# Capstone Project - 5
# Live Class Monitoring system (Face Emotion Recognition)
## DL+MLE


**Team**

**Kanishka Raj, Rakhi Kumari, Raushan Kumar, Sridhar Nagar**

# Content

- Introduction
- Defining Problem Statement
- Data Summary
- Dependency
- Modelling Creation
- Model evaluation
- Real time Face Detection
- Deployment
- Challenges
- Conclusion

# Introduction

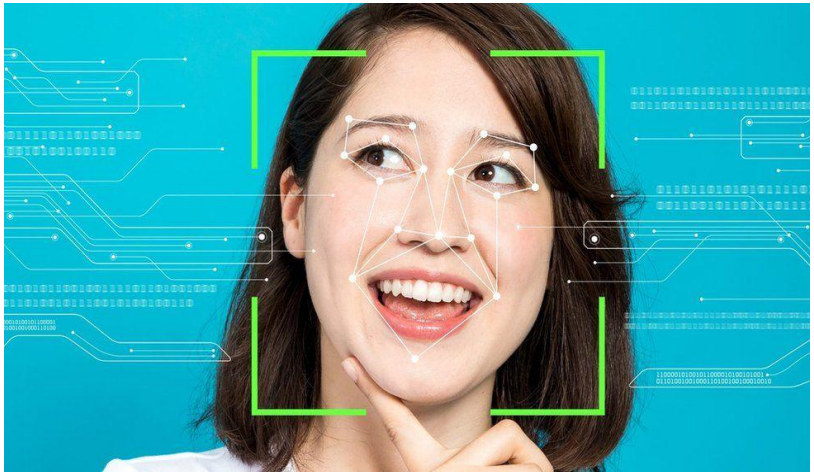
Human emotions and intentions are expressed through f**acial expressions and deriving angle.** Efficient and effective feature is the fundamental component of facial expression system.

Facial expressions convey non-verbal cues, which play an important role in interpersonal relations. **Automatic recognition of facial expressions can be an important component of natural human-machine interfaces.**
An automatic Facial Expression Recognition system needs to solve the following problems: **detection and location of faces in a cluttered scene, facial feature extraction, and facial expression classification.**

# Defining Problem Statement

There are major challenges associated with digital learning when compared with brick and mortar classrooms. One of many challenges is how to **ensure quality learning for students.**

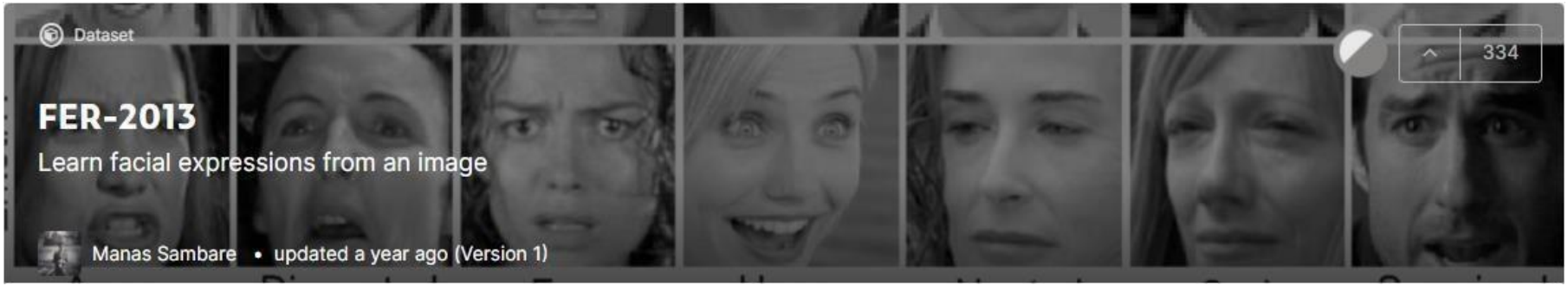
**Digital platforms might overpower** physical classrooms in terms of content quality but when it comes to understanding **whether students are able to grasp the content in a live class scenario** is yet an **open-end challenge**.



# Data Summary

## Data Set link

https://www.kaggle.com/msambare/fer2013




**This dataset contains 35887 grayscale 48x48 pixel face images with seven emotions.**

# Data Summary

We have develop some deep learning model which detect the emotion of the students **during the live class by the use of webcam so the teacher can understand that student are able to understand the topic or not.** we take emotion of the student and deploy that emotion into the model our model is trained by **FER2013** dataset which available on the kaggle.

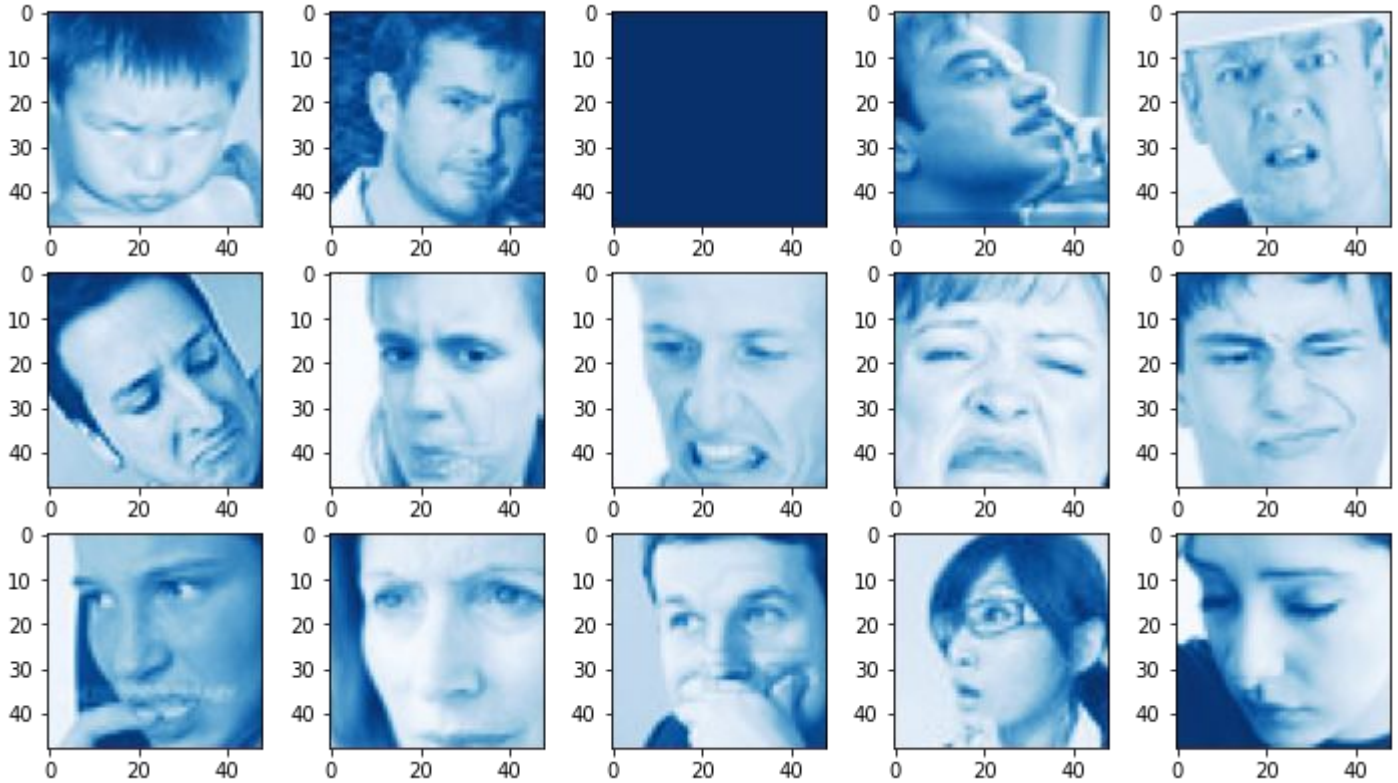
| Label | Emotion | No. of images for Training | No. of images for Testing |
|---|---|---|---|
| 0 | Angry | 3995 | 958 |
| 1 | Disgust | 436 | 111 |
| 2 | Fear | 4097 | 1024 |
| 3 | Happy | 7215 | 1774 |
| 4 | Sad | 4830 | 1247 |
| 5 | Surprised | 3171 | 831 |
| 6 | Neutral | 4965 | 1233 |

# Data Summary

Angry

Disgust

Fear

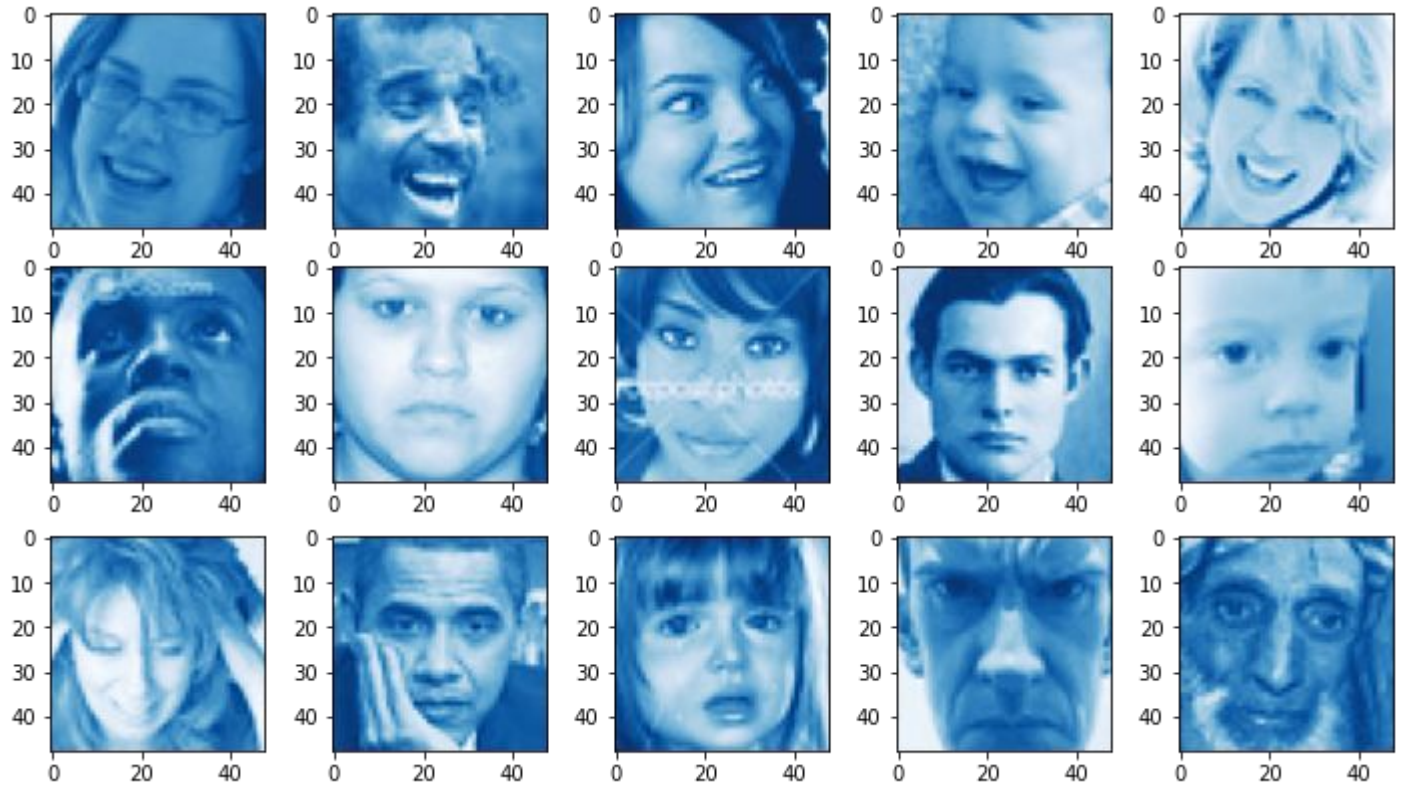


# Data Summary

Happy

Sad

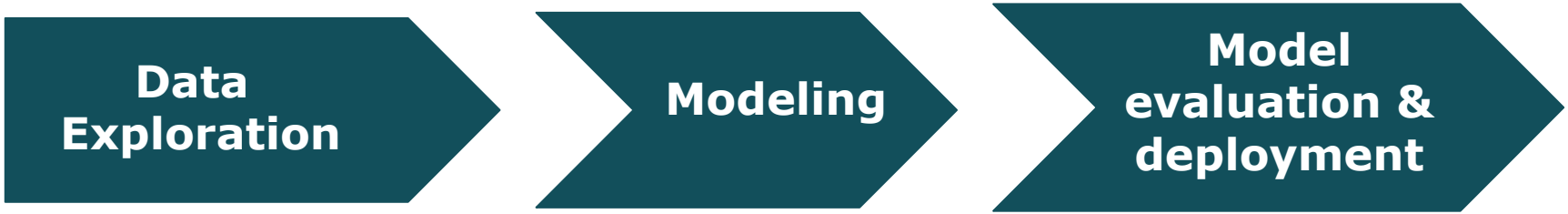
Surprised



# Project Dependency

- Python 3.0
- Tensorflow 2.0
- OpenCv
- GPU must for faster computation
- Google Colab and VS Code for application code
- Streamlit for the application deployment
- Heroku for the application deployment

# Pipeline




## Understanding the data

- Types of emotions
- Images in each category
- Their properties

## Modeling structures

- DeepFace
- Transfer learning
- CNN

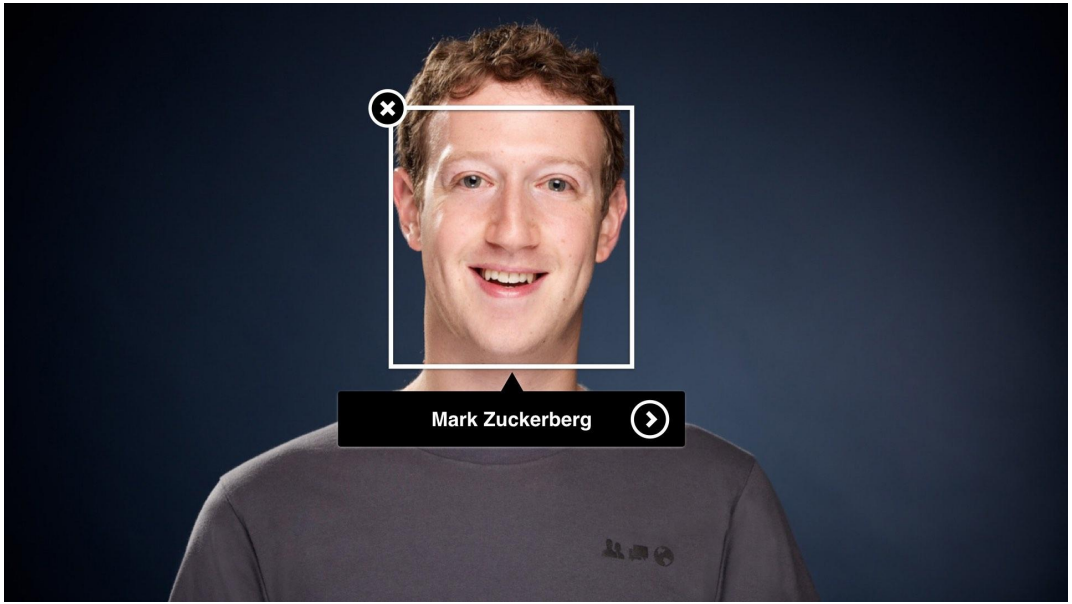
## Graphs and applications

- Loss & accuracy plots
- Confusion matrix (Heatmap)
- Streamlit
- Heroku

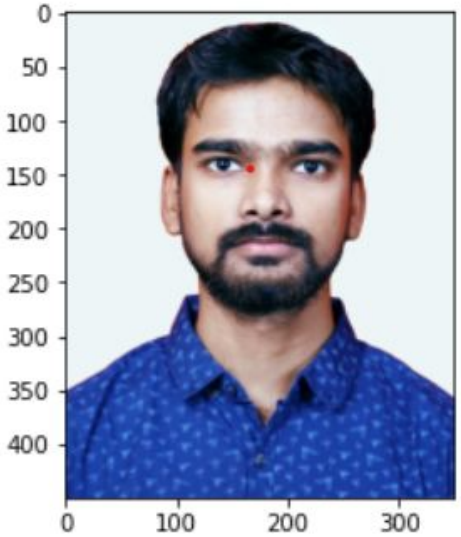
# Model Creation

1) Deep face model

**Deepface is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python.**



# Deep face Prediction




```
# result
prediction1
```

```
{'age': 32,
 'dominant_emotion': 'neutral',
 'dominant_race': 'indian',
 'emotion': {'angry': 0.007850777183193713,
  'disgust': 6.796536947667465e-16,
  'fear': 2.7680073344527045e-06,
  'happy': 0.00030318649351102067,
  'neutral': 99.98117089271545,
  'sad': 0.010663949797162786,
  'surprise': 1.0387559257196699e-05},
 'gender': 'Man',
 'race': {'asian': 0.000716245077291671,
  'black': 0.18748602457094632,
  'indian': 99.73831172078461,
  'latino hispanic': 0.07023896997411012,
  'middle eastern': 0.0023746925461487554,
  'white': 0.0008750781235424799},
 'region': {'h': 212, 'w': 212, 'x': 74, 'y': 57}}
```

```
[ ] # predict dominant emotion of image
    prediction1['dominant_emotion']
```

```
'neutral'
```

```
[ ] # race of image
    prediction1['dominant_race']
```

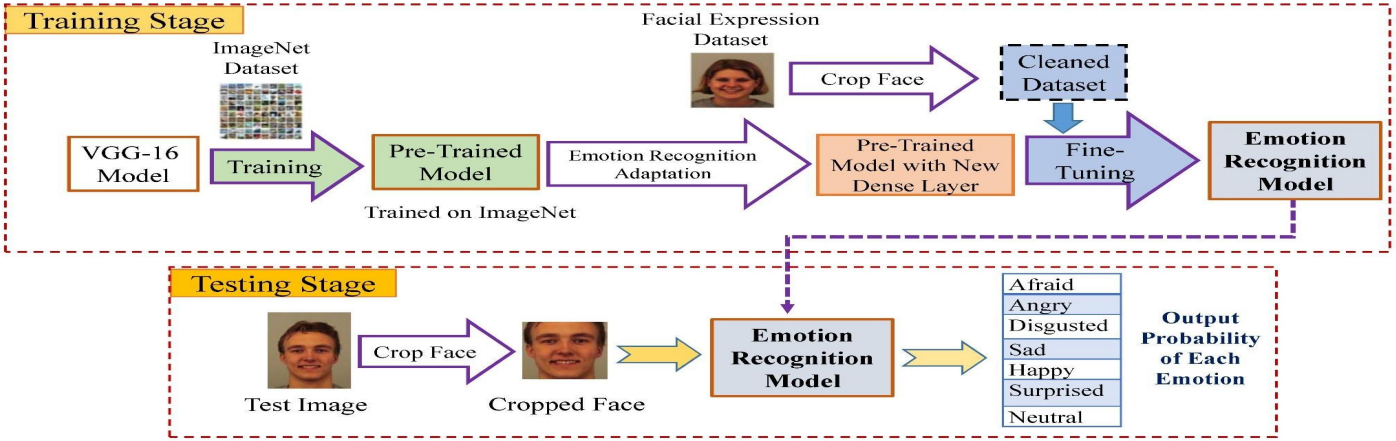
```
'indian'
```

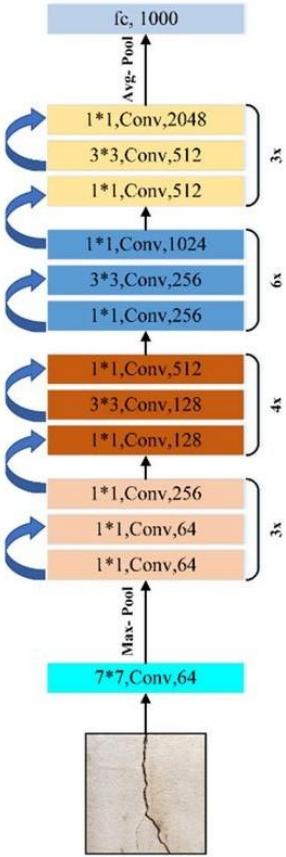
**Deep face model detect correct emotion of the Picture. Model showing an Indian man with Neutral emotion which looks correct as per expression.**

# Model Creation

## 2) Transfer Learning (ResNet50)

- ResNet, short for **Residual Networks** is a classic neural network used as a backbone for many **computer vision tasks**.
- This model was the **winner of ImageNet challenge in 2015**. The fundamental breakthrough with ResNet was it allowed us to train extremely deep neural networks with 150+layers successfully.
- ResNet makes it possible to **train up to hundreds or even thousands of layers and still achieves compelling performance.**



# Modeling Steps


- Pre trained 50 conv layers
- Flatten layer
- FCL - 512 units
- FCL - 256 units
- FCL - 7 units

- Activation Function - ReLu, Softmax
- Epoch - 50
- Optimizer - Adam
- Batch size -32
- Callbacks- EarlyStopping, ReduceLROnPlateau

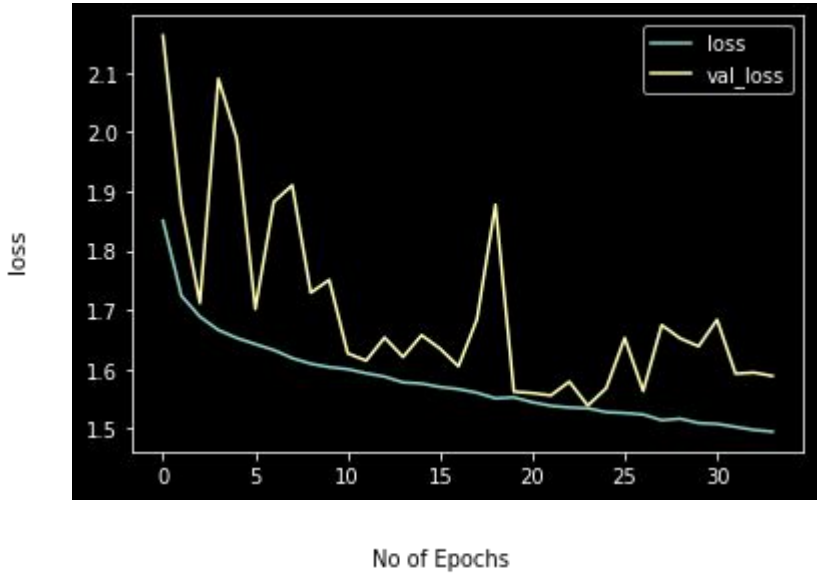
- Loss and accuracy plots
- Heatmap of confusion matrix

**Also we use some common techniques for each layer**

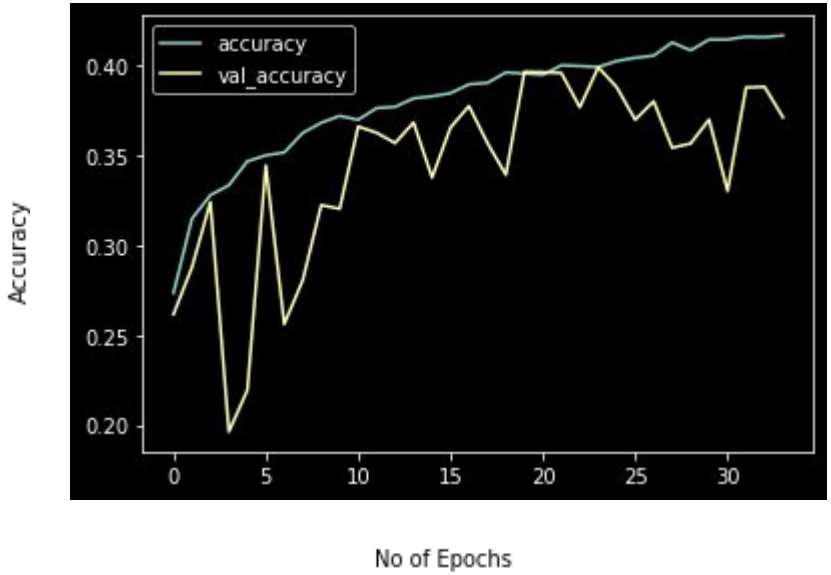
- Batch normalization
- Dropout

# Model Evaluation

## Categorical Cross Entropy



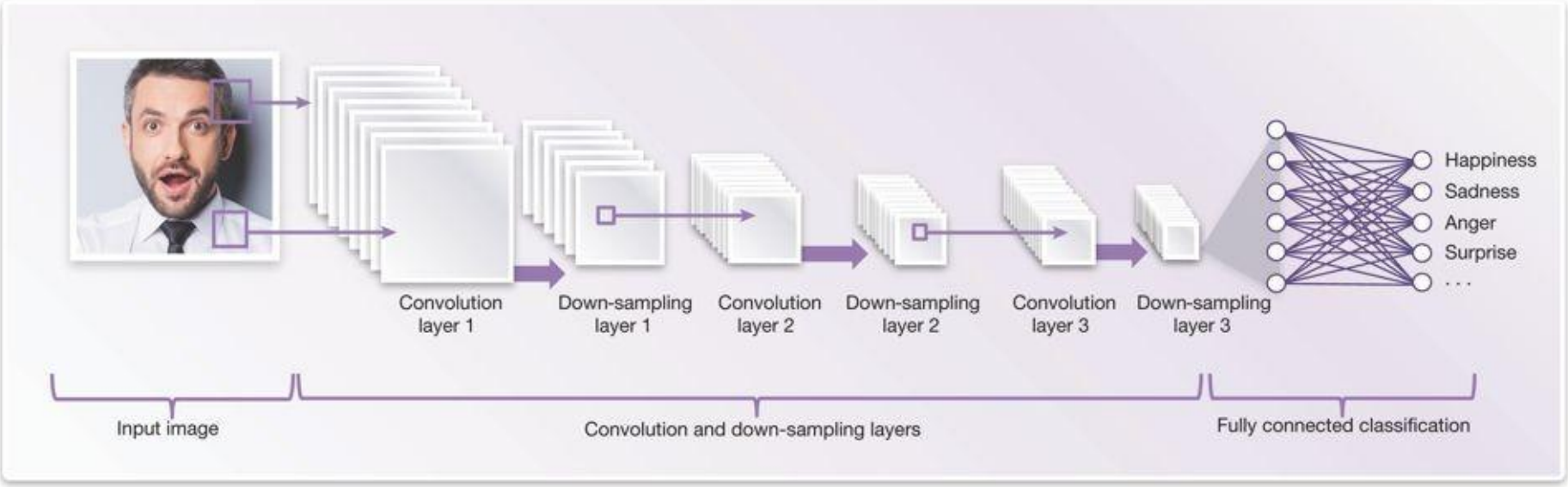

## Accuracy



# Model Creation

## 3) Custom CNN Model

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.



# Modeling Steps


- Layer 1- 3*3,Conv,64
- Layer 2- 3*3,Conv,128
- Layer 3- 3*3,Conv,254
- Layer 4- 3*3,Conv,512
- Flatten layer
- FC - 512 units
- FC - 256 units
- FC - 7 units

- Activation Function - ReLu, Softmax
- Epoch - 50
- Optimizer - Adam
- Batch size -32
- Callbacks- EarlyStopping, ReduceLROnPlateau

- Loss and accuracy plots
- Heatmap of confusion matrix

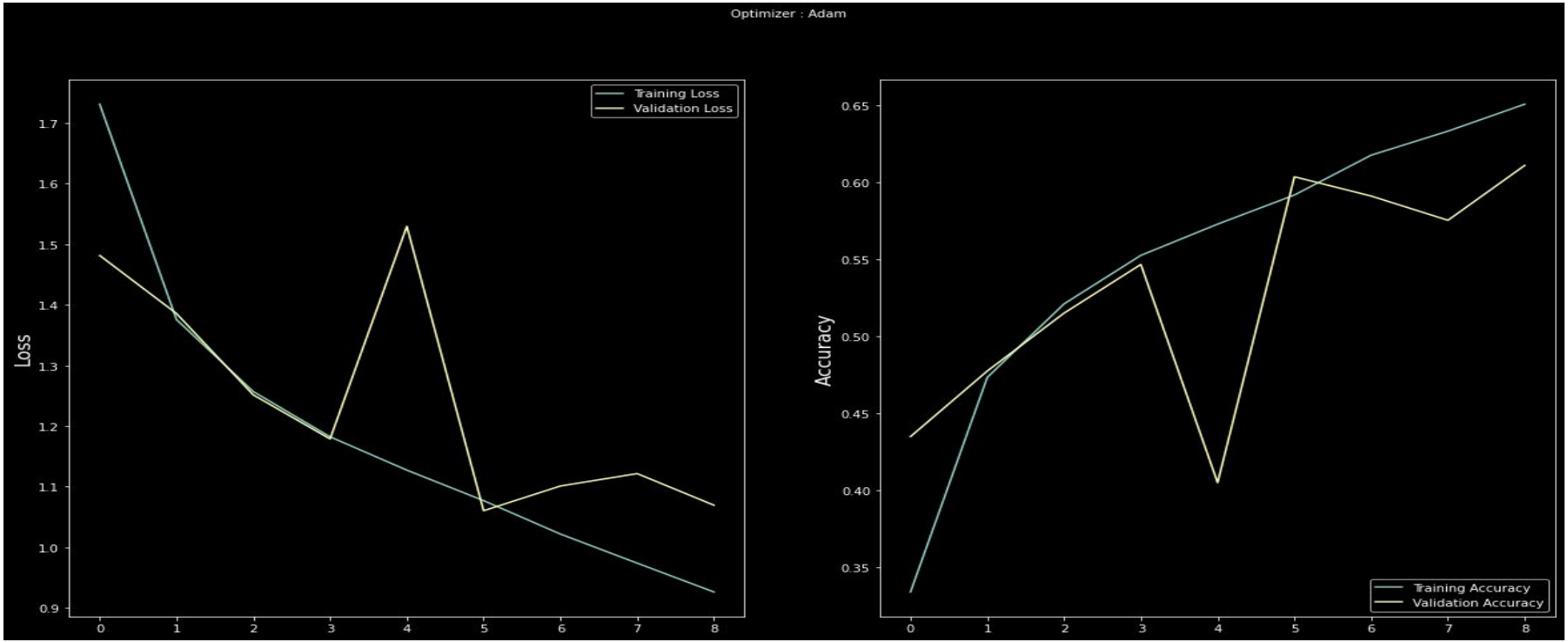
**Also we use some common techniques for each layer**

- Batch normalization
- Dropout

# Model Evaluation

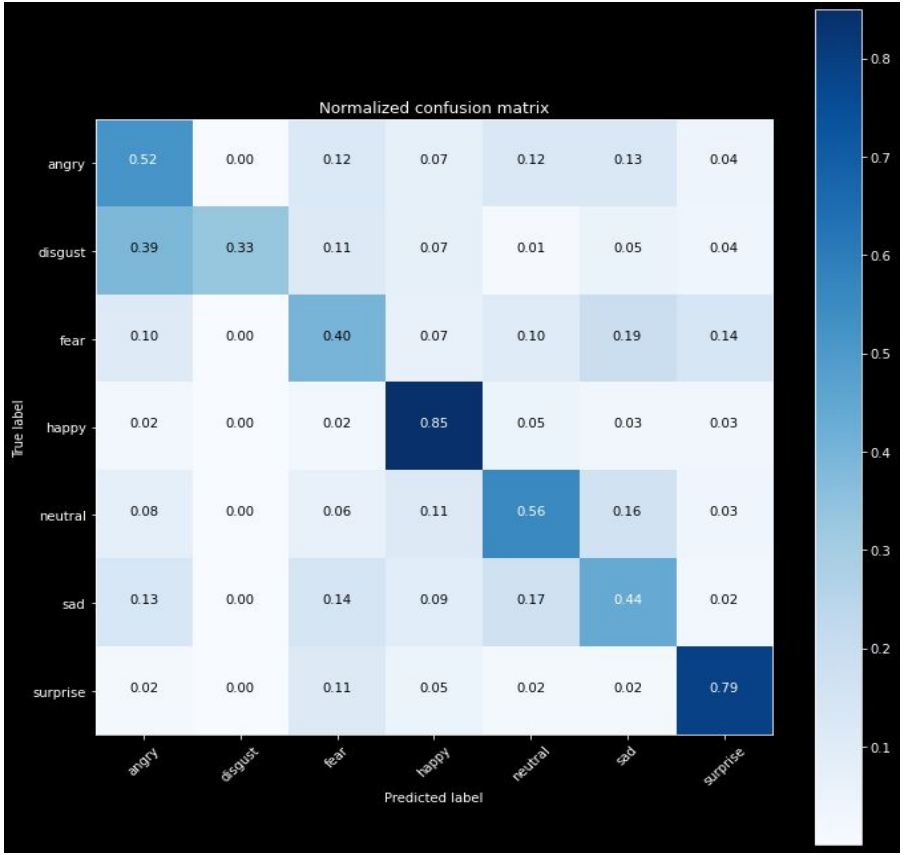
## Categorical Cross Entropy

## Accuracy



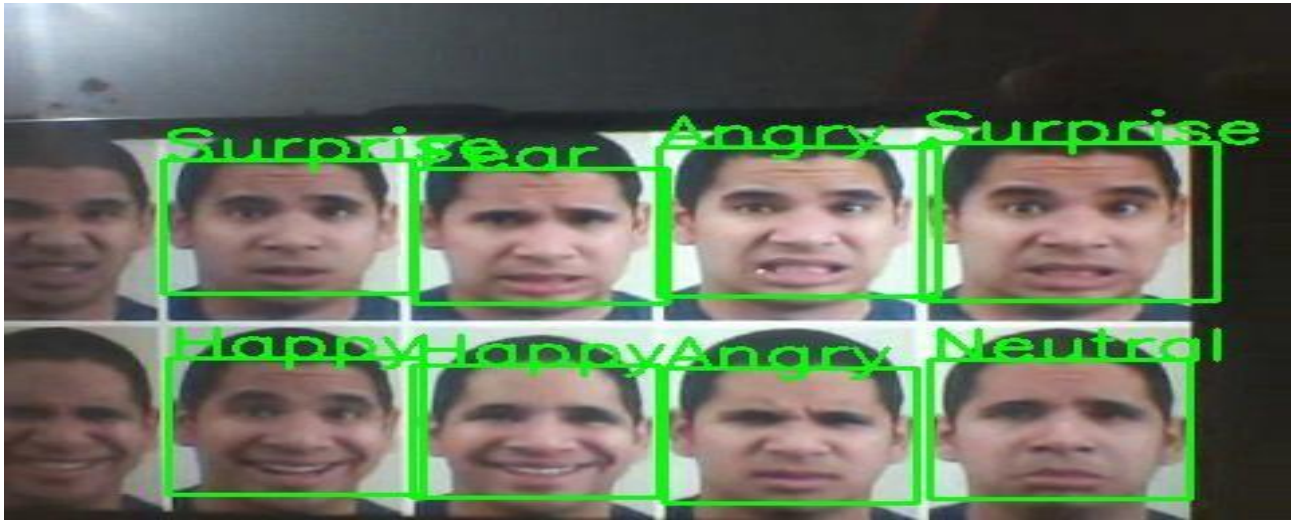
# Model Evaluation

## Confusion matrix (Heatmap)



# Real time local video face emotion detection

- We created some pattern for detecting and prediction the emotion **for single as well for the multiple face** by the **use of OpenCV video capture in local system.**
- For webapp OpenCv can't use so we use **streamlit-webrtc for Front end application**.



# Deployment of Web App in Heroku and Streamlit:

- We deployed the app in Heroku if you saw in the starting section of github repo you see the all the requirement files are there for creating an app on Heroku of name “face-emotion-recog-app ”.

- But due to high slug size the buffering takes time so we have ran our app working on local and it ran properly and app is also fine also we’ve included video on github repo.

***Heroku Link***:
https://face-emotion22.herokuapp.com/
***Streamlit Link***:
https://share.streamlit.io/codewithsridhar/live-class-monitoring-system-face-emotion-recognition-/main/app.py

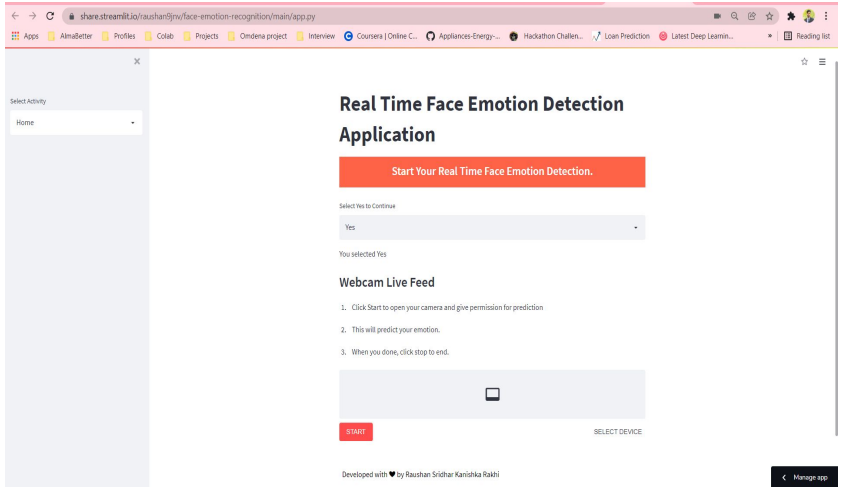
# GitHub Repo


Github Repo link : [Sridhar/Live-Class-Monitoring-System-Face-Emotion-Recognition- (github.com)]

# Deployment

## Creating Web App Using Streamlit

Streamlit is an open-source python framework for building web apps for Machine Learning and Data Science. We can instantly develop web apps and deploy them easily using Streamlit. Streamlit allows you to write an app the same way you write a python code. Streamlit makes it seamless to work on the interactive loop of coding and viewing results in the web app.

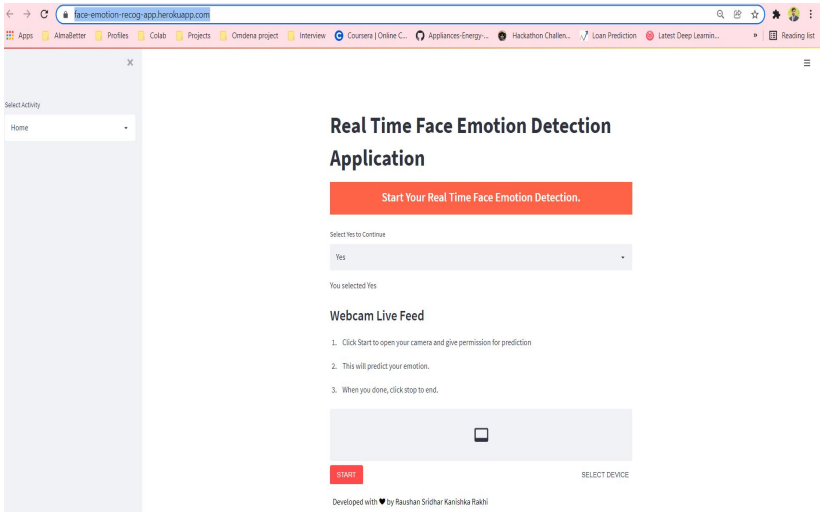



***Streamlit Link:***
https://share.streamlit.io/codewithsridhar/live-class-monitoring-system-face-emotion-recognition-/main/app.py

# Deployment

## Deployment in cloud platform

Heroku is a container-based cloud Platform as a Service (PaaS) supporting several programming languages as Java, Node.js, Scala, Python, PHP, and Go.




***Heroku Link***:
https://face-emotion22.herokuapp.com/

# Challenges

- **Large image dataset** to handle
- **limited GPU** access on Google colab notebook
- Every training epoch takes too much time **so experimenting with different data split which takes more time to give useful result.**
- **Hyperparameter tuning is very sensitive** for these Neural Network model. So we make changes very careful.
- After seeing all the model score we need to find the best model based on different matrices score **which is time consuming** because **models takes lot of time to compute the result.**
- Running **webcam on Google Colab**
- **Deployment**

# Conclusion

- We build the **WebApp using streamlit and deployed in Heroku and Streamlit Sharing.**
- The model which was created by custom CNN model gave **training accuracy of 74% and test accuracy of 60%**.which is **maximum from the rest of all four models we used in this project.**
- we have also included the **video of my WebApp working in Local**. **Local face detection help us to evaluate the performance after every small change in the code cell.**
- **Codes**, which we deployed are **present in Github Repository.**
- It was such an amazing and interesting project. We learnt a lot from this. We did lots of **experiments with respect to data set distribution** for the model and saw the effects on the performance.

# Future Scope

- The future of facial recognition technology is bright.
- Security and surveillances are the major segments which will be deeply influenced.
- It will also be adopted by retailers and banking systems in coming years to keep fraud in debit/credit card purchases and payment especially the ones that are online.
- Technology would fill in the loopholes of largely prevalent inadequate password system.
- In the long run, robots using facial recognition technology may also come to foray.

# Thank You