EMOTION RECOGNITION FROM AUDIO

12/13/2024

**Submitted By:**

Obiaku Valentine Chike

Ramandeep Singh

Karmjeet Kaur

Janvi

**Submitted To:**

Ishant Gupta

2024F-T3 AML 3104 - Neural Networks and Deep Learning 01

**Table of Contents**

**1. Introduction**

1.1 Background

1.2 Objective

1.3 Scope

**2. Dataset Description**

2.1 Dataset Overview

2.2 Data Structure

2.3 Emotion and Gender Mapping

**3. Preprocessing and Feature Extraction**

3.1 Data Cleaning

3.2 Feature Extraction

3.3 Label Encoding

**4. Data Exploration and Visualization**

4.1 Statistical Overview

4.2 Emotion Distribution

4.3 Correlation Heatmap

4.4 PCA Visualisation

**5. Model Design and Implementation**

5.1 Model Selection

5.2 LSTM Model Architecture

5.3 GRU Model Architecture

5.4 Model Training

**6. Model Evaluation**

**7. Hyperparameter Tuning**

7.1 Hyperparameter Tuning with KerasTuner

7.2 Hyperparameter Search Results

**8. Detailed Code Explanation**

8.1 Streamlit UI Code

8.2 Model Loading and Prediction Code

8.3 Audio Feature Extraction Code

**9. Conclusion and Future Work**

9.1 Summary of Results

9.2 Future Improvements

**10. Deployment on Streamlit Cloud**

10.1 Deployment Process

10.2 Deployment Challenges

**11. Conclusion**

**12. References**

**1. Introduction**

**1.1 Background**

Emotion detection from audio is a subfield of speech recognition that focuses on analyzing the emotional state of the speaker based on their voice. It has a wide range of applications, including in customer service, mental health monitoring, entertainment, and human-computer interaction.

**1.2 Objective**

This report aims to demonstrate the process of emotion detection using audio files. The primary goal is to extract features from the audio and train a machine learning model (LSTM) to classify different emotions (e.g., happy, sad, angry) based on these features.

**1.3 Scope**

The scope of this project includes:

* Audio data preprocessing and feature extraction.
* Data exploration and visualization.
* Building an LSTM (Long Short-Term Memory) model for emotion classification.
* Hyperparameter tuning for optimizing model performance.
* Evaluation of model performance using various metrics.

**2. Dataset Description**

**2.1 Dataset Overview**

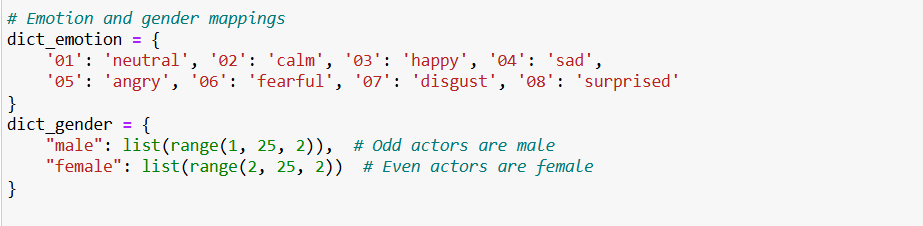
The dataset used for this project is the RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song), which contains 24 actors (12 male and 12 female) who speak 8 different emotional categories (neutral, calm, happy, sad, angry, fearful, disgust, and surprised).

**2.2 Data Structure**

* Number of Samples: Approximately 2,400 audio samples.
* File Format: WAV.
* Attributes: Each file consists of audio data, where each emotion is labeled in the filename.
  + Example: Actor\_01\_01.wav indicates Actor 1 performing emotion "neutral".

**2.3 Emotion and Gender Mapping**

* Emotions: 'neutral', 'calm', 'happy', 'sad', 'angry', 'fearful', 'disgust', 'surprised'.
* Gender Mapping: Odd-numbered actors are male, even-numbered actors are female.



**3. Preprocessing and Feature Extraction**

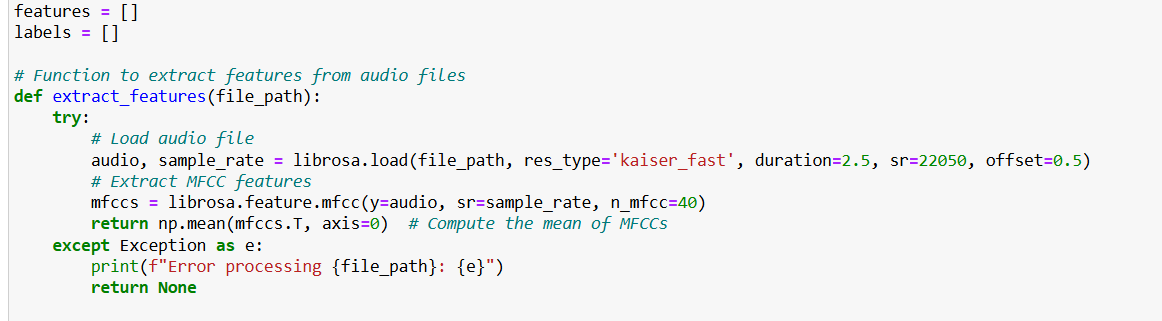
**3.1 Data Cleaning**

* Before extracting features, the dataset undergoes cleaning where:
* Non-audio files are filtered out.
* Invalid or corrupted audio files are discarded.

**3.2 Feature Extraction**

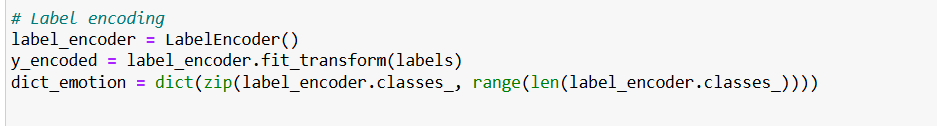
We used Mel-frequency cepstral coefficients (MFCCs) for feature extraction, which are commonly used in speech and audio recognition tasks. The features are extracted as follows:

* Load audio using librosa library.
* Extract MFCCs using librosa.feature.mfcc.
* The MFCCs are averaged over time to obtain a fixed-length feature vector for each sample.



**3.3 Label Encoding**

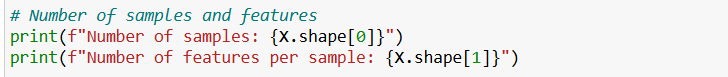
The emotion labels are encoded using LabelEncoder to convert categorical labels into numeric format, which is required for training machine learning models.



**4. Data Exploration and Visualization**

**4.1 Statistical Overview**

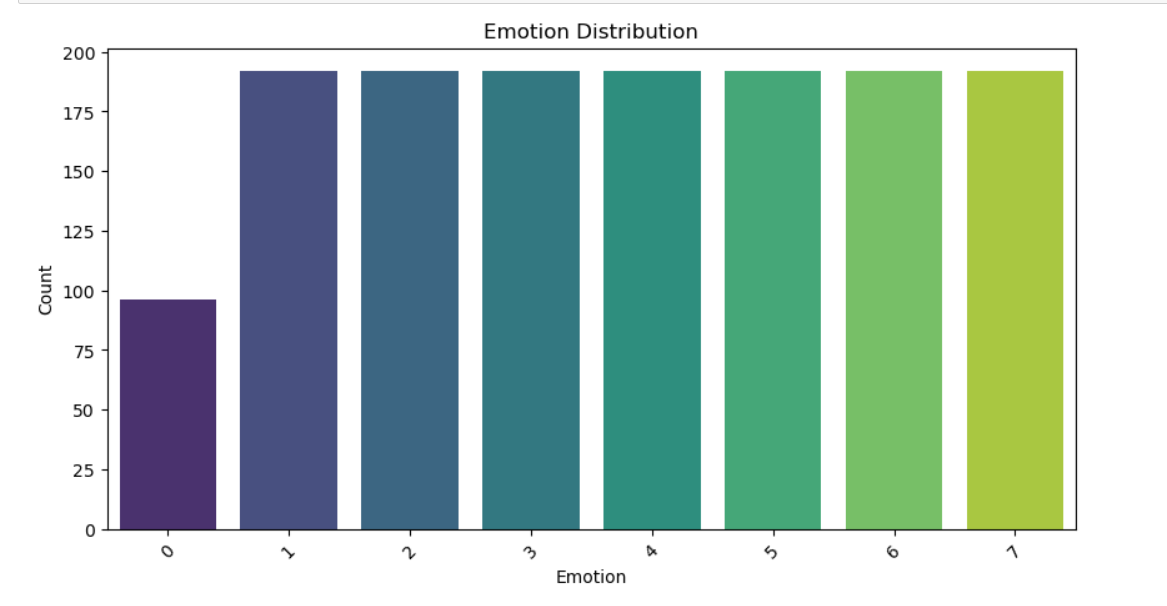
* The number of samples and features are displayed to understand the dataset better.





**4.2 Emotion Distribution**

A bar plot is generated to show the distribution of different emotions in the dataset.



The bar chart titled "Emotion Distribution" illustrates the counts of various emotions, categorized from 0 to 6. Key observations from the analysis are as follows:

**Distribution Overview:**

The chart shows a clear representation of seven distinct emotions, with counts ranging from emotion 0 to emotion 6.

**Count Comparison:**

* Emotions 1 through 6 exhibit a relatively uniform distribution, each showing counts around or exceeding 100. This suggests a balanced representation of these emotions within the dataset.
* In contrast, emotion 0 displays a significantly lower count, indicating that this emotion is less prevalent.

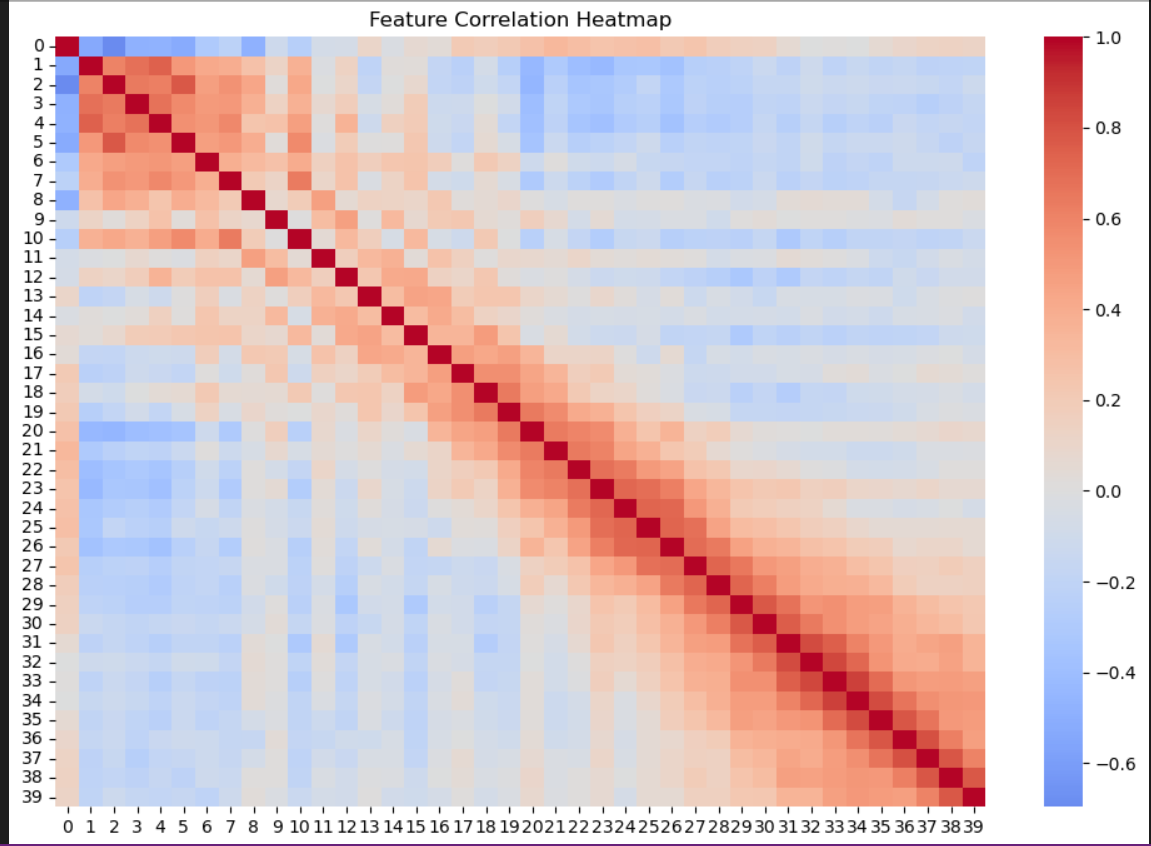
**Implications:**

The even distribution among emotions 1 to 6 may imply that the dataset captures a range of active or positive emotional states.

The low count for emotion 0 raises questions about its representation, suggesting that it may correspond to a less common emotional response or a neutral state.

**4.3 Correlation Heatmap**

A heatmap of feature correlations helps understand the relationships between different features in the dataset.



The heatmap titled "Feature Correlation Heatmap" visualizes the correlation between various features in a dataset, providing a comprehensive overview of how these features relate to one another. The analysis is structured as follows:

**Heatmap Structure**

* The heatmap is color-coded to represent correlation coefficients, ranging from -1 to 1.
  + Positive correlations (red hues) indicate a direct relationship, where increases in one feature correspond to increases in another.
  + Negative correlations (blue hues) indicate an inverse relationship, where increases in one feature correspond to decreases in another.
  + A correlation of 0 indicates no relationship between the features.

**Diagonal Dominance**

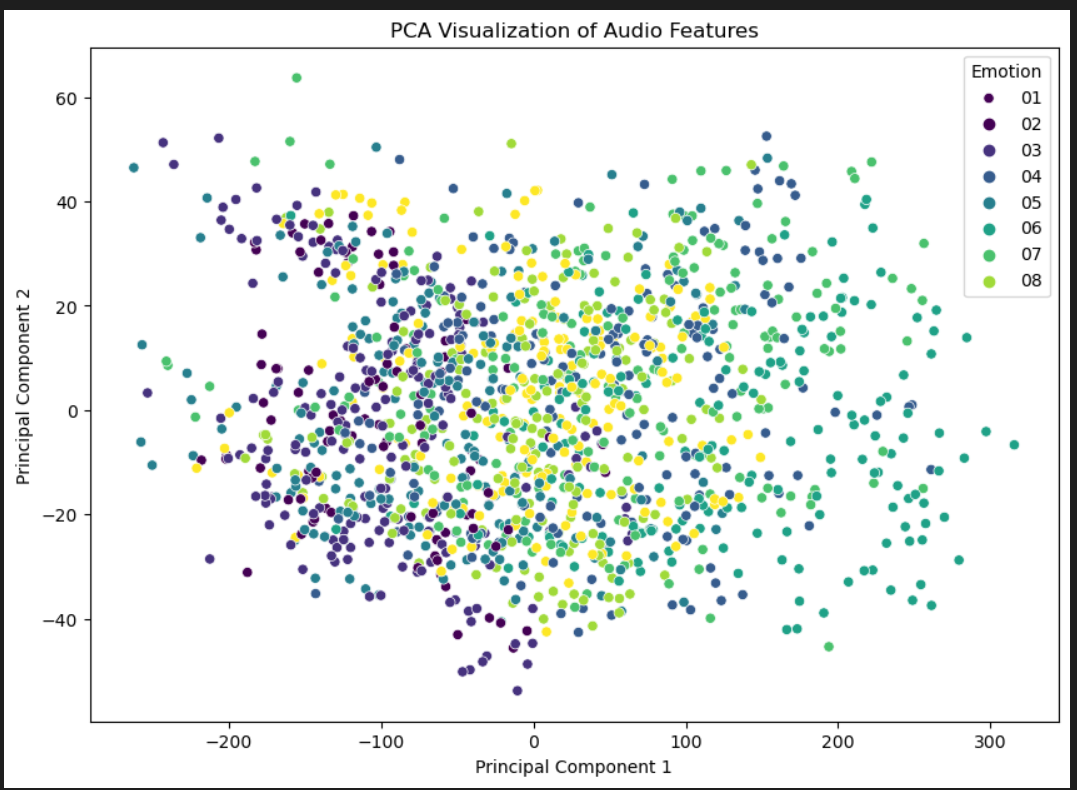
* The diagonal (from the top left to the bottom right) represents the correlation of each feature with itself, which is always 1. This is expected and serves as a reference point for comparison with off-diagonal correlations.

**Key Observations**

* Strong Correlations: Certain features exhibit strong positive correlations, indicated by darker red colors. This suggests that these features may share underlying relationships or patterns, which could be critical for predictive modeling or feature selection.
* Moderate to Weak Correlations: Many features show moderate to weak correlations, as indicated by lighter shades of red and blue. These features may have less direct influence on each other, which could imply independent behavior.
* Negative Correlations: There may be features that exhibit negative correlations, represented by blue shades. This indicates that as one feature increases, the other tends to decrease, which could be significant in understanding the interactions between features.

**4.4 PCA Visualization**

PCA (Principal Component Analysis) is used for dimensionality reduction to visualize the features in a 2D space.



The scatter plot titled "PCA Visualization of Audio Features" illustrates the results of a Principal Component Analysis (PCA) applied to a dataset of audio features, with each point representing an individual data sample colored by its associated emotion category. The analysis is structured as follows:

**PCA Overview**

* PCA is a dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional space while preserving as much variance as possible. In this plot, the first two principal components (PC1 and PC2) are displayed on the x- and y-axes, respectively.

**Data Distribution**

* The scatter plot displays a wide distribution of data points across both principal components. This suggests that the audio features contain a variety of information and that the samples cover a broad spectrum of emotional expressions.

**Emotion Representation**

* Each point is color-coded according to its associated emotion (labeled as 01 to 08). Notable observations include:
  + Cluster Formation: Certain emotions may show clustering tendencies, indicating that samples with similar emotional content are grouped together in the PCA space. This could imply shared characteristics among those audio features.
  + Overlap: Some emotion categories appear to overlap significantly, suggesting that the audio features for these emotions may not be distinctly separable. This might indicate that certain emotions share similar acoustic properties.

**Key Observations**

* Diverse Emotional Representation: The presence of diverse colors across the plot indicates a variety of emotional expressions captured within the dataset.
* Potential Overlap Between Emotions: The overlapping regions of different colors suggest that some emotions may not be easily distinguishable based solely on audio features, which could present challenges for classification tasks.

**5. Model Design and Implementation**

**5.1 Model Selection**

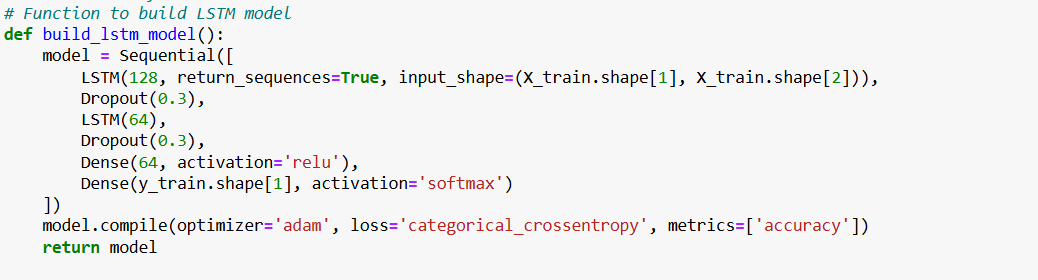
For the emotion detection task, we tested both LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) architectures. Both are types of recurrent neural networks (RNNs) designed to handle sequential data, such as audio features. The primary difference between the two lies in their architecture: GRU is considered a simpler and faster alternative to LSTM due to having fewer gates, which may reduce training time and complexity while achieving comparable performance.

**5.2 LSTM Model Architecture**

The LSTM model consists of:

* LSTM Layers: Designed to capture long-term dependencies in sequential data.
* Dropout Layers: To reduce overfitting during training.
* Dense Layers: For the final classification output.

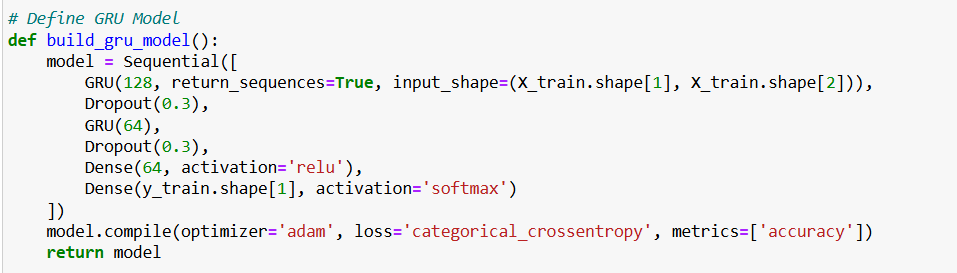
The architecture of the LSTM model is as follows:

****

**5.3 GRU Model Architecture**

In addition to the LSTM model, a GRU (Gated Recurrent Unit) model was implemented. GRU operates similarly to LSTM but with a simpler structure. It uses a combination of gates (reset and update) to manage the flow of information and capture dependencies in sequential data. The GRU model was chosen to evaluate whether a simpler architecture could achieve comparable results to LSTM.

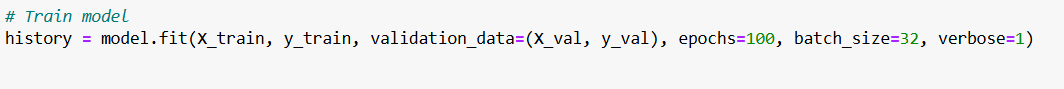
The architecture of the GRU model is as follows:

****

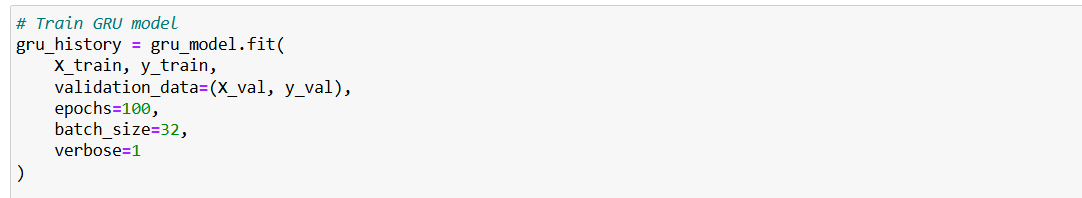
**5.4 Model Training**

Both the LSTM and GRU models were trained on the same dataset, using the same parameters (batch size of 32, 100 epochs, etc.). The training process was monitored using the accuracy and loss metrics, and early stopping was employed to avoid overfitting.

**Train LSTM model:**

****

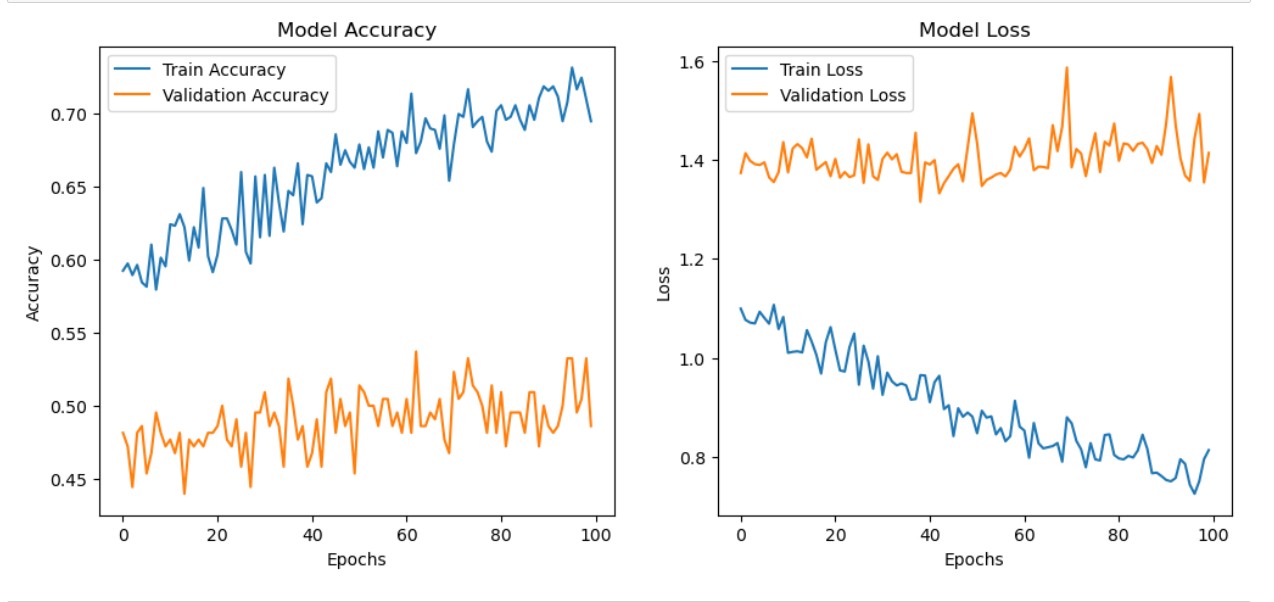
**Train GRU model:**

****

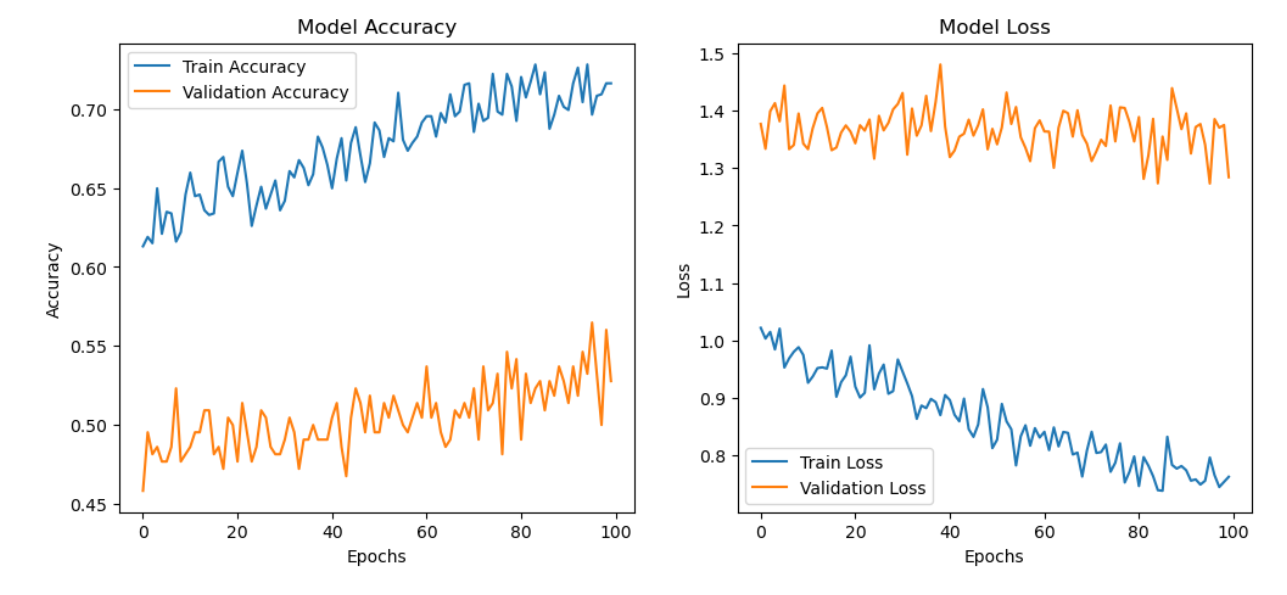
**6 Model Evaluation Plots**

The evaluation curves for both models are plotted below to show the progression of accuracy and loss during training.

Plotting training and validation accuracy for LSTM

****

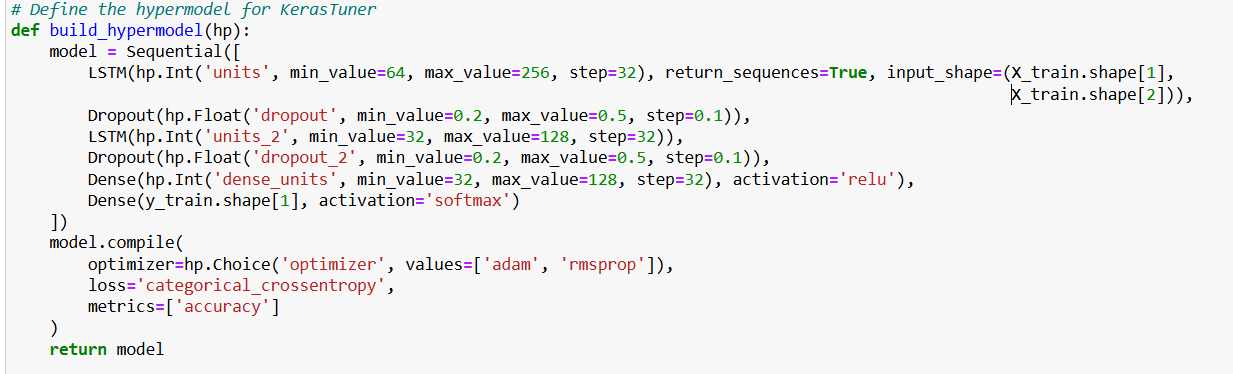
Plotting training and validation accuracy for GRU

****

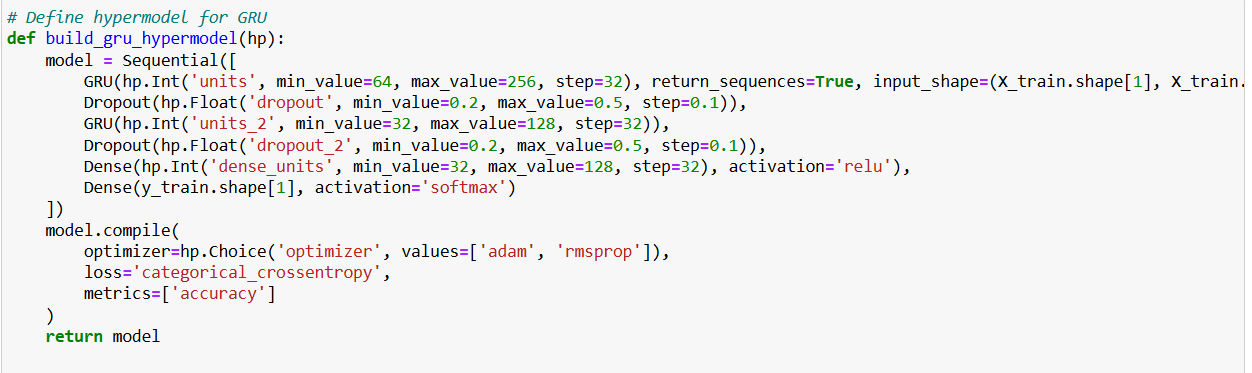
**7. Hyperparameter Tuning**

**7.1 Hyperparameter Tuning with KerasTuner**

To further optimize the performance of both models, hyperparameter tuning was performed using KerasTuner. We tuned parameters such as the number of hidden units, dropout rates, and the learning rate for both the LSTM and GRU models. The search space for hyperparameters was set as follows:

****

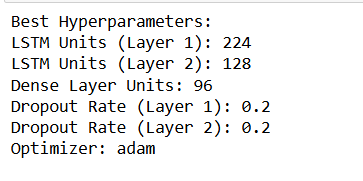
Similar tuning was performed for the GRU model.

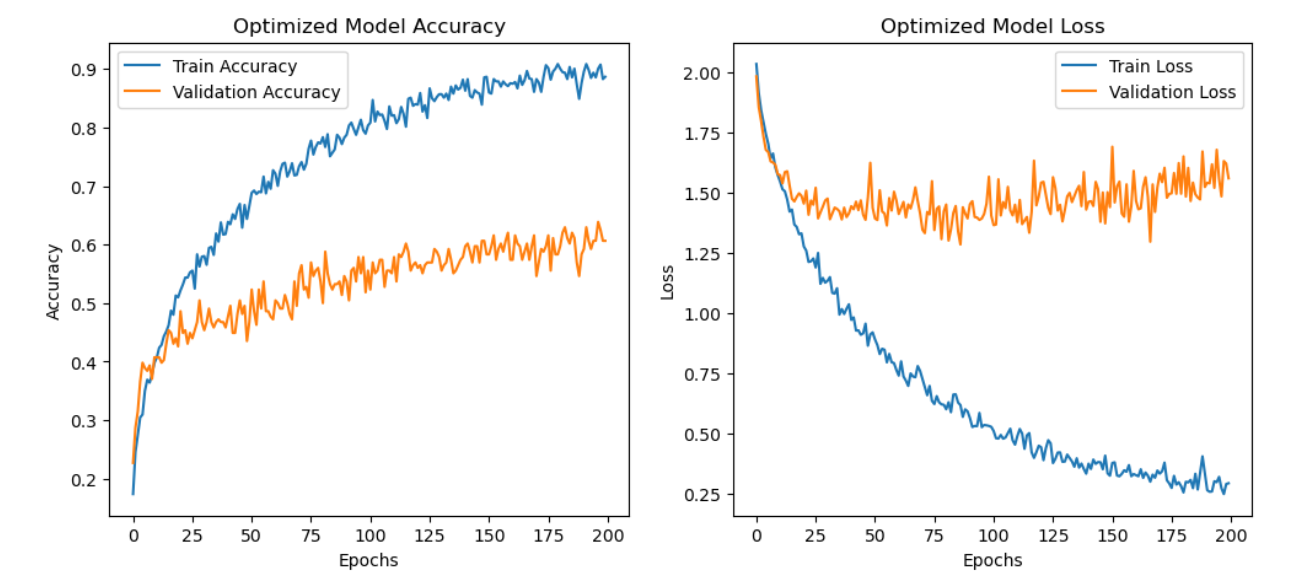
****

**7.2 Hyperparameter Search Results**

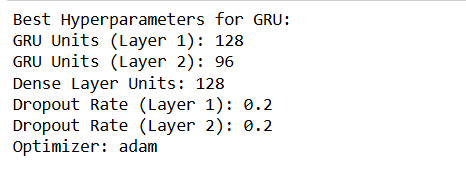
KerasTuner provided the best hyperparameters for both models. After tuning, the models showed a slight improvement in accuracy and reduced overfitting compared to their default configurations.

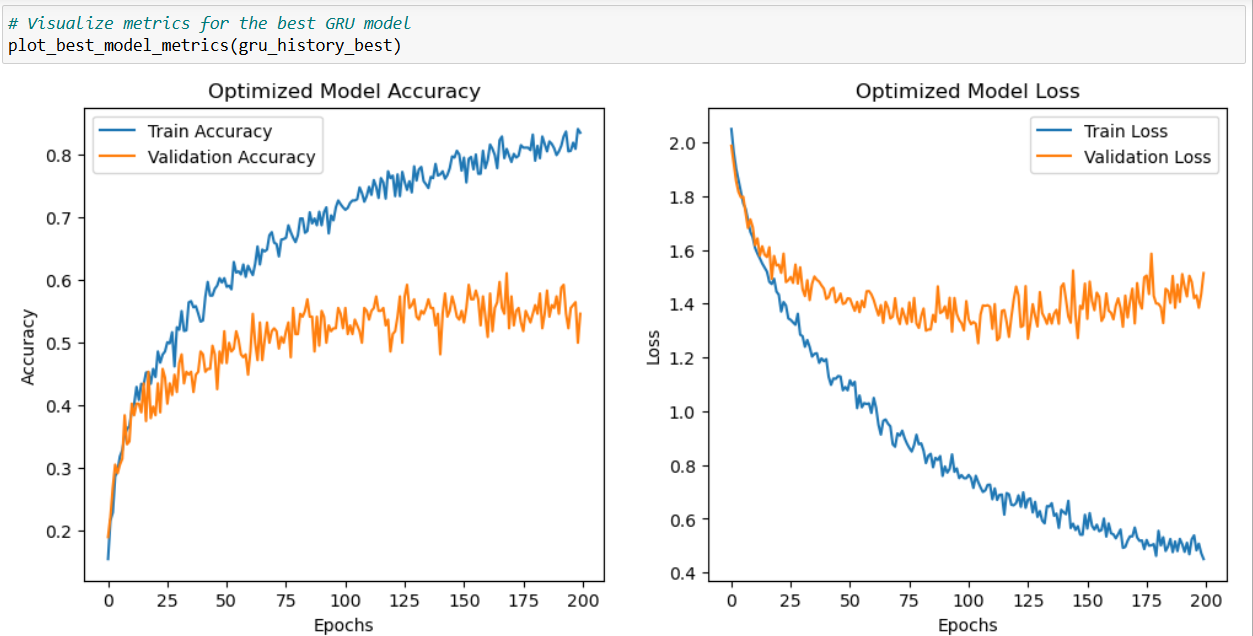
**For LSTM:**

****

****

**For GRU:**

****

****

**8. Detailed Code Explanation**

**8.1 Streamlit UI Code**

The Streamlit UI provides an interactive web interface where users can upload an audio file. Upon uploading, the app preprocesses the file, extracts the features, and feeds them into the trained model to predict the emotion. The result is displayed on the UI.

**8.2 Model Loading and Prediction Code**

The trained LSTM model is saved as a .h5 file and loaded using Keras for prediction. The model takes the extracted MFCC features from the uploaded audio file and outputs a predicted emotion.

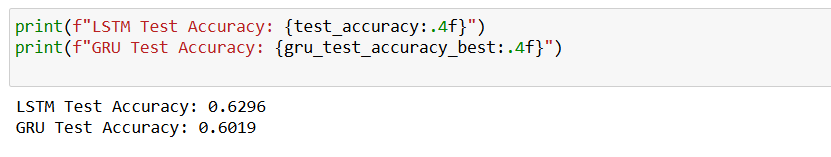
**8.3 Audio Feature Extraction Code**

The feature extraction code utilizes librosa to load the audio file, extract the MFCCs, and reshape the features into a format suitable for model input. This includes handling mono audio files and padding sequences to ensure consistent input length.

**9. Conclusion and Future Work**

**9.1 Summary of Results**

Both LSTM and GRU models were successful in classifying emotions from audio data with reasonable accuracy. The LSTM model performed slightly better in terms of accuracy

****

**9.2 Future Improvements**

Experiment with different audio feature extraction methods, such as spectral features or Chroma features, to further improve classification accuracy. Incorporate data augmentation techniques to increase the robustness of the model.

**10. Deployment on Streamlit Cloud**

**10.1 Deployment Process**

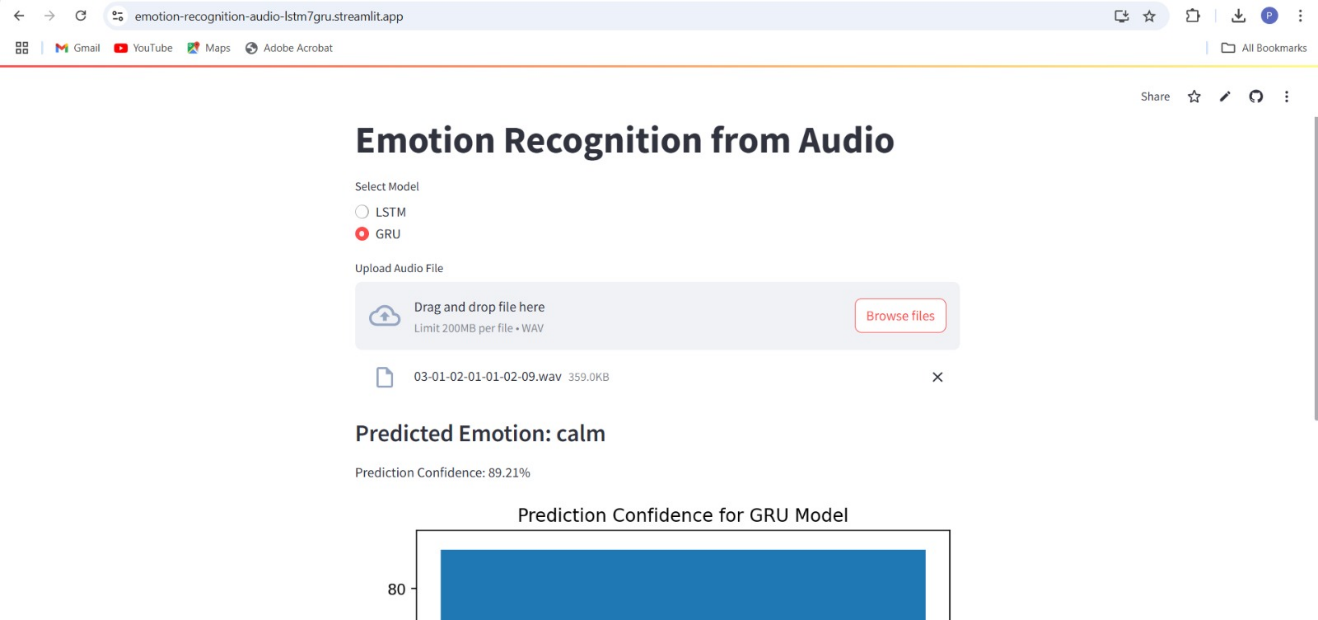
The model was deployed on Streamlit Cloud, enabling real-time predictions from the web interface. The deployment involved pushing the code to a GitHub repository, linking it with Streamlit, and configuring the app for public access.

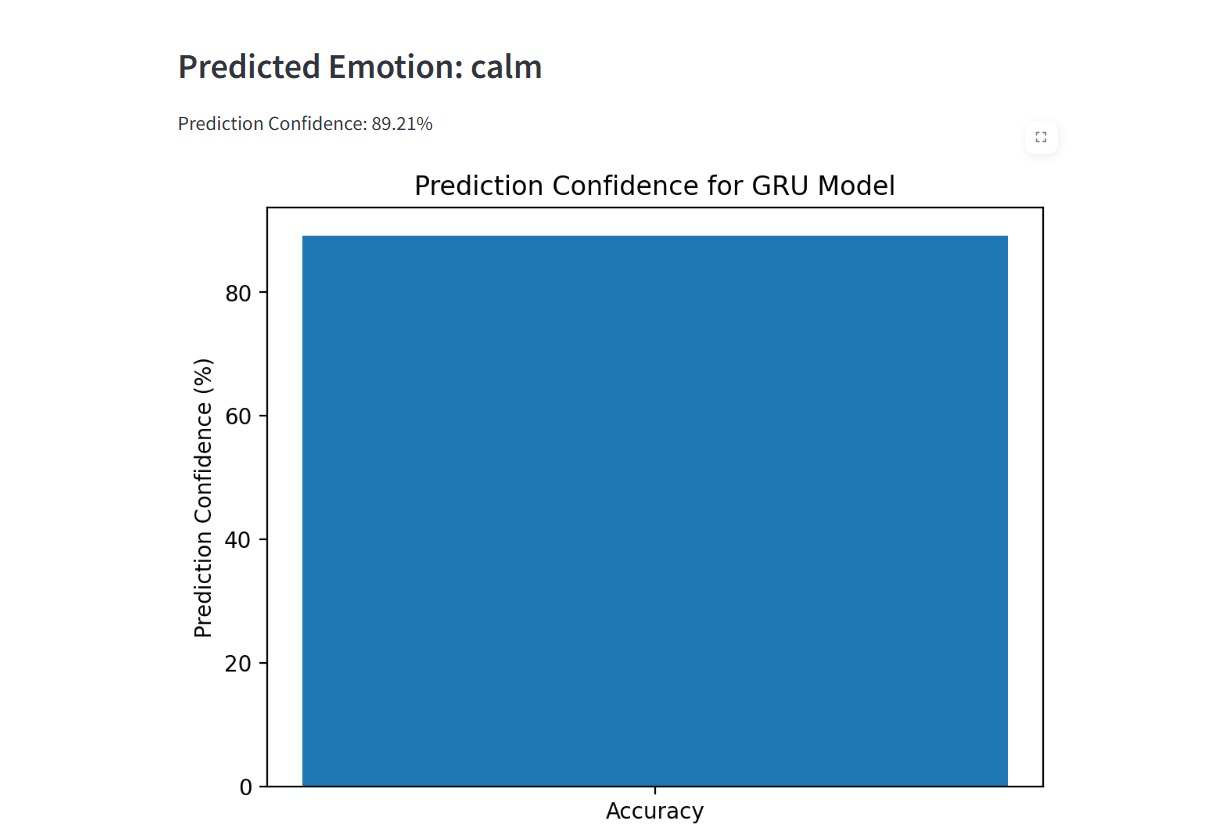
**10.2 Deployment Challenges**

UI Link-[View our UI](https://emotion-recognition-audio-lstm7gru.streamlit.app/%20)

Git Hub link - [Git\_hub\_link](https://github.com/Karm17/emotion-recognition-audio)

One of the main challenges during deployment was ensuring that the audio file uploads were processed efficiently in the cloud environment. Optimizations were made to ensure the app could handle large audio files without timing out.





**11. Conclusion**

The project successfully built and deployed an emotion recognition system using audio signals. The LSTM-based model demonstrated promising results in emotion classification. Future improvements could focus on enhancing the model with more advanced features, handling background noise better, and further tuning hyperparameters.

**12.** **References**

* Librosa documentation: https://librosa.org/doc/latest/
* Keras documentation: <https://keras.io/>
* Streamlit documentation: https://streamlit.io/docs
* TensorFlow documentation: <https://www.tensorflow.org/>