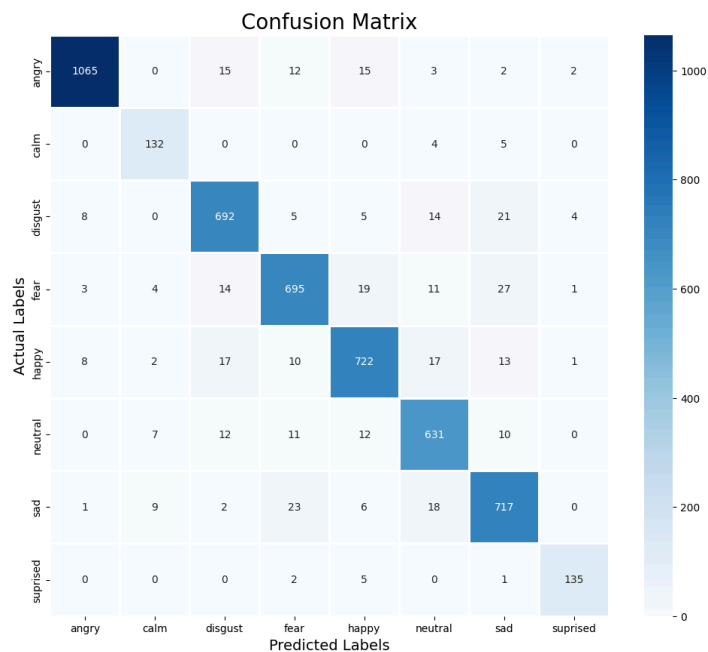


Fig: 39: Confusion Matrix for CNN Model

Output in the form of classification of rows and columns

```
[ ] print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
angry	0.98	0.96	0.97	1114
calm	0.86	0.94	0.89	141
disgust	0.92	0.92	0.92	749
fear	0.92	0.90	0.91	774
happy	0.92	0.91	0.92	790
neutral	0.90	0.92	0.91	683
sad	0.90	0.92	0.91	776
surprised	0.94	0.94	0.94	143
accuracy			0.93	5170
macro avg	0.92	0.93	0.92	5170
weighted avg	0.93	0.93	0.93	5170

Fig: 40: Classification report (CNN)

3.2.15. Support Vector Machine

visualize confusion matrix for SVM model

```
[ ] ## saving model
import os
from datetime import datetime
def save_model(model , suffix = None) :
    model_dir = os.path.join('/content/drive/MyDrive/SPEECH EMOTION RECONITION',datetime.now().strftime('%Y%m%d-%H%M%S'))
    model_path = model_dir + '-' + suffix + '.h5'
    print(f'saving model into {model_path}')
    model.save(model_path)
    return model_path

[ ] save_model(my_model,suffix = 'speech_emotion_recognition_model')

saving model into /content/drive/MyDrive/SPEECH EMOTION RECONITION/20231023-131929-speech_emotion_recognition_model.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3000: UserWarning: You are saving your model as an H
saving_api.save_model(
'/content/drive/MyDrive/SPEECH EMOTION RECONITION/20231023-131929-speech_emotion_recognition_model.h5'

[ ] #Visualize Confusion Matrix For SVM model
result_cf = confusion_matrix (y_test,y_pred)
sns.set (font_scale=1.2)
plt.figure(figsize=(8,8))
label_name = np.unique(y_test)
sns.heatmap (result_cf,annot=True,cmap='Blues',fmt='g',xticklabels=label_name, yticklabels=label_name)
sns.set (font_scale=1.2)

plt.title ('Confusion Matrix For SVM Model')
plt.xlabel ('Predicted Label')
plt.ylabel ('True Label')
plt.show ()
```

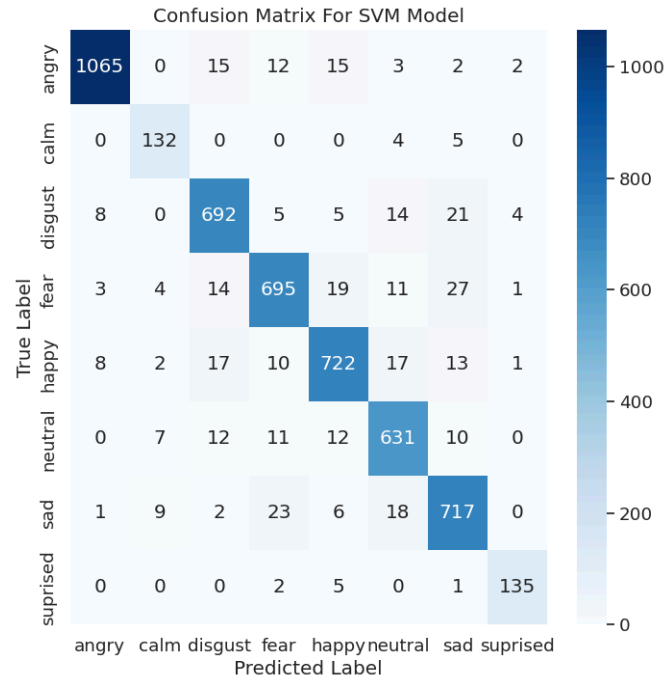


Fig: 41: Confusion Matrix for SVM Model

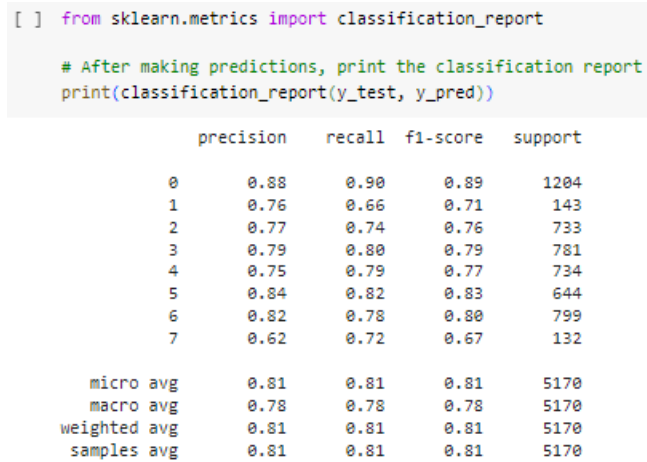


Fig: 42: Classification report (SVM)

3.2.16. Random Forest Classifier

By importing randomforestclassifier we can predict the accuracy of our project

```
[ ] from sklearn.ensemble import RandomForestClassifier
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import accuracy_score
    #from sklearn.feature_extraction import FeatureExtractor

    # Load dataset, where X contains audio features and y contains emotion labels
    # replace this with your actual data loading code.
    #X, y = load_data()

    # Split the 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)

    # Initialize and train the Random Forest classifier
    clf = RandomForestClassifier(n_estimators=100, random_state=42)
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    # Calculate the accuracy of the model
    accuracy = accuracy_score(y_test, y_pred)
    print ("Accuracy:", (accuracy*100))
```

Fig: 43: Random Forest Classifier Model

The total accuracy got for our project is:

Accuracy: 89.69052224371373

3.3 DATA PREPARATION

Data preparation is a critical stage in speech emotion recognition (SER), ensuring the quality and relevance of input data for accurate analysis. In SER, data preparation involves several key steps. Firstly, raw audio files are collected from various sources, such as recordings or databases, and preprocessed to remove noise and normalize audio levels. Feature extraction follows, where relevant features like pitch, intensity, and spectral characteristics are extracted from the audio signals using techniques like Mel-frequency cepstral coefficients (MFCCs) or prosodic features.

Next, the data is labeled with corresponding emotion categories, a process that may involve manual annotation or automated algorithms for labeling. Data augmentation techniques like adding background noise or altering pitch and speed can help diversify the dataset and improve model robustness.

Overall, thorough data preparation is essential for enhancing the effectiveness of SER models by ensuring they are trained on high-quality, diverse, and properly labeled datasets, ultimately improving their ability to accurately recognize and classify emotions in speech signals.

RAVDESS DATA:

The RAVDESS contains 1440 files: 60 trials per actor x 24 actors = 1440. The RAVDESS contains 24 professional actors (12 female, 12 male), vocalizing two lexically-matched statements in a neutral North American accent. Speech emotions include calm, happy, sad, angry, fearful, surprise, and disgust expressions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression.

File naming convention

Each of the 1440 files has a unique filename. The filename consists of a 7-part numerical identifier (e.g., 03-01-06-01-02-01-12.wav). These identifiers define the stimulus characteristics:

Filename identifiers

1. Modality (01 = full-AV, 02 = video-only, 03 = audio-only).
2. Vocal channel (01 = speech, 02 = song).
3. Emotion (01 = neutral, 02 = calm, 03 = happy, 04 = sad, 05 = angry, 06 = fearful, 07 = disgust, 08 = surprised).

4. Emotional intensity (01 = normal, 02 = strong). NOTE: There is no strong intensity for the 'neutral' emotion.
5. Statement (01 = "Kids are talking by the door", 02 = "Dogs are sitting by the door").
6. Repetition (01 = 1st repetition, 02 = 2nd repetition).
7. Actor (01 to 24. Odd numbered actors are male, even numbered actors are female).

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

1. Audio-only (03)
2. Speech (01)
3. Fearful (06)
4. Normal intensity (01)
5. Statement "dogs" (02)
6. 1st Repetition (01)
7. 12th Actor (12)

Female, as the actor ID number is even.

TESS DATA

A set of 200 target words were spoken in the carrier phrase "Say the word _" by two actresses (aged 26 and 64 years) and recordings were made of the set portraying each of seven emotions (anger, disgust, fear, happiness, pleasant surprise, sadness, and neutral). There are 2800 data points (audio files) in total.

CREMA-D DATA

CREMA-D is a data set of 7,442 original clips from 91 actors. These clips were from 48 male and 43 female actors between the ages of 20 and 74 coming from a variety of races and ethnicities (African American, Asian, Caucasian, Hispanic, and Unspecified). Actors spoke from a selection of 12 sentences. The sentences were presented using one of six different emotions (Anger, Disgust, Fear, Happy, Neutral, and Sad) and four different emotion levels (Low, Medium, High, and Unspecified).

CHAPTER 4

RESULTS AND DISCUSSION

The field of SER has been thoroughly researched over the past 20 years. Many methods for extracting speech features have been investigated and put into practice. For efficient emotion recognition, a variety of supervised and unsupervised classification algorithms have been tested. However, Emotions are subjective, so there's no consensus on how to measure or categorize them.

Recently, there have been some encouraging results in the recognition of emotions using deep learning techniques. Because of their advantages over traditional techniques, such as scalability, all-purpose parameter fitting, and infinite flexibility, they are thought to be the most suitable for the SER framework.

Traditional classifiers, however, in contrast, are quick because they require less training data. In SER, machine learning techniques have been widely used. These include more sophisticated approaches like neural networks and more conventional support vector machines (SVM) classifiers.

4.1. Convolutional Neural Network (CNN)

Speech data from the training set is used to train a convolutional neural network (CNN). The accuracy of the CNN, RNN, and MLP models' training and verification sets is shown in Figure. Figure illustrates how accuracy tends to rise with increasing iteration times for both the training and verification sets, but especially for the training set. The training set's accuracy is over 99% and the verification set's accuracy is over 87% after 200 iterations. Compared to RNN and MLP, the CNN model created in this work has higher accuracy in both the training and verification sets.

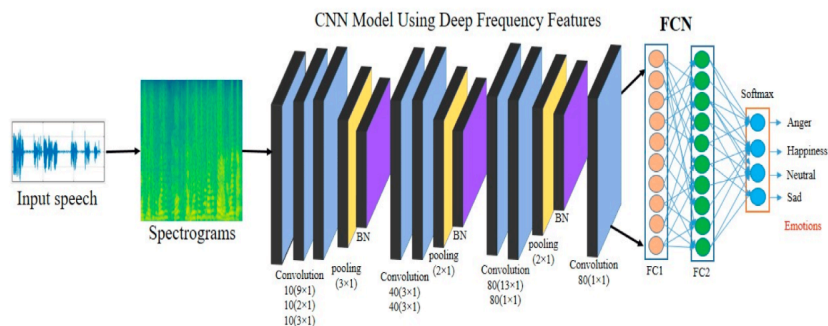


Fig: 44: CNN Model Using Deep Frequency Features

The training and testing data is labeled as actual labels and predicted labels from the extracted features using MFCC. This allows the CNN algorithm to calculate the precision, recall, f1-score, and support. The confusion matrix and accuracy for the recognized emotions is shown below:

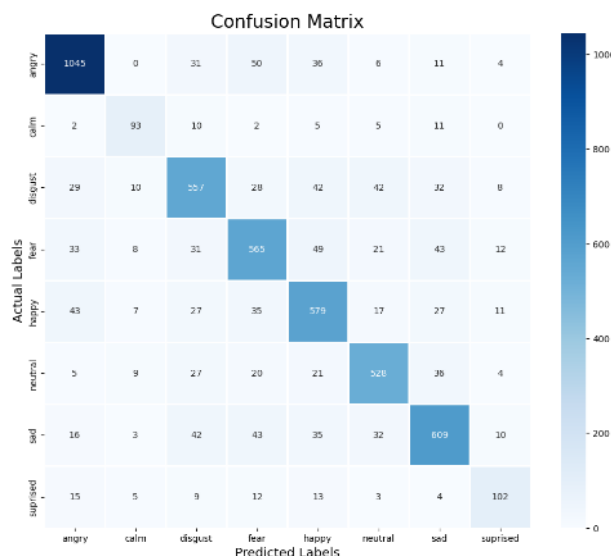


Fig: 45: Confusion Matrix (CNN)

	Precision	recall	f1-score	support
angry	0.98	0.96	0.97	1114
calm	0.86	0.94	0.89	141
disgust	0.92	0.92	0.92	749
fear	0.92	0.90	0.91	774
happy	0.92	0.91	0.92	790
neutral	0.90	0.92	0.91	683
sad	0.90	0.92	0.91	776
surprised	0.94	0.94	0.94	143
accuracy			0.93	5170
macro avg	0.92	0.93	0.92	5170
weighted avg	0.93	0.93	0.93	5170

Fig: 46: CNNs Classification report

Therefore the overall accuracy for all emotions is 93% which is getting the most accuracy then all the algorithms like SVM and MLP.

4.2. Decision Tree Classifier:

For the decision tree the input space is recursively divided into regions, and each region is given a label or value based on the average value or majority class of the training samples in that region. This is how it operates. A Decision Tree Classifier can be described as follows in brief: In addition, overfitting issues with individual Decision Trees are frequently addressed by ensemble techniques such as Random Forests. The training score for the dataset using decision

tree classifier after prediction gives the accuracy with 0.99% and for the test score the accuracy is 0.78%.

4.3. MLP Classifier

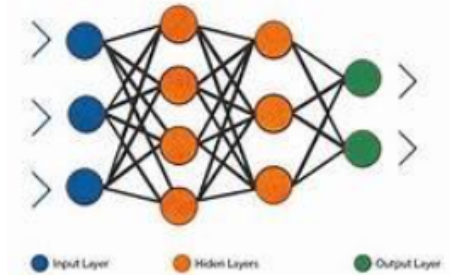


Fig: 47: MLP Classifier Model

In SER, MLP classifiers are used to identify and categorize spoken language emotions. Sentiment analysis, affective computing, human-computer interaction, and other fields can all benefit from the model's ability to grasp the intricate relationships between acoustic properties and emotional states. The training set's accuracy used in the task of classification gets 0.99% and test score gets an accuracy value of 0.87%.

4.4. Support Vector Machine(SVM)

SVMs can use the kernel trick to convert the input features into a higher-dimensional space, which enables the algorithm to identify more intricate relationships in the data. Radial basis function (RBF) kernels, polynomial, and linear kernels are examples of common kernel functions. The data points that are closest to the decision boundary (hyperplane) are known as support vectors.

	Precision	recall	f1-score	support
0	0.87	0.88	0.88	1152
1	0.71	0.77	0.74	121
2	0.73	0.76	0.75	743
3	0.76	0.79	0.77	769
4	0.81	0.76	0.78	834
5	0.83	0.81	0.82	639
6	0.81	0.78	0.79	793
7	0.61	0.76	0.68	119
micro avg	0.80	0.80	0.80	5170
macro avg	0.77	0.79	0.78	5170
weighted avg	0.80	0.80	0.80	5170
samples avg	0.80	0.80	0.80	5170

Fig: 48: SVMs Classification report

The accuracy for the support vector machine of SER for the labeled emotions is 80%

4.5. Random Forest

The Random Forest classifier is a powerful and widely used machine learning algorithm known for its robustness and versatility in classification tasks. It belongs to the ensemble learning methods, where it combines multiple decision trees to make predictions.

In a Random Forest, a multitude of decision trees are trained on random subsets of the training data, using a technique known as bagging (Bootstrap Aggregating). Each tree is trained independently, and during prediction, the class with the most votes from all trees is selected as the final prediction. This ensemble approach helps reduce overfitting and increases the overall accuracy and stability of the model.

Random Forests offer several advantages, including:

4.5.1. Robustness to Overfitting: By averaging predictions over multiple trees, Random Forests are less prone to overfitting compared to individual decision trees.

Handling large datasets: Random Forests can efficiently handle large datasets with high dimensionality and mixed data types.

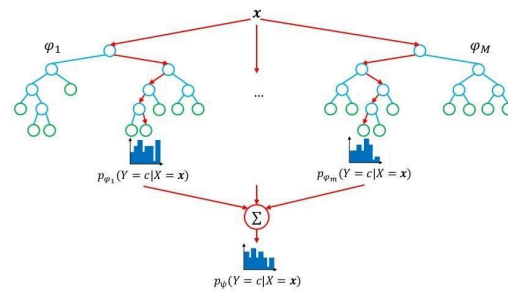
4.5.2. Feature Importance: They provide a measure of feature importance, allowing users to identify which features are most influential in making predictions.

4.5.3. Out-of-bag (OOB) error estimation: Random Forests use OOB samples for internal validation, providing an estimate of the model's performance without the need for a separate validation set.

Overall, the Random Forest classifier is a versatile and effective algorithm suitable for various classification tasks, from healthcare and finance to image recognition and natural language processing. Its simplicity, scalability, and interpretability make it a popular choice among data scientists and machine learning practitioners.

The capacity of Random Forest to measure the significance of various features in the classification task is one of its benefits. Mel Frequency Cepstral Coefficients (MFCCs), pitch, intensity, and other acoustic properties could all be considered features in the context of SER. The Random Forest algorithm offers valuable insights into the features that are most important for accurately classifying emotions. The random forest algorithm has an 89% accuracy rate.

Random forests



Randomization

- Bootstrap samples
- Random selection of $K \leq p$ split variables
- Random selection of the threshold

} Random Forests

} Extra-Trees

14 / 39

Fig: 49: Random Forest Model

The random forest algorithm has an 89% accuracy rate.

4.6. The Heat Map For Labeled MFCC Coefficients

A heatmap uses a color gradient to graphically depict the intensity of a variable, in this case, the MFCC coefficients. Typically, the time or frames are displayed on the x-axis, and the individual MFCC coefficients are displayed on the y-axis. Over time, different emotional states may show up as different patterns of MFCC coefficients. Variations in the intensity of particular coefficients at various points during the utterance can be displayed on a heatmap. The heatmap can be used to visualize consistent patterns in the MFCC coefficients that may be related to specific emotions. Heatmaps are a useful tool for assessing SER model performance.

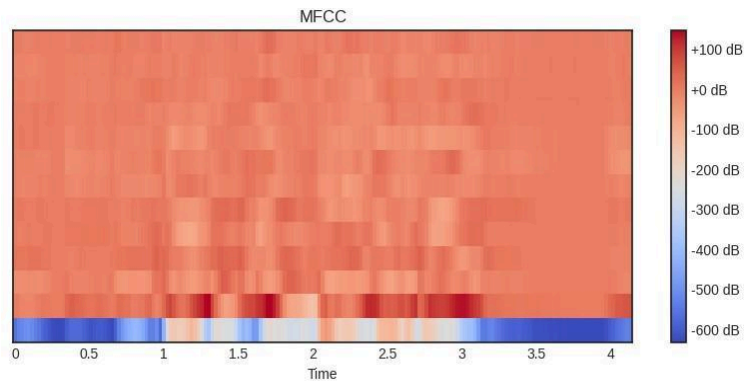


Fig: 50: Heat map for labeled MFCC coefficients

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1. CONCLUSION

In recent years, researchers both domestically and internationally have been very interested in One of the fundamental technologies in systems for human-computer interaction, SER technology, because of its capacity to precisely identify feelings as well as and thereby enhance the level of interaction between humans and computers. We present a fused feature SER using a deep learning algorithm in this paper.

We found it easy to classify and feature because of its noiseless data.Utilizing CNNs for sequence coding transformation and spatial feature representation allowed us to achieve a 93% accuracy on the RAVDESS dataset's holdout test set. The results analysis leads to the consideration of using semantics in conjunction with speech to text conversion for the purpose of identifying emotions. With an overall emotion detection accuracy of 80%, the SVM model outperforms the CNN methods by almost 13%. When speech features are combined with speech transcriptions, better outcomes are seen. The class accuracy for the Random Forest model is 89%, and the class accuracy for the MLPclassifier model is 87%. In contrast, the Decision Tree Classifier Model yields an accuracy of 0.99% for training scores and 0.78% for test scores overall. Speech sentiment and emotion recognition can improve communication in emotion-related applications such as social and conversational robots, which can benefit from the use of the suggested models.

5.2. FUTURE SCOPE

In Future,

1. *Healthcare Applications:* Speech emotion recognition has potential applications in healthcare, particularly in mental health assessments. Future projects might aim to develop tools for early detection of emotional disorders or to assist in therapeutic interventions.
2. *Real-time Applications:* There is a growing demand for real-time emotion recognition applications in various domains, including customer service, human-computer interaction, and mental health. Future projects may emphasize developing systems capable of providing instantaneous feedback.
3. *Human-Robot Interaction:* Emotion recognition in speech can play a vital role in improving human-robot interaction. Future projects may explore how robots and virtual agents can better understand and respond to human emotions, enhancing the overall user experience.
4. *Ethical Considerations and Bias Mitigation:* As with any AI technology, addressing ethical concerns and mitigating biases in emotion recognition systems will be crucial. Future projects may focus on developing fair and unbiased models to avoid reinforcing stereotypes or perpetuating discrimination.

BIBLIOGRAPHY

- [1] Speech Emotion Recognition Using Deep Learning Techniques: A Review, published by the Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah (IEEE), 2019.
- [2] The Malaysian Ministry of Education, under Grant FRGS/1/2018/ICT02/UIAM/02/4, "A Comprehensive Review of Speech Emotion Recognition Systems," (IEEE), 2021, funded by Fundamental Research Grant, FRGS19-076-0684.
- [3] Preethi Jeevan, Professor, Department of Computer Science and Engineering, SNIST, Hyderabad-501301, India, Speech Emotion Recognition Using Machine Learning (IRJET), 2023.
- [4] In the 2018 International Conference on Network Infrastructure and Digital Content (IC-NIDC), pages 193–197, G. Liu, W. He, and B. Jin present their work on "Feature fusion of speech emotion recognition based on deep Learning."
- [5] Emotion recognition: a survey by Matilda S. 2015;3(1):14–19; International Journal of Advanced Computer Research.
- [6] Rao KS, Koolagudi SG. A review of speech recognition for emotions. International Journal of Speech Recognition 2012.
- [7] "Emotion Recognition and Affective Computing on Vocal Social Media," Inf. Manag., Feb. 2015, W. Dai, D. Han, Y. Dai, and D. Xu.
- [8] "Emotional speech recognition: Resources, features, and methods," D. Ververidis and C. Kotropoulos, Speech Commun., vol. 48, no. 9, pp. 1162–1181, 2006.
- [9]. Mandar Gilke , Pramod Kachare , Rohit Kothalikar , Varun Pius Rodrigues and Madhavi Pednekar. MFCCbased vocal emotion recognition using ANN, International Conference on Electronics Engineering and Informatics, 150-154, 2012.
- [10] Jouni Pohjalainen and Paavo Alku. Multi-scale modulation filtering in automatic detection of emotions in telephone speech. International Conference on Acoustic, Speech and Signal Processing, 980-984, 2014.

SESHADRI RAO GUDLAVALLERU ENGINEERING COLLEGE

(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)
Seshadri Rao Knowledge Village, Gudlavalleru

Department of Computer Science and Engineering

Program Outcomes (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions., component, or software to meet the desired needs.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs)

PSO1 : Design, develop, test and maintain reliable software systems and intelligent systems.

PSO2 : Design and develop web sites, web apps and mobile apps.

PROJECT PROFORMA

Classification of Project	Application	Product	Research	Review
	√			

Note: Tick Appropriate category

Project Outcomes	
Course Outcome (CO1)	Identify and analyze the problem statement using prior technical knowledge in the domain of interest.
Course Outcome (CO2)	Design and develop engineering solutions to complex problems by employing systematic approach.
Course Outcome (CO3)	Examine ethical, environmental, legal and security issues during project implementation.
Course Outcome (CO4)	Prepare and present technical reports by utilizing different visualization tools and evaluation metrics.

Mapping Table

CS3518 : MAIN PROJECT															
Course Outcomes	Program Outcomes and Program Specific Outcome														
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12		PSO 1	PSO 2
CO1	3	3	1					2	3	2				1	1
CO2	2	2	2	3	3			3	2	2		3		3	3
CO3	2	3	3	2	2	3	3	2	3	2	2			2	
CO4	3		2		3				2	3	2	2		2	3

Note: Map each project outcomes with POs and PSOs with either 1 or 2 or 3 based on level of mapping as follows:

1-Slightly (Low) mapped 2-Moderately (Medium) mapped 3-Substantially (High) mapped

Acceptance Letter form IEEE International Conference on Emerging Systems and Intelligent Computing 2024:

Dear TUMMALACHARLA DIVYA,

Greetings from International Conference on Emerging Systems and Intelligent Computing!

Congratulations! We are pleased to inform you that your Paper ID "219" Titled "Recognition of emotional speech using MFCC and Machine Learning Technique" has been accepted provisionally for Oral Presentation / Poster and possible inclusion in the "The 4th International Conference on Emerging Systems and Intelligent Computing (ESIC- 2024) " to be organized at School of Computer Engineering, KIIT Deemed-to-be University, Bhubaneswar scheduled to be held during 9th to 10th February 2024.

Authors are requested to follow the instructions below for Final Camera Ready Paper Submission:-

1. The authors are urged to ensure that the camera ready paper strictly follows the IEEE Template given on the conference website. The camera ready papers that do not meet the template requirements may not be submitted for further consideration to IEEE Xplore.
2. Please make sure that the Plagiarism of your final camera ready paper should be below 20%, using standard plagiarism checking software, before submitting the camera ready paper from your end to confirm the inclusion of the paper in IEEE Xplore.
3. The number of pages in Final Camera Ready Paper must be within 06 Pages strictly.
4. IEEE is very strict about the compliance requirements for PDF files for inclusion in the IEEE Xplore® Digital Library. We strongly recommend using IEEE PDF eXpress® site <https://www.pdf-express.org/> to check your final version of the paper. Please log in using the conference code ID: 60604x
5. At least one author must register with Full Author Registration and present the accepted paper at International Conference on Emerging Systems and Intelligent Computing, in order to the paper to be considered for inclusion in IEEE Xplore.
6. The authors will have to submit a camera-ready version as per the guidelines given on the conference website on or before 10th January 2024 (The upload link will be send to you soon).

7. IEEE Copyright Form also need to be signed and uploaded in the corresponding section. The link for IEEE copyright will be sent to you after registration process is over.

8. For paper registration follow the link: <https://icesic.com/registration>

Please check the conference website <https://icesic.com/> for regular updates regarding submission guidelines for the camera ready paper, registration details and other important information.

Congratulations and Best of Luck !

With Best Regards,

Organizing Committee

International Conference on Emerging Systems and Intelligent Computing

To stop receiving conference emails, you can check the 'Do not send me conference email' box from your User Profile.

Microsoft respects your privacy. To learn more, please read our Privacy Statement.

Microsoft Corporation

One Microsoft Way

Redmond, WA 98052