Capstone Project

Face-Emotion-Recognition

Deep learning and MLE

Submitted by - Avilash Srivastava

Contents

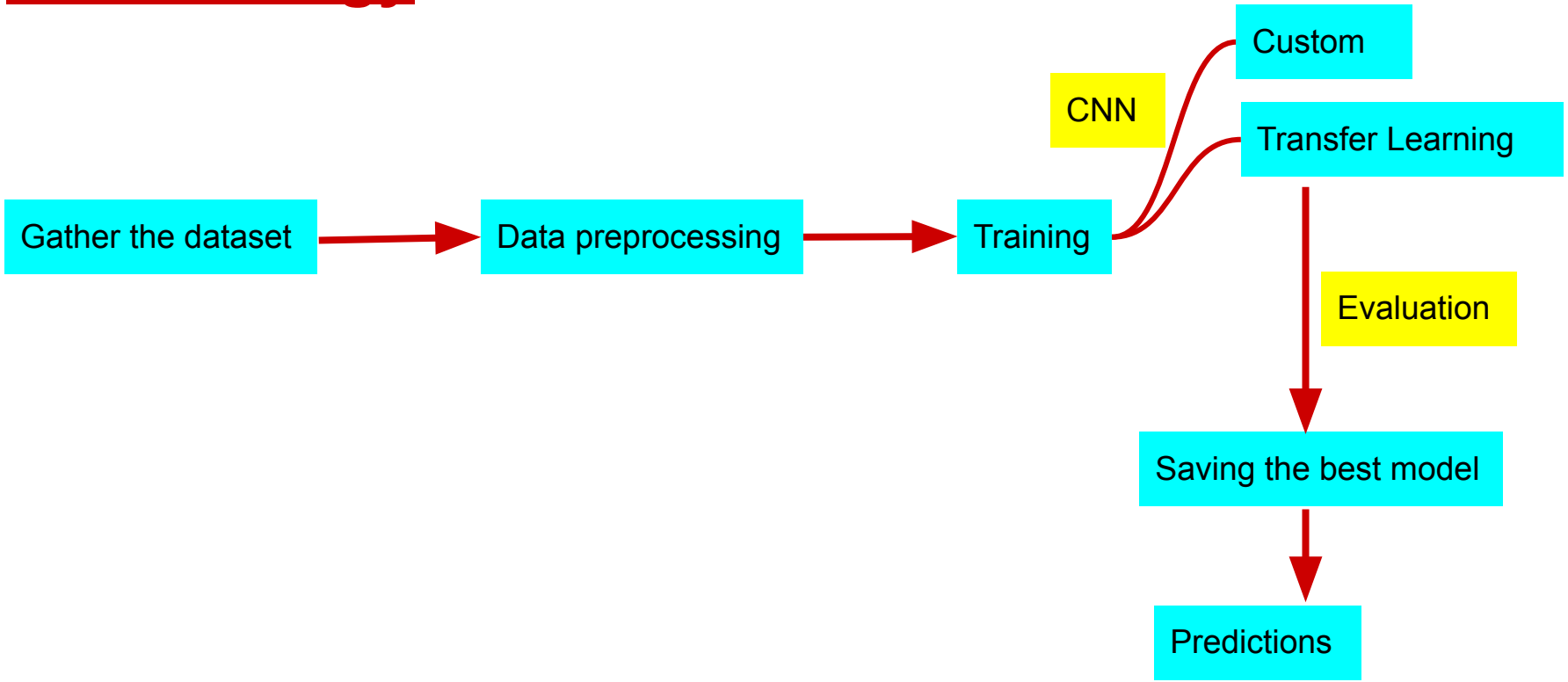
Project Introduction

- The Indian education landscape has been undergoing rapid changes for the past 10 years owing to the advancement of web-based learning services, specifically, eLearning platforms.
- Digital platforms to conduct live classes are proving their worth in terms of quality content and resources.
- We now have a load of data in terms of video, audio, texts.
- Using this data we can build numerous deep learning projects which will help in improving digital learning.

Problem Statement

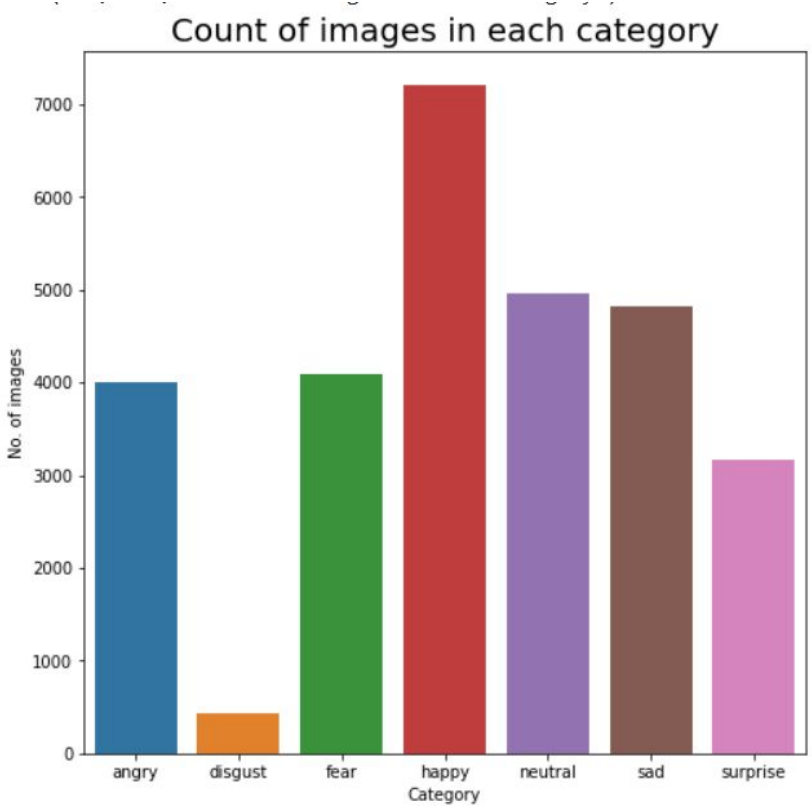
- One of many challenges is how to ensure quality learning for students.
- Difficult to understand whether students are able to grasp the topics
- In a physical classroom during a lecturing teacher can see the faces and assess the emotion of the class and tune their lecture accordingly, whether he is going fast or slow. He can identify students who need special attention.
- Digital classrooms are conducted via video telephony software program (exZoom) where it's not possible for medium scale class (25-50) to see all students and access the mood.
- Because of this drawback, students are not focusing on content due to lack of surveillance.
- Deep learning backed system not only solves the surveillance issue, but it also removes the human bias from the system, and all information is no longer in the teacher's brain rather translated in numbers that can be analysed and tracked.
- We will solve the above-mentioned challenge by applying deep learning algorithms to live video data. The solution to this problem is by recognizing facial emotions.

Methodology



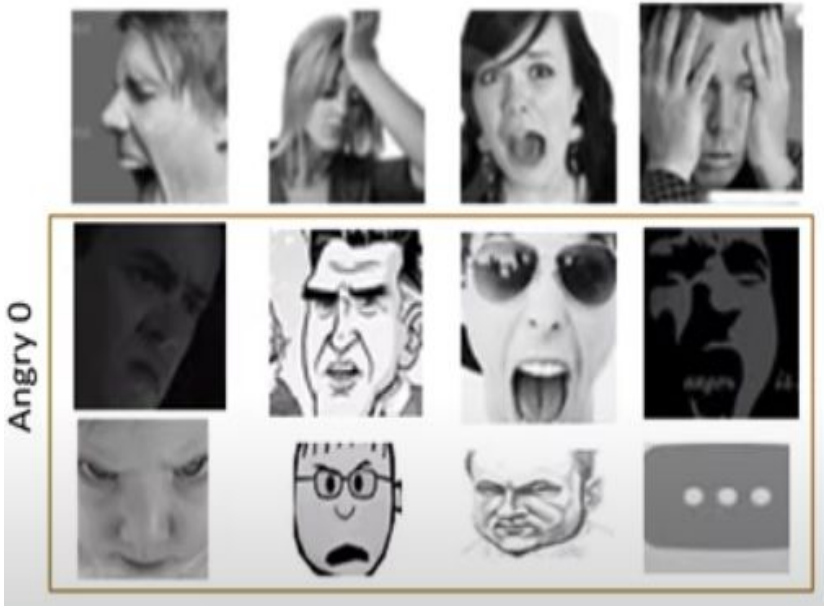
Dataset

- The model has been trained on “FER - 2013” dataset which was published on international conference on machine Learning(ICML).
- Consists of 35887 grayscale, 48x48 sized face images with seven emotions.
- Emotions are angry, disgust, fear, happy, neutral, sad, surprise.
- Disgust has very less examples, so model might not perform good on disgust images.



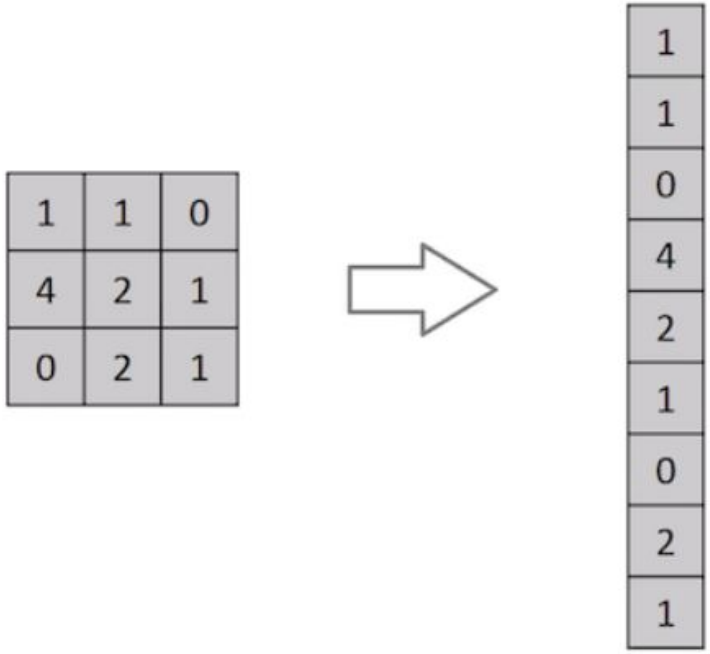
Problems in Dataset

- Imbalance problem
- Intra class variation
- Occlusion
- Contrast
- Sunglasses
- Outliers



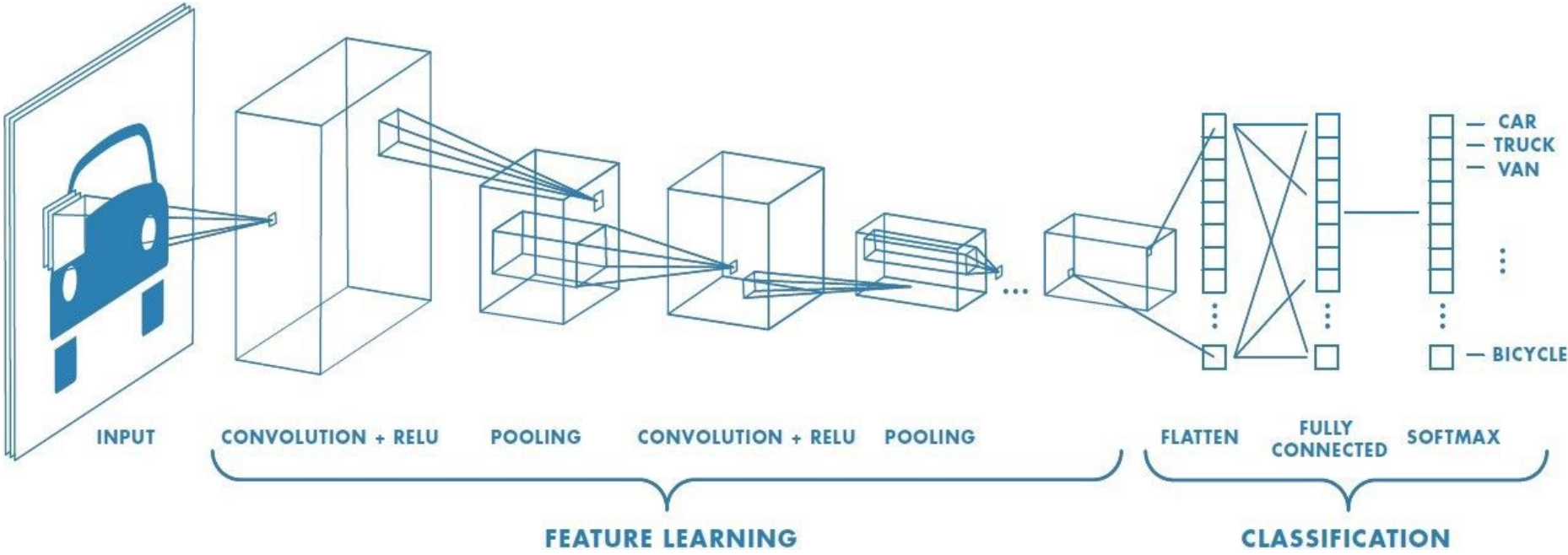
CNN Architecture

- Better option than simple neural networks.
- CNN is better as number of parameters reduces and weights can be reused.
- Made of multiple layers to capture low level features to high level features.
- Convolution layer
- Pooling layer
- Fully connected layer
- Activation layers
- Batch normalization
- Dropout layers
- Transfer learning can be applied from pre built models such as VGG, Alexnet, Resnet, Mobilenet etc.



Flattening of a 3×3 image matrix into a 9×1 vector

CNN Architecture

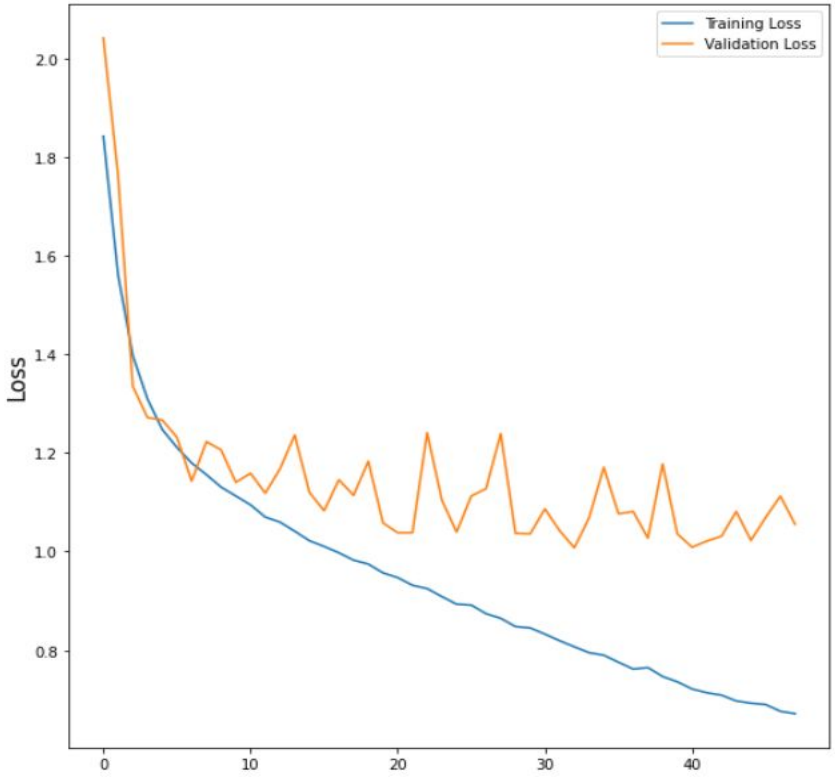


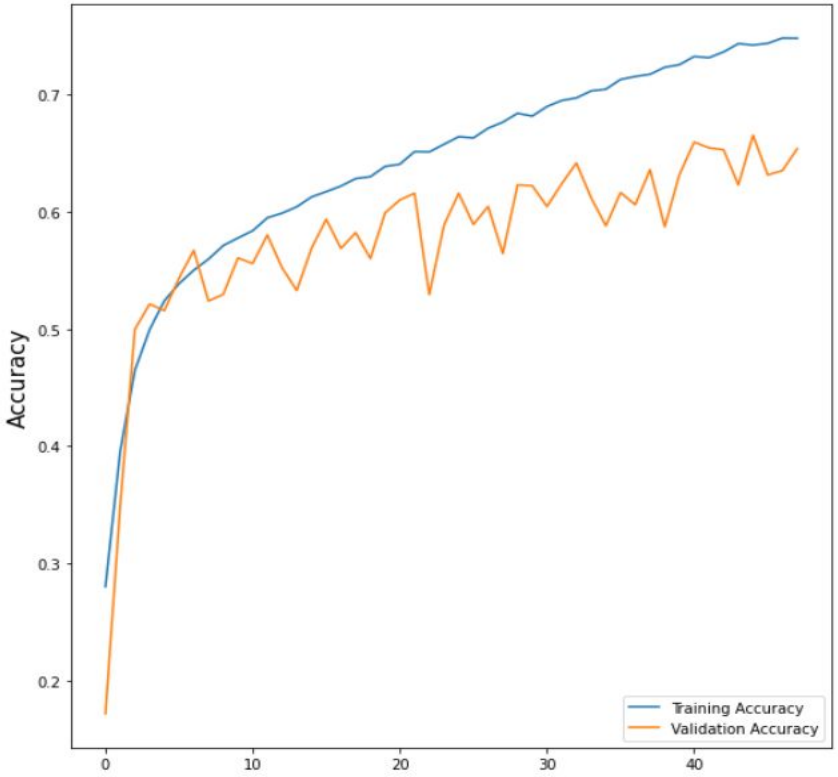
Training the model

- Training set accuracy : 74%
- Validation set accuracy : 67% at 45 epochs

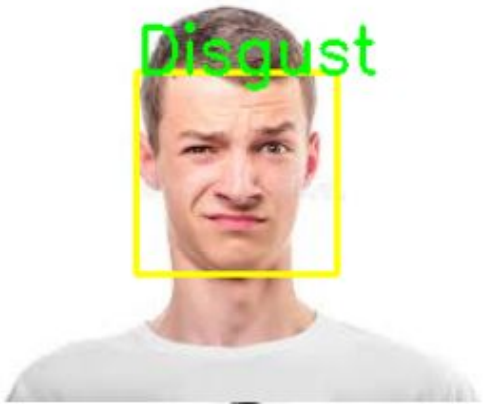
```
Epoch 00042: val_accuracy did not improve from 0.65910
Epoch 43/48
225/225 [==============================] - 598s 3s/step - loss: 0.7086 - accuracy: 0.7359 - val_loss: 1.0312 - val_accuracy: 0.6525

Epoch 00043: val_accuracy did not improve from 0.65910
Epoch 44/48
225/225 [==============================] - 599s 3s/step - loss: 0.6969 - accuracy: 0.7430 - val_loss: 1.0813 - val_accuracy: 0.6226

Epoch 00044: val_accuracy did not improve from 0.65910
Epoch 45/48
225/225 [==============================] - 600s 3s/step - loss: 0.6924 - accuracy: 0.7418 - val_loss: 1.0227 - val_accuracy: 0.6649

Epoch 00045: val_accuracy improved from 0.65910 to 0.66495, saving model to model.h5
Epoch 46/48
225/225 [==============================] - 598s 3s/step - loss: 0.6897 - accuracy: 0.7433 - val_loss: 1.0693 - val_accuracy: 0.6311

Epoch 00046: val_accuracy did not improve from 0.66495
Epoch 47/48
225/225 [==============================] - 599s 3s/step - loss: 0.6760 - accuracy: 0.7477 - val_loss: 1.1126 - val_accuracy: 0.6349

Epoch 00047: val_accuracy did not improve from 0.66495
Epoch 48/48
225/225 [==============================] - 598s 3s/step - loss: 0.6711 - accuracy: 0.7476 - val_loss: 1.0554 - val_accuracy: 0.6534

Epoch 00048: val_accuracy did not improve from 0.66495
```

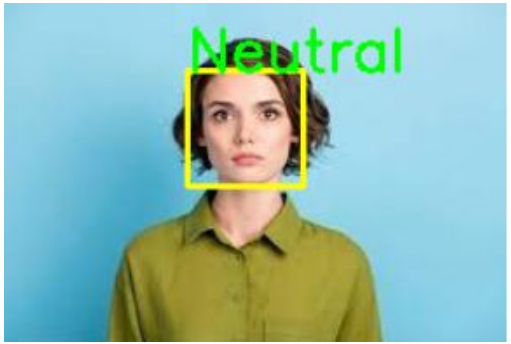
Evaluation


Training Loss
Validation Loss
Loss
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0
10
20
30
40


Accuracy
0.7
0.6
0.5
0.4
0.3
0.2
Training Accuracy
Validation Accuracy
0
10
20
30
40

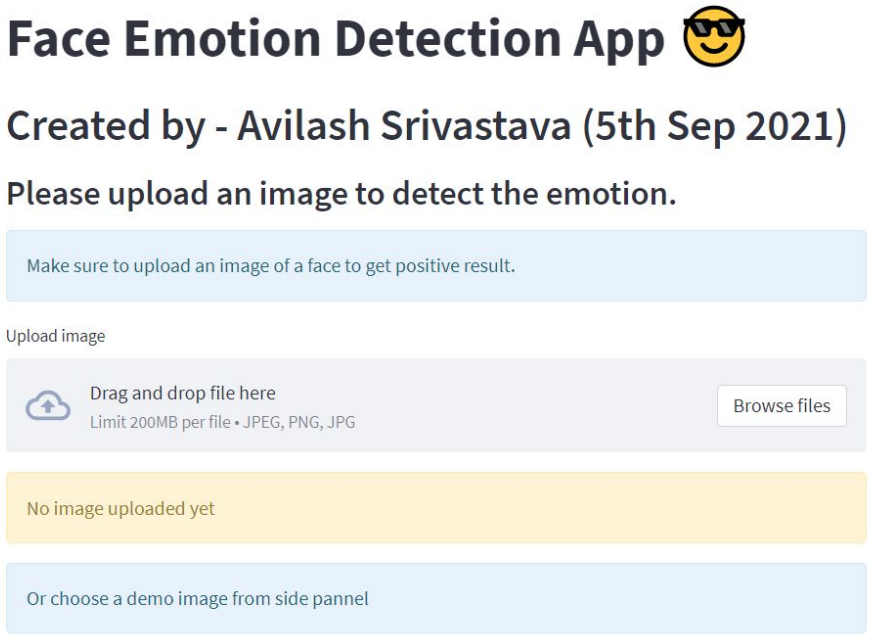
Predicted Examples





Streamlit App

- Built a streamlit python script.
- Made a docker image
- Deployed on azure cloud and streamlit cloud.
- Provides end to end solution.



Live Demo

Challenges

- Tried Batch Gradient Descent but failed due to low compute resources.
- Overfitting occurred on training set more often. Applied dropout
- Too many errors while experimenting real time. Learnt from every errors and improved.
- Gathering codes for real time demo on streamlit app was haunting as local code did not run on cloud servers.
- Streamlit app is still slow due to free account and less resources.

Conclusion

- Came long way, starting from integers, floats and booleans in python to being able to understand the beauty of artificial intelligence and building a complete project.
- Got to learn how everything works from development to production and deployment.
- Model gave an accuracy of 74% for training set and 67% for test set.
- Can be further improved if we have more data, more compute resources, trying different combinations of cnn layers, transfer learning.
- App can run smoothly when compute resources are better.