

A

# PROJECT SCHOOL REPORT

on

## SPEECH EMOTION RECOGNIZER USING CNN

Submitted By

Abhijeet Gowlikar	245322733129
Akanksha Rondla	245322733132
Gangula Venkata Raja Vineela	245322733143
Indrakar Gaurav	245322733150
Shaik Ayisha	245322733180
Udata Lekhana Surya Bhanu	245322733188

Under the guidance of

**P V N Balarama Murthy**



**NEIL GOGTE INSTITUTE OF TECHNOLOGY**

Kachavanisingaram Village, Hyderabad, Telangana 500058.



**NEIL GOGTE INSTITUTE OF TECHNOLOGY**

A Unit of Keshav Memorial Technical Education (KMTES)

Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

## **CERTIFICATE**

*This is to certify that the project work entitled “Speech Emotion Recognition” is a bonafide work carried out by ABHIJEET GOWLIKAR (245322733129), AKANKSHA RONDLA (245322733132), GANGULA VENKATA RAJA VINEELA (245322733143), INDRAKAR GAURAV(245322733150), SHAIK AYISHA(245322733180), UDATA LEKHANA SURYA BHANU(245322733188) of III year V semester Bachelor of Engineering in CSE during the academic year 2024-2025 and is a record of bonafide work carried out by them.*

**P V N Balarama Murthy**

Project Mentor

# ABSTRACT

Speech Emotion Recognition (SER) is a vital task in various domains, ranging from human-computer interaction (HCI) and customer service to mental health analysis. Speech Emotion Recognition has become one of the most important yet challenging task in real world and it has also gained popularity in recent years. Initially we started with the Deep neural network models which include CNN's (Convolutional Neural Network ), ANN's(Artificial Neural Network), LSTM(Long Short-Term Memory), ResNet9(Residual Network) , RNN(Residual neural network) from scratch using pytorch.

CNN is a powerful deep learning architecture often used in the image processing. While CNN is primarily used in image processing, their ability to extract local features and recognize patterns in 2D data makes them highly effective for our project we used a input 1D matrix which captures the essential elements to detect an emotion from speech signal . In our project we used CNN to detect and classify emotions from speech signals. The system extracts features from audio signal using librosa and an input feature vector is fed to the CNN's first layer. Our model uses convolutional layers to extract localized emotional features, such as variations in pitch and intensity, from time-frequency representation of speech data.

This work paves way for more robust and adaptive SER systems, which can be applied to a wide range of fields such as virtual assistants, healthcare, automated call centers, and interactive entertainment, providing valuable insights into emotional states and facilitating more personalized user experiences.

# TABLE OF CONTENTS

S. NO.	TITLE	PAGE NO
	<b>ABSTRACT</b>	3
	<b>TABLE OF CONTENTS</b>	5
	<b>LIST OF FIGURES</b>	6
<b>1</b>	<b>Introduction</b>	7
	1.1 Problem Statement and Objectives	
	1.2 Motivation	
	1.3 Scope	
<b>2</b>	<b>Literature Survey</b>	10
<b>3</b>	<b>PROPOSED WORK , ARCHITECTURE,TECHNOLOGY STACK &amp; IMPEMENTATION DETAILS</b>	14
	3.1 Proposed Work	
	3.2 Technology Stack	
	3.3 Implementation Details	
<b>4</b>	<b>Results &amp; Discussions</b>	27
	4.1 Result	
	4.2 Output screens	
<b>5</b>	<b>Conclusion &amp; Future Scope</b>	33
<b>6</b>	<b>References</b>	35

# LIST OF FIGURES

<b>Fig. No.</b>	<b>Figure Name</b>	<b>Page No.</b>
<b>1</b>	Accuracy Graph	13
<b>2</b>	Emotion classes in Dataset	19
<b>3</b>	CNN Model Architecture	26
<b>4</b>	Confusion Matrix	28
<b>5</b>	Train and validation accuracy curves	29
<b>6</b>	Precision recall and F1 score	30
<b>7</b>	UI HomePage	31
<b>8</b>	SignUp Page	31
<b>9</b>	SignIn Page	32
<b>10</b>	Predicted Output	33

# LIST OF TABLES

Table No.	Table Name	PageNo.
1	Data preprocessing summary	19

## CHAPTER 1

# INTRODUCTION

### 1.1 Problem Statement and Objectives

**Statement:** Modeling human emotions in speech signals is challenging due to high computational demands and limited emotion-labeled data. This project aims to build a Transformer model for Speech Emotion Recognition. The focus is on creating a scalable, accurate solution for real-world applications.

**Objectives:**

1. To build a Transformer model for accurate Speech Emotion Recognition (SER)
2. Overcome data scarcity with techniques like data augmentation.
3. Ensure Scalability.
4. Testing the model with the different audio-clips.
5. Optimize the resource utilization for real-world deployment.
6. Evaluate the model using metrics to validate its accuracy and performance.

### 1.2 Motivation

The ability to accurately recognize emotions from speech is a powerful tool with wide-ranging applications in both personal and professional contexts. Speech Emotion Recognition (SER) enhances the Human-Computer interaction by making systems emotionally aware, enabling more natural communication. It plays a crucial role in mental health monitoring by detecting emotional patterns in speech. SER improves customer service by analyzing caller emotions and providing real-time feedback. It aids in security, forensics recognizing stress,

excitement. Additionally, SER supports education and accessibility, personalizing learning and helping those with speech disorders convey emotions.

The motivation for this work stems from the need to enhance the performance of existing models in recognizing emotions from speech data. With the rapid advancements in artificial intelligence and machine learning, the demand for real-time applications of emotion recognition in various industries is increasing. Virtual assistants, mental health apps, and customer service automation systems all require more accurate and adaptive models to cater to user's emotional needs.

### **1.3 Scope**

This project focuses on implementation of the Speech Emotion Recognition using the various Deep learning models such as CNN, ANN, LSTM, RNN, ResNet 9 and Transformer based model for the variable length and also uses the Spectrogram analysis.

The scope of this project is to enhance Speech Emotion Recognition (SER) by refining a Conv1D-based model that directly analyzes raw speech signals, focusing on features such as tone, pitch, rhythm rather than relying on text transcripts. Most existing models in SER typically convert speech into text through speech-to-text (STT) systems before analyzing the emotional content

Speech-to-text models can often struggle with accuracy, particularly in noisy environments or when the speaker has a non-standard accent or dialect. Errors in transcription can lead to misinterpretation of the emotional content, as emotions in speech are often conveyed through tone, pitch, and rhythm rather than the literal meaning of words.



This model can be optimized for real-time applications where immediate emotional feedback is critical. For instance, virtual assistants, automated customer service systems, or mental health apps can benefit from faster and more accurate emotion recognition based on tone and speech dynamics rather than relying on the delay introduced by transcription.

The scope of our project can be further expanded by exploring the combination of audio-based emotion recognition and facial expression recognition to create a more comprehensive multimodal system. This would provide even deeper insights into emotional states, leading to better user experiences in applications like mental health support, virtual assistants.

## CHAPTER 2

# LITERATURE SURVEY

### 1.1 Existing Models

There are several existing models for Speech Emotion Recognition such as **Wav2Vec 2.0**, **HuBERT**.

Wav2Vec.2.0 is a pre-trained self-supervised model developed by Facebook AI, it was originally used for the speech recognition but adaptable for emotion recognition. It learns from directly raw audio, removing the need for extensive labeled data. The model consist of CNN feature extractor followed by a Transformer encoder, allowing it to capture both local and global dependencies in speech. It handles the noise perfectly and genralizes it across all languages. The model can classify the Emotions like happiness, anger, and sadness effectively. It is easily adaptable for various speech tasks. It is pre-trained on large speech datasets like LibriSpeech and CommonVoice, containing hours of unlabeled speech for SER it is fine-tuned on labeled emotion datasets such as RAVDESS and TESS.

HuBERT is also one of the popular model used for the Speech Emotion Recognition which is also developed by the Facebook AI, is a self-supervised model trained to predict hidden units from masked audio segments. It learns speech representations without labeled data by clustering audio fearures iteratively. This model uses the combination of the CNNs and Transformers, making it efficient for capturing both short-term and long-term dependencies in speech. It performes well in the less resource scenarios making it effective for the tasks SER with limited emotion labled datasets. It is designed to deal with the audio in the noisy environment. It can classify the emotions such as happiness,

sadness , anger , fear, surprise, disgust, neutral. It is also pre-trained on datasets like LibriSpeech and CommonVoice and fine-tuned for SER on datasets like RAVDESS and CREMA-D.

## **1.2 Key Techniques and Algorithms:**

We experimented with ANN where it takes 40 input features, such as Mel-Frequency Cepstral Coefficients (MFCCs), and predicts one of six emotional labels: happy, sad, neutral, disgust, fear or anger. It consists of the 3 fully connected layers. The first layer maps the 40 input features to 128 units, the second layer further processes the output with 128 units, and the third layer generates a probability distribution over the five emotion classes. It also uses the ReLU as the activation function.

We experimented with a custom RNN model, a deep learning model it consists of 3 bidirectional Recurrent Neural Network (RNN) layers followed by a fully connected layer for output generation. Each RNN layer incorporates dropout regularization to avoid overfitting. The model has been designed to handle sequential data, learning features across multiple RNN layers to generate accurate predictions.

We evaluated LSTM Model. This model consists of an LSTM layer followed by a fully connected layer. The LSTM layer has been designed to handle sequential inputs, with configurable parameters for input size, number of layers, and a dropout rate of 0.2 to reduce overfitting. The output of the LSTM has been passed through the fully connected layer to generate predictions corresponding to emotion classes.

We experimented with SVC (Support Vector Classifier) model, a machine learning model that employs a linear kernel to classify data by finding the optimal hyperplane that separates classes. The model was trained using  $X_{train}$  and  $y_{train}$ , and predictions were made on  $X_{test}$ . It uses the hyperparameter  $C=1$  to balance the margin and misclassification.

We evaluated a RandomForestClassifier, an ensemble learning method that builds multiple decision trees and combines their outputs for classification. The model was trained with 100 estimators and a fixed random state for reproducibility. It leverages the power of multiple trees to improve accuracy and reduce the risk of overfitting.

We experimented with ResNet9 model, a deep learning architecture that incorporates residual connections to enhance training and mitigate vanishing gradient issues. This model consists of initial convolutional layers followed by three residual blocks, each containing two convolutional layers. These residual blocks allow the network to learn identity mappings, making it easier to train deeper networks. After applying max pooling and dropout for regularization, the output is flattened and passed through a fully connected layer for classification.

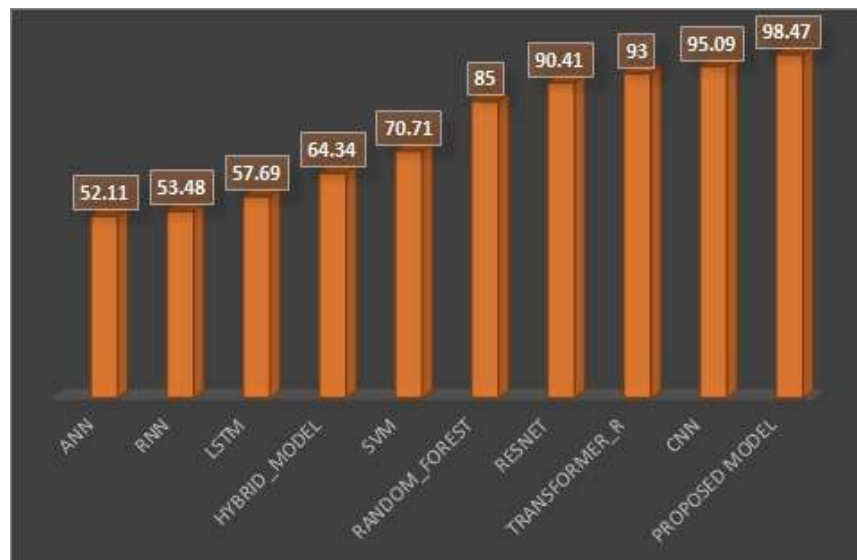
In order to capture the long-distance data points we used the transformer based classification model which uses sequential input processing. The model consist of an embedding layer followed by a Transformer encoder with two layers to capture complex temporal dependencies. The transformer encoder utilizes multi-head self.attention mechanism and it includes the feed-forward and liner classifier which maps the transformed representations to the target number of classes.

### 1.3 Justification for the Proposed Work:

The selection of the best-performing model was based on extensive experimentation with ten architectures, highlighting a data-driven approach to model selection. Through thorough evaluation process ,we ensured final model's reliability and scalability, addressing gaps such as noise sensitivity, overfitting and misclassification in existing solutions.

Our dataset(Telugu emotion dataset) consists of less number of files. CNN1D performs well with small to moderate datasets whereas transformer requires large datasets to effectively learn attention weights, as the self attention mechanism is prone to overfitting making CNN our choice for proposed model.

As in the below bar graph in contrast to other models the proposed model demonstrated exceptional performance, achieving the highest accuracy (98.47%) among all tested architectures.



**Fig1:** Accuracy graph

## CHAPTER 3

# PROPOSED WORK , ARCHITECTURE,TECHNOLOGY STACK & IMPEMENTATION DETAILS

### 3.1 Proposed Work

The primary objective of this proposed work is to enhance the performance of Conv1D model in the domain of Speech Emotion Recognition (SER). The focus is on refining the model architecture to improve feature extraction, reduce overfitting, and enable better generalization across various speech emotions. The proposed enhancements aim to address the limitations of the existing model while maintaining both efficiency and scalability. A more robust and efficient model is essential to accurately capture the intricate patterns within speech data, enabling the model to deliver higher classification accuracy.

Although the Conv1D model's current architecture has proven to be effective, there are several areas in which enhancements can be made to better handle the complex nature of speech signals. The following proposed improvements will aim to strengthen the model's capability to understand speech characteristics and provide more precise emotion classifications:

1. Refined Feature Extraction:

The current feature extraction pipeline, relying on techniques such as MFCCs, chroma, and Mel spectrograms, has provided useful input to the model. However, by exploring additional features like pitch, tone

the model could capture more subtle and emotionally relevant features of speech.

2. Dependence on ASR Accuracy:

Errors in speech-to-text conversion, especially in noisy environments, can lead to poor emotion recognition performance.

3. Regularization and Overfitting Mitigation:

Overfitting is a common issue in deep learning, especially with limited data. To address this, techniques like dropout, L2 regularization, or data augmentation (specifically for audio data) will be employed. Dropout can prevent the model from becoming too reliant on specific neurons, while L2 regularization will penalize large weights to promote generalization. Augmenting the training dataset through time-stretching, pitch-shifting, and noise addition will also expose the model to a wider range of speech data, improving its ability to generalize to unseen data.

4. Model Evaluation and Optimization:

The model's evaluation metrics, such as accuracy, precision, recall, and F1-score, will be closely monitored during training. By performing thorough cross-validation and hyperparameter tuning, the model will be optimized to balance performance and computational efficiency. Techniques such as grid search or Bayesian optimization can be explored to identify the best combination of hyperparameters, ensuring that the model performs optimally across different speech emotions.

### 5. Scalability and Real-Time Performance:

As the project progresses, ensuring that the model is scalable and can handle large-scale datasets efficiently will be crucial. The optimization of the model for inference speed and memory usage will be prioritized, allowing for real-time or near-real-time emotion recognition in speech data. This could be accomplished by reducing the model's size through techniques like model pruning or quantization without sacrificing its predictive performance.

In conclusion, by implementing these enhancements, the Conv1D model will become more robust in recognizing and classifying speech emotions accurately, addressing current limitations while ensuring that the model is both scalable and efficient for real-world applications.

## 3.2 Technology Stack

### • Programming Language

Python serves as the backbone to our project, chosen for its extensive support of libraries and tools that are specifically designed for machine learning tasks. Its rich set of libraries have been used for model training, evaluation, and many more. Python is an ideal language for implementing and scaling machine learning solutions efficiently.

### • Libraries and Frameworks

#### 1. PyTorch:

PyTorch is the core framework used in our project, powering the design and implementation of the Conv1D model, as well as the creation of DataLoaders and the execution of training and inference pipelines. The



optimization of the model is performed using the Adam optimizer, while loss computation during training is handled by CrossEntropyLoss, a commonly used loss function for classification tasks.

## 2. **Librosa:**

Audio signal processing plays a crucial role in feature extraction in our project, and Librosa is the popular library for this purpose. It allows for efficient extraction of Mel-frequency cepstral coefficients (MFCCs), chroma features, and Mel spectrograms, all of which are essential for training and inference. These features form the input to the neural network, enabling it to learn from the audio data effectively.

## 3. **NumPy:**

NumPy is the fundamental library for numerical computations in Python. It has been heavily used in preprocessing and feature extraction stages to manipulate data, perform mathematical operations, and handle arrays efficiently. The integration of NumPy ensures that the data pipeline remains performant and optimized for machine learning tasks.

## 4. **Pandas:**

Pandas is the primary library used for data management and preprocessing. It has been used for handling tabular data, making it easier to clean, filter, and manipulate datasets before they are fed into machine learning models. The use of DataFrames allows for efficient handling of large datasets in a structured format.

## 5. **Scikit-learn:**

Scikit-learn has been utilized for various preprocessing tasks, such as

splitting the dataset into training and testing sets, as well as encoding target labels. Additionally, it provides robust methods for evaluating the performance of machine learning models, including the calculation of metrics like accuracy, precision, recall, and F1-score.

#### 6. **Matplotlib/Seaborn:**

For visualizing data and results, Matplotlib and Seaborn have been used for creating informative and insightful plots. These include visualizations of training trends, such as loss and accuracy curves over epochs, as well as plots like the confusion matrix to assess model performance in terms of true and false classifications.

### 3.3 Implementation Details

**1. Load Dataset:** We analyzed a total of **783 audio files** after applying data augmentation techniques. These audio files are distributed across **6 emotion classes**: happy, fear, sad, angry, disgust, and neutral. The original dataset consisted of **269 audio files**, but through data augmentation, which included adding **white noise** and applying **spectral shifts**, the number of files was increased to **783**.

The below table gives data processing summary.

Step	Details
Original Dataset Size	269 audio files
Augmented Dataset Size	783 audio files (after noise addition and spectral shifting)
Feature Types	MFCC, Chroma, Mel Spectrogram
Train-Test Split	- <b>Training Set:</b> 587 samples (75%) - <b>Testing Set:</b> 196 samples (25%)
Final Dataset Dimensions	- <b>x_train:</b> 587 samples - <b>x_test:</b> 196 samples

**Table1:** Data preprocessing summary

To implement this project we used telugu emotion dataset saved inside ‘emotions\_dataset’ folder and below screenshot shows various classes of emotions that have been used in our project.

Shared with me > emotions_dataset					✓	☰	ⓘ
Type ▾	People ▾	Modified ▾	Source ▾				
Name ↑	Owner	Last modified ▾	File size				
angry_modified	kotilokyaan@gmail.com	19 Dec 2024 kotilokyaan@g...	—	👤	⬇	✎	☆
disgust	Abhijeet Gowlikar	12 Dec 2024 Abhijeet Gowli...	—				
fear	akshaya12300@gmail.com	19 Dec 2024 akshaya12300...	—				
happy	sandhyavlogs283	9 Dec 2024 sandhyavlogs283	—				
neutral	Abhijeet Gowlikar	12 Dec 2024 Abhijeet Gowli...	—				
sad	me	9 Dec 2024 me	—				

**Fig2:** Emotion classes in Dataset

## 2. Model Architecture

The model, Conv1DModel, has been built to process 1D feature sequences derived from audio signals. Its architecture follows a hierarchical approach, leveraging

convolutional and pooling layers to extract meaningful patterns in the data. Below are the key components:

- **Convolutional Layers:** The model consists of eight convolutional layers, which are the core building blocks for feature extraction from the input audio features. These layers have the following characteristics:
  - **Kernel Size:** Each convolutional layer uses a kernel size of 5, which strikes a balance between capturing local and broader patterns in the input signal. A smaller kernel (like 3) might miss some contextual information, while a larger kernel (like 7 or more) could over-smooth the signal, making it harder to detect subtle nuances in the speech patterns. So, kernel size of 3 has been used.
  - **Padding:** Padding is applied to the input to ensure that the spatial dimensions (sequence length) are preserved after convolution. We used kernel size of 3 so padding has been set to 2 to ensure that the output feature map retains the same length as input.
  - **Channels:** The output channels are progressively increased as the model deepens. The first two layers have 16 and 32 channels, respectively, which increases to 128 channels by the final convolutional layers. We started with fewer channels to avoid computation overhead in early layers when processing raw input data. As we progress deeper into the network the extracted features become more abstract, so we increased the number of channels to allow the network to learn richer set of features. Once the network has enough capacity to capture complex features we kept the channels constant to avoid overloading the model.

- **Pooling Layers:** Our model incorporates five max-pooling layers with a kernel size of 2. Pooling has been applied after every two convolutional layers to:
  - **Dimensionality Reduction:** Pooling halves the sequence length, reducing the number of parameters in subsequent layers.
  - **Feature Selection:** Max-pooling helps capture the most relevant features from the convolutions, emphasizing important patterns while discarding irrelevant details.
- **Dropout Layers:** Dropout is used to regularize the model and prevent overfitting, which is essential in deep learning models with many parameters. The model includes two dropout layers:
  - **Dropout1:** Applied after the initial layers, with a dropout rate of 0.1. With dropout rate of 0.1, 90% of the neurons remain active during training allowing the network to learn from sufficient number of features. Starting with a low dropout rate also prevents model from underfitting.
  - **Dropout2:** Applied before the final fully connected layer with a dropout rate of 0.5. We increased the dropout rate to 0.5 because as the network becomes deeper there arises a higher risk of overfitting. So, applying higher dropout rate helps in stronger regularization and helps the model to generalize better.
- **Fully Connected Layer:** After the feature extraction phase, our model uses a fully connected (linear) layer to map the hierarchical features to the final emotion classes. The number of output classes being 6, corresponding to the number of emotions being predicted by our model.

### 3. Data Preparation

The feature extraction process is essential to convert raw audio data into a form that the model can process. Several audio features have been extracted and preprocessed to capture different aspects of the audio signal:

- **MFCC (Mel-Frequency Cepstral Coefficients):**
  - **Purpose:** MFCCs capture the short-term power spectrum of sound, which is highly effective for speech and emotion recognition tasks.
  - **Extraction:** Our model extracts 40 MFCC coefficients from the audio and computes the mean of each coefficient across the time frames.
- **Chroma Features:**
  - **Purpose:** Chroma features reflect the intensity of the 12 pitch classes and are often used in music analysis and audio classification tasks.
  - **Extraction:** These features have been computed using the short-time Fourier transform (STFT), which represents the frequency content of the audio over time.
- **Mel Spectrogram:**
  - **Purpose:** The Mel spectrogram captures the frequency content of the audio signal using the Mel scale, which aligns more closely with human perception of pitch.
  - **Extraction:** The mean of the Mel spectrogram is calculated to reduce the dimensionality and ensure a fixed-size representation for each audio file.
- **Feature Concatenation:**

- The MFCC, chroma, and Mel spectrogram features are concatenated into a single feature vector, which serves as the input to the convolutional layers.

- **Reshaping Data:**

- The feature vectors are reshaped to match the model's expected input format: (batch\_size, channels, sequence\_length).

#### 4. Data Augmentation

To improve the robustness and generalization of our model, data augmentation techniques have been applied to the audio data:

- **Noise Addition:**

- **Function:** noise(data, noise\_factor)
- **Description:** Random Gaussian noise has been added to the audio signal, simulating real-world conditions where background noise can affect the quality of recordings.

- **Time Shifting:**

- **Function:** shift(data, sampling\_rate, shift\_max, shift\_direction)
- **Description:** This technique has been used to shift the audio in time, either left or right, which helps the model become more invariant to slight variations in timing. The maximum shift amount is controlled by the shift\_max parameter, and the direction of the shift is specified by the shift\_direction.

#### 4. Data Loading

- **Function:** load\_data(save=False)

- **Description:** This function traverses the directories containing audio files, extracts the relevant features (MFCC, chroma, Mel spectrograms), applies the necessary augmentations, and prepares the dataset for training.
- This function ensures that the features are stored in a format compatible with the model, either in memory or saved to disk for later use.

## 5. Train-Test Split

- **Function:** `train_test_split` from `scikit-learn`
  - **Purpose:** The dataset has been divided into training and testing subsets to evaluate the model's performance. 75% of the data is allocated to the training set, and 25% is reserved for testing. This split ensures that the model is trained on a large enough dataset while being evaluated on an independent set to check for overfitting.

## 6. Evaluation Metrics

Our model's performance has been assessed using several standard metrics like:

- **Accuracy:** The proportion of correct predictions compared to the total number of predictions. It has been calculated as the number of correct predictions divided by the total number of samples.
- **Loss:** The loss function (cross-entropy loss) measures the error between the predicted class probabilities and the actual class labels. Our model aims to minimize this loss during training.
- **Confusion Matrix:** A confusion matrix provides a detailed breakdown of the model's predictions, showing the number of true positives, false positives, true negatives, and false negatives for each emotion class. This matrix is useful for



identifying which classes the model performs well on and which ones it struggles with.

## 7. Model Training

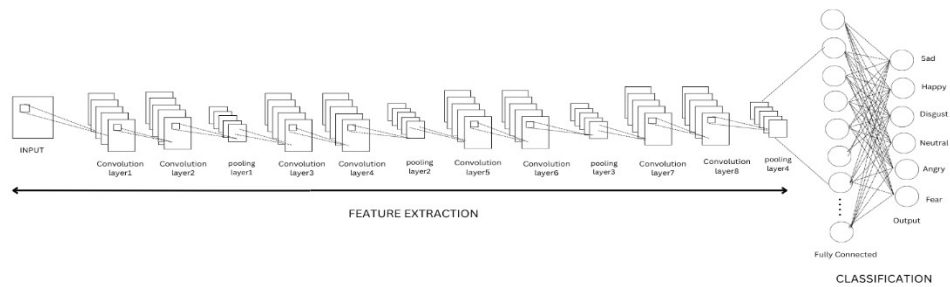
- **Training Loop:** The model has been training for a specified number of epochs(60). In each epoch:
  - The model's parameters are updated based on the loss computed from the training data.
  - After every epoch, the model is evaluated on the validation set to track performance.
  - The train\_loss, train\_accuracy, val\_loss, and val\_accuracy metrics are logged to monitor the training progress.
- **Optimization:** Our model uses the Adam optimizer with a learning rate of 0.001 and a weight decay of  $1e-6$  to optimize the parameters. This optimizer is well-suited for tasks with sparse gradients, like the ones encountered in deep learning models.
- **Evaluation:** The model's performance on the validation set is evaluated after every epoch, and the training and validation metrics are printed to track the progress.

## 8. Inference Pipeline

- **Prediction Function:** A separate inference pipeline has been provided to make predictions on unseen audio data. The process involves:
  - **Audio Loading:** The audio file is loaded, and features are extracted.

- **Feature Transformation:** The extracted features are reshaped to match the model's expected input format.
- **Model Evaluation:** The model is set to evaluation mode, and the features are passed through the model to obtain predictions.
- **Class Mapping:** The predicted class is mapped to the corresponding emotion label using a label encoder.

### Model Architecture:



**Fig3:** CNN Model Architecture

The above architecture is of a Convolutional Neural Network (CNN) designed for emotion classification. The process begins with an input layer that receives raw data. This input has been passed through eight convolutional layers, each extracting hierarchical features. Early layers identify basic patterns, while deeper layers capture more complex and abstract representations of the data. Pooling layers downsample the feature maps, reducing spatial dimensions and computational complexity while retaining essential information. Our architecture follows two convolution layers and

a pooling layer and this is repeated four times. After feature extraction, the processed data is flattened and passed into a fully connected layer, which maps the features to specific emotion classes. The model predicts one of six emotions: Sad, Happy, Disgust, Neutral, Angry, or Fear.

## **Backpropagation Derivation for CNN with Cross-Entropy Loss and ReLU Activation**

### **1. Forward Pass:**

- The input is passed through the layers: Convolutional layers (conv1, conv2, ...), followed by ReLU activations (F.relu), and MaxPooling layers (pool1, pool2, ...).
- The output from the last pooling layer is flattened and passed through a fully connected layer (fc).

### **2. Loss Function:**

- The model's final output is compared to the ground truth using Cross-Entropy Loss. This loss is computed as:

$$\text{Loss} = - \sum_{i=1}^C y_i \log(\hat{y}_i)$$

where  $C$  is the number of classes,  $y_i$  is the ground truth for class  $i$ , and  $\hat{y}_i$  is the predicted probability for class  $i$  after applying Softmax.

### **Backward Pass (Backpropagation):**

- The gradients are computed with respect to the weights and biases in each layer. The key steps for each layer:
  - a. For Fully Connected Layer (fc):

- The loss is backpropagated to the fully connected layer. The gradient of the loss with respect to the output is calculated, which is:

$$\frac{\partial Loss}{\partial z} = \hat{y} - y$$

where  $Z$  is the input to the last layer (before the activation function, Softmax is implicitly used in Cross-Entropy), and  $y$  is the true label.

b. For ReLU Activation (F.relu):

- The gradient of ReLU with respect to its input is:

$$\frac{\partial \text{ReLU}(x)}{\partial x} = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

- This gradient is applied element-wise to the gradients coming from the next layer.

c. For Convolutional Layers (conv1, conv2, etc.):

- The gradients of the loss with respect to the convolutional weights are calculated using the chain rule. The convolution operation is differentiable, and the gradient of the loss with respect to the filters is computed by:

$$\frac{\partial Loss}{\partial W} = \sum_{i,j} \frac{\partial Loss}{\partial O} \cdot \frac{\partial O}{\partial W}$$

- This involves computing the gradient of the loss with respect to the output, followed by the gradient with respect to the filters using the input data.

Optimization (Adam):

After calculating gradients through backpropagation, Adam optimizer is used to update the model's weights. Adam combines momentum and RMSProp to adapt the learning rate for each parameter.

1. Momentum: Computes a moving average of past gradients, helping the optimizer to continue in the same direction even with noisy updates.
2. RMSProp: Adapts the learning rate based on the magnitude of recent gradients.

The Adam update rule is:

$$W_t = W_{t-1} - \frac{\alpha \cdot \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

where:

- $\hat{m}_i$  and  $\hat{v}_i$  are corrected moving averages of gradients
- $\alpha$  is the learning rate
- $\epsilon$  is a small constant to prevent division by zero.

This allows the model to update weights efficiently and converge faster while avoiding overfitting (with weight decay).

## CHAPTER 4

# Results and Discussions

### 4.1 Results

The below table will provide a comprehensive overview of the different models experimented in finalizing the proposed model for our project. Each row in the table represents a model, and the columns include the following details:

1. **Model Name:** This column lists the name or type of the model, such as CNN, Transformer, or ResNet, to identify the specific architecture being discussed.
2. **Description:** A short summary of the model's structure and its key components, such as the number of layers, activation functions, or any unique characteristics.
3. **Accuracy:** It includes Training Accuracy and Test Accuracy. Training Accuracy reflects how well the model performs on the training data, and Test Accuracy indicates the model's performance on unseen test data.
4. **Metrics:** This column will provide additional performance metrics that might include Precision, Recall, F1-Score, or other relevant statistics that give a more comprehensive evaluation of the model's performance.
5. **Flaws:** This column will list any potential limitations of the model, such as issues with overfitting, computational inefficiencies, or difficulties handling specific types of data (example: long-range dependencies in

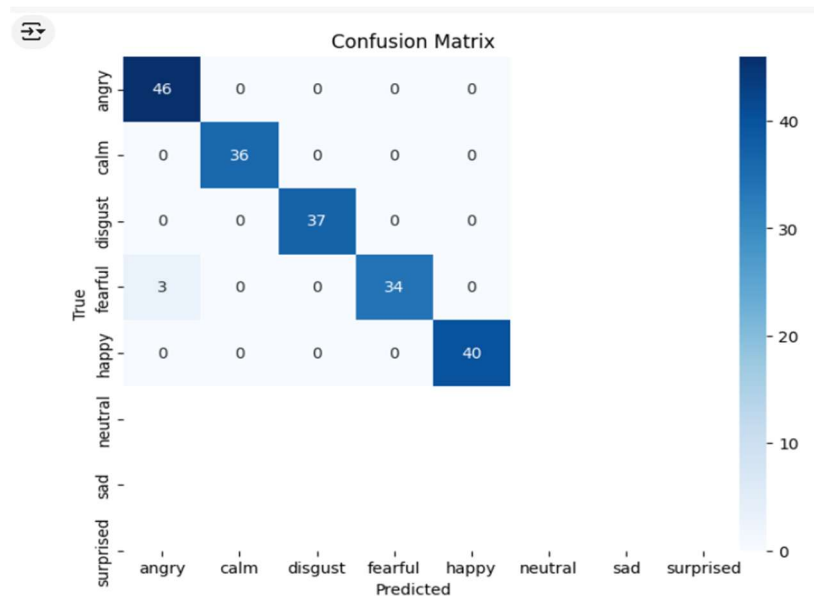
sequential

data).

Overview of Models				
MODEL	Description	Accuracy	Metrics	Flaws
SVM	Support Vector Classifier with a linear kernel	Training: 94.23% Testing: 70.71%	Confusion matrix	Likely to overfitting
Random_Forest	Random Forest classifier trained on MFCC features extracted	Testing: 85%	F1-score, precision	unable to predict for emotion fear
RNN	custom RNN-based model with three bidirectional RNN layers	Training : 65.38% Testing : 53.48%	Accuracy	poor generalization likely to overfit
ANN	A neural network model uses fully connected architecture with 40 MFCC	Training: 54.41% Testing: 52.11%	Accuracy	Limited by the simple architecture
LSTM	A sequence-based model using LSTM layers	Training: 58% Testing: 57.69%	Confusion Matrix	Struggles to achieve high accuracy
CNN 1 4SETS	A deep 1D convolutional it uses 8 convolution layers and 4 pooling layers and dropout	Training: 99.83% Testing: 98.47%	Confusion matrix	may overfit for large datasets
CNN	it consist of 1D 4 convolution layers with ReLu as activation	Training: 96.97% Testing: 95.09%	confusion matrix	may overfit for large datasets
Transformer	It consists of an embedding layer, two transformer encoders with multi-head attention and feed-forward layers, followed by a classifier layer	Training: 93.52% Testing: 93%	confusion matrix	inefficiency in training due to complexity
ResNet9	Consists of convolution layers, residual blocks, pooling, dropout, and a fully connected layer.	Training: 93.83% Testing: 90.41%	confusion matrix	vanishing gradients, overfitting, and limited model expressiveness
Audio Transformer	It consists of an embedding layer, positional encoding, two transformer encoders with multi-head attention and feed-forward layers, followed by a classifier layer	Training: 98.08% Testing: 98.47%	confusion matrix	high computational cost, overfitting, and difficulty with long-range dependencies.

The CNN-based Speech Emotion Recognition model demonstrates strong performance across all metrics, achieving high accuracy in emotion classification.

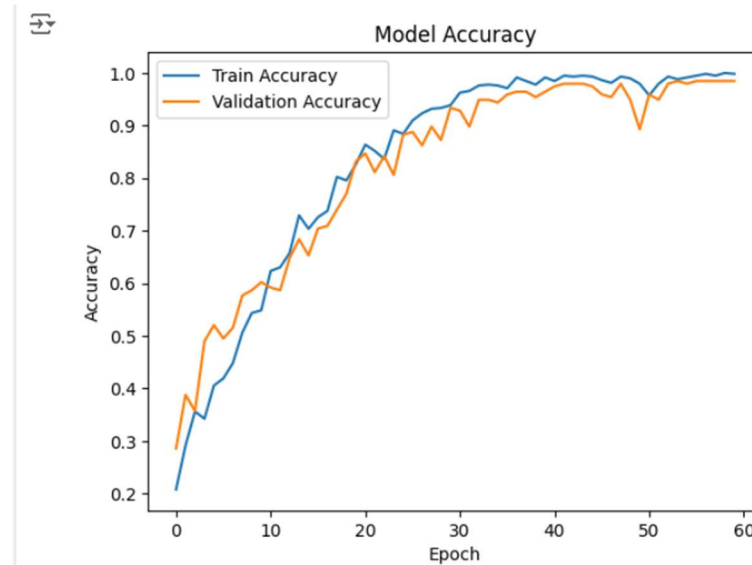
The screenshot of confusion matrix highlights minimal misclassifications, with particularly strong performance in predicting classes like "angry," "disgust," and "happy," where there are no false positives or negatives. Misclassifications, such as a small overlap between "fearful" and "calm," are notably rare and do not significantly impact the model's overall reliability.



**Fig4:** Confusion Matrix

The training and validation accuracy curves as in below screenshot demonstrate the CNN model's effective learning and convergence, with both curves reaching high accuracy over 60 epochs, indicating minimal overfitting and consistent performance.





**Fig5:** Train and validation accuracy curves

The architecture, consisting of multiple Conv1D layers, MaxPooling, and dropout for regularization, ensures the model's robustness and generalization. The classification results confirm that the model maintains high precision, recall, and F1-scores across all emotion classes, making it a reliable tool for practical applications in SER tasks. This consistency across metrics and class distributions establishes the model as an effective and dependable solution for emotion detection in audio data.

```

⇒ Number of predictions: 196
Number of true labels: 196
Sample predictions: [1 2 0 3 2 3 2 1 0 4]
Sample true labels: [1 2 0 3 2 3 2 1 0 4]
Confusion Matrix:
[[46  0  0  0  0]
 [ 0 36  0  0  0]
 [ 0  0 37  0  0]
 [ 3  0  0 34  0]
 [ 0  0  0  0 40]]
Precision: 0.9856
Recall: 0.9847
F1 Score: 0.9846

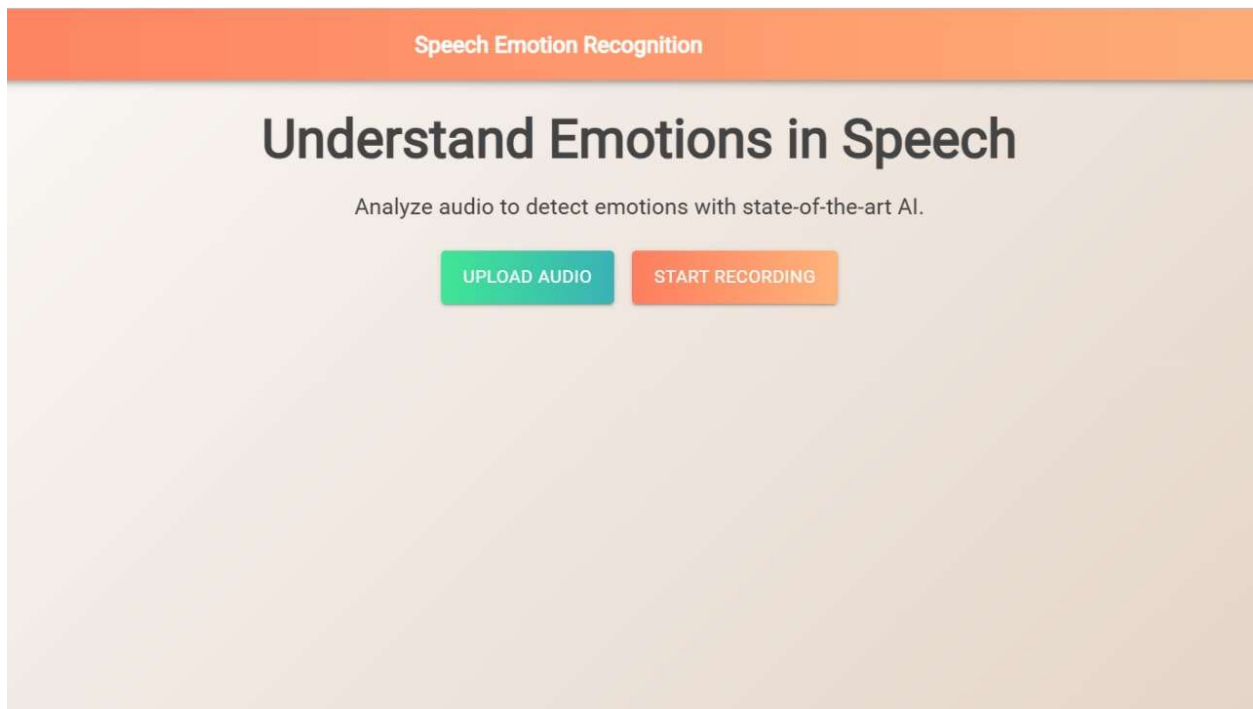
```

**Fig6:** Precision recall and F1 score

## 4.2 Output Screens

Our application Speech Emotion Recognition provides an intuitive and user-friendly interface for users to interact with the application. It includes a main page, a sign-in page, and a sign-up page.

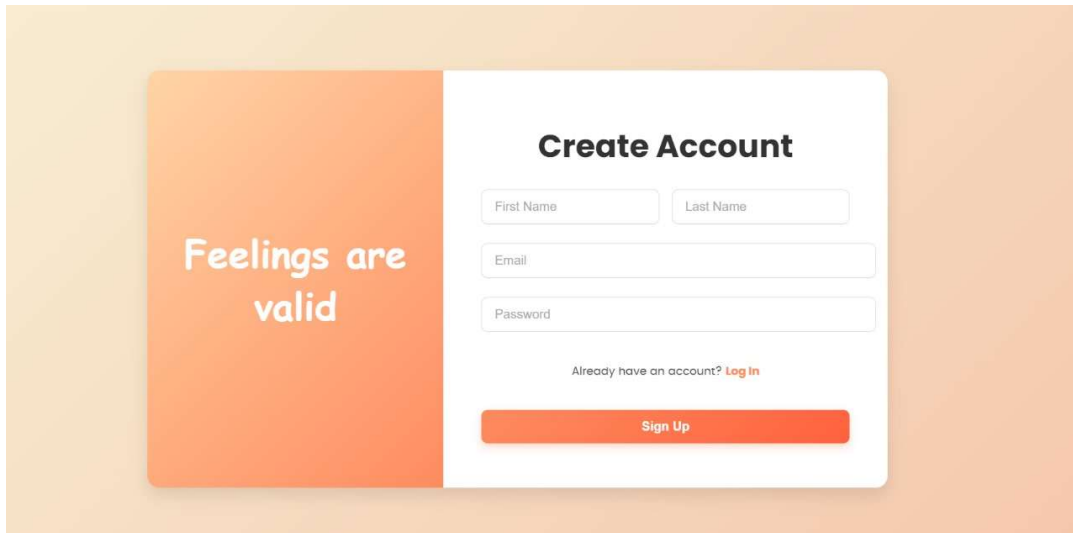
This is the **home page** of our UI where a user can login or signup to use our services



**Fig7:** UI HomePage

Sign Up Page:

When a user clicks on signup they will get navigate to this page where user has to provide information like name, email, password

A screenshot of a 'Create Account' form. On the left, an orange square contains the text 'Feelings are valid'. To the right, a white form box has the title 'Create Account'. It includes input fields for 'First Name', 'Last Name', 'Email', and 'Password'. Below these fields is a link 'Already have an account? Log In' and a red 'Sign Up' button.

Feelings are valid

### Create Account

First Name Last Name

Email

Password

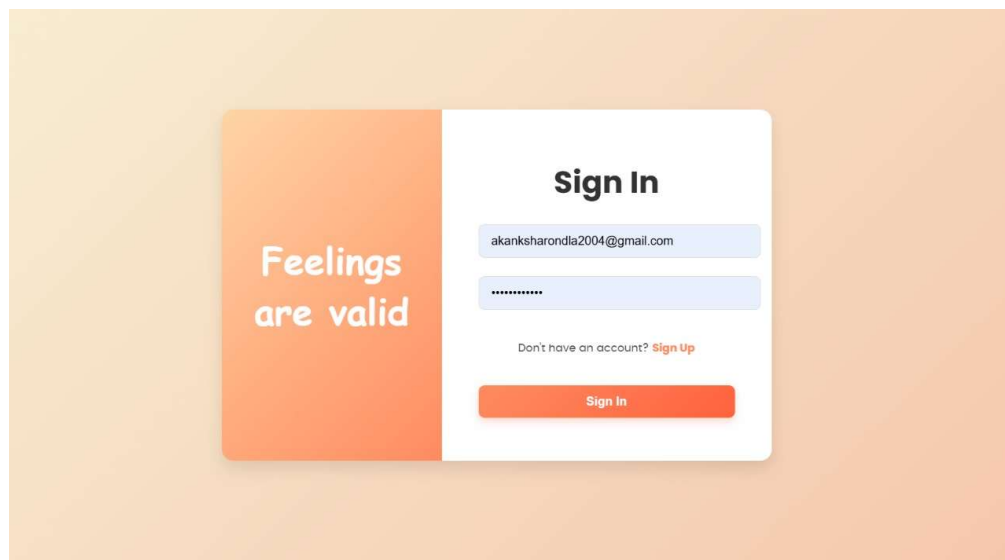
Already have an account? [Log In](#)

Sign Up

**Fig8:** SignUp Page

Sign In page:

After signup user has to login by providing the same email and password which the user provided while signup.

A screenshot of a 'Sign In' form. On the left, an orange square contains the text 'Feelings are valid'. To the right, a white form box has the title 'Sign In'. It includes input fields for email (containing 'akanksharondia2004@gmail.com') and password (masked with dots). Below these fields is a link 'Don't have an account? Sign Up' and a red 'Sign In' button.

Feelings are valid

### Sign In

akanksharondia2004@gmail.com

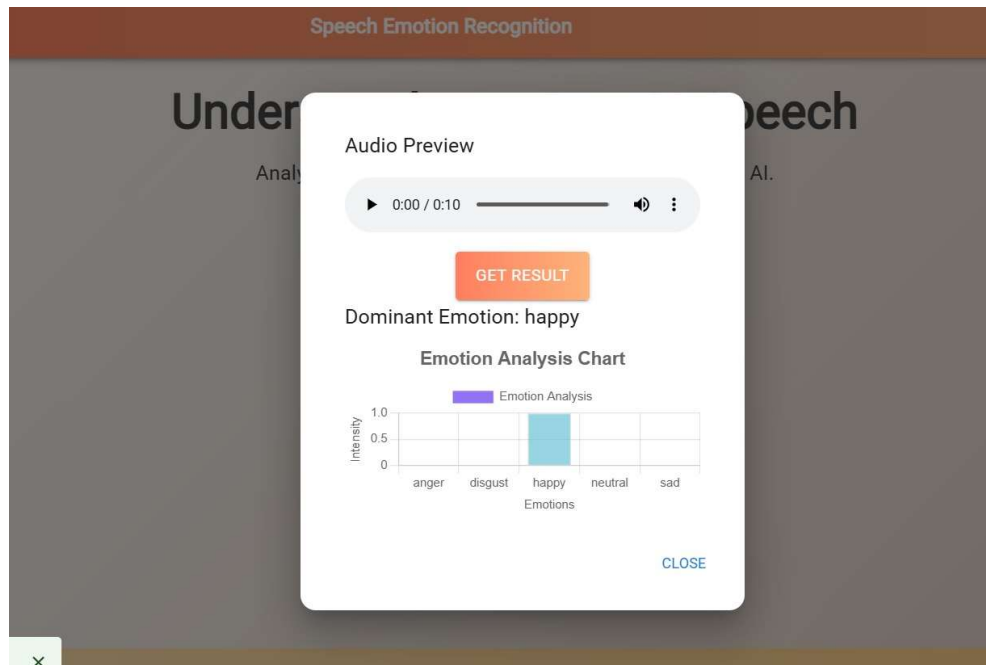
.....

Don't have an account? [Sign Up](#)

Sign In

**Fig9:** SignIn Page

After signing in user can either drag and drop audio files or upload files from their device. User can also choose to record their voice. Upon the file being successfully uploaded, we give the predicted emotion.



**Fig10:** Predicted Output

## Chapter 5

# Conclusion & Future Scope

In this proposed work, we aim to enhance the performance of the Conv1D model in the domain of Speech Emotion Recognition (SER) by addressing key limitations of the current architecture. By refining the feature extraction process, implementing model architecture improvements, reducing overfitting, and optimizing for better generalization, we anticipate significant improvements in the model's ability to accurately classify emotions from speech data. These enhancements will provide the Conv1D model with a more robust and scalable framework to handle the intricate patterns of speech, ensuring better classification performance and generalization across different speech emotions.

Through these proposed changes, the Conv1D model will become more efficient and adaptable, enabling it to be deployed in real-world applications that require accurate and real-time emotion recognition. The improvements outlined in this work not only aim to enhance model performance but also ensure that it remains computationally feasible and scalable for large-scale datasets and practical use cases.

## Future Scope:

While the current Speech Emotion Recognition (SER) model provides a strong foundation, several areas remain for further exploration and enhancement.

Incorporating datasets from different languages will enable the model to recognize emotions in multilingual contexts. This would be crucial for applications in global markets and diverse user bases.

Combining speech data with facial expression recognition could improve emotion classification accuracy. This multi-modal approach would allow the model to utilize visual cues (from videos or images) along with audio features, offering more reliable emotion recognition in real-world settings.

Model architecture can be further improved by using hybrid models like combination of CNNs for feature extraction with sequential modeling using RNNs or Transformers.

For a Speech Emotion Recognition (SER) project, incorporating advanced audio transformers in the future can expand the system's performance and usability for large datasets:

- 1. Audio Transformers:** Employ **Wav2Vec2** and **HuBERT** to extract robust pre-trained embeddings, enhancing emotion recognition in diverse audio conditions.
- 2. Audio Classification Transformers:** Use **Audio Spectrogram Transformer (AST)** to improve emotion detection through spectrogram-based feature analysis.

**3. OpenAI Transformers:** Integrate **Whisper** for multilingual emotion recognition and enhanced transcription capabilities in SER systems.

While six basic emotions can be effective for general applications, a larger set of emotion categories could improve **emotion granularity**. Recognizing emotional nuances, such as distinguishing between **surprise** and **confusion**, or between **happiness** and **pride**, can provide more accurate insights into a person's emotional state, enhancing the performance of **human-computer interaction** and **mental health monitoring** systems.

The current system could be adapted to work in real-time applications such as virtual assistants (e.g., Siri, Alexa), mental health monitoring tools, or customer service chatbots. Efficient inference pipelines, optimized for lower latency, could allow the model to predict emotions during ongoing conversations.

## Chapter 5

# References:

1. **Han K, Yu D, and Tashev.** This reference discusses speech emotion recognition using deep neural networks (DNNs) and extreme learning machines, aligning well with and ANN focus.
2. **Lin Zhen-Tao et al.** Explores SER based on formant characteristics and phoneme type convergence, which may provide insights into feature extraction techniques relevant to CNN and Transformer-based models.
3. **Farooq Misbah et al.** Focuses on the impact of feature selection algorithms in SER using deep convolutional neural networks, directly relevant to CNNs.
4. **Vaswani A, Shazeer N, Parmar N, et al.** This reference introduces the Transformer model ("Attention is All You Need"), critical for applying Transformers in SER tasks.
5. **Issa D, Demir M F, and Yasici A.** Focuses specifically on speech emotion recognition using deep convolutional neural networks, directly relevant to your CNN-based approach.