

Capstone Project – 5

Verbal Communication Quality Monitoring & Feedback System [Speech Emotion Recognition]

Team Members

Asim Siddiqui

Sagar Rokad

Suraj Kumar Mishra

Content

- Introduction
- Problem Statement
- Data Summary
- Data Preprocessing
- EDA
- Model Implementation
- Deployment
- Challenges
- Conclusion



Introduction

Q.1) What is Speech Emotion Recognition?

Speech Emotion Recognition (SER) can be defined as extraction of the emotional state of the speaker from his or her speech signal

Q.2) Why it is important?

Nowadays, more and more intelligent systems are using emotion recognition models to improve their interaction with humans. This is important, as the systems can adapt their responses and behavioral patterns according to the emotions of the humans and make the interaction more natural.

Q.3) What are the scopes of Speech Emotion Recognition?

Security, medicine, entertainment, education etc.

Problem Statement

Verbal Communication is valuable and sought after in workplace and classroom environments alike. There is no denying the notion that Indians lack verbal communication and consequently lag behind in the workplace or classroom environments. Clear and comprehensive speech is the vital backbone of strong communication and presentation skills.

AIM:- We will solve the above-mentioned problem by applying deep learning algorithms to audio/speech data. The solution will be to identify emotions in speech.

Data Summary

We have built a deep learning model which detects emotions of speech. For this purpose we've used Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)-2018 dataset. The description of dataset is as follows :

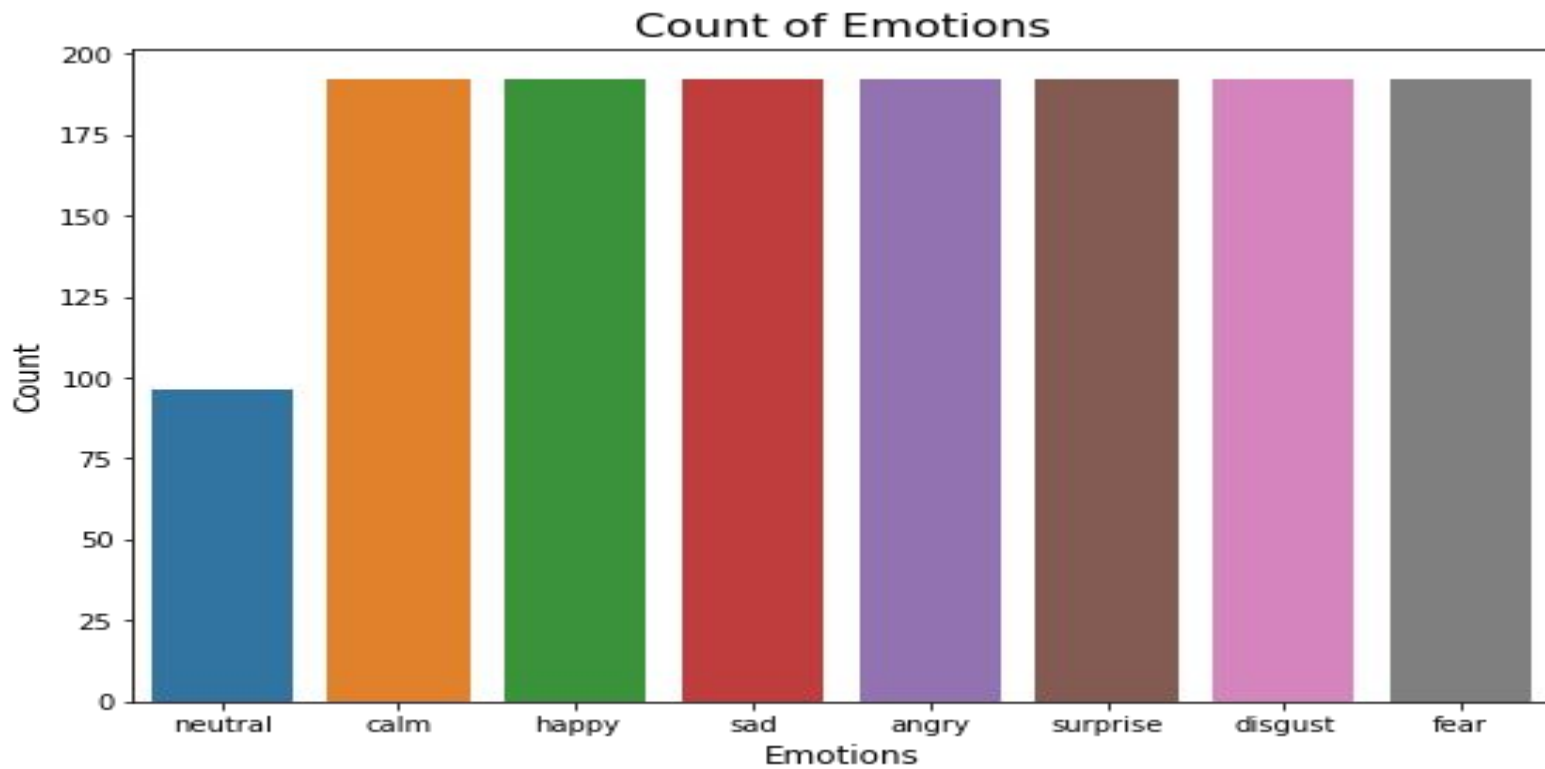
- RAVDESS contains 1440 files: 60 trials per actor x 24 actors = 1440. The RAVDESS contains 24 professional actors (12 female, 12 male), vocalizing two lexically-matched statements in a neutral North American accent.
- Speech emotions includes 8 emotions such as neutral, calm, happy, sad, angry, fearful, surprise, and disgust expressions.

Data Preprocessing

- **Extracting features using LIBROSA (librosa is a python package for music and audio analysis).**
- **Converting audio files into waveforms and spectrograms.**
- **Creating dataframe for extracted features for audio files.**
- **Data augmentation.**

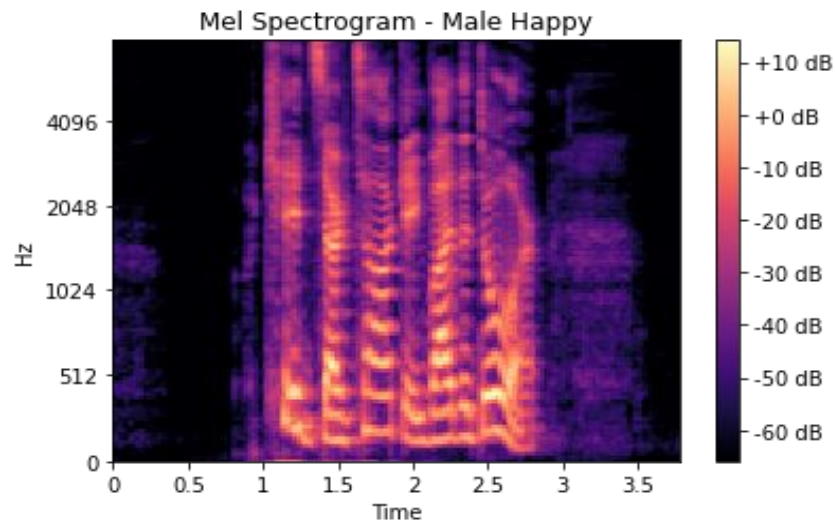
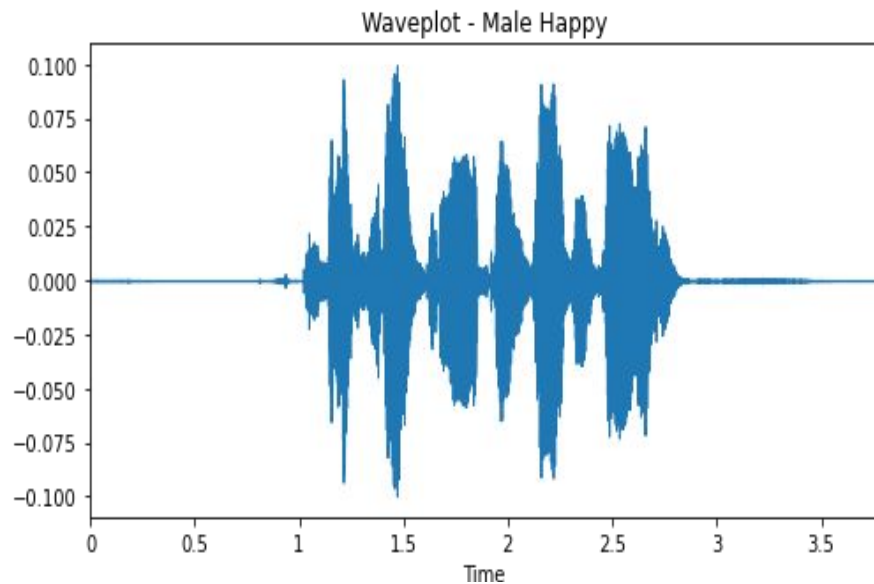
Exploratory Data Analysis

- How many emotions are there for each class in audio dataset?



EDA(continued)

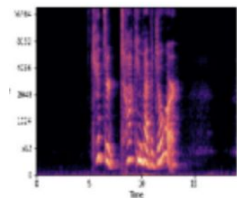
- A waveform is (mathematics) the shape of a wave function represented by a graph showing some dependent variable as function of an independent variable.
- A spectrogram is a visual representation of the spectrum of a sound changing through time.



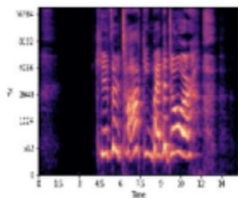
EDA(continued)

How different emotions looks in spectrogram?

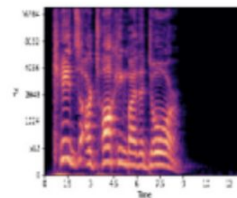
Surprised



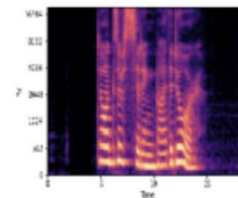
Fearful



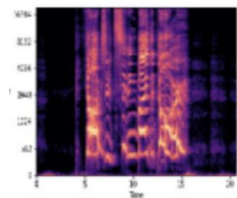
Angry



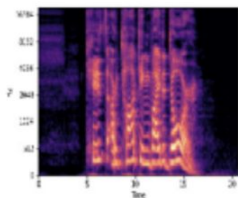
Calm



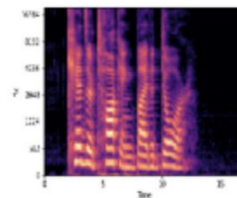
Fearful



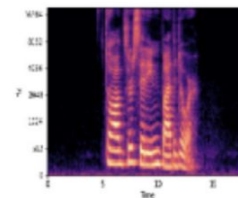
Happy



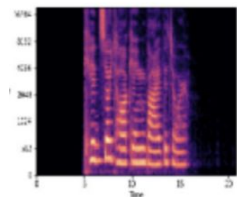
Angry



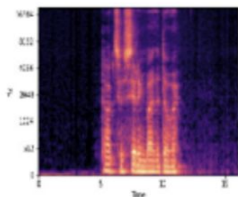
Neutral



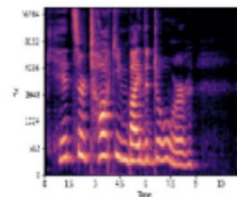
Calm



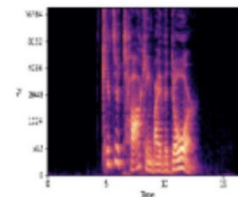
Fearful



Fearful

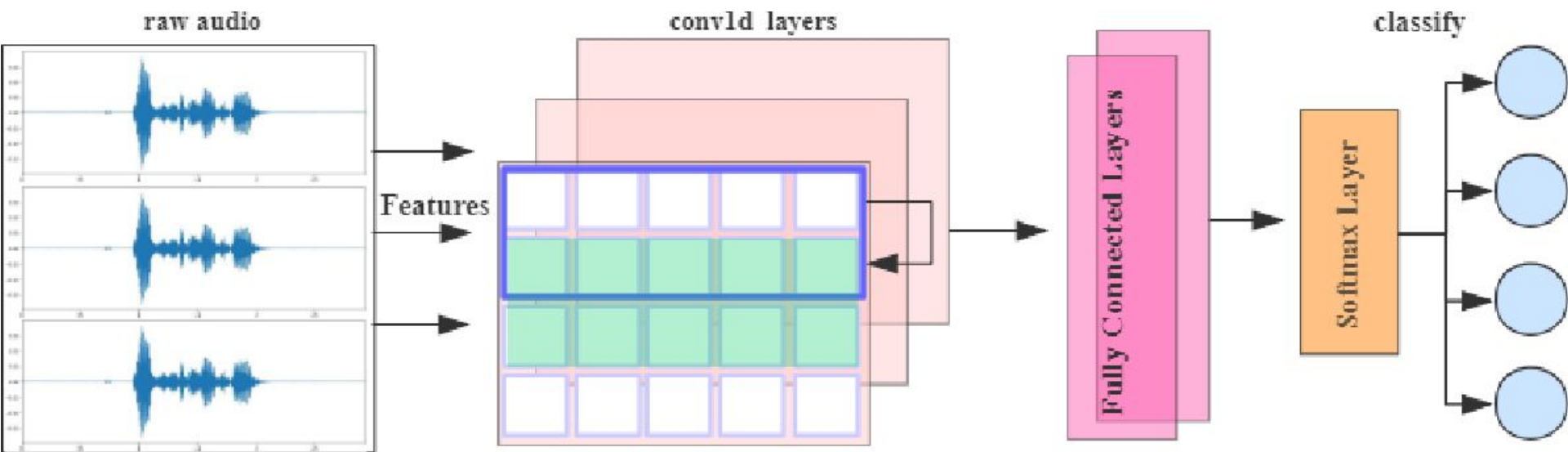


Disgust



Model implementation

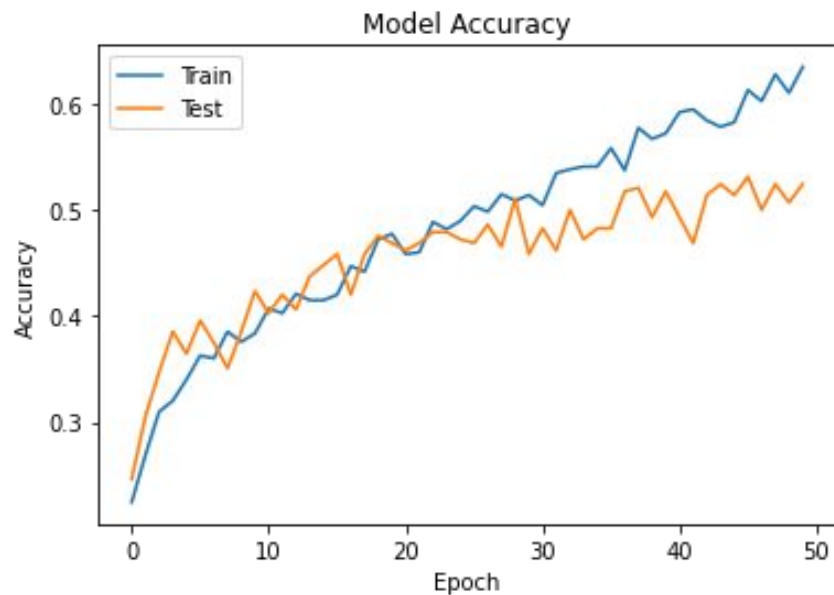
Base model CONV1D:



Model implementation (Conv1D contd..)



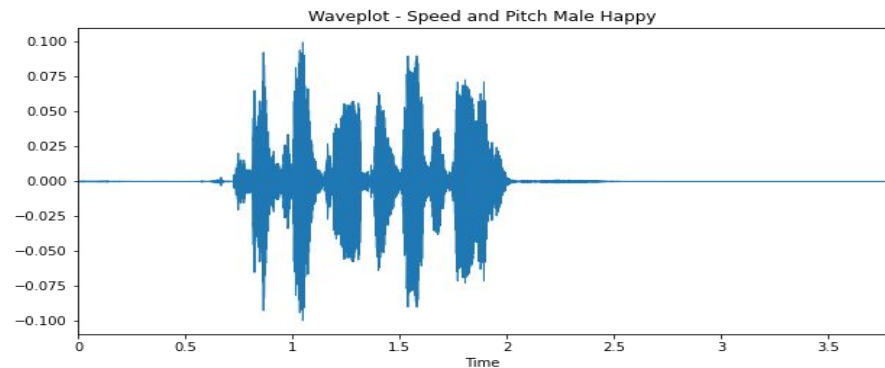
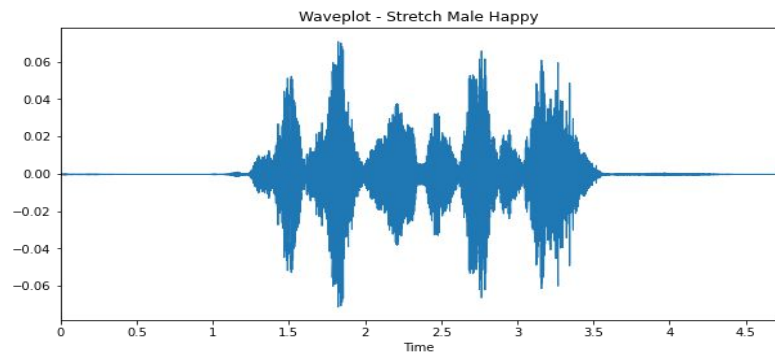
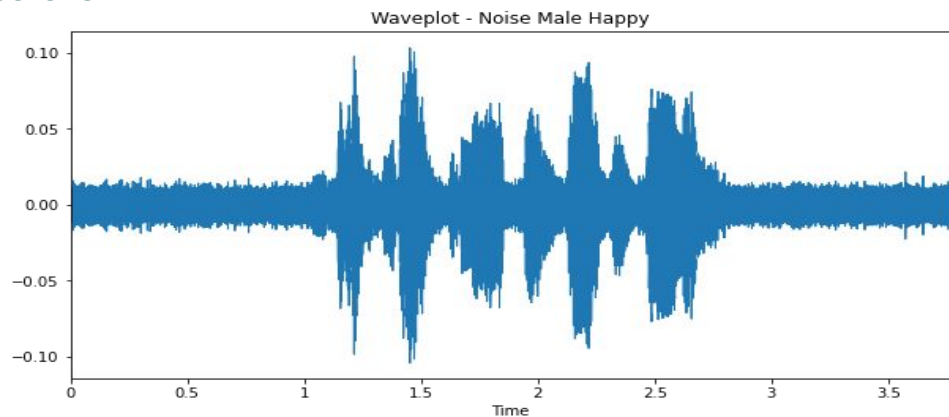
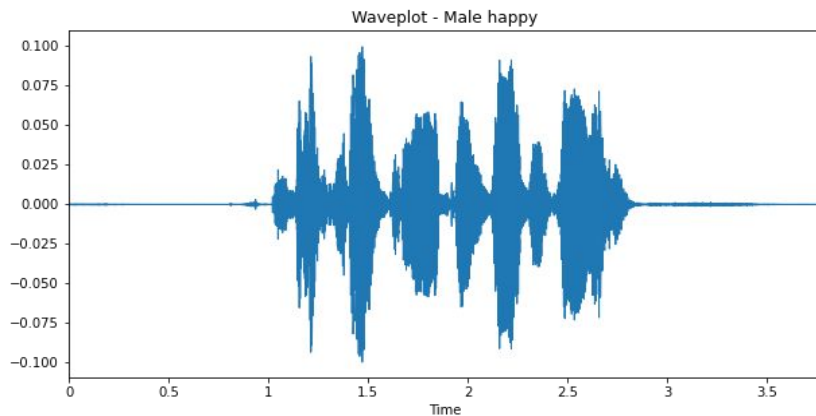
Base model Conv1D performance:



| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| angry | 0.50 | 0.51 | 0.51 | 39 |
| calm | 0.61 | 0.74 | 0.67 | 38 |
| disgust | 0.71 | 0.68 | 0.69 | 37 |
| fear | 0.73 | 0.21 | 0.33 | 38 |
| happy | 0.38 | 0.51 | 0.43 | 35 |
| neutral | 0.45 | 0.54 | 0.49 | 24 |
| sad | 0.40 | 0.47 | 0.43 | 38 |
| surprise | 0.62 | 0.54 | 0.58 | 39 |
| accuracy | | | | 288 |
| macro avg | 0.55 | 0.53 | 0.52 | 288 |
| weighted avg | 0.56 | 0.52 | 0.52 | 288 |

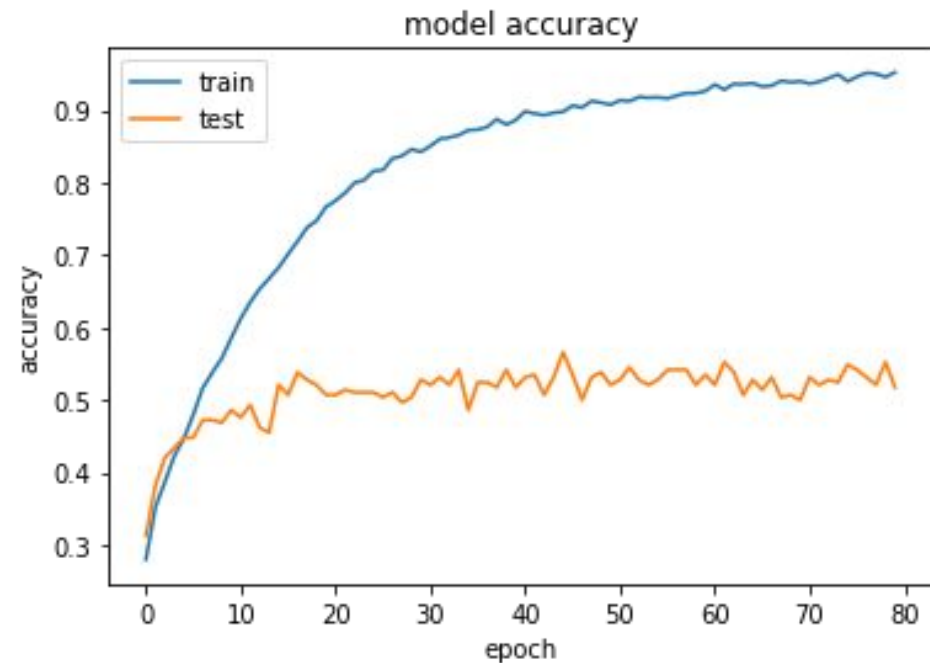
Model implementation (contd..)

How audio files looks after **data augmentation**?



Model implementation(Conv1D contd...)

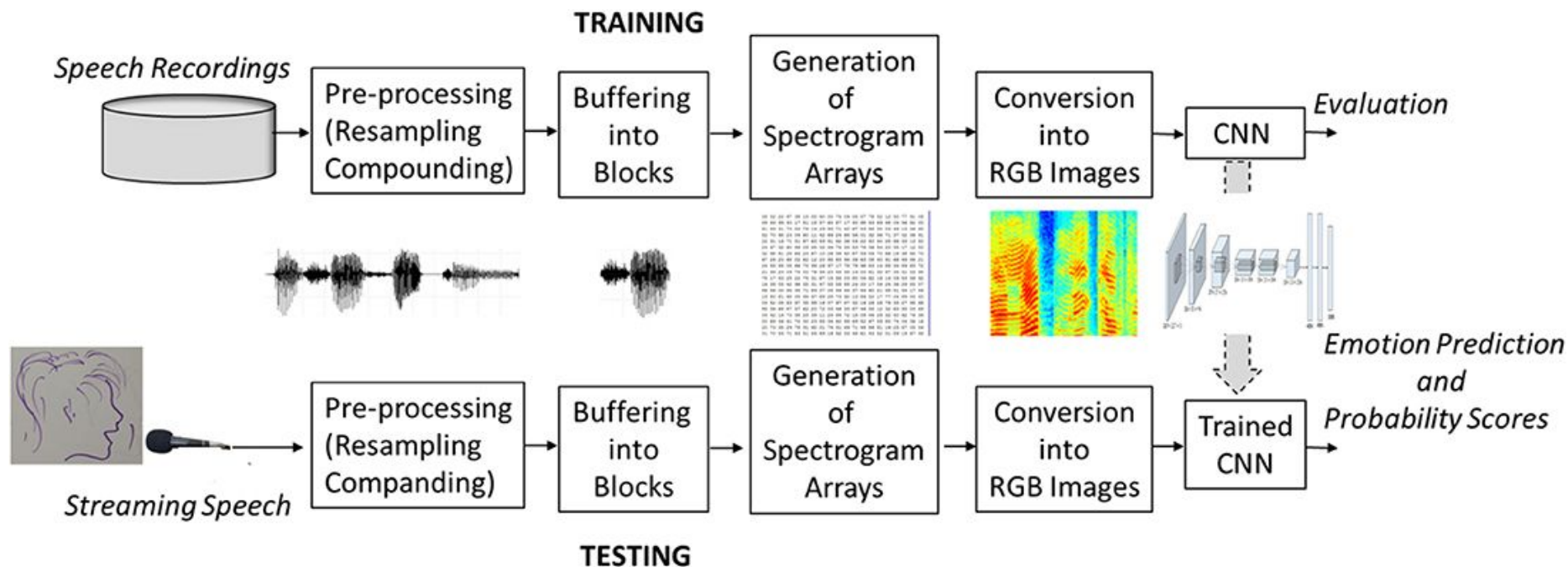
Implementing our base model Conv1D on augmented data:-



| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| angry | 0.58 | 0.56 | 0.57 | 39 |
| calm | 0.48 | 0.68 | 0.57 | 38 |
| disgust | 0.51 | 0.73 | 0.60 | 37 |
| fear | 0.59 | 0.42 | 0.49 | 38 |
| happy | 0.50 | 0.49 | 0.49 | 35 |
| neutral | 0.42 | 0.58 | 0.49 | 24 |
| sad | 0.47 | 0.21 | 0.29 | 38 |
| surprise | 0.59 | 0.49 | 0.54 | 39 |
| accuracy | | | 0.52 | 288 |
| macro avg | 0.52 | 0.52 | 0.50 | 288 |
| weighted avg | 0.52 | 0.52 | 0.51 | 288 |

Model Implementation(VGG-16 contd..)

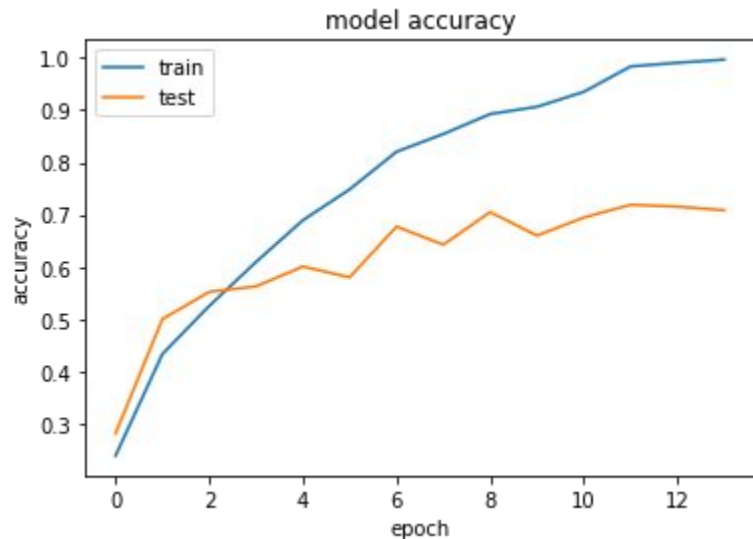
We then used transfer learning concept with VGG-16 Model



Model implementation (VGG-16 contd..)



VGG-16 performance:

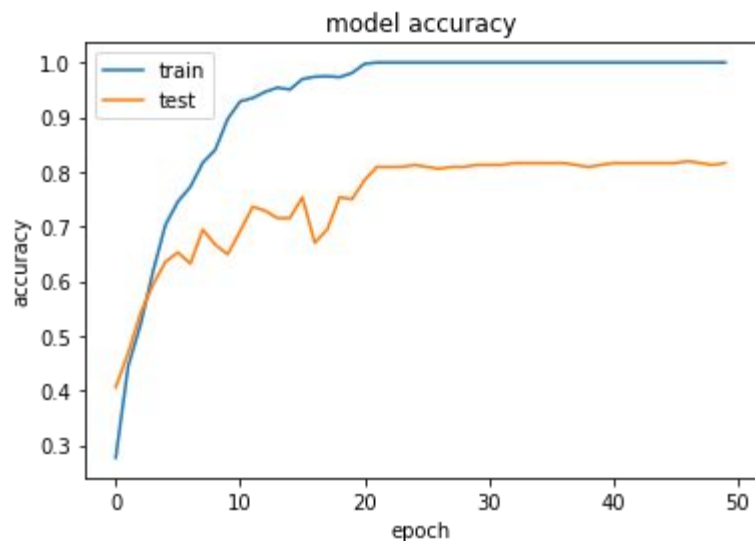


| | precision | recall | f1-score | support | |
|--|--------------|--------|----------|---------|-----|
| | angry | 0.87 | 0.67 | 0.75 | 39 |
| | calm | 0.81 | 0.79 | 0.80 | 38 |
| | disgust | 0.72 | 0.76 | 0.74 | 37 |
| | fearful | 0.71 | 0.71 | 0.71 | 38 |
| | happy | 0.54 | 0.63 | 0.58 | 35 |
| | neutral | 0.65 | 0.71 | 0.68 | 24 |
| | sad | 0.57 | 0.55 | 0.56 | 38 |
| | surprised | 0.82 | 0.85 | 0.84 | 39 |
| | accuracy | | | 0.71 | 288 |
| | macro avg | 0.71 | 0.71 | 0.71 | 288 |
| | weighted avg | 0.72 | 0.71 | 0.71 | 288 |

Model implementation (VGG-19 contd..)



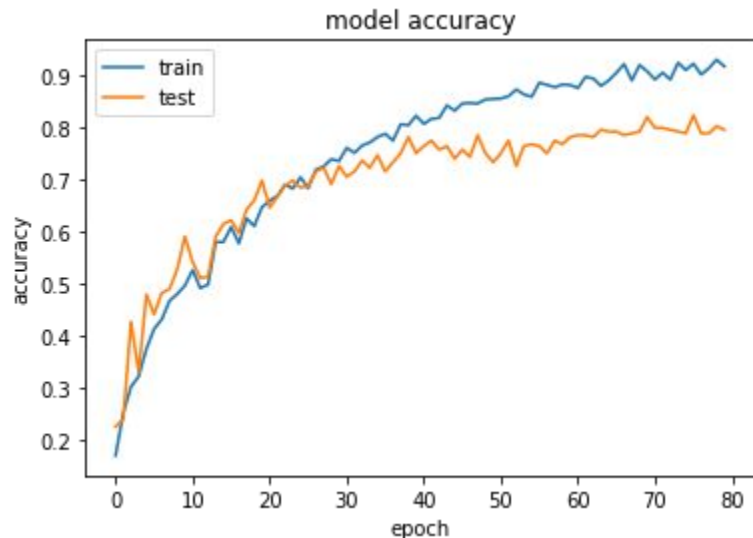
VGG-19 (Fine Tuning) performance:



| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| angry | 0.93 | 0.72 | 0.81 | 39 |
| calm | 0.87 | 0.87 | 0.87 | 38 |
| disgust | 0.82 | 0.89 | 0.86 | 37 |
| fearful | 0.82 | 0.71 | 0.76 | 38 |
| happy | 0.71 | 0.86 | 0.78 | 35 |
| neutral | 0.81 | 0.88 | 0.84 | 24 |
| sad | 0.72 | 0.76 | 0.74 | 38 |
| surprised | 0.87 | 0.87 | 0.87 | 39 |
| accuracy | | | 0.82 | 288 |
| macro avg | 0.82 | 0.82 | 0.82 | 288 |
| weighted avg | 0.82 | 0.82 | 0.82 | 288 |

Model implementation (VGG-19 contd..)

VGG-19 (Fine Tuning + Data Augmentation) performance:



| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| angry | 0.78 | 0.82 | 0.80 | 39 |
| calm | 0.76 | 0.89 | 0.82 | 38 |
| disgust | 0.77 | 0.92 | 0.84 | 37 |
| fearful | 0.90 | 0.71 | 0.79 | 38 |
| happy | 0.79 | 0.66 | 0.72 | 35 |
| neutral | 0.86 | 0.75 | 0.80 | 24 |
| sad | 0.77 | 0.61 | 0.68 | 38 |
| surprised | 0.79 | 0.97 | 0.87 | 39 |
| accuracy | | | 0.80 | 288 |
| macro avg | 0.80 | 0.79 | 0.79 | 288 |
| weighted avg | 0.80 | 0.80 | 0.79 | 288 |

Model implementation (contd..)



We saved our implemented models with best values using checkpoints

| Models (CNN) | Train accuracy | Test accuracy |
|--|----------------|---------------|
| Conv 1D | 0.95 | 0.51 |
| VGG-16 Fine tuning | 0.98 | 0.71 |
| VGG-19 Fine tuning | 1.00 | 0.81 |
| VGG-19 (Fine tuning + Augmentation) | 0.91 | 0.82 |
| Inception | 0.97 | 0.73 |

Deployment

AI

Tools used in deployment of deep learning project:-

Streamlit- Streamlit is an open-source python library that is useful to create and share data web apps.



GCP - Google Cloud Platform, commonly referred to as GCP, is a cloud computing service created by Google for building, testing, deploying, and managing applications and services through Google-managed data centers.

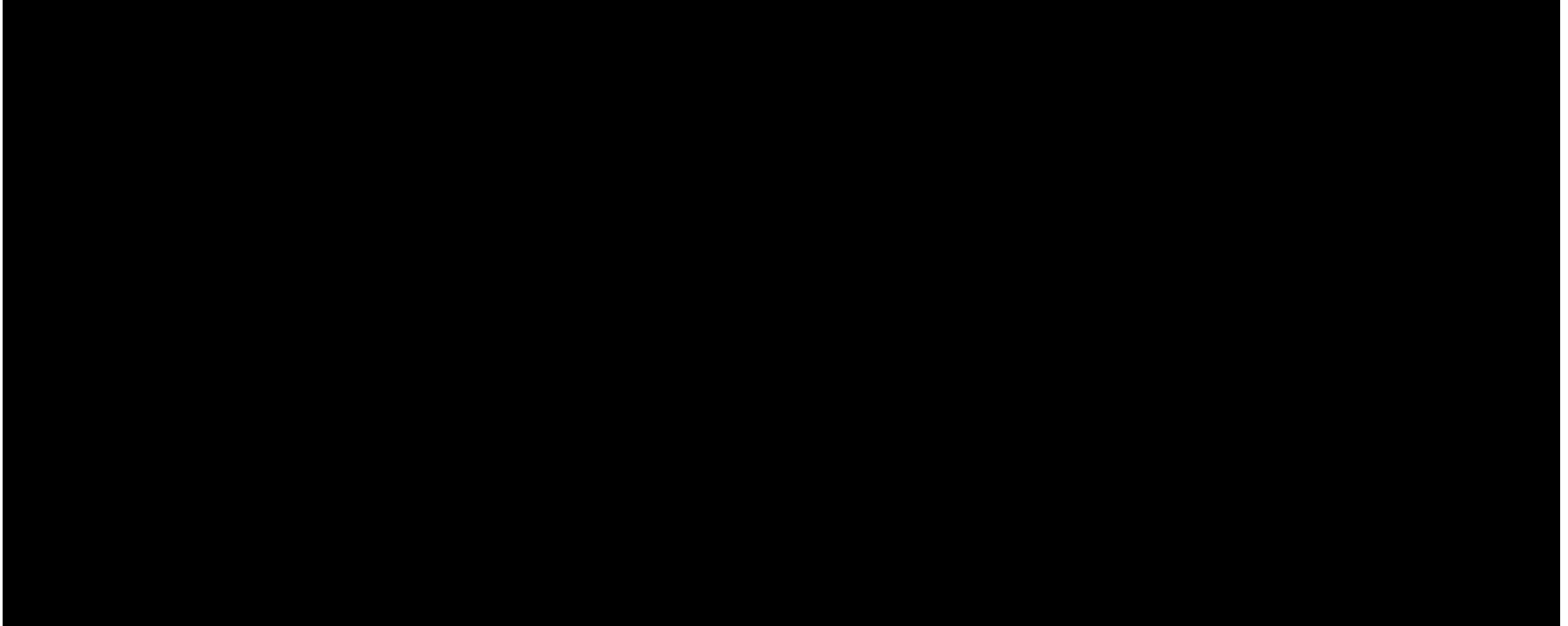


Google Cloud Platform

Deployment(contd..)

Implementation video of our project with a web app.

URL- <https://ser-app-324611.ue.r.appspot.com/>



Challenges

- As this project is not a stand-alone data science project, it required little to medium domain knowledge. Understanding Mfccs, Mel scale is very important for feature selection.
- Limitations include not using feature selection to reduce the dimensionality of our augmented CNN which may have improved learning performance.
- During deployment in Heroku and Azure, there were problems as some additional libraries needed to be installed. After installing these libraries, some of the previous requirements had to be downgraded in order to be compatible with the installed ones, that's why we deployed on GCP.

Conclusion

- VGG19 (fine tuning + augmentation) was giving the best accuracy score of 82% and solved the problems like overfitting to some extent.
- It's quite difficult to get the accuracy of more than 90% due to lack of data.
- To solve problems like over-fitting that we had seen in almost every model, we need more real time data.
- Noise Adding ,Pitching and Shifting for the imbalanced data was helping in achieving a better result.
- Computational cost was much high resulting in several runtime crashes but we're able to get our best model for deployment.

Thank You

