UNIVERSITY OF MASSACHUSETTS DARTMOUTH

DEPARTMENT OF COMPUER & INFORMATION SCIENCE

"FACIAL EMOTION RECOGNITION"

CIS 600: MASTER'S PROJECT

FINAL PROJECT REPORT

PROJECT ADVISOR

DR. PAUL BERGSTEIN

PRITKUMAR PATEL

01822653

Fall 2020

Table of Contents

Abstract **4**

Chapter 1: Introduction **5**

Chapter 2: Literature Review / Background Research **7**

Face-API.js 7

Available Models 7

Face Detection Models 7

68 Point Face Landmark Detection Models 7

Face Recognition Model 8

Face Expression Recognition Model 8

Depthwise Separable Convolutions 8

Chapter 3: Design / Models Used **9**

Model Design 9

Model Process Flow 10

Application Design 10

Chapter 4: Implementation and Results **11**

Implementation 11

Results 14

Conclusion **15**

Future Work **15**

References **16**

Table of Figures

Figure 1: Depth Wise Convolution

Figure 2: Model Summary

Figure 3: Emotion Data

Figure 4: Login Page

Figure 5: Dashboard

Figure 6: Your Emotion Data

Figure 7: Neutral

Figure 8: Happy

Figure 9: Surprise

Figure 10: Sad

Figure 11: Angry

Figure 12: Fear

Figure 13: Disgust

Abstract

Facial recognition [1] is the process of finding human emotions from face expression. The human brain automatically detects emotions, and software has been developed to detect instances. This technology is always accurate, and it will eventually be able to learn emotions as our brains do. Understanding contextual emotion has widespread consequences for society and business.

The aim of the project is to create a real time web base facial emotion recognition application which recognize most basic human expressions: Disgust, Fear, Anger, Happy, Neutral, Sad, Surprise. This project can be used to predict the expressions of both real time videos and still images. Although, we will have to provide the image to model in both cases. In case of real time video, the images will be taken at any instance and it will be provided to the model to predict the facial expression. The application will automatically detect face using tiny face detector model then it crops and resize the image to specific size and give it to emotion model for prediction of expression, which will generate seven probability values corresponds to seven basic expressions, value with highest probability value corresponding to expression will be predicted expression for that image. As it is a web base application there are some extra modules added to this project which allows user to login into their account and user can store there emotion data to the database by hitting the send button provided on the interface.

Chapter 1: Introduction

Facial recognition [1] is the process of finding human emotions from face expression. The human brain automatically detects emotions, and software has been developed to detect instances. This technology is always accurate, and it will eventually be able to learn emotions as our brains do. Understanding contextual emotion has widespread consequences for society and business. In the public sphere, governmental organizations could make good use of the ability to detect emotions like guilt, fear, and uncertainty. Companies have also been taking advantage of emotion recognition to drive business outcomes. Companies such as Disney [2] have plans of using facial recognition to judge the emotion response of audience. Also, apple release the feature on iPhone called Animoji [3], where user can get computer simulated emoji to mimic user's expression.

Facial Emotion Recognition as the same suggest the aim of the project is to create a real time web base application that recognize most basic human expressions: Disgust, Fear, Happy, Neutral, Surprise, Sad.

This application can be used to predict the expression of real time videos, by providing image to the already trained model. In case of real time video prediction of expression images will be provide to the model to predict the expression at the same time. The application will automatically detect any number of faces using tiny face detector model then it will crop and resize the image to specific size and give it to emotion model for prediction of expression which will generate seven probability values corresponding to seven basic expression. Further, value which have the highest probability value corresponding to expression will be predicted expression for that image. This web base application mainly relies on face-api.js which is a JavaScript API to detect face and recognize face on web browser, which is implemented on top of tensorflow.js core API [4].

The face-api.js have four available models:

- Face Detection Model
- 68 Point Face Landmark Detection Model
- Face Recognition Model
- Face Expression Recognition Model

The models used for this project are Face Expression Recognition Model which have achieved the accuracy of 99.38% and Face Detection Model, both models are trained on WIDERFACE dataset [5].

As it is a web base application there are some extra modules added to this project which allows user to login into their account and user can store there emotion data to the database by hitting the send button provided on the interface.

Chapter 2: Literature Review / Background Research

Face-API.js

The face-api.js [6] is a JavaScript module which implements CNN [7][8] to solve for detecting face, recognition of face and landmark detection, optimized for web and mobile devices. This module is built on top on top of tensorflow.js core. For detecting face, face-api.js implements the model Tiny Face Detector, SSD MonilenetV1, and experimental MTCNN.

Available Models

Face Detection Models

SSD Mobilenet V1

Single Shot Multibox Detector (SSD) MobileNet V1 is model based on MobileNet V1 with goal of obtaining high accuracy in detecting face bounding boxes [6]. This model is basically used to compute location of each face in given image and will return bounding boxes with its probability for each face detected.

Tiny Face Detector

Tiny Face Detector [6] is a model for real- time face detection, compare to SSD model it is faster, smaller and consumes less resources. This model is extremely mobile and web friendly. The size of this quantized model is only 190 KB. This model is trained on custom dataset of more than 14,000 images labeled with bounding boxes. Thus, it will predict the bounding boxes, which entirely covers facial feature points.

MTCNN

This model is experimental model known as Multi-Task Cascade Convolution Neural Networks) [9][10], it also represents an alternative face detector to SSD MobileNet V1 and Tiny Face Detector model. By tuning the input parameter, MTCNN is able to detect wide range of face bounding boxes. It is a 3-stage cascade CNN, which consequently returns 5 face landmark points alongside with bounding boxes and probability for each face. This model is only size of 2 MB.

68 Point Face Landmark Detection Models

This package is used to implement very lightweight and fast, also accurate 68-point face landmark detector. The model (face_landmark_68_model) has a size of 350kb and

tiny model ((face_landmark_68_model) is only 80kb in size. Both of this model are based on ideas of depth wise separable convolution as well as densely connected blocks. These models are trained on more than 35,000 face images labeled with 68 face landmarks.

Face Recognition Model

For recognition of face, an architecture like ResNet-34 is implemented to compute a face descriptor from given face image, used to describe characteristics of a person face. This model can be used for face recognition of any person. The determination of similarity of two arbitrary faces is done by comparing face descriptors, for example by computing Euclidean distance or other classifiers. The size of the model is around 6.2 MB (face_recognition_model) [6].

Face Expression Recognition Model

This model is fast light weight and provides good accuracy [6]. It has a size of 310kb based on depth wise separable convolutions and densely connected blocks, trained on different images from publicly available datasets and images scrapped form web.

Depthwise Separable Convolutions

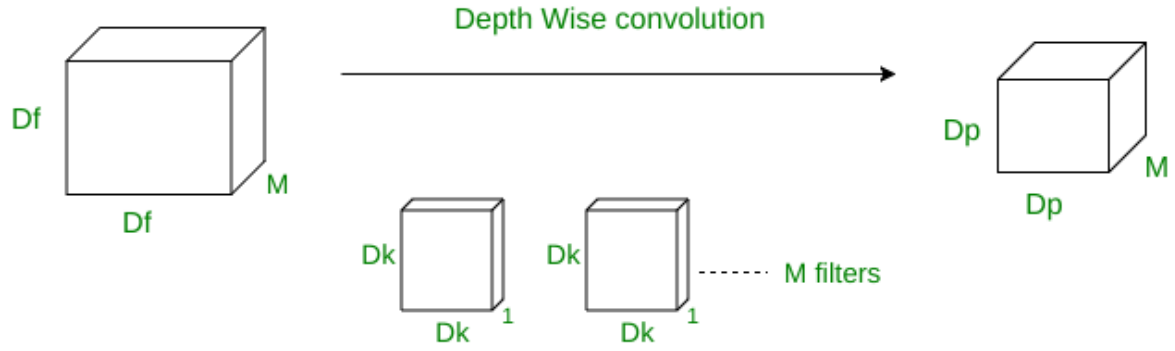

Figure 1: Depth Wise Convolution

Depthwise separable convolutional networks [11] are widely used because they have a smaller number of parameters to adjust compared to standard CNN's and they are computationally cheaper because of fewer computations, making them suitable for web base and mobile version applications.

In this convolution model, the convolution is applied to single channel at a given time compared to standard CNN's in which it is done for all M channels. In turn here filter kernels will be of size **Dk x Dk x 1.** There are M channels in the input data, then M such filters will be required. Output will be as of size **Dp x Dp x M.**

Chapter 3: Design / Models Used

Model Design

Mainly used MobileNetV1 TensorFlow application and Tiny Face Detector models. Where, MobileNetV1 [12] is a tensorflow application suitable for mobile and web base applications, platform where there is lack of computational power. Difference between traditional CNN's and MobileNetV1 is that instead of 3 x 3 convolutional layer, MobileNetV1 splits the convolutional into 3 x 3 depthwise convolutional and 1 x 1 pointwise convolutional [11]. MobileNetV1 are small, low powered, and low latency models, which are parameterized to meet resource constraints of different uses. MobileNetV1 are used for detection, classification, and segmentation, which is like other large-scale models as of Inspection [12]. MobileNetV1 have two parameters which can be tuned to fit accuracy/resource trade off. Two parameters are width multiplier which allow to thin the network and resolution multiplier which change input dimension of image.

Tiny Face Detector [4] is a model for real- time face detection, compare to SSD model it is faster, smaller and consumes less resources. This model is extremely mobile and web friendly. The size of this quantized model is only 190 KB. This model is trained on custom dataset of more than 14,000 images labeled with bounding boxes. Thus, it will predict the bounding boxes, which entirely covers facial feature points.

In Emotion Detection model mobileNetV1 is loaded with pretrained weights of "ImageNet" with top layer not included from application. I added average pooling with dense layers for the model prediction. Below is the model summary of our model:

```
model.summary()----------------------------------
_________________________________________________________________
Layer (type)                 Output shape              Param #
=================================================================
image_input (InputLayer)     [null,100,100,3]          0
_________________________________________________________________
mobilenet_1.00_224 (Model)   multiple                  3228864
_________________________________________________________________
global_average_pooling2d_1 ( [null,1024]               0
_________________________________________________________________
dense_1 (Dense)              [null,1024]               1049600
_________________________________________________________________
predictions_mobilenet (Dense [null,7]                  7175
=================================================================
Total params: 4285639
Trainable params: 4263751
Non-trainable params: 21888
_________________________________________________________________
undefined
send domMutated http://localhost/face_app/dashboard
```

Figure 2: Model Summary

Model Process Flow

1. Gather the data (images which are extracted from video)

I am using the live feed camera to gather the images

2. Preprocess

Tiny face detector is used to preprocess image. It takes the image as input, detect the faces and facial points in the image and draw box around the face.

3. Pass the preprocess faces to Emotion Recognition Model

For each detected face, it is passed to emption recognition model. And predicted result on that face is appended to box around the face.

Application Design

This web-based application consists of login, registration page which lets user create account and login. Once, the user is logged in successfully the main page will initialize the camera, which will predict the emotion of active user in a real time. Right above the camera frame send button is provided, using which user will be able to store their data on database. User also have privilege to see the past emotion records.

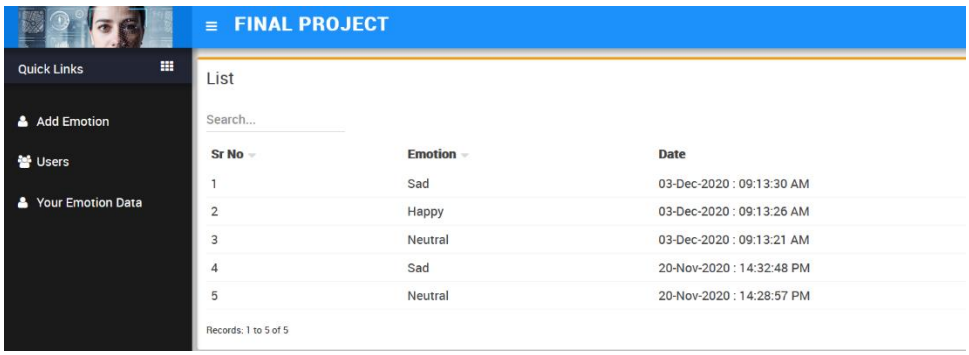

Figure 3: Emotion Data

Also, SQL database is used in this project to perform operations like login to the application and storing user data to the table. There is total 13 tables in the database, among them 2 tables user data and logins are the main tables used most frequently by the application.

Chapter 4: Implementation and Results

Implementation

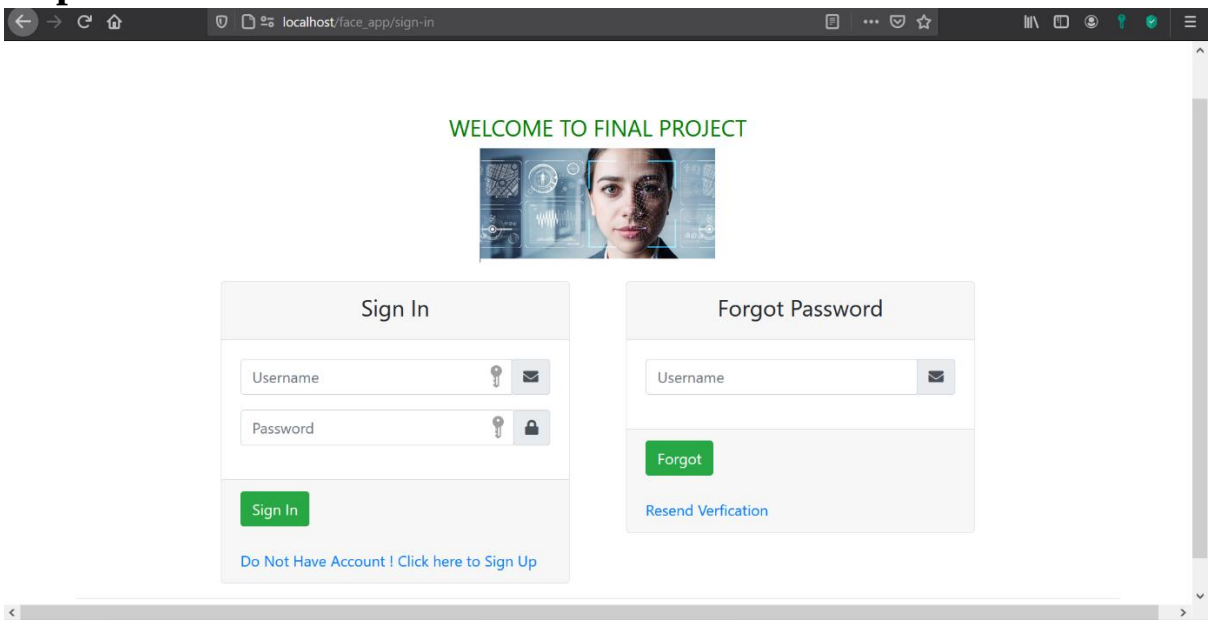

Figure 4: Login Page

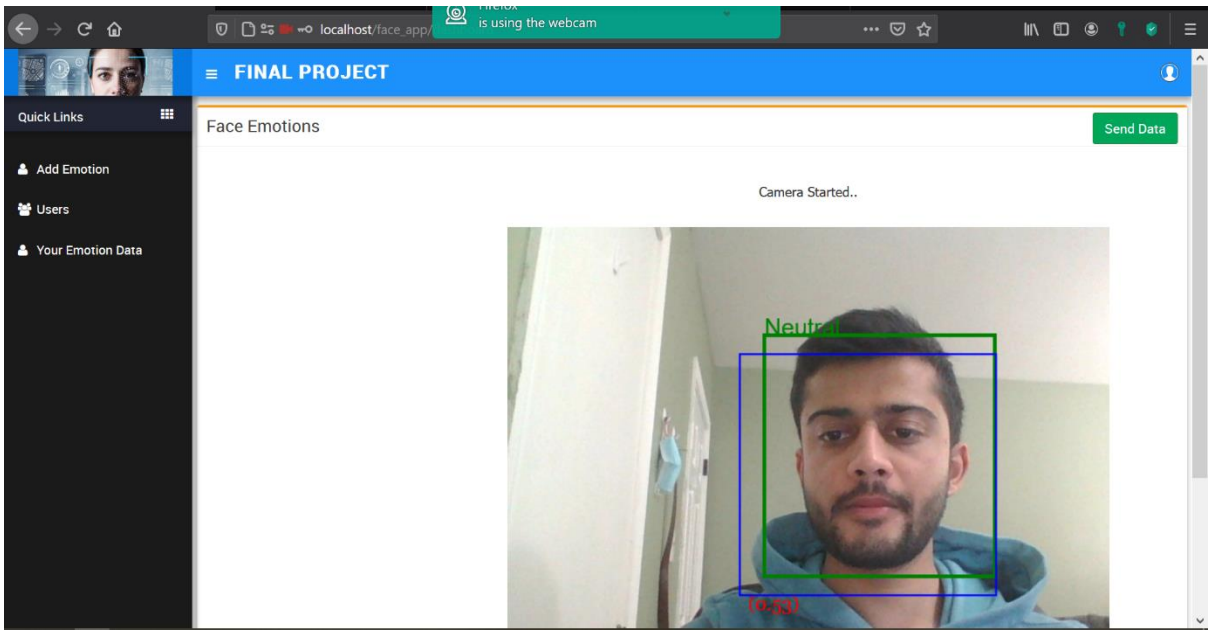

Figure 5: Dashboard

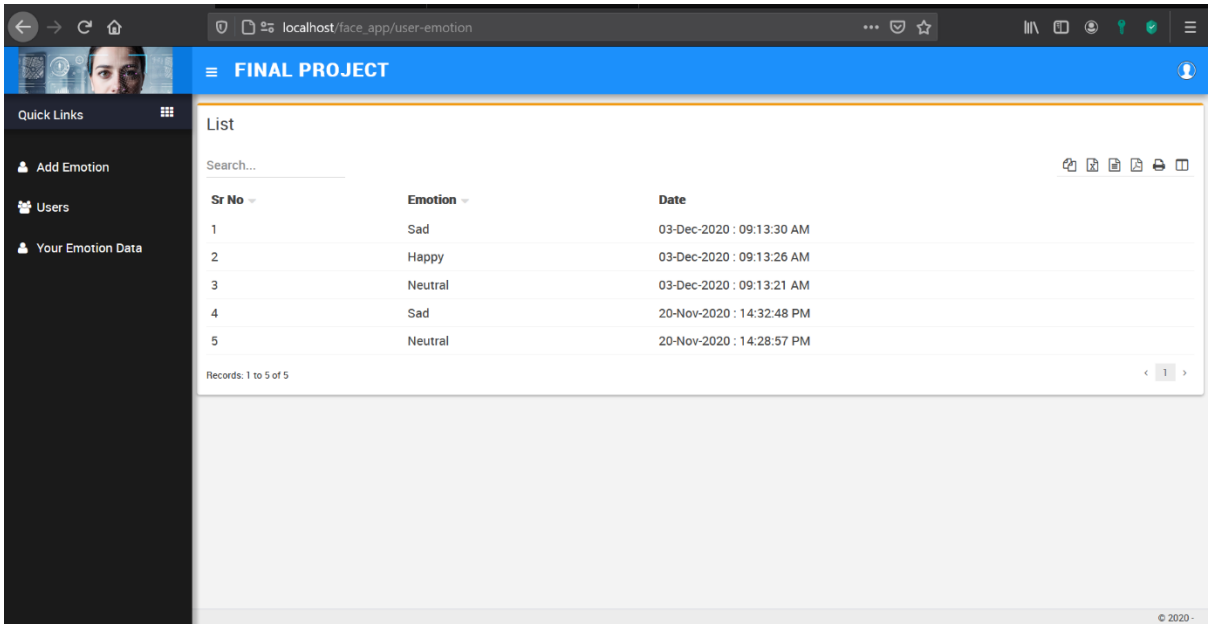

Figure 6: Your Emotion Data

Prediction Images

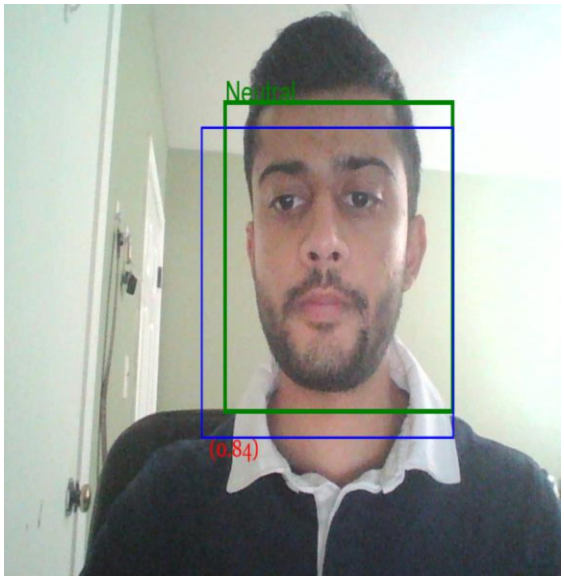

Figure 7: Neutral

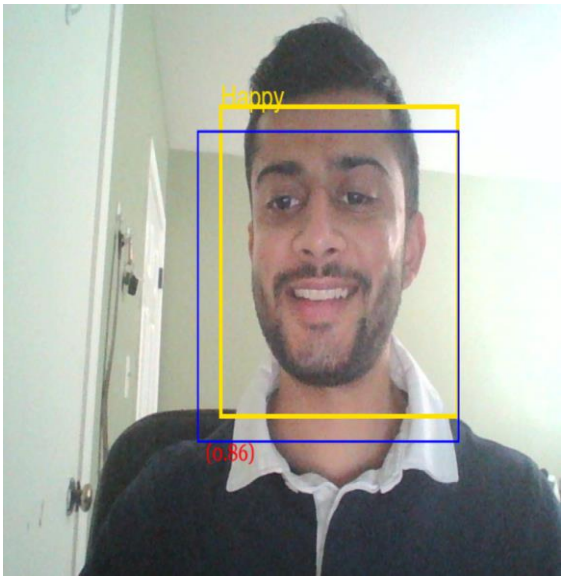

Figure 8: Happy

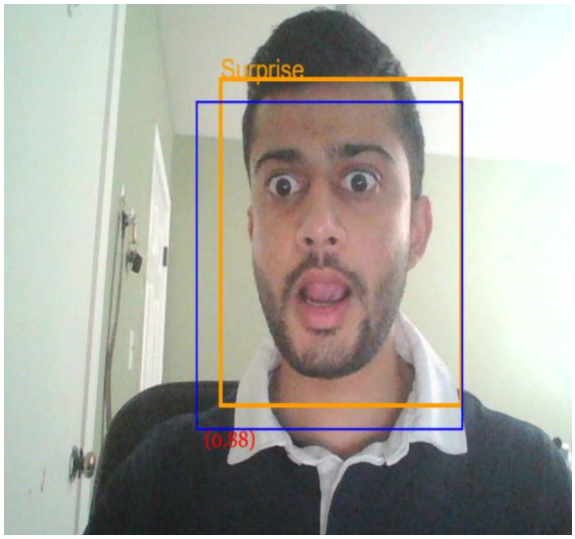

Figure 9: Surprise

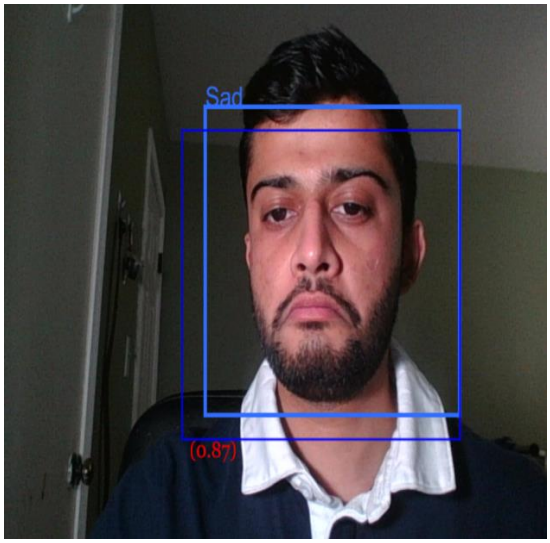

Figure 10: Sad

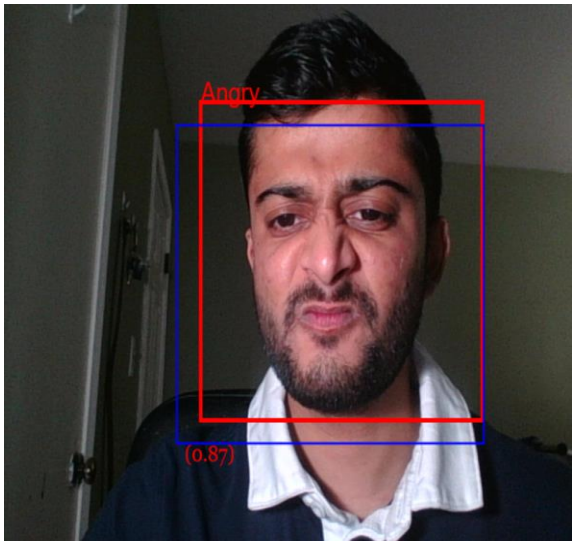

Figure 11: Angry

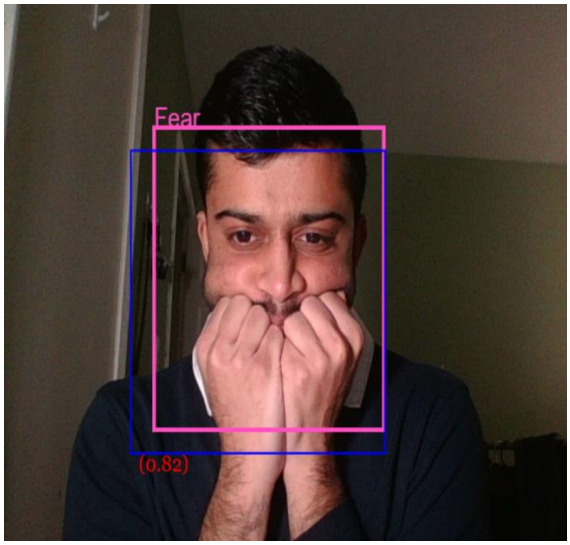

Figure 12: Fear

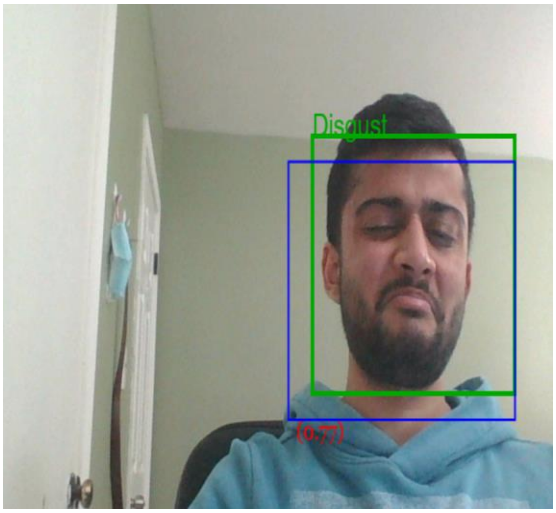

Figure 13: Disgust

Results

There is total seven expressions and the project is able to predict the results with the accuracy of average 83%-87%.

Conclusion

Physiological characteristics of human face with various expressions such as happiness, fear, sadness, anger, surprise, and disgust, all are associated with geometrical structure which resort a base matching template for this recognition system. In this project when model is predicting incorrectly, the correct label is often most likely the second emotion. In term of side face or when the image is not providing proper front face view the algorithm and methods will give less accurate recognition of model. Mostly, the model is given the accuracy of average 83%-87% for all emotions. Additionally, implementing this project I learned to develop web base facial emotion recognition application using face-api.js.

Future Work

I started this project to create something fun work and by taking lots of existing, open-source works. Which allowed me to spin-up important pieces of project quickly, which allowed me to learn how open-source modules work and learned the best ways to glue them together. This project is giving good results but still there is a room to improve this project by:

- Using face detection models which are pretrained as base model (not image net)
- By doing more experimentation with fine tuning steps
- And by using facial landmark data/model in tandem with the face images.

Further, I can also improve this project by using Text Analytics API provided by Oxford [13].

References

1. https://algorithmia.com/blog/introduction-to-emotion-recognition
2. https://www.pocket-lint.com/apps/news/apple/142230-what-are-animoji-how-to-use-apple-s-animated-emoji
3. https://www.pocket-lint.com/apps/news/apple/142230-what-are-animoji-how-to-use-apple-s-animated-emoji
4. https://justadudewhohacks.github.io/face-api.js/docs/index.html
5. http://shuoyang1213.me/WIDERFACE/
6. https://justadudewhohacks.github.io/face-api.js/docs/index.html
7. https://machinelearningmastery.com/how-to-develop-convolutional-neural-network-models-for-time-series-forecasting/
8. https://web.stanford.edu/class/cs231a/lectures/intro_cnn.pdf
9. https://kpzhang93.github.io/MTCNN_face_detection_alignment/
10. Joint Face Detection and Alignment using Multi-task Cascaded Convolutional NetworksKaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Senior Member, IEEE, and Yu Qiao, Senior Member, IEEE
11. Depthwise Separable Convolutions for NeuralMachine Translation, Łukasz Kaiser, Aidan N. Gomez, François Chollet
12. https://arxiv.org/pdf/1704.04861.pdf
13. https://cran.r-project.org/web/packages/mscstexta4r/README.html