Capstone Project-5

Speech Emotion Recognition

Deep Learning & ML Engineering Project

Palash Pathak

OVERVIEW

1. Defining Problem Statement

2. Data Collection

3. Exploratory Data Analysis

4. Data augmentation

5. Extracting features from audio

4. Processing features

5. Defining model

6. Training and validating Model

7. Select best model

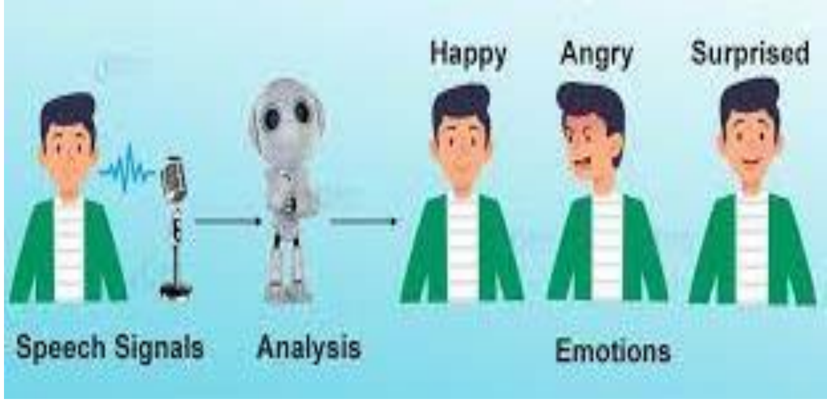
8. Deploy model

Overview of the Problem Statement

- Speech Emotion Recognition, abbreviated as SER, is the act of attempting to recognize human emotion and affective states from speech. This is capitalizing on the fact that voice often reflects underlying emotion through tone and pitch.

- Tasks for the Project:

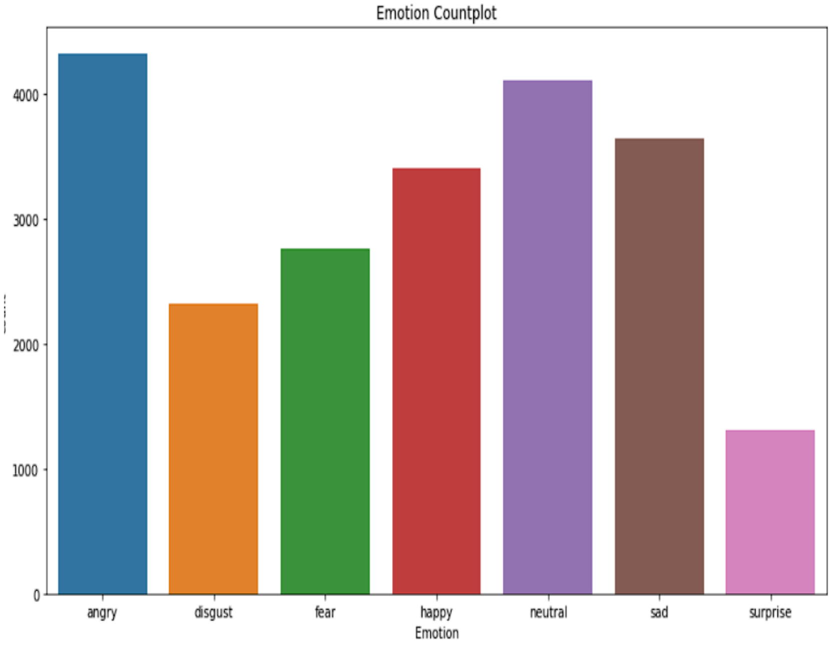
1. Collect data from public databases.
2. Process audio data for applying model.
3. Identify tone/emotion from audio.
4. Deploy model on Azure platform.

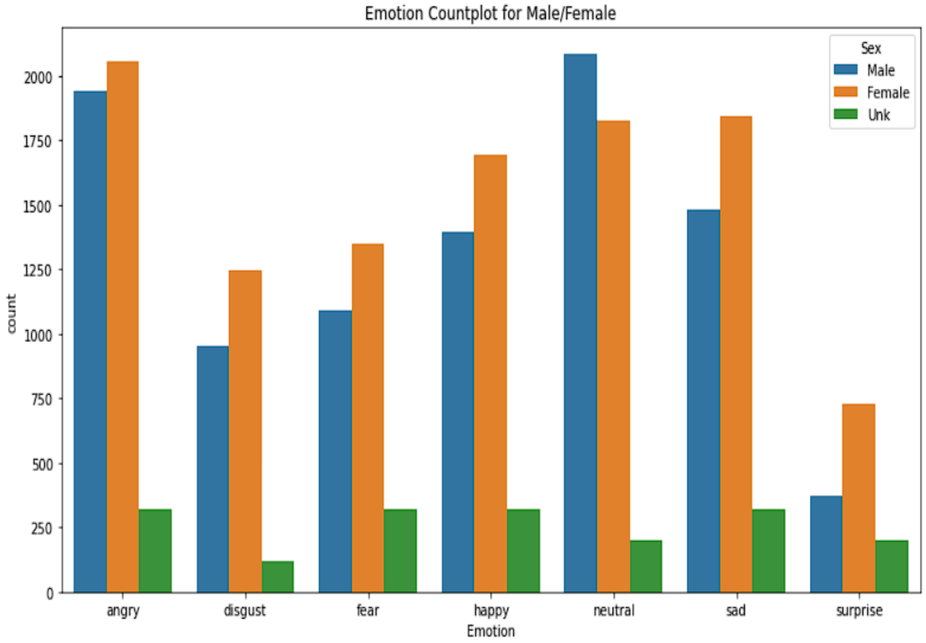


Dataset used:

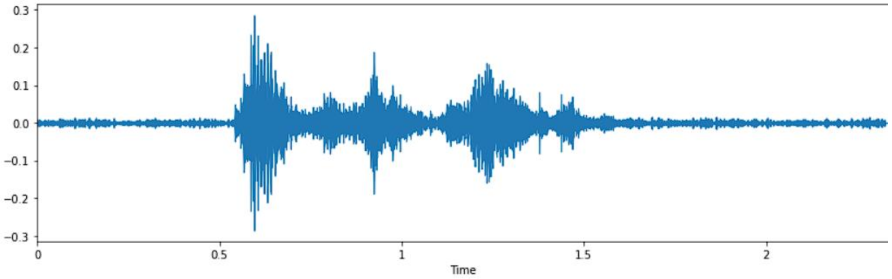
Name	Language	Gender	Source link	Description
CREMA-D	English	Both	https://github.com/CheyneyComputerScience/CREMA-D	7,442 original clips from 48 male and 43 female actors spoken in 7 diff emotions.
TESS	English	Female	https://tspace.library.utoronto.ca/handle/1807/24487	Toronto Emotional Speech Set: 2 female speakers (young and old), 2800 audio files, random words were spoken in 7 different emotions.
SAVEE	English	Male	https://www.kaggle.com/datasets/ejlok1/surrey-audiovisual-expressed-emotion-savee	Surrey Audio-Visual Expressed Emotion: 4 male speakers, 480 audio files, same sentences were spoken in 7 different emotions.
RAVDEES	English	Male	https://zenodo.org/record/1188976#.YntXEehBxPY	2452 audio files, with 12 male speakers and 12 Female speakers, speaking only 2 statements of equal lengths in 8 different emotions by all speakers.
BERLIN	German	Both	https://www.kaggle.com/datasets/piyushagni5/berlin-database-of-emotional-speech-emodb	5 male and 5 female speakers, 535 audio files, 10 different sentences were spoken in 7 different emotions.
EMOVO	Italian	Both	http://voice.fub.it/activities/corpora/emovo/index.html	It is a database built from the voices of 3 male and 3 female actors who played 14 sentences simulating 6 emotional states.
CASIA	Chinese	Both	http://shachi.org/resources/27	Chinese Emotional Speech Corpus Four professional speakers are required to utter 500 sentences in 6 emotions.
SHEMO	Persian	Both	https://github.com/mansourehk/ShEMO	Sharif Emotional Speech Database: 3000 utterances,87 native-Persian speakers for five basic emotions.
CaFE	Canadian French	Both	https://zenodo.org/record/1478765#.Yntal-hBxPY	Canadian French Emotional contains six different sentences, pronounced by 6 male and 6 female actors, in 7 basic emotions.
AESDD	GREEK	Both	http://m3c.web.auth.gr/research/aesdd-speech-emotion-recognition/	Acted Emotional Speech Dynamic Database: 3 female and 2 male actors were recorded. The actors acted these 19 utterances in 5 chosen emotions.
J L Corpus	English	Both	https://www.kaggle.com/datasets/tli725/jl-corpus	2400 recording of 240 sentences by 2 males and 2 female actors in 5 emotions.

Emotions Distribution:

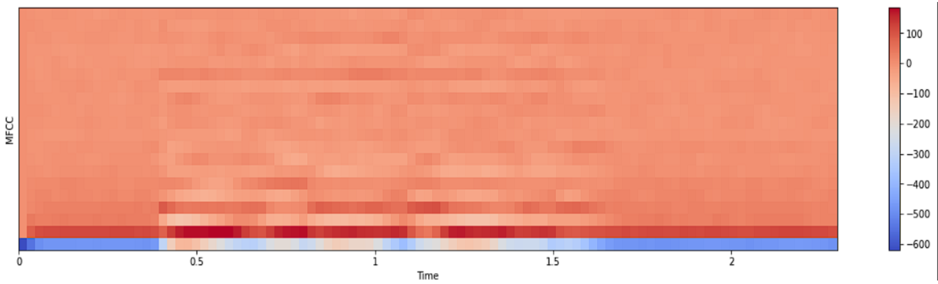




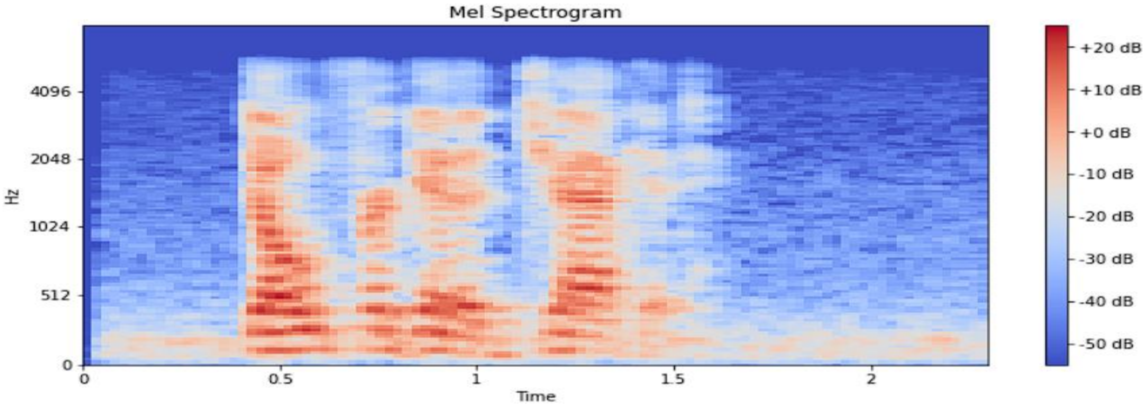
Audio features:



Waveform

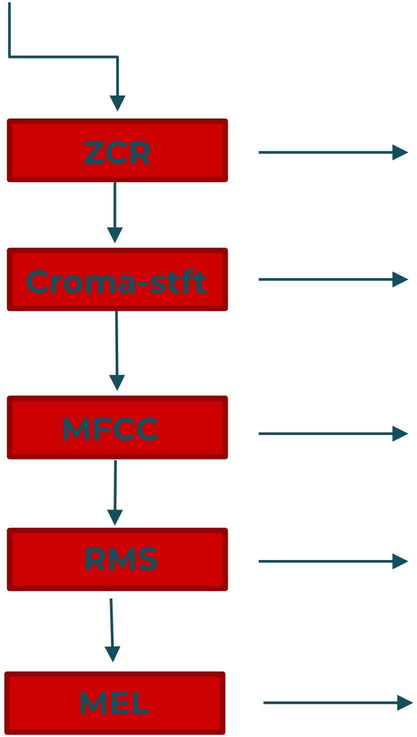


MFCC



Mel spectrogram

Extracting features from Audio data:



ZCR → The zero-crossing rate (ZCR) is the rate at which a signal transitions from positive to zero to negative or negative to zero to positive.

Croma-stft → Compute a chromagram from a waveform. Chromagram is defined as the whole spectral audio information mapped into one octave.

MFCC → Mel-frequency cepstral coefficients (MFCCs) are coefficients that collectively make up an MFC.

RMS → Compute root-mean-square (RMS) value for each frame from the audio sample.

MEL → Compute a mel-scaled spectrogram

Data Augmentation:

Add Noise

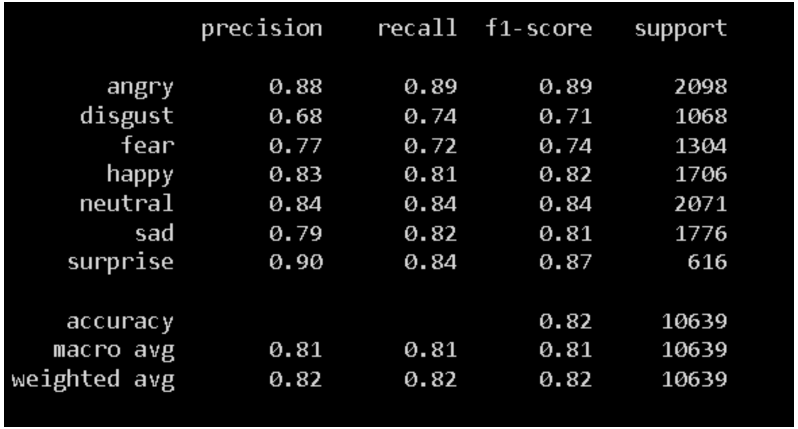
Stretch

Shift

Alter Pitch

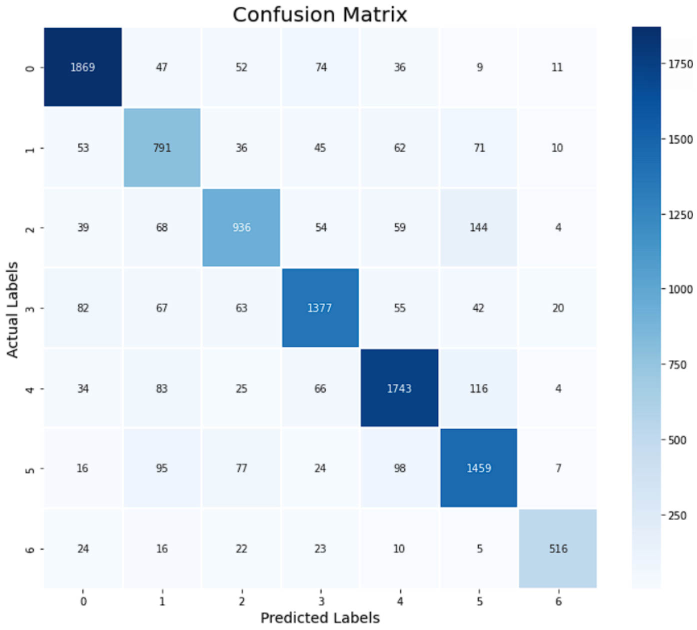
Model-1 : MLPClassifier

MLPClassifier from sklearn-neural network.
Hidden layer sizes = (256,256,64)
It works well, lets keep this as base model.
Accuracy on train/test set is 91% / 81%.
From the classification report on test set,
Its evident that model is performing
poor on ‘disgust’ and ‘fear’ emotions.



```
              precision    recall  f1-score   support

       angry       0.88      0.89      0.89      2098
     disgust       0.68      0.74      0.71      1068
        fear       0.77      0.72      0.74      1304
       happy       0.83      0.81      0.82      1706
     neutral       0.84      0.84      0.84      2071
         sad       0.79      0.82      0.81      1776
    surprise       0.90      0.84      0.87       616

    accuracy                           0.82     10639
   macro avg       0.81      0.81      0.81     10639
weighted avg       0.82      0.82      0.82     10639
```

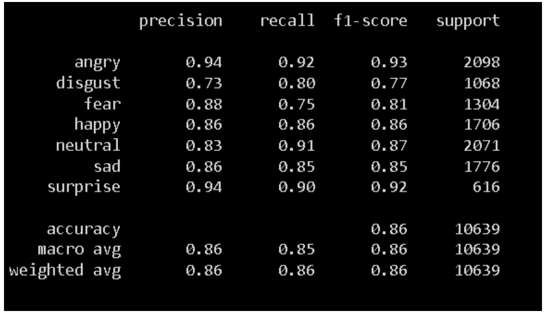
Classification Report
on Test set



CM for test set

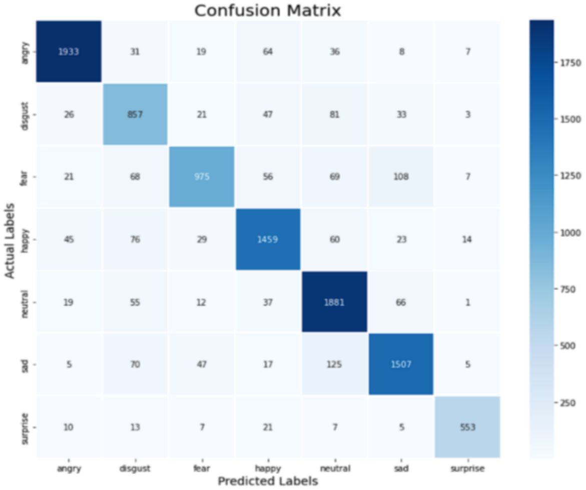
Model-2 : CNN

A Custom CNN network .
Dropout technique is used to reduce overfitting.
Accuracy on train/test set is 87% / 86%.
Total parameters: 6L

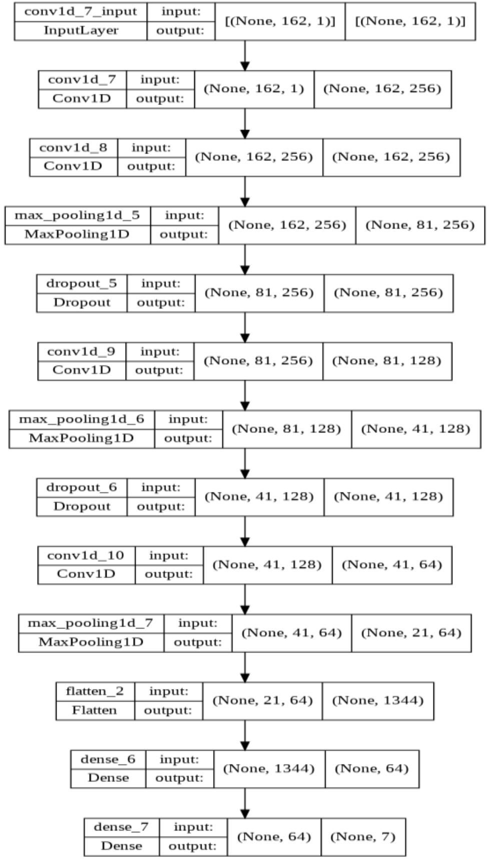


```
              precision    recall  f1-score   support

       angry       0.94      0.92      0.93      2098
     disgust       0.73      0.80      0.77      1068
        fear       0.88      0.75      0.81      1304
       happy       0.86      0.86      0.86      1706
     neutral       0.83      0.91      0.87      2071
         sad       0.86      0.85      0.85      1776
    surprise       0.94      0.90      0.92       616

    accuracy                           0.86     10639
   macro avg       0.86      0.85      0.86     10639
weighted avg       0.86      0.86      0.86     10639
```

CM for test set



Model



conv1d_7_input | InputLayer | input: [(None, 162, 1)] | output: [(None, 162, 1)]

conv1d_7 | Conv1D | input: (None, 162, 1) | output: (None, 162, 256)

conv1d_8 | Conv1D | input: (None, 162, 256) | output: (None, 162, 256)

max_pooling1d_5 | MaxPooling1D | input: (None, 162, 256) | output: (None, 81, 256)

dropout_5 | Dropout | input: (None, 81, 256) | output: (None, 81, 256)

conv1d_9 | Conv1D | input: (None, 81, 256) | output: (None, 81, 128)

max_pooling1d_6 | MaxPooling1D | input: (None, 81, 128) | output: (None, 41, 128)

dropout_6 | Dropout | input: (None, 41, 128) | output: (None, 41, 128)

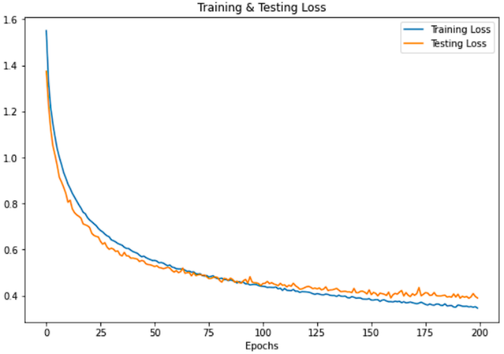
conv1d_10 | Conv1D | input: (None, 41, 128) | output: (None, 41, 64)

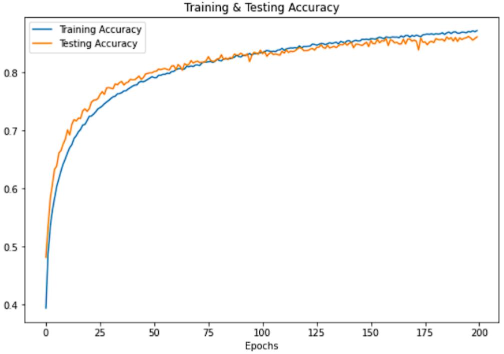
max_pooling1d_7 | MaxPooling1D | input: (None, 41, 64) | output: (None, 21, 64)

flatten_2 | Flatten | input: (None, 21, 64) | output: (None, 1344)

dense_6 | Dense | input: (None, 1344) | output: (None, 64)

dense_7 | Dense | input: (None, 64) | output: (None, 7)

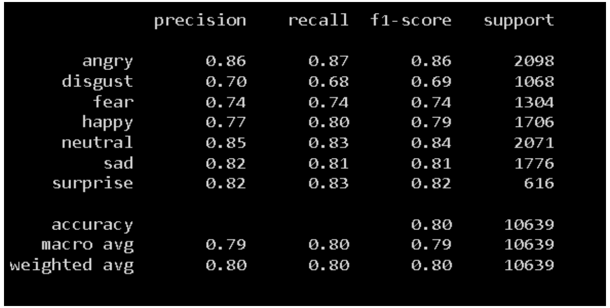




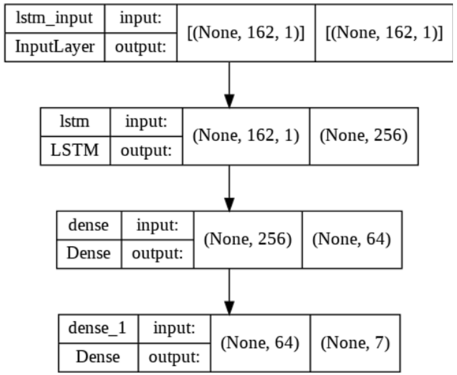
Model performance

Model-3 : LSTM

A Custom LSTM network .
There is overfitting.
Accuracy on train/test set is 98% / 80%.
Parameters : 2.8L

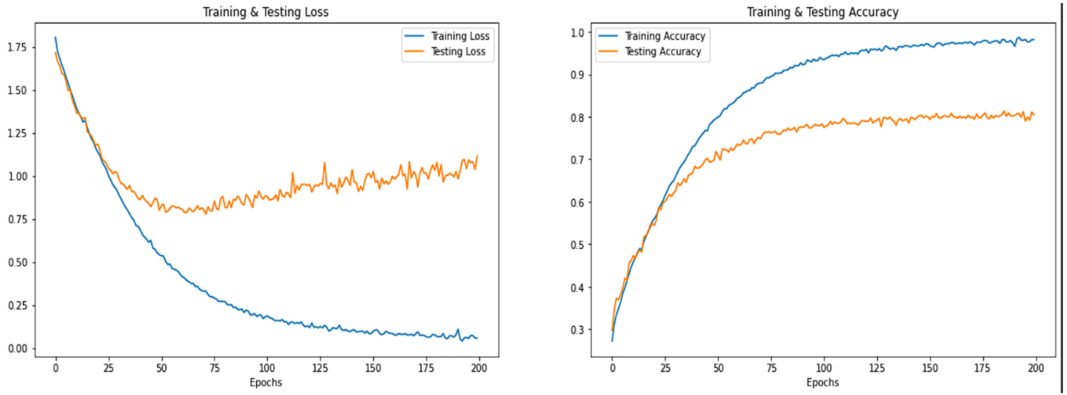


```
              precision    recall  f1-score   support

       angry       0.86      0.87      0.86      2098
     disgust       0.70      0.68      0.69      1068
        fear       0.74      0.74      0.74      1304
       happy       0.77      0.80      0.79      1706
     neutral       0.85      0.83      0.84      2071
         sad       0.82      0.81      0.81      1776
    surprise       0.82      0.83      0.82       616

    accuracy                           0.80     10639
   macro avg       0.79      0.80      0.79     10639
weighted avg       0.80      0.80      0.80     10639
```

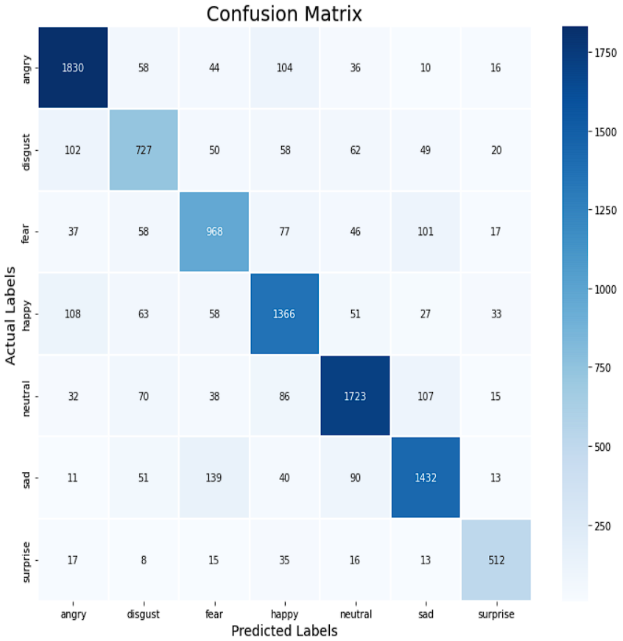


lstm_input InputLayer	input: output:	[(None, 162, 1)]	[(None, 162, 1)]
lstm LSTM	input: output:	(None, 162, 1)	(None, 256)
dense Dense	input: output:	(None, 256)	(None, 64)
dense_1 Dense	input: output:	(None, 64)	(None, 7)

Model



Model performance

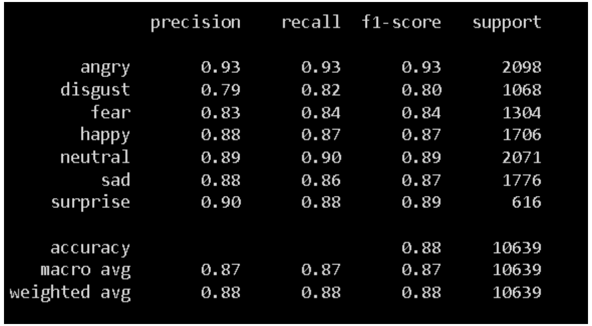


	angry	disgust	fear	happy	neutral	sad	surprise
angry	1830	58	44	104	36	10	16
disgust	102	727	50	58	62	49	20
fear	37	58	968	77	46	101	17
happy	108	63	58	1366	51	27	33
neutral	32	70	38	86	1723	107	15
sad	11	51	139	40	90	1432	13
surprise	17	8	15	35	16	13	512

CM for test set

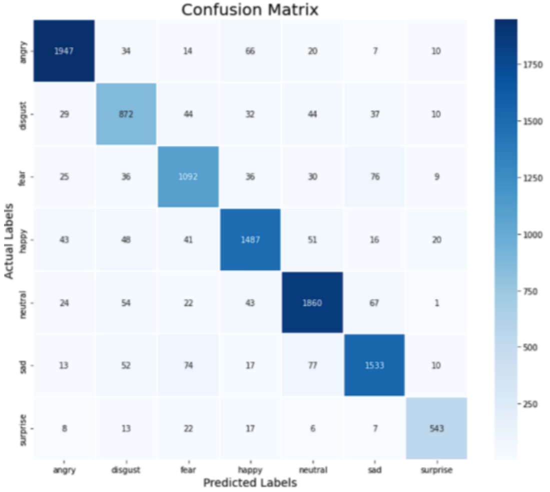
Model-5 : CNN+ LSTM

Adding LSTM to CNN network .
Dropout technique is used to reduce overfitting.
Accuracy on train/test set is 80% / 71%.
Parameters: 6.3L



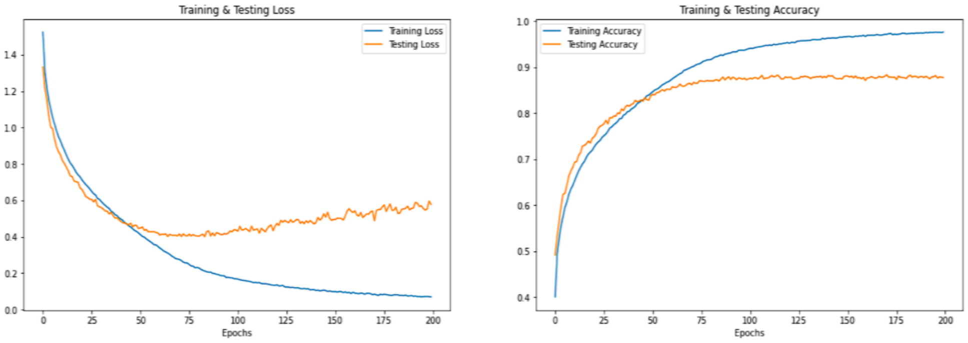
```
              precision    recall  f1-score   support

       angry       0.93      0.93      0.93      2098
     disgust       0.79      0.82      0.80      1068
        fear       0.83      0.84      0.84      1304
       happy       0.88      0.87      0.87      1706
     neutral       0.89      0.90      0.89      2071
         sad       0.88      0.86      0.87      1776
    surprise       0.90      0.88      0.89       616

    accuracy                           0.88     10639
   macro avg       0.87      0.87      0.87     10639
weighted avg       0.88      0.88      0.88     10639
```

CM for test set



Confusion Matrix

	angry	disgust	fear	happy	neutral	sad	surprise
angry	1947	34	14	66	20	7	10
disgust	29	872	44	32	44	37	10
fear	25	36	1092	36	30	76	9
happy	43	48	41	1487	51	16	20
neutral	24	54	22	43	1860	67	1
sad	13	52	74	17	77	1533	10
surprise	8	13	22	17	6	7	543

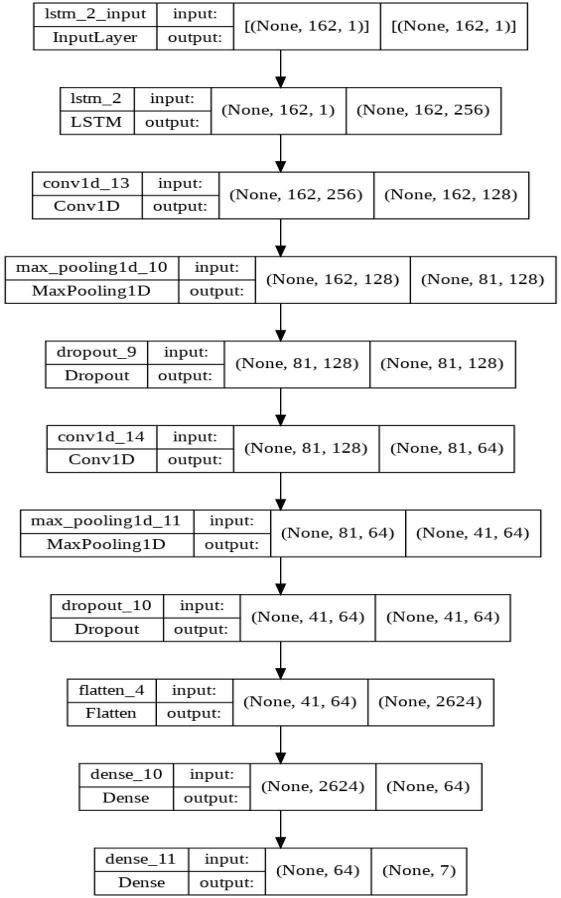


Training & Testing Loss

Training & Testing Accuracy

Model performance

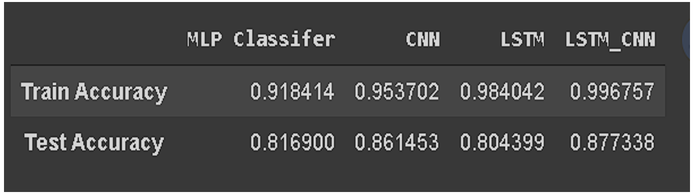
Model



Model Selection

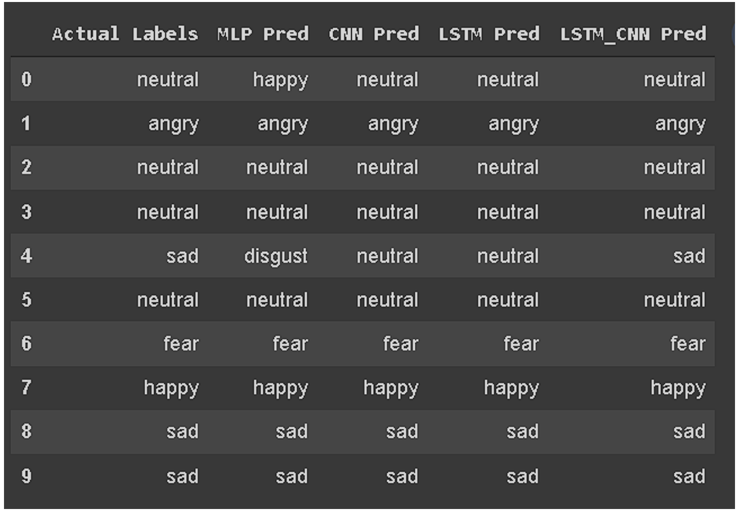
1. MLP Classifier performed well on the data with 91% and 81% accuracy on Train/Test sets resp. Handling overfitting is a challenge for a ANN network.
2. CNN model with around 6L parameters resulted in accuracy of 95%/86% on train/test sets. Maxpool layer and dropout is utilized in training.
3. LSTM model with over 2.8L parameters resulted in accuracy of 98%/80% on train/test set showing overfitting.
4. A combination of LSTM and CNN helped reducing overfitting and resulted in 99%/88% train/test accuracy. Hence, I have selected this model for deployment.

Accuracy Table



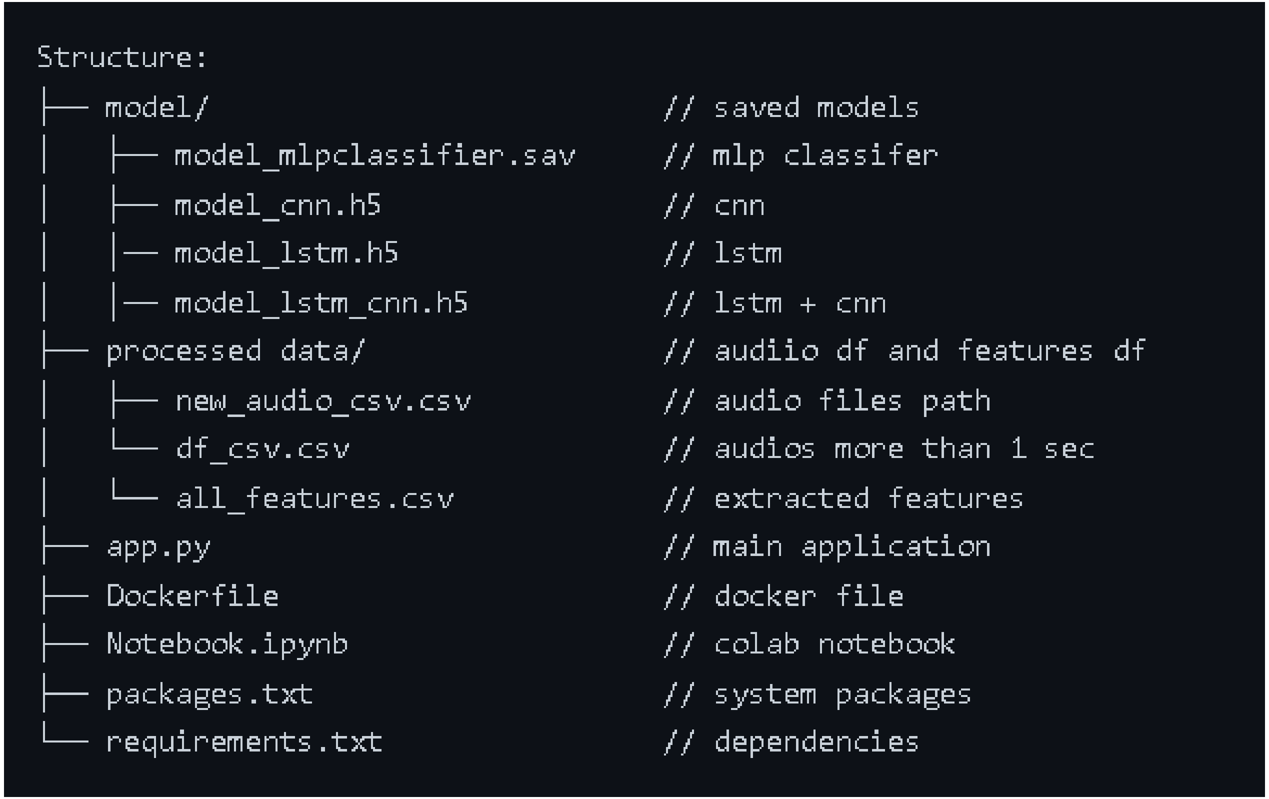
	MLP Classifer	CNN	LSTM	LSTM_CNN
Train Accuracy	0.918414	0.953702	0.984042	0.996757
Test Accuracy	0.816900	0.861453	0.804399	0.877338

Prediction Table



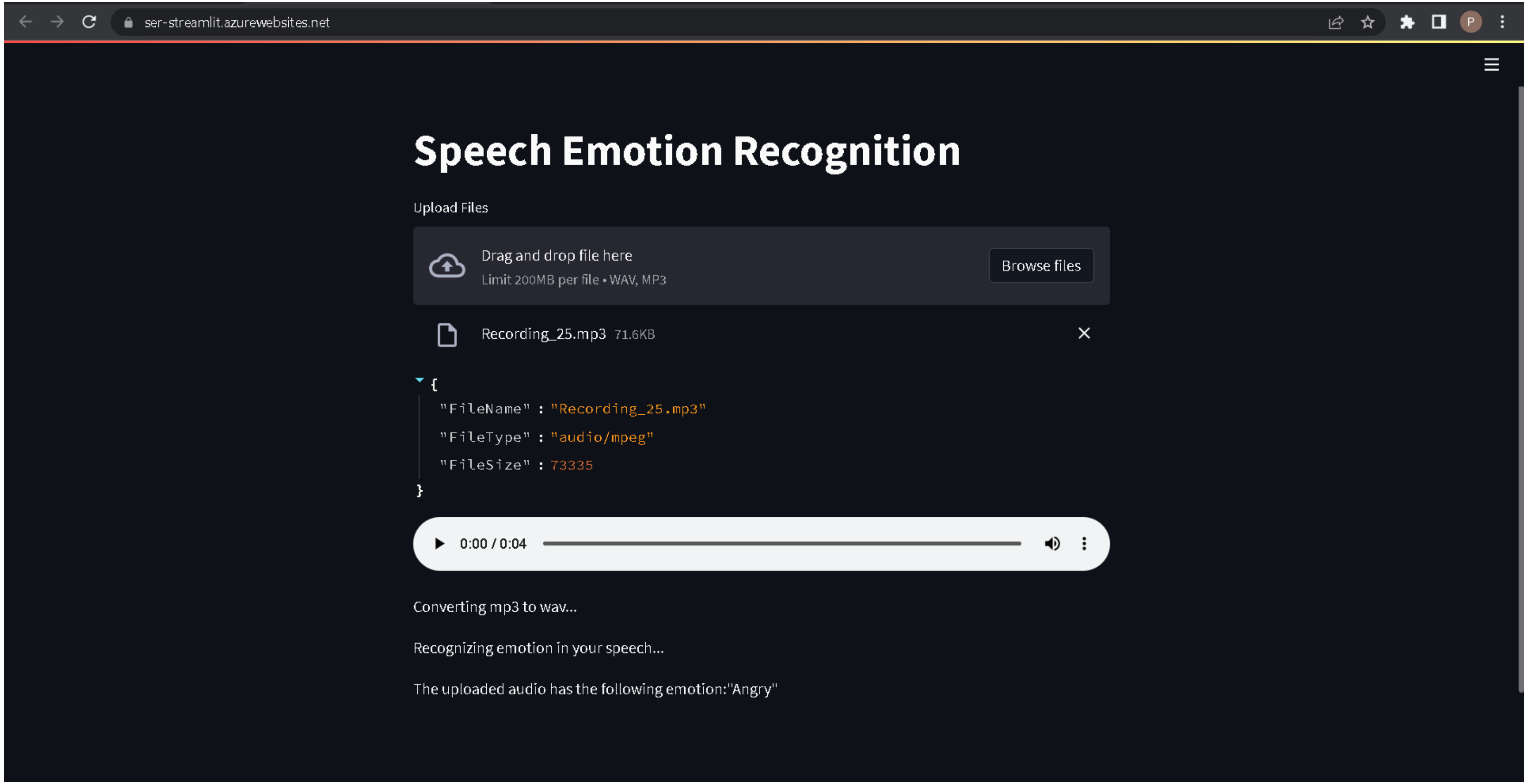
	Actual Labels	MLP Pred	CNN Pred	LSTM Pred	LSTM_CNN Pred
0	neutral	happy	neutral	neutral	neutral
1	angry	angry	angry	angry	angry
2	neutral	neutral	neutral	neutral	neutral
3	neutral	neutral	neutral	neutral	neutral
4	sad	disgust	neutral	neutral	sad
5	neutral	neutral	neutral	neutral	neutral
6	fear	fear	fear	fear	fear
7	happy	happy	happy	happy	happy
8	sad	sad	sad	sad	sad
9	sad	sad	sad	sad	sad

Project Structure



```
Structure:
├── model/                          // saved models
│   ├── model_mlpclassifier.sav    // mlp classifer
│   ├── model_cnn.h5                // cnn
│   │── model_lstm.h5               // lstm
│   │── model_lstm_cnn.h5           // lstm + cnn
├── processed data/                 // audiio df and features df
│   ├── new_audio_csv.csv           // audio files path
│   └── df_csv.csv                  // audios more than 1 sec
│   └── all_features.csv            // extracted features
├── app.py                          // main application
├── Dockerfile                      // docker file
├── Notebook.ipynb                  // colab notebook
├── packages.txt                    // system packages
└── requirements.txt                // dependencies
```

Deployed App



Speech Emotion Recognition

Upload Files

Drag and drop file here

Limit 200MB per file • WAV, MP3

Recording_25.mp3 71.6KB

```
{
  "FileName" : "Recording_25.mp3"
  "FileType" : "audio/mpeg"
  "FileSize" : 73335
}
```

Converting mp3 to wav...

Recognizing emotion in your speech...

The uploaded audio has the following emotion:"Angry"

Summary

Started with...

- Gathering wide range of properly labelled speech recordings in different languages and accent to make sure model generalizes well on real world data.
- Selecting best number of emotions to be classified. Selected emotions are Happy, Sad, Angry, Surprise, Disgust, Fear, Neutral.
- Augmenting Data to generate more data. Techniques used are Noise Insertion, Shifting, Stretching and changing Pitch.
- Extract all important audio features that can be learned by model.
- Trying different neural network models like MLP Classifier, 1-d CNN network, LSTM, LSTM+CNN combination & selecting best model(LSTM+CNN).
- Keeping a check on overfitting while training model by using techniques such as Dropout.
- Dockerizing and creating application using Streamlit.
- Deploying application using Azure web-apps services.
- All models performed poorly on Disgust & Fear emotion as compared to other emotions.
- Using multiple datasets of different properties like gender, language, accent, recording environment is essential in getting a more generalized model.
- Data Augmentation techniques proved to be useful in improving model performance

Thank You