Face Recognition Home Security

Prepared by

Slman Ali

Abdullah

Mahmood Yousaf

Sabir Mehdi

December 22, 2021

1. Abstract

The main aim of this project is to use computer vision for face recognition. For face recognition, the dataset was the primary challenge for this project.Dataset is generated using opencv by using webcam. Using 2000 images with 4 classes are used for model training and testing. CNN architecture is trained.After training, model is tested with unseen dataset. For evaluation of classification models, the most common measures like precision, recall and f1 score are calculated.

2. Introduction

Living in this period, we should concede the way that conveying the actual key has turned into an exceptionally feverish thing. Some way or another, assuming that you lose it, you will really place yourself in hot water. Aside from it, there are security issues inseparably connected with the conventional lock system. This project is basically designed for Face ID Door lock. This is basically a computer vision and IOT based project. In this project we have covered the computer vision and face recognition model of the project. This project uses a camera to capture the face. It works in an extremely productive and secure manner to conclude whose face it is. The facial acknowledgment framework essentially catches the essence of the individual and afterward stores it in the database. Afterward, when the individual comes to open the door, the facial acknowledgment innovation filters the picture, and afterward it will check with the information base.

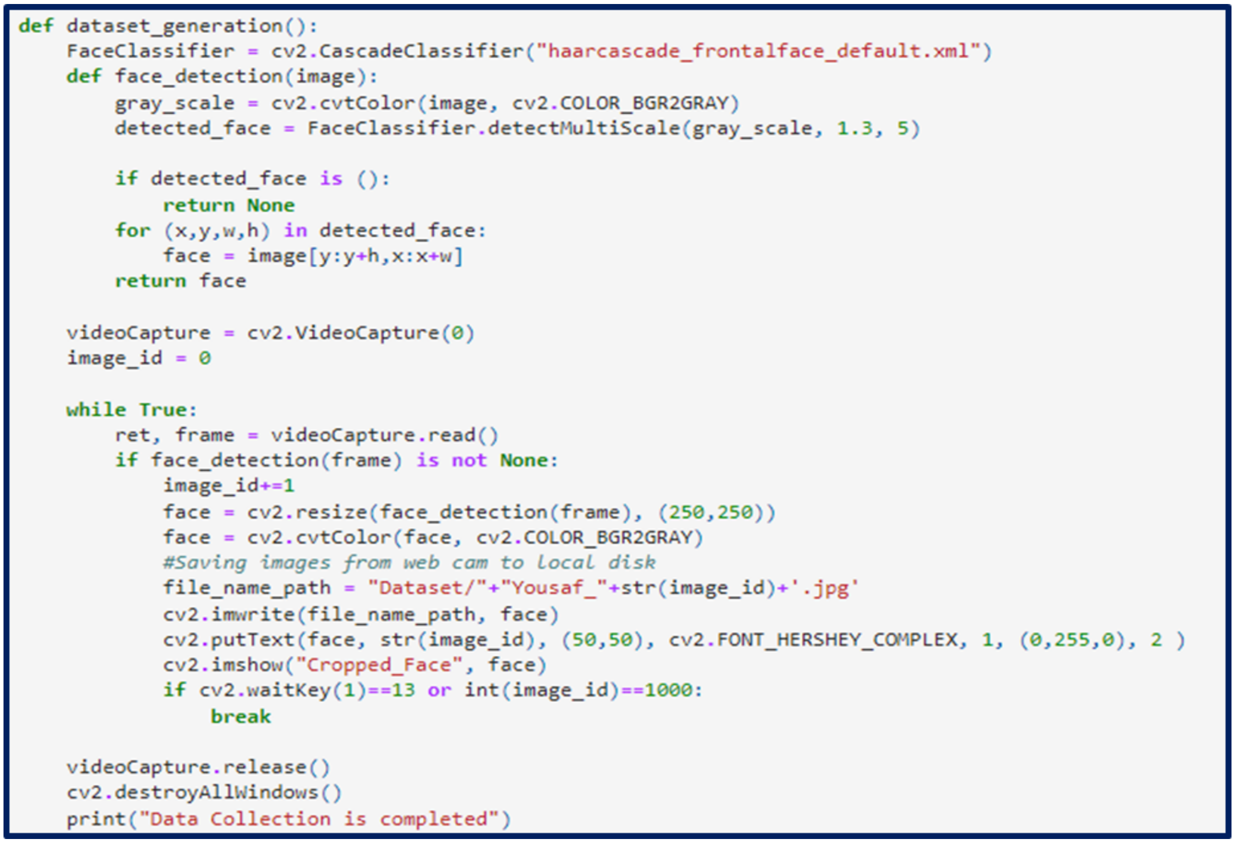
In the event that the picture matches the put away information base, the door will be opened. If not, it will send a gatecrasher caution to the owner. Face ID acknowledgment innovation is, without a doubt, probably the most effective way of biometric confirmation. It is dependable and secure. These days, we can see diverse shrewd devices producing organizations are coordinating this innovation.

3. Data Generation

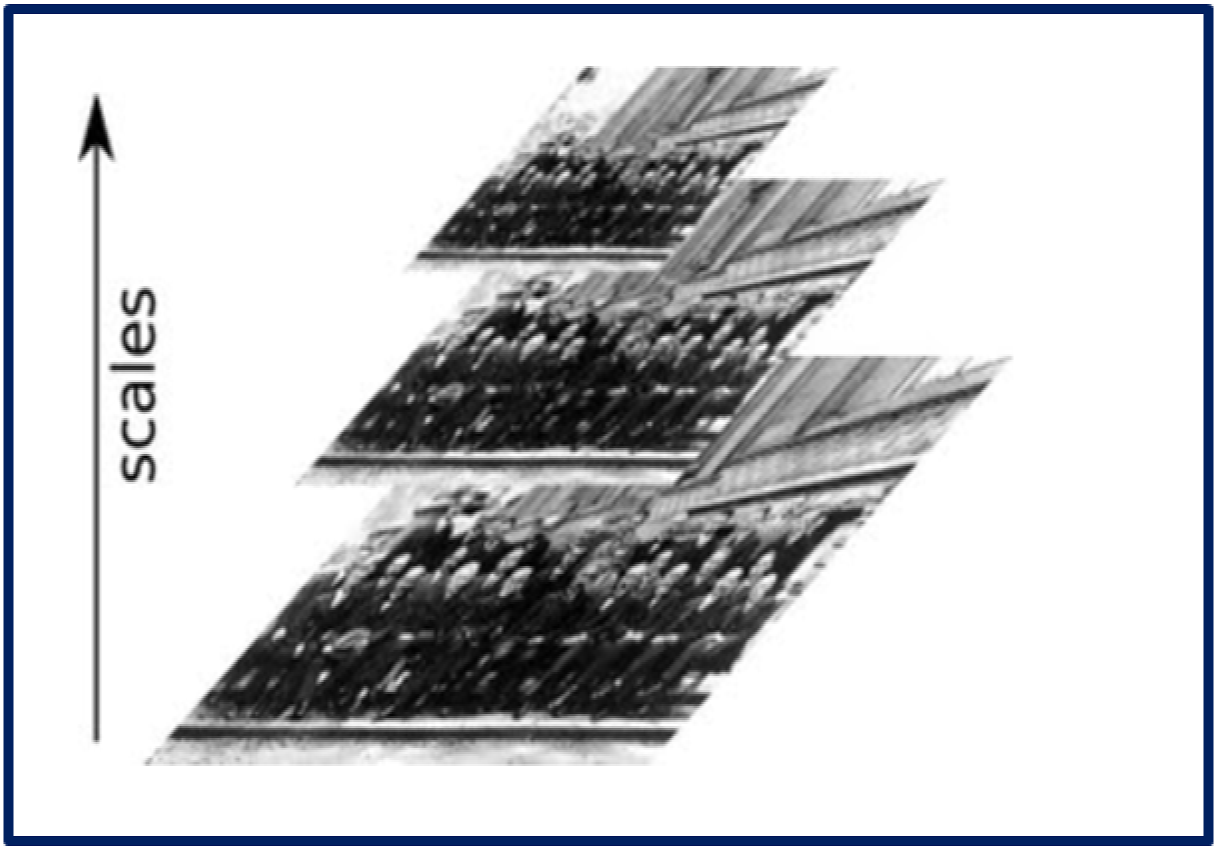
Data generation is done locally by taking pictures of our colleagues. As it is face recognition, we took almost 500 pics of a single person from different angles so that our model would predict it correctly in different circumstances. we took 4 people (Abdullah, Manzoor, Qazi and Mahmood) and got 200 images (each of which are 500).

As we are taking pictures from the live cam so we need to get pics of only faces and not the background so for this we used haarcascade frontal face classifier file and passed it to the cascade classifier, so after performing this we were able to get only a face rather than both face and background.
Basically, the frontal face classifier actually uses an algorithm known as viola jones algorithm, which is tree based algo which predicts whether something is face or not.

After that we performed some scaling and ultimately generated our dataset successfully.



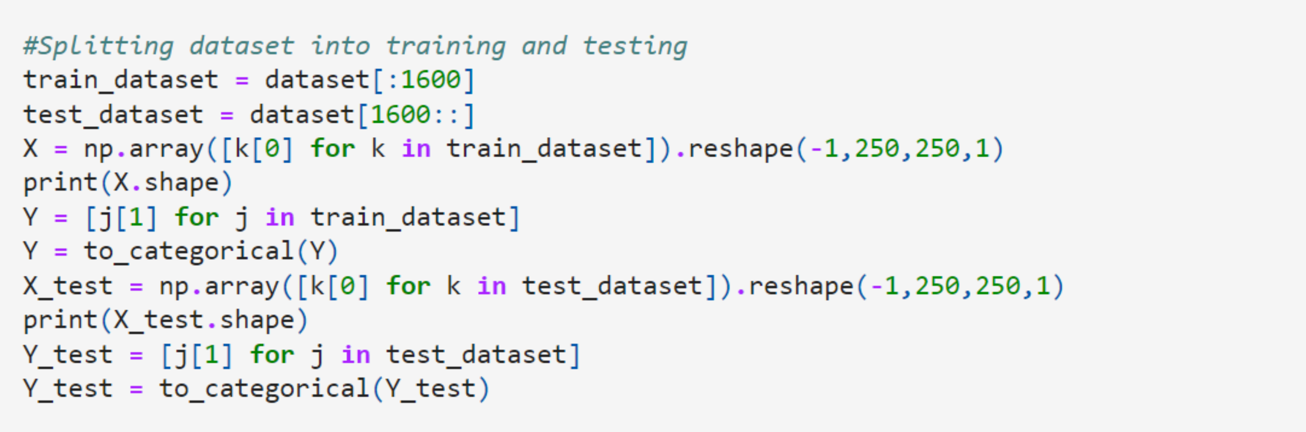
```
def dataset_generation():
    FaceClassifier = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
    def face_detection(image):
        gray_scale = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        detected_face = FaceClassifier.detectMultiScale(gray_scale, 1.3, 5)

        if detected_face is ():
            return None
        for (x,y,w,h) in detected_face:
            face = image[y:y+h,x:x+w]
        return face

    videoCapture = cv2.VideoCapture(0)
    image_id = 0

    while True:
        ret, frame = videoCapture.read()
        if face_detection(frame) is not None:
            image_id+=1
            face = cv2.resize(face_detection(frame), (250,250))
            face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)
            #Saving images from web cam to local disk
            file_name_path = "Dataset/"+"Yousaf_"+str(image_id)+'.jpg'
            cv2.imwrite(file_name_path, face)
            cv2.putText(face, str(image_id), (50,50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0), 2 )
            cv2.imshow("Cropped_Face", face)
            if cv2.waitKey(1)==13 or int(image_id)==1000:
                break

    videoCapture.release()
    cv2.destroyAllWindows()
    print("Data Collection is completed")
```



4. Data Pre-Processing

In data preprocessing first of all we have to label the data as for now we are having data labeled as (yousaf,abdullah,qazi and manzor). Now at this part we will label the data with numerical data because we performed numerical operations on the data. Once we have labeled the data, the next step is to minimize the computation cost. For this purpose we will divide the image by 256 for lowering the computation power. That’s how we will save power consumption. We have a dataset which is in patterns and to produce randomness in data we will use the shuffle library to increase the randomness in our dataset.

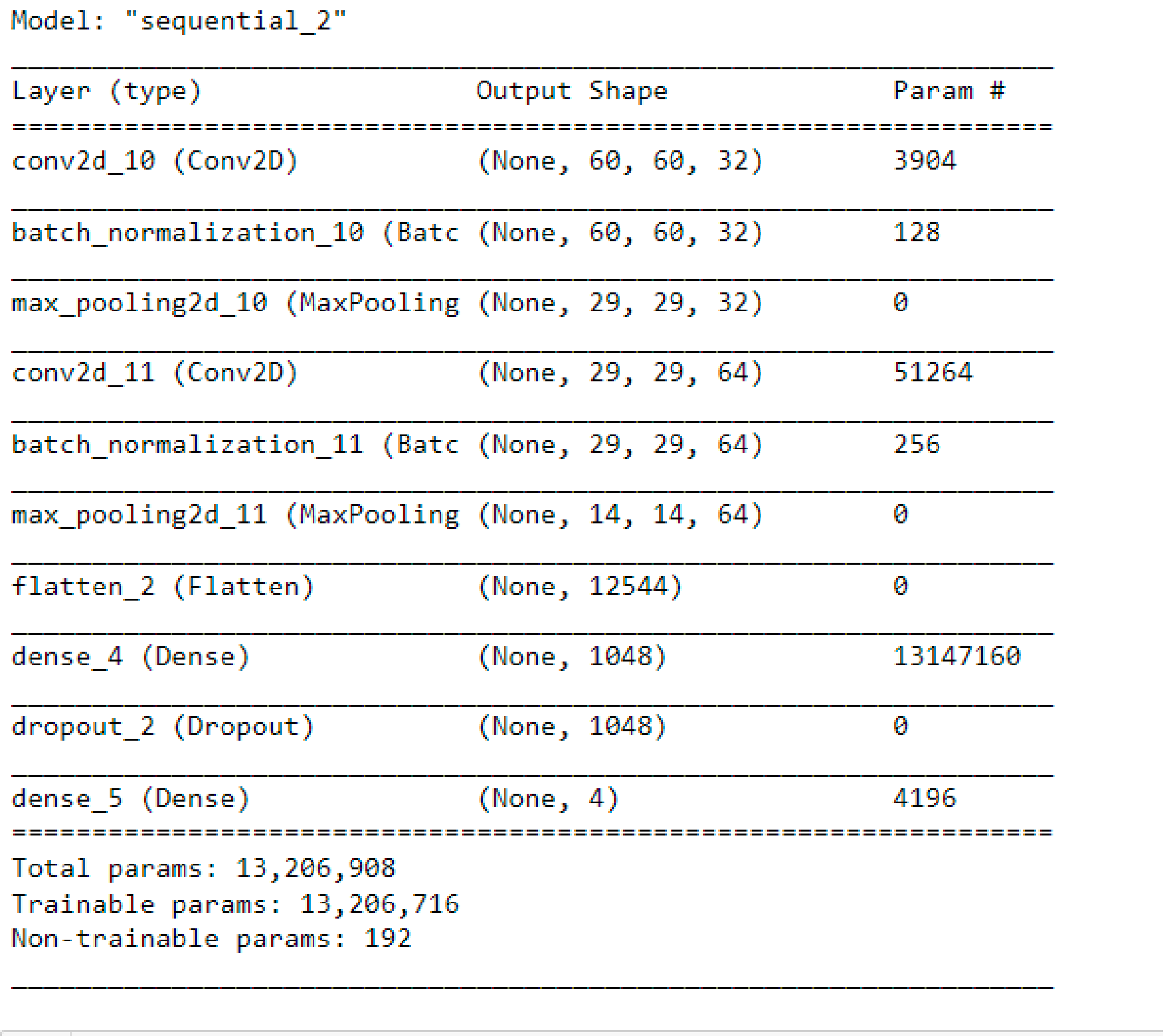
In our model training we will split the dataset into 1600 images for training and 400 images for testing data(unseen). As our images are 256x256 we will also have to do a reshaping by using the reshape function according to our dataset. In the final step we will divide the dataset into categorical data. As we have four types of images so we will divide the dataset into four categories.



```
#Splitting dataset into training and testing
train_dataset = dataset[:1600]
test_dataset = dataset[1600::]
X = np.array([k[0] for k in train_dataset]).reshape(-1,250,250,1)
print(X.shape)
Y = [j[1] for j in train_dataset]
Y = to_categorical(Y)
X_test = np.array([k[0] for k in test_dataset]).reshape(-1,250,250,1)
print(X_test.shape)
Y_test = [j[1] for j in test_dataset]
Y_test = to_categorical(Y_test)
```

5. Methodology

After data preprocessing, data is prepared for machine learning models. CNN models are used for training and testing. CNN architecture does feature extraction automatically. Following CNN architecture is used for model training and testing.



```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_10 (Conv2D)           (None, 60, 60, 32)        3904      
_________________________________________________________________
batch_normalization_10 (Batc (None, 60, 60, 32)        128       
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 29, 29, 32)        0         
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 29, 29, 64)        51264     
_________________________________________________________________
batch_normalization_11 (Batc (None, 29, 29, 64)        256       
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 14, 14, 64)        0         
_________________________________________________________________
flatten_2 (Flatten)          (None, 12544)             0         
_________________________________________________________________
dense_4 (Dense)              (None, 1048)              13147160  
_________________________________________________________________
dropout_2 (Dropout)          (None, 1048)              0         
_________________________________________________________________
dense_5 (Dense)              (None, 4)                 4196      
=================================================================
Total params: 13,206,908
Trainable params: 13,206,716
Non-trainable params: 192
_________________________________________________________________
```

Fig 1: CNN Model Architecture

Softmax activation function is used at the last layer because we have a multi-class classification problem. We have four classes and the softmax function will predict probability for each class.

6. Results

Model Accuracy

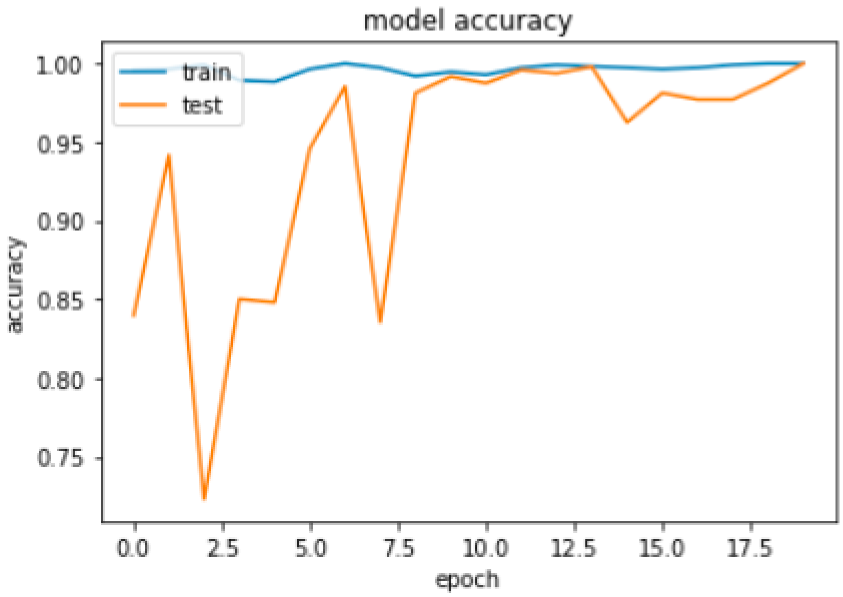


Fig 2: Training and testing accuracy

Model Loss

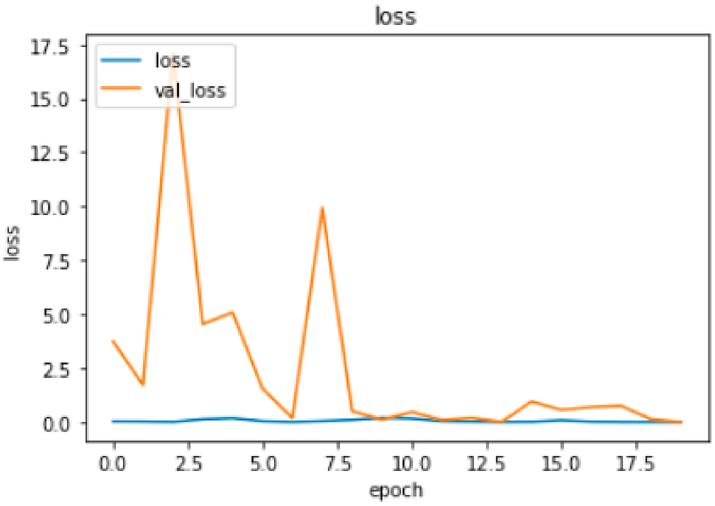


Fig 3: Training and testing loss

Performance measures for CNN Model

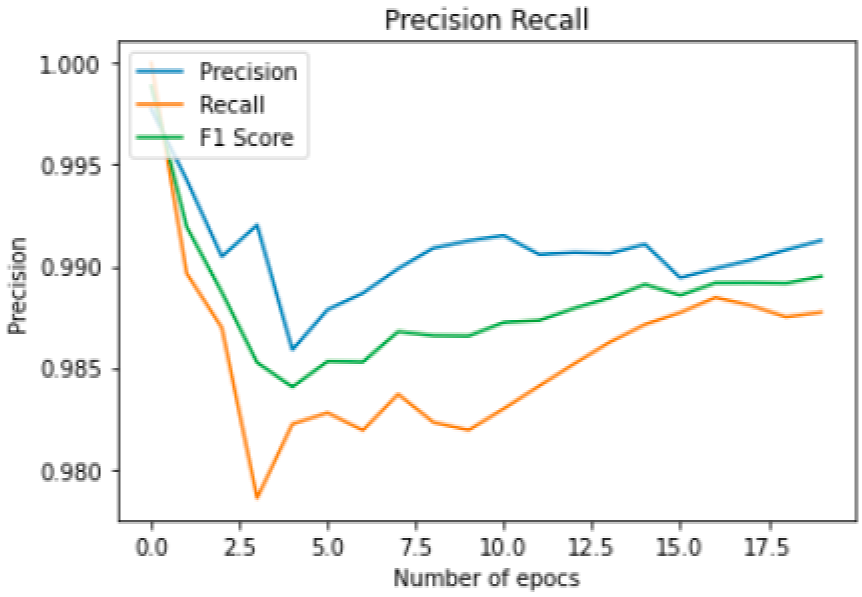
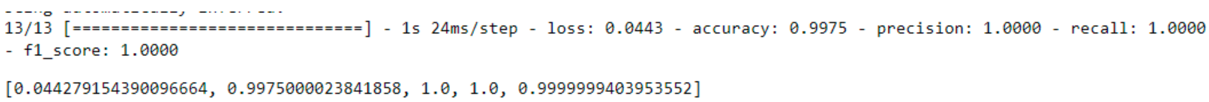


Fig 4: Performance measures for multi-class classification problem

Results on unseen dataset



```
13/13 [==============================] - 1s 24ms/step - loss: 0.0443 - accuracy: 0.9975 - precision: 1.0000 - recall: 1.0000
- f1_score: 1.0000

[0.044279154390096664, 0.9975000023841858, 1.0, 1.0, 0.9999999403953552]
```

Fig 5: Results on unseen dataset

Source Code:

Github Link: [Click here to view the project on github]

7. Recommendation

This project can proceed with IOT Face detection projects such as car doors and home doors etc. This model can be more efficient for live image capturing. This model can detect authorized users. This model learns from data. This project is basically useful for disable people who can't unlock doors even with fingerprints who can't even use finger print either they lost their hands or they cannot access fingerprint devices. So, they can use their face ID to unlock the doors easily and there will be no security issues. This is more helpful to them.

Face ID acknowledgment innovation is, without a doubt, probably the most effective way of biometric confirmation. It is dependable and secure. These days, we can see diverse shrewd devices producing organizations are coordinating this innovation. There are a lot of benefits of Face ID door lock such as providing touchless access, no need to carry key, highly secure and automation.

8. References

- Facial Recognition History - Mobbeel
- jQuery Image Cropping Plugins | jQuery Script
- Cropper.js - Plugin of jQuery For Image Cropping | jQuery Plugins (jquerypost.com)