FACE RECOGNITION

Team Members:-

Aakash Murugan

Aishwary Jagetia

Animesh Nema

Onkar Trivedi

INTRODUCTION:

The goal of this assignment is to implement a neural network using deep learning for the purpose of face recognition.

Our Dataset comprises of the above mentioned team member's faces (and a classmate, not part of this group, so that we have a versatile dataset).

The neural network was trained on a specific percentage of this data and was tested on the rest. The code and the results are described later in this report:-

CREATING THE DATASET.

1. Utilize the videos folder to generate a folder "dataset" with the names of project group members, each denoting the location of parsed images for each member.

2. **face_cascade=cv2.CascadeClassifier("haarcascade_frontalface_default.xml")**
   We used the above code using OpenCV. This helped us focus and detect the "face-like" regions of the parsed images and center the image around these features to provide better training to the model.

3. OpenCV was used to parse the video into numerous frames (or images).

4. This image is then turned into a Grey image and subjected to Cascade Classification, which detects objects of different sizes in the input image. The detected objects are returned as a list of rectangle, by the command:
   faces = face_cascade.detectMultiScale(gray, 1.3, 5).
   We resized the images to 50x50 pixels.

5. This image is then stored in the dataset, in an appropriate folder named after the name of the group member.

6. The pathnames are then “jumbled” to avoid any un-needed bias by the neural network at training time, avoiding any potential inaccuracy.

7. The final CSV file generated, gives us the pathnames for the generated dataset as input and the corresponding label of the image for training the network.

The Dataset was then loaded and the model was trained on 67% of the data and tested on the rest of the 33%.

The results were as follows:-

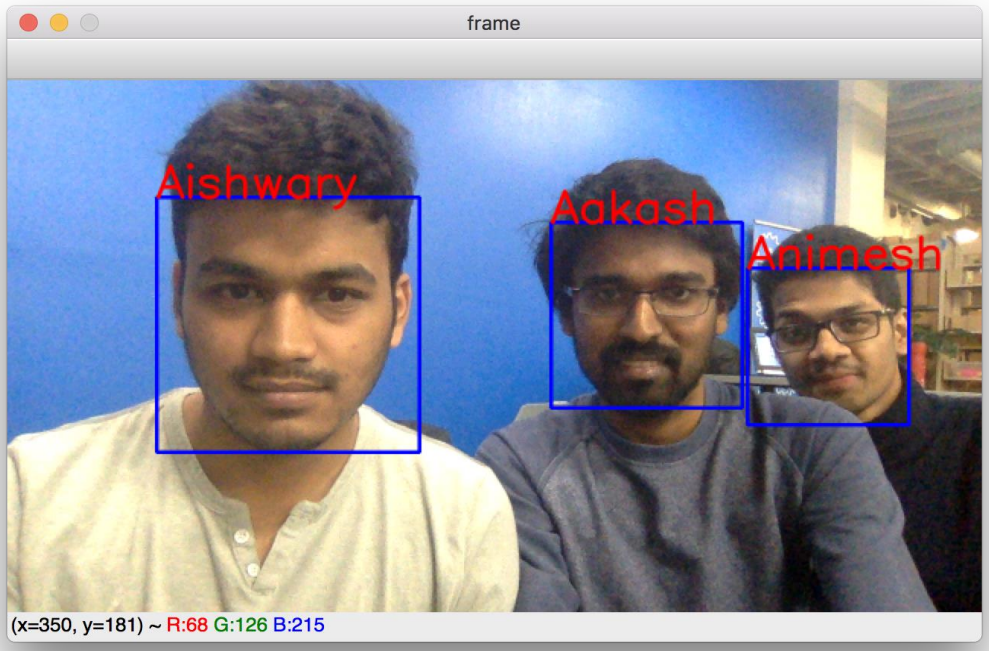
```
Train on 16750 samples, validate on 8250 samples
Epoch 1/10
55s - loss: 0.4573 - acc: 0.8141 - val_loss: 0.0144 - val_acc: 0.9964
Epoch 2/10
51s - loss: 0.0119 - acc: 0.9962 - val_loss: 0.0086 - val_acc: 0.9978
Epoch 3/10
51s - loss: 0.1323 - acc: 0.9610 - val_loss: 0.0244 - val_acc: 0.9915
Epoch 4/10
51s - loss: 0.0068 - acc: 0.9979 - val_loss: 0.0014 - val_acc: 0.9998
Epoch 5/10
51s - loss: 0.0040 - acc: 0.9986 - val_loss: 0.0017 - val_acc: 0.9998
Epoch 6/10
51s - loss: 0.0014 - acc: 0.9996 - val_loss: 0.0017 - val_acc: 0.9995
Epoch 7/10
51s - loss: 0.0011 - acc: 0.9997 - val_loss: 0.0041 - val_acc: 0.9990
Epoch 8/10
51s - loss: 3.2289e-04 - acc: 0.9999 - val_loss: 0.0054 - val_acc: 0.9984
Epoch 9/10
51s - loss: 1.3914e-04 - acc: 0.9999 - val_loss: 0.0016 - val_acc: 0.9998
Epoch 10/10
51s - loss: 9.7641e-06 - acc: 1.0000 - val_loss: 0.0018 - val_acc: 0.9998
Error: 0.02%
```

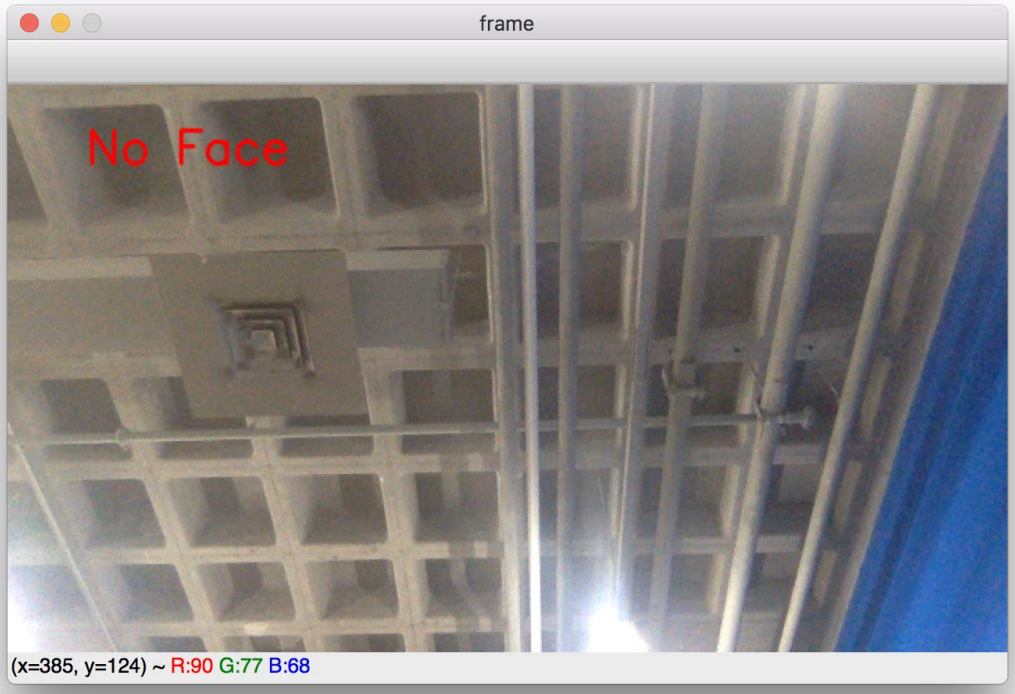
THE TEST DATA ACCURACY WAS **99.98 %**

A STEP FURTHER…

Using OpenCV, VideoCapture command we enabled our pretrained model to detect faces using live video using the camera of the laptop.

The results can be seen below.


frame
Aishwary
Aakash
Animesh
(x=350, y=181) ~ R:68 G:126 B:215


frame
No Face
(x=385, y=124) ~ R:90 G:77 B:68

HIERARCHY OF THE HOMEWORK