A Project/Dissertation Review- Report
On

Face Detection

Submitted in partial fulfillment of the requirements
for the award of degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

Under the supervision of
Name of Supervisor: Mr.V.Gokul Rajan
Designation: Professor

Submitted By

Rahul Verma (18SCSE1010173)
Raushan (18SCSE1010511)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING / DEPARTMENT OF COMPUTERAPPLICATION
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA
DECEMBER, 2021

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"FACE DETECTION"** in partial fulfillment of the requirements for the award of the B.TECH submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of JUNE 2021 to DECEMBER 2021,

under the supervision of Mr.V.Gokul Rajan, Professor, Department of Computer Science and Engineering/Computer Application and Information and Science, of School of Computing Science and Engineering , Galgotias University, Greater Noida.

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

Rahul Verma (18SCSE1010173)

Raushan(18SCSE1010511)

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Mr.V.Gokul Rajan

Professor

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of Rahul Verma (18SCSE1010173) and Raushan (18SCSE1010511) has been held on and his/her work is recommended for the award of B. Tech(CSE)

Signature of Examiner(s) **Signatureof Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date: 20th December 2021
Place: Greater Noida

ACKNOWLEDGEMENT

In the accomplishment of completion of my project on Malware Scanner I would like to convey my special gratitude to Mr.V.Gokul Rajan of Computer Science Department.

Your valuable guidance and suggestions helped me in various phases of the completion of this project. I will always be thankful to you in this regard.

I am ensuring that this project was finished by me and not copied.

Rahul Verma (18SCSE1010173)

Raushan (18SCSE1010511)

ABSTRACT

In this project, a secure face recognition system is presented, in which face detection is performed with skin color detection followed by light normalization and normalized cross correlation. Principal component analysis (PCA) is used for face verification. Due to the rising concern about the security and privacy of the biometric system, we offer a secure storage for user records by encrypting them using Advanced Encryption Standard (AES). A different encryption/ decryption key is used for each user, and that key is not stored in the database, it is extracted by expanding the submitted user identification (ID). A simulation of AES algorithm using field programmable gate array (FPGA) and the very high speed integrated circuit hardware description language (VHDL) is performed.

Keywords: Opencv , Pandas , Numpy , Machine Learing , face recognition , Advanced Encryption Standard (AES).

Table Of Contents

LIST OF FIGURES

CHAPTER I

Introduction

Machine learning is a subfield of Artificial Intelligence. The trends in advancement of AI pertains its success to the computation capabilities of the modern day CPUs and GPUs which can run large datasets and give us results. The smart phones of today, without an exception have a selfie camera, which generally has a rectangular boundary around the face. The selfie camera has certain features, like it may display the age of the person, the gender of the person. The accuracy of the features displayed by the selfie camera depends on several factors. Such as light, angle, beauty mode enabled etc. The smart phone industry, uses a pretrained model to recognize the gender and age of the person. The pre-trained model uses deep learning to identify features using which it predicts the gender and age of the person based on identifying and analyzing facial features.

FORMULATION OF PROBLEM

The problem statement is to recognize the face that is feeded into the model via webcam. The image will be feeded to the pre-trained model via a webcam. This venture was finished with this incredible "Open Source Computer Vision Library", the OpenCV. On this instructional exercise, we will concentrate on Raspberry Pi (along these lines, Raspbian as OS) and Python, yet I additionally tried the code on My Windows and it likewise works fine. OpenCV was intended for computational proficiency and with a solid spotlight on continuous applications. In this way, it's ideal for ongoing face acknowledgment utilizing a camera.

SCOPE/OBJECTIVE

In addition to being used for security systems, authorities have found a number of other applications for facial recognition systems. While earlier post-9/11 deployments were well-publicized trials, more recent deployments are rarely written about due to their covert nature.

There are also a number of potential uses for facial recognition that are currently being developed. For example, the technology could be used as a security measure at ATMs. Instead of using a bank card or personal identification number, the ATM would capture an image of the customer's face, and compare it to the account holder's photo in the bank database to confirm the customer's identity. Facial recognition systems are used to unlock software on mobile devices. An independently developed Android Marketplace app called VisidonApplock makes use of the phone's built-in camera to take a picture of the user. Facial recognition is used to ensure only this person can use certain apps that they choose to secure.

Face detection and facial recognition are integrated into the iPhoto application for Macintosh, to help users organize and caption their collections.

Some cameras, in addition, incorporate a smile shutter or take automatically a second picture if someone closed their eyes during exposure.

Because of certain limitations of fingerprint recognition systems, facial recognition systems are used as an alternative way to confirm employee attendance at work for the claimed hours.

Another use could be a portable device to assist people with prosopagnosia in recognizing their acquaintances.

TOOLS AND TECHNOLOGY USED

- **Video Capture**
- **OpenCV**
- **Python**
- **Libraries/Packages:**
  i) numpy
  ιι) **ios**
  ιιι) **Pandas**
- **Face API**
- **Cascade Classifier**

CHAPTER 2

LITERATURE SURVEY

1. What the original author is doing is that they are trying to prove the feasibility and the effectiveness of machine learning methods for computer-aided diagnosis (CAD) of lymph node metastasis in gastric cancer using clinical GSI data. In this paper, we employed a simple and classic algorithm called KNN that combines several feature selection algorithms and metric learning methods. The experimental results show that our scheme outperforms traditional diagnostic means (e.g., EUS and MDCT).

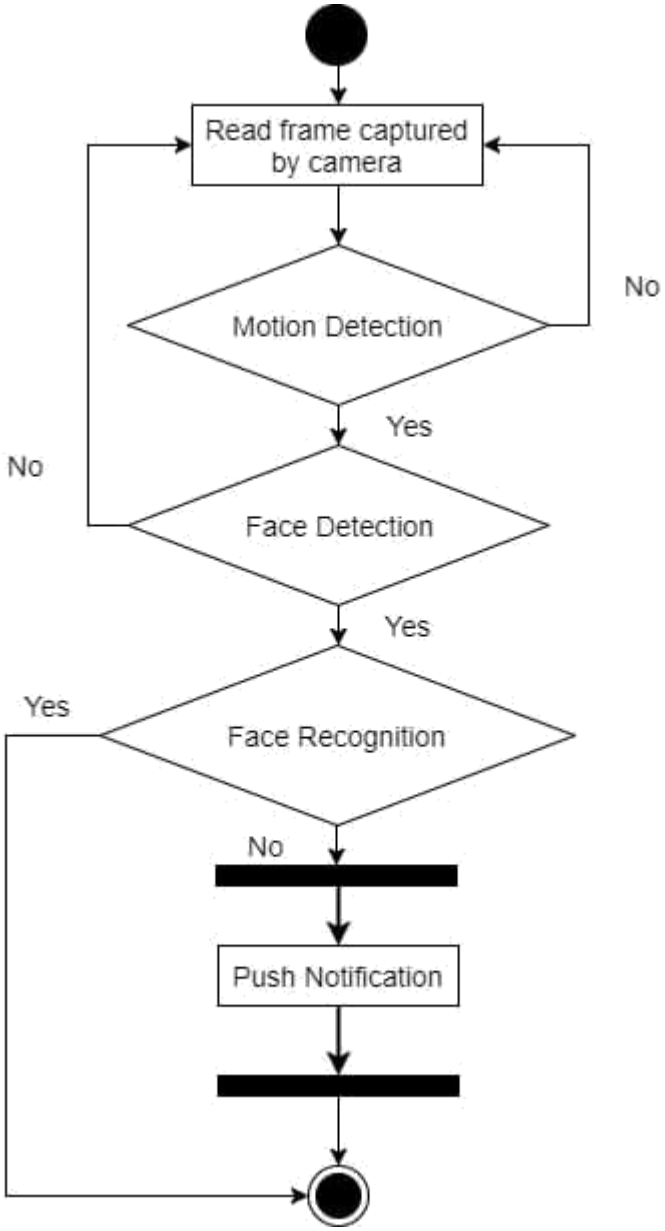
2. The limitation of the research is the insufficient number of clinical cases. Thus, in the future work, we will conduct more experiments on clinical data to improve further the efficiency of the proposed scheme and to explore more useful and powerful machine learning methods for CAD in clinical.

3. This literature review suggests that designing a suitable image-processing procedure is a prerequisite for a successful classification of remotely sensed data into a thematic map. Effective use of multiple features of remotely sensed data and the selection of a suitable classification method are especially significant for improving classification

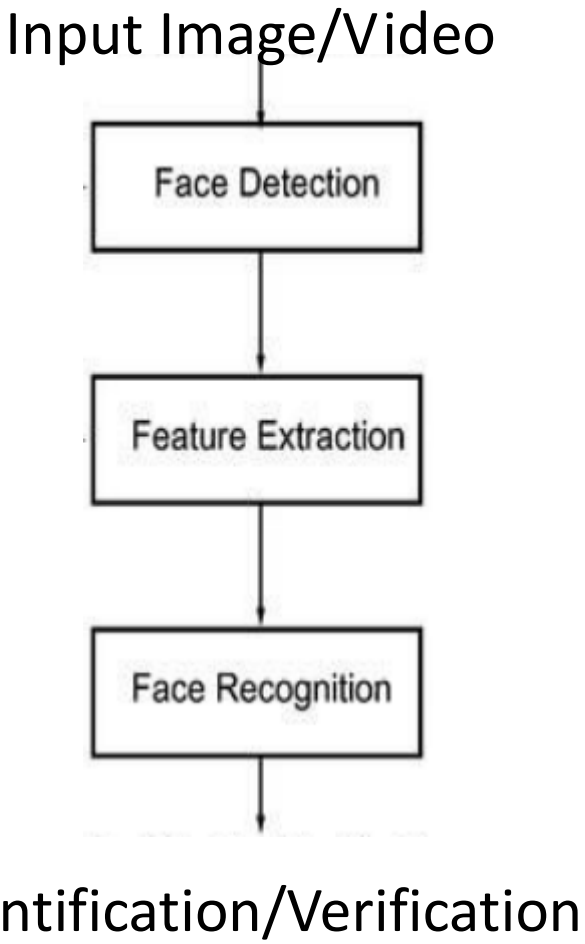
Architecture diagram for proposed system

An architecture diagram is a graphical representation of a set of the concepts, that are part of an architecture, including their principles and elements and components. It actually helps the viewer to analyze that how the proposed system is working in it.

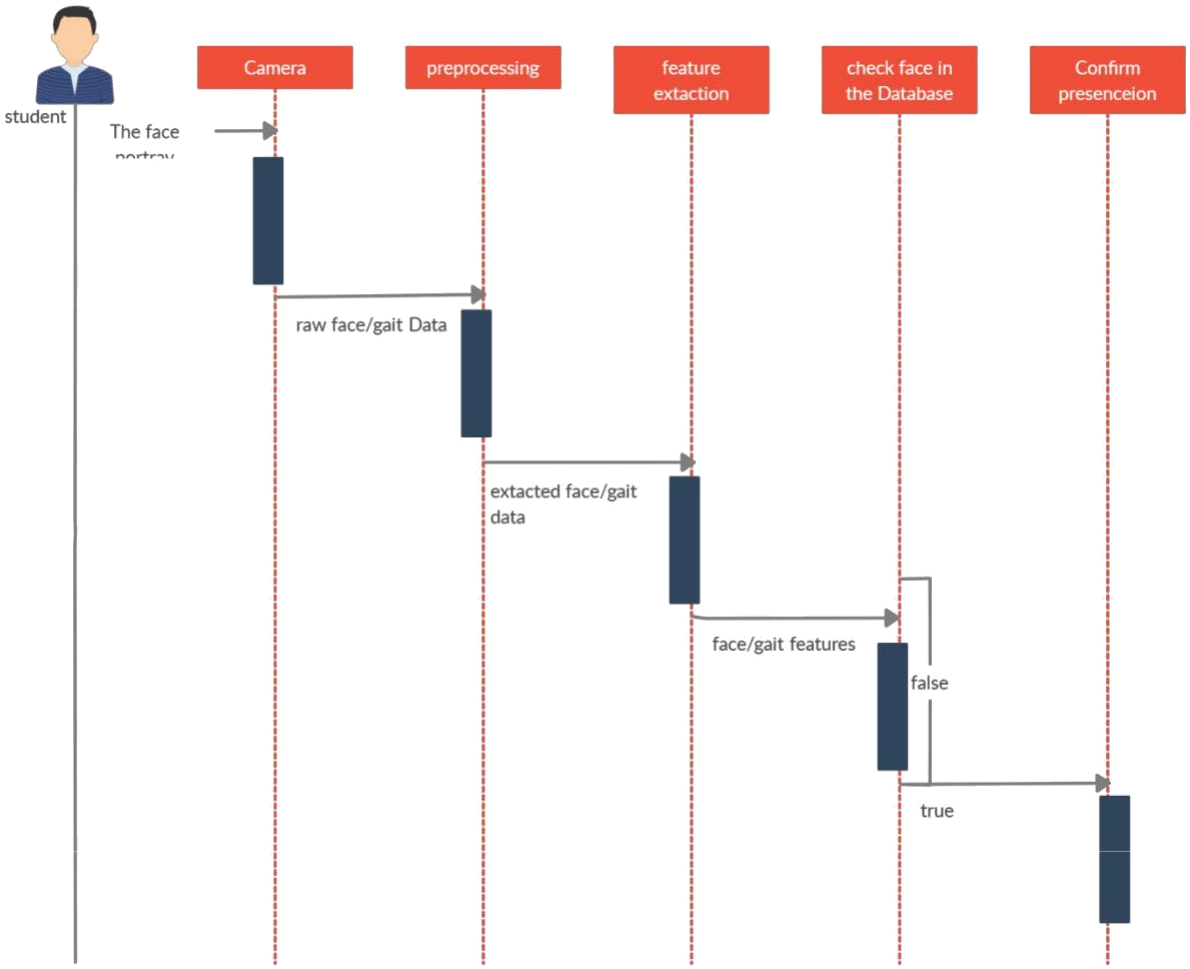
Activity Diagram –



User Diagram –



Sequence Diagram –



CHAPTER 3

FUNCTIONALITY/ WORKING OF PROJECT

We have proposed one modified algorithm which uses the underlying principles of KLT Algorithm and Viola-Jones Algorithm, but it works somehow better than that and it has been proved. We have used the concept of finding the eigen vectors for the live stream and this amount of substantial work hasn't been done before for real time systems, but for the video stream, research has been done.

1. Detect the Face- First, you must detect the face. Use the vision.CascadeObjectDetector System object to detect the location of a face in a video frame. The cascade object detector uses the Viola-Jones detection algorithm and a trained classi-fication model for detection. By default, the detector is configured to detect faces, but it can be used to detect other types of objects.To track the face over time, our implementation uses the Kanade-Lucas-Tomasi (KLT) algorithm.
2. Identify Facial Features to Track- The KLT algorithm tracks a set of feature points across the video frames. Once the detection locates the face, it identifies feature points that can be reliably tracked.
3. Initialize a Tracker to Track the Points - With the feature points identified, you can now use the vision.PointTracker System object to track them. For each point in the previous frame, the point tracker attempts to find the corresponding point in the current frame.

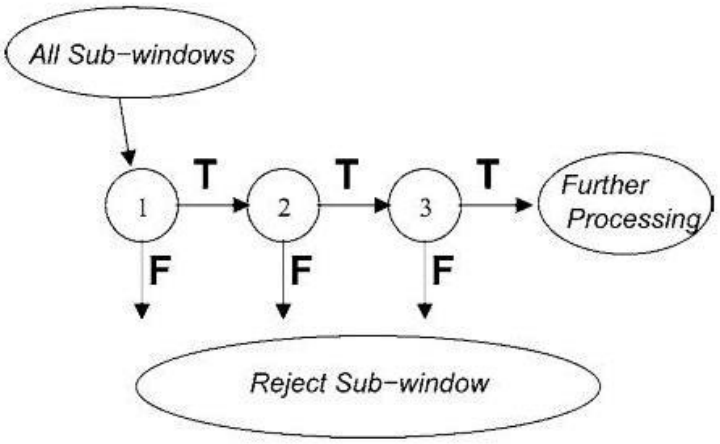


Figure – Detectors Cascade

4. Initialize a Video Player to Display the Results
5. Track the Face

MERITS OF PROPOSED SYSTEM

* No Training Period: KNN is called Lazy Learner (Instance based learning). It does not learn anything in the training period. It does not derive any discriminative function from the training data. In other words, there is no training period for it. It stores the training dataset and learns from it only at the time of making real time predictions. This makes the KNN algorithm much faster than other algorithms that require training e.g. SVM, Linear Regression etc.

* Since the KNN algorithm requires no training before making predictions, new data can be added seamlessly which will not impact the accuracy of the algorithm.

* KNN is very easy to implement. There are only two parameters required to implement KNN i.e. the value of K and the distance function (e.g. Euclidean or Manhattan etc.)

WORKING OF PROJECT AND CODE IMPLEMENTATION

```
// video
stream
from web
camera
        import cv2
        cap=cv2.VideoCapture(0)
        whileTrue:
                ret,frame=cap.read()
                gray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
                if ret==False:
                        Continue
                cv2.imshow('video frame',frame)
                cv2.imshow('gray frame',gray)
                // wait for user input-q to stop the capture
                key_pressed=cv2.waitKey(1) &0xFF
                ifkey_pressed==ord('q'):
                        Break
        cap.release()
        cv2.destroyAllWindows()
```

Reading Video Stream

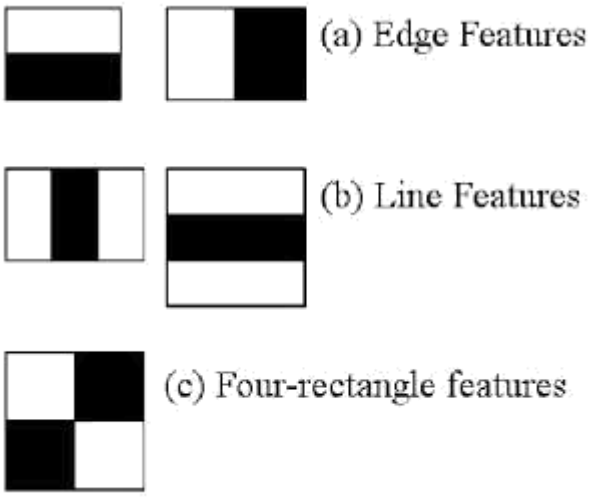
It conatins python code for classifying the name of the user after detecting their face.

```
#
reading
// video
stream from
web camera
        import cv2
        cap=cv2.VideoCapture(0)
        whileTrue:
            ret,frame=cap.read()
            gray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
            if ret==False:
                Continue
            cv2.imshow('video frame',frame)
            cv2.imshow('gray frame',gray)
            // wait for user input-q to stop the capture
            key_pressed=cv2.waitKey(1) &0xFF
            ifkey_pressed==ord('q'):
                Break
        cap.release()
        cv2.destroyAllWindows()
```

Face detection

Face detection has gained a lot of attention due to its real-time applications. A lot of research has been done and still going on for improved and fast implementation of the face detection algorithm. Why is face detection difficult for a machine? Face detection is not as easy as it seems due to lots of variations of image appearance, such as pose variation (front, non-front), occlusion, image orientation, illumination changes and facial expression.
OpenCV contains many pre-trained classifiers for face, eyes, smile etc.

In this project I have used the haarcascade classifier which uses it'shaar features to detect face.



These are some Haar features which are similar to convolutional kernals(used in convolutional neural network).

```
import cv2
cap=cv2.VideoCapture(0)
face_cascade=cv2.CascadeClassifier('haarcascade_frontalface_alt.xml ')
whileTrue:
        ret,frame=cap.read()
        gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
        if ret==False:
                Continue
        faces=face_cascade.detectMultiScale(gray,1.3,5)
        forx,y,w,hin faces:
                cv2.rectangle(frame,(x,y),((x+w),(y+h)),(255,0,0),2)
cv2.imshow('video frame',frame)
        Key_pressed=cv2.waitKey(1) &0xFF
        ifkey_pressed==ord('a'):
                break
cap.release()
cv2.destroyAllWindows()
```

face_data_collection

```
import cv2
import numpy as np
cap=cv2.VideoCapture(0)
face_cascade=cv2.CascadeClassifier('haarcascade_frontalface_alt.xml') skip=0
face_data=[]
dataset_path='./data/'
file_name=input("Enter the name of the person :")
while True:
        ret,frame=cap.read()
        gray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
        if ret==False:
                continue
        faces=face_cascade.detectMultiScale(gray,1.3,5)
        if len(faces)==0:
                continue
        faces=sorted(faces,key=lambda f:f[2]*f[3])

        * picking the last face which has largest area f[2]*f[3] for
        face in faces[-1:]:
                x,y,w,h=face
                cv2.rectangle(frame,(x,y),((x+w),(y+h)),(0,255,255),2)
                  extract(crop out the required face) :region of interest(ROI)
                off_set=10 face_region=frame[y-off_set:y+h+off_set,x-
                off_set:x+w+off_set]
                face_region=cv2.resize(face_region,(100,100))
                skip+=1
                  capturing every 10th frame
                if skip%10==0:
                        face_data.append(face_region)
                        print(len(face_data))


        cv2.imshow('frame',frame)
        cv2.imshow('face section',face_region)
        key_pressed=cv2.waitKey(1) & 0xFF
        if key_pressed==ord('q'):
                break
```

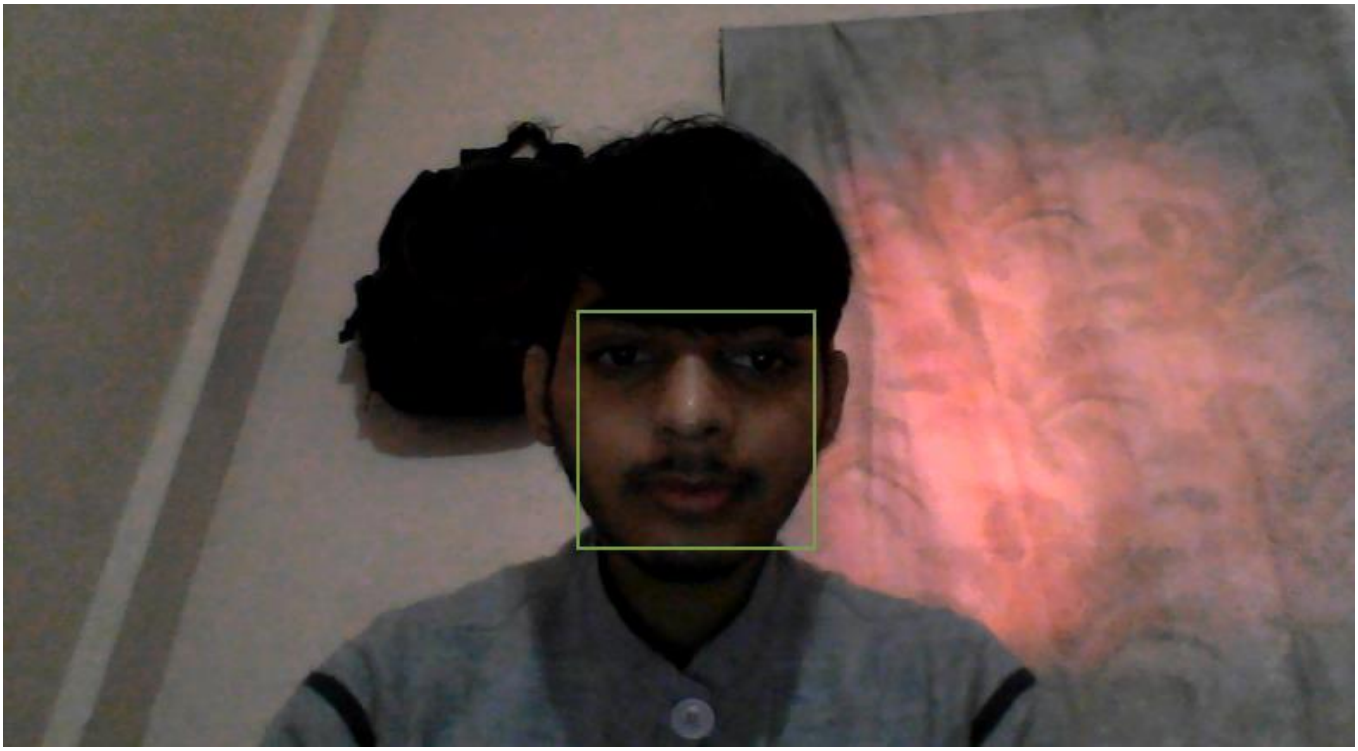
```
* convert our face list into numpy array
face_data=np.array(face_data)
face_data=face_data.reshape((face_data.shape[0],-1))
print(face_data.shape)
* save this data into file system
np.save(dataset_path+file_name+'.npy',face_data)
print("data saved sucessfully at "+dataset_path+file_name+'.npy')
cap.release()
cv2.destroyAllWindows()
```

Face Recognition

```
import cv2
import numpy as np
import os
#######################          KNN CODE  ######################
def distance(x1,x2):
        return np.sqrt(((x1-x2)**2).sum())
def KNN(train,test,k=5):
        dist=[]
        for ix in range(train.shape[0]):
                x=train[ix,:-1]
                y=train[ix,-1]
                d=distance(test,x)
                dist.append((d,y))
        # print(dist)
        dist=sorted(dist,key=lambda x:x[0])
        dist=dist[:k]
        labels=np.array(dist)
        labels=labels[:,-1]
        output=np.unique(labels,return_counts=True)
        index=np.argmax(output[1])
        return output[0][index]
#################################################################
import face_recognition as fr
import cv2
import numpy as np
import os

path = "./train/"
```

```
known_names = []
known_name_encodings = []

images = os.listdir(path)
for _ in images:
    image = fr.load_image_file(path + _)
    image_path = path + _
    encoding = fr.face_encodings(image)[0]

    known_name_encodings.append(encoding)

known_names.append(os.path.splitext(os.path.basename(image_path))[0].capitalize()
)

print(known_names)

test_image = "./test/test.jpg"
image = cv2.imread(test_image)
# image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

face_locations = fr.face_locations(image)
face_encodings = fr.face_encodings(image, face_locations)

for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
    matches = fr.compare_faces(known_name_encodings, face_encoding)
    name = ""

    face_distances = fr.face_distance(known_name_encodings, face_encoding)
    best_match = np.argmin(face_distances)

    if matches[best_match]:
        name = known_names[best_match]

    cv2.rectangle(image, (left, top), (right, bottom), (0, 0, 255), 2)
    cv2.rectangle(image, (left, bottom - 15), (right, bottom), (0, 0, 255), cv2.FILLED)
    font = cv2.FONT_HERSHEY_DUPLEX
    cv2.putText(image, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255), 1)


cv2.imshow("Result", image)
cv2.imwrite("./output.jpg", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output –



Haarcascades code for Face –

-<opencv_storage>

-<cascade type_id="opencv-cascade-classifier">

<stageType>BOOST</stageType>

<featureType>HAAR</featureType>

<height>20</height>

<width>20</width>
-<stageParams>

<maxWeakCount>213</maxWeakCount>

</stageParams>
-<featureParams>

<maxCatCount>0</maxCatCount>

</featureParams>

<stageNum>22</stageNum>
-<stages>
-<_>

<maxWeakCount>3</maxWeakCount>

<stageThreshold>8.2268941402435303e-01</stageThreshold>
-<weakClassifiers>
-<_>

<internalNodes>0 -1 0 4.0141958743333817e-03</internalNodes>

<leafValues>3.3794190734624863e-02 8.3781069517135620e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 1 1.5151339583098888e-02</internalNodes>

<leafValues>1.5141320228576660e-01 7.4888122081756592e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 2 4.2109931819140911e-03</internalNodes>
<leafValues>9.0049281716346741e-02 6.3748198747634888e-01</leafValues>

</_>

</weakClassifiers>

</_>
-<_>

<maxWeakCount>16</maxWeakCount>

<stageThreshold>6.9566087722778320e+00</stageThreshold>
-<weakClassifiers>

-<_>

<internalNodes>0 -1 3 1.6227109590545297e-03</internalNodes>

<leafValues>6.9308586418628693e-02 7.1109461784362793e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 4 2.2906649392098188e-03</internalNodes>

<leafValues>1.7958030104637146e-01 6.6686922311782837e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 5 5.0025708042085171e-03</internalNodes>

<leafValues>1.6936729848384857e-01 6.5540069341659546e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 6 7.9659894108772278e-03</internalNodes>

<leafValues>5.8663320541381836e-01 9.1414518654346466e-02</leafValues>

</_>
-<_>

<internalNodes>0 -1 7 -3.5227010957896709e-03</internalNodes>

<leafValues>1.4131669700145721e-01 6.0318958759307861e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 8 3.6667689681053162e-02</internalNodes>

<leafValues>3.6756721138954163e-01 7.9203182458877563e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 9 9.3361474573612213e-03</internalNodes>

<leafValues>6.1613857746124268e-01 2.0885099470615387e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 10 8.6961314082145691e-03</internalNodes>

<leafValues>2.8362309932708740e-01 6.3602739572525024e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 11 1.1488880263641477e-03</internalNodes>

<leafValues>2.2235809266567230e-01 5.8007007837295532e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 12 -2.1484689787030220e-03</internalNodes>

<leafValues>2.4064640700817108e-01 5.7870548963546753e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 13 2.1219060290604830e-03</internalNodes>

<leafValues>5.5596548318862915e-01 1.3622370362281799e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 14 -9.3949146568775177e-02</internalNodes>

<leafValues>8.5027372837066650e-01 4.7177401185035706e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 15 1.3777789426967502e-03</internalNodes>

<leafValues>5.9936738014221191e-01 2.8345298767089844e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 16 7.3063157498836517e-02</internalNodes>

<leafValues>4.3418860435485840e-01 7.0600342750549316e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 17 3.6767389974556863e-04</internalNodes>

<leafValues>3.0278879404067993e-01 6.0515749454498291e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 18 -6.0479710809886456e-03</internalNodes>

<leafValues>1.7984339594841003e-01 5.6752568483352661e-01</leafValues>

</_>

</weakClassifiers>

</_>
-<_>

<maxWeakCount>21</maxWeakCount>

<stageThreshold>9.4985427856445312e+00</stageThreshold>
-<weakClassifiers>

-<_>

<internalNodes>0 -1 19 -1.6510689631104469e-02</internalNodes>

<leafValues>6.6442251205444336e-01 1.4248579740524292e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 20 2.7052499353885651e-03</internalNodes>

<leafValues>6.3253521919250488e-01 1.2884770333766937e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 21 2.8069869149476290e-03</internalNodes>

<leafValues>1.2402880191802979e-01 6.1931931972503662e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 22 -1.5402400167658925e-03</internalNodes>

<leafValues>1.4321430027484894e-01 5.6700158119201660e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 23 -5.6386279175058007e-04</internalNodes>

<leafValues>1.6574330627918243e-01 5.9052079916000366e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 24 1.9253729842603207e-03</internalNodes>

<leafValues>2.6955071091651917e-01 5.7388240098953247e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 25 -5.0214841030538082e-03</internalNodes>

<leafValues>1.8935389816761017e-01 5.7827740907669067e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 26 2.6365420781075954e-03</internalNodes>

<leafValues>2.3093290627002716e-01 5.6954258680343628e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 27 -1.5127769438549876e-03</internalNodes>

<leafValues>2.7596020698547363e-01 5.9566420316696167e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 28 -1.0157439857721329e-02</internalNodes>

<leafValues>1.7325380444526672e-01 5.5220472812652588e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 29 -1.1953660286962986e-02</internalNodes>

<leafValues>1.3394099473953247e-01 5.5590140819549561e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 30 4.8859491944313049e-03</internalNodes>

<leafValues>3.6287039518356323e-01 6.1888492107391357e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 31 -8.0132916569709778e-02</internalNodes>

<leafValues>9.1211050748825073e-02 5.4759448766708374e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 32 1.0643280111253262e-03</internalNodes>

<leafValues>3.7151429057121277e-01 5.7113999128341675e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 33 -1.3419450260698795e-03</internalNodes>

<leafValues>5.9533137083053589e-01 3.3180978894233704e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 34 -5.4601140320301056e-02</internalNodes>

<leafValues>1.8440659344196320e-01 5.6028461456298828e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 35 2.9071690514683723e-03</internalNodes>

<leafValues>3.5942441225051880e-01 6.1317151784896851e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 36 7.4718717951327562e-04</internalNodes>

<leafValues>5.9943532943725586e-01 3.4595629572868347e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 37 4.3013808317482471e-03</internalNodes>

<leafValues>4.1726520657539368e-01 6.9908452033996582e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 38 4.5017572119832039e-03</internalNodes>

<leafValues>4.5097151398658752e-01 7.8014570474624634e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 39 2.4138500913977623e-02</internalNodes>

<leafValues>5.4382127523422241e-01 1.3198269903659821e-01</leafValues>

</_>

</weakClassifiers>

</_>
-<_>

Haarcascade code for Eye Detection-

-<opencv_storage>

-<cascade type_id="opencv-cascade-classifier">

<stageType>BOOST</stageType>

<featureType>HAAR</featureType>

<height>20</height>

<width>20</width>

-<stageParams>

<maxWeakCount>93</maxWeakCount>

</stageParams>

-<featureParams>

<maxCatCount>0</maxCatCount>

</featureParams>

<stageNum>24</stageNum>

-<stages>

-<_>

<maxWeakCount>6</maxWeakCount>

<stageThreshold>-1.4562760591506958e+00</stageThreshold>

-<weakClassifiers>

-<_>

<internalNodes>0 -1 0 1.2963959574699402e-01</internalNodes>

<leafValues> -7.7304208278656006e-01 6.8350148200988770e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 1 -4.6326808631420135e-02</internalNodes>

<leafValues>5.7352751493453979e-01 -4.9097689986228943e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 2 -1.6173090785741806e-02</internalNodes>

<leafValues>6.0254341363906860e-01 -3.1610709428787231e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 3 -4.5828841626644135e-02</internalNodes>

<leafValues>6.4177548885345459e-01 -1.5545040369033813e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 4 -5.3759619593620300e-02</internalNodes>

<leafValues>5.4219317436218262e-01 -2.0480829477310181e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 5 3.4171190112829208e-02</internalNodes>

<leafValues> -2.3388190567493439e-01 4.8410901427268982e-01</leafValues>

</_>

</weakClassifiers>

</_>
-<_>

<maxWeakCount>12</maxWeakCount>

<stageThreshold>-1.2550230026245117e+00</stageThreshold>
-<weakClassifiers>

-<_>

<internalNodes>0 -1 6 -2.1727620065212250e-01</internalNodes>

<leafValues>7.1098899841308594e-01 -5.9360730648040771e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 7 1.2071969918906689e-02</internalNodes>

<leafValues> -2.8240481019020081e-01 5.9013551473617554e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 8 -1.7854139208793640e-02</internalNodes>

<leafValues>5.3137522935867310e-01 -2.2758960723876953e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 9 2.2333610802888870e-02</internalNodes>

<leafValues> -1.7556099593639374e-01 6.3356137275695801e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 10 -9.1420017182826996e-02</internalNodes>

<leafValues>6.1563092470169067e-01 -1.6899530589580536e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 11 2.8973650187253952e-02</internalNodes>

<leafValues> -1.2250079959630966e-01 7.4401170015335083e-01</leafValues>

</_>

-<_>

<internalNodes>0 -1 12 7.8203463926911354e-03</internalNodes>

<leafValues>1.6974370181560516e-01 -6.5441650152206421e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 13 2.0340489223599434e-02</internalNodes>

<leafValues> -1.2556649744510651e-01 8.2710450887680054e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 14 -1.1926149949431419e-02</internalNodes>

<leafValues>3.8605681061744690e-01 -2.0992340147495270e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 15 -9.7281101625412703e-04</internalNodes>

<leafValues> -6.3761192560195923e-01 1.2952390313148499e-01</leafValues>

</_>

-<_>

<internalNodes>0 -1 16 1.8322050891583785e-05</internalNodes>

<leafValues> -3.4631478786468506e-01 2.2924269735813141e-01</leafValues>

</_>
-<_>

<internalNodes>0 -1 17 -8.0854417756199837e-03</internalNodes>

<leafValues> -6.3665801286697388e-01 1.3078659772872925e-01</leafValues>
</_>

</weakClassifiers>

</_>
-<_>

Haarcascade Code for Hand –

<?xml version="1.0"?>
-<opencv_storage>
-<haarcascade_hand type_id="opencv-haar-classifier">
<size>48 48</size>
-<stages>
-<_>
<!-- stage 0 -->
-<trees>
-<_>
<!-- tree 0 -->
-<_>
<!-- root node -->
-<feature>
-<rects>
<_>19 19 3 4 -1.</_>
<_>20 19 1 4 3.</_>
</rects>
<tilted>0</tilted>
</feature>
<threshold>-1.0453539434820414e-003</threshold>
<left_val>0.8817573189735413</left_val>
<right_node>1</right_node>
</_>
-<_>
<!-- node 1 -->

-<feature>
-<rects>
<_>19 20 3 2 -1.</_>
<_>20 20 1 2 3.</_>
</rects>
<tilted>0</tilted>
</feature>
<threshold>7.2231667581945658e-004</threshold>

<left_val>-0.6242303848266602</left_val>

<right_val>0.8814914226531982</right_val>

</_>
</_>
-<_>
<!-- tree 1 -->
-<_>
<!-- root node -->
-<feature>
-<rects>
<_>27 21 2 2 -1.</_>
<_>28 21 1 1 2.</_>
<_>27 22 1 1 2.</_>
</rects>
<tilted>0</tilted>
</feature>
<threshold>-5.3812549595022574e-005</threshold>
<left_node>1</left_node>
<right_val>-0.0745572820305824</right_val>

</_>
-<_>
<!-- node 1 -->
-<feature>
-<rects>
<_>28 16 3 2 -1.</_>

<_>29 17 1 2 3.</_>
</rects>
<tilted>1</tilted>
</feature>
<threshold>-9.1646931832656264e-004</threshold>
<left_val>0.6321527957916260</left_val>
<right_val>-0.0809556171298027</right_val>
</_>
</_>
-<_>
<!-- tree 2 -->

-<_>
<!-- root node -->
-<feature>
-<rects>
<_>19 21 2 2 -1.</_>
<_>19 21 1 1 2.</_>
<_>20 22 1 1 2.</_>
</rects>

<tilted>0</tilted>

</feature>

<threshold>1.4717580052092671e-004</threshold>

<left_node>1</left_node>

<right_val>0.8000996112823486</right_val>

</_>
-<_>
<!-- node 1 -->
-<feature>
-<rects>
<_>20 16 2 3 -1.</_>
<_>19 17 2 1 3.</_>
</rects>
<tilted>1</tilted>
</feature>
<threshold>-9.4311492284759879e-004</threshold>
<left_val>0.8731678724288940</left_val>
<right_val>-0.4287275969982147</right_val>
</_>
</_>
-<_>
<!-- tree 3 -->

-<_>

<!-- root node -->
-<feature>
-<rects>
<_>28 16 3 2 -1.</_>
<_>29 17 1 2 3.</_>
</rects>

<tilted>1</tilted>

</feature>

<threshold>-4.7741498565301299e-004</threshold>

<left_node>1</left_node>
<right_val>-0.1412670016288757</right_val>
</_>
-<_>
<!-- node 1 -->
-<feature>
-<rects>

<_>29 21 4 2 -1.</_>

<_>31 21 2 1 2.</_>

<_>29 22 2 1 2.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>-4.1668509948067367e-004</threshold>

<left_val>0.5653824210166931</left_val>

<right_val>-0.0391297787427902</right_val>

</_>

</_>

-<_>

<!-- tree 4 -->

-<_>

<!-- root node -->
-<feature>
-<rects>

<_>20 16 2 3 -1.</_>

<_>19 17 2 1 3.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>7.3959620203822851e-004</threshold>

<left_node>1</left_node>

<right_val>0.6774765253067017</right_val>

</_>

-<_>

<!-- node 1 -->
-<feature>

-<rects>

<_>21 19 1 3 -1.</_>

<_>20 20 1 1 3.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>-6.4050592482089996e-004</threshold>

<left_val>0.7523996829986572</left_val>

<right_val>-0.4327284991741180</right_val>

</_>

</_>

</trees>

<stage_threshold>-1.5993740558624268</stage_threshold>

<parent>-1</parent>

<next>-1</next>

</_>

-<_>

<!-- stage 1 -->

-<trees>
-<_>

<!-- tree 0 -->
-<_>

<!-- root node -->
-<feature>
-<rects>

<_>4 22 36 11 -1.</_>

<_>13 22 18 11 2.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>0.0675685107707977</threshold>

<left_node>1</left_node>

<right_val>0.9078469872474670</right_val>

</_>

-<_>

<!-- node 1 -->
-<feature>
-<rects>

<_>1 20 42 21 -1.</_>

<_>15 27 14 7 9.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>-0.3119403123855591</threshold>

<left_val>0.8980209827423096</left_val>

<right_val>-0.4678015112876892</right_val>

</_>

</_>

-<_>

<!-- tree 1 -->
-<_>

<!-- root node -->
-<feature>
-<rects>

<_>24 23 4 2 -1.</_>

<_>24 23 2 2 2.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>-1.1808789568021894e-003</threshold>

<left_val>-0.6527258753776550</left_val>

<right_node>1</right_node>

</_>

-<_>

<!-- node 1 -->

-<feature>
-<rects>

<_>24 23 4 2 -1.</_>

<_>24 23 2 2 2.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>-1.1808789568021894e-003</threshold>

<left_val>-0.6527258753776550</left_val>

<right_val>0.2073729932308197</right_val>

</_>

</_>

-<_>

<!-- tree 2 -->
-<_>

<!-- root node -->
-<feature>
-<rects>

<_>20 18 2 3 -1.</_>

<_>19 19 2 1 3.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>1.9879010505974293e-003</threshold>

<left_node>1</left_node>

<right_val>0.8133239150047302</right_val>

</_>

-<_>

<!-- node 1 -->
-<feature>
-<rects>

<_>19 21 2 4 -1.</_>

<_>19 21 1 2 2.</_>

<_>20 23 1 2 2.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>-6.7020917776972055e-004</threshold>

<left_val>0.7097799777984619</left_val>

<right_val>-0.4328291118144989</right_val>

</_>

</_>

-<_>

<!-- tree 3 -->

-<_>

<!-- root node -->
-<feature>
-<rects>

<_>24 23 4 2 -1.</_>

<_>24 23 2 2 2.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>1.4331060228869319e-003</threshold>

<left_val>0.0968235135078430</left_val>

<right_node>1</right_node>

</_>

-<_>

<!-- node 1 -->
-<feature>
-<rects>

<_>24 23 4 2 -1.</_>

<_>24 23 2 2 2.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>1.4331060228869319e-003</threshold>

<left_val>0.0968235135078430</left_val>

<right_val>-0.6457396745681763</right_val>

</_>

</_>

-<_>

<!-- tree 4 -->
-<_>

<!-- root node -->
-<feature>
-<rects>

<_>24 23 2 4 -1.</_>

<_>24 23 2 2 2.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>-6.9399538915604353e-004</threshold>

<left_val>-0.8354082703590393</left_val>

<right_node>1</right_node>

</_>

-<_>

<!-- node 1 -->
-<feature>
-<rects>

<_>24 23 2 4 -1.</_>

<_>24 23 2 2 2.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>6.1676127370446920e-004</threshold>

<left_val>0.4531211853027344</left_val>

<right_val>-0.6836237907409668</right_val>

</_>

</_>

-<_>

<!-- tree 5 -->
-<_>

<!-- root node -->
-<feature>
-<rects>

<_>25 19 3 6 -1.</_>

<_>26 19 1 6 3.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>-1.0453870054334402e-003</threshold>

<left_node>1</left_node>

<right_val>-0.2099598944187164</right_val>

</_>

-<_>

<!-- node 1 -->
-<feature>
-<rects>

<_>26 19 3 2 -1.</_>

<_>27 19 1 2 3.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>-6.5541139338165522e-004</threshold>

<left_val>0.5076360106468201</left_val>

<right_val>-0.1917399019002914</right_val>

</_>

</_>

-<_>

<!-- tree 6 -->
-<_>

<!-- root node -->
-<feature>
-<rects>

<_>20 18 3 8 -1.</_>

<_>21 18 1 8 3.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>1.2647550320252776e-003</threshold>

<left_node>1</left_node>

<right_val>0.7455921769142151</right_val>

</_>

-<_>

<!-- node 1 -->
-<feature>
-<rects>

<_>20 16 4 7 -1.</_>

<_>21 16 2 7 2.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>-1.6928729601204395e-003</threshold>

<left_val>0.6885066032409668</left_val>

<right_val>-0.3846471905708313</right_val>

</_>

</_>

</trees>

<stage_threshold>-0.8838403224945068</stage_threshold>

<parent>0</parent>

<next>-1</next>

</_>

-<_>

<!-- stage 2 -->
-<trees>
-<_>

<!-- tree 0 -->

-<_>

<!-- root node -->
-<feature>
-<rects>

<_>23 17 13 15 -1.</_>

<_>18 22 13 5 3.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>0.0667461231350899</threshold>

<left_node>1</left_node>

<right_val>0.9426509141921997</right_val>

</_>

-<_>

<!-- node 1 -->
-<feature>
-<rects>

<_>22 23 4 3 -1.</_>

<_>23 23 2 3 2.</_>

</rects>

<tilted>0</tilted>

</feature>

<threshold>-3.7081871414557099e-004</threshold>

<left_val>0.4906210899353027</left_val>

<right_val>-0.5502368807792664</right_val>

</_>

</_>

-<_>

<!-- tree 1 -->
-<_>
<!-- root node -->

-<feature>

-<rects>

<_>34 18 3 4 -1.</_>

<_>35 19 1 4 3.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>-2.4800749961286783e-003</threshold>

<left_val>0.6330623030662537</left_val>

<right_node>1</right_node>

</_>

-<_>

<!-- node 1 -->

-<feature>

-<rects>

<_>34 18 3 5 -1.</_>

<_>35 19 1 5 3.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>-2.7521960437297821e-003</threshold>

<left_val>0.6847578287124634</left_val>

<right_val>-0.057887702355385</right_val>

</_>

</_>

-<_>

<!-- tree 2 -->

-<_>

<!-- root node -->

-<feature>

-<rects>

<_>14 18 5 3 -1.</_>

<_>13 19 5 1 3.</_>

</rects>

<tilted>1</tilted>

</feature>

<threshold>2.9267109930515289e-003</threshold>

<left_node>1</left_node>

<right_val>0.9005092978477478</right_val>

</_>

CHAPTER 4

PROBLEM FORMATION

1. Illumination

Illumination changes the face appearance drastically. It has been found that the difference between two same faces with different illuminations is higher than two different faces taken under same illumination.

2. Pose

Facial Recognition Systems are highly sensitive to pose variations. The pose of a face varies when the head movement and viewing angle of the person changes.

It may result in faulty recognition or no recognition if the database only has the frontal view of the face.

3. Occlusion

Occlusion means blockage, and it occurs when one or other parts of the face are blocked and whole face is not available as an input image. Occlusion is considered one of the most critical challenges in face recognition system.

4. Expressions

Face is one of the most crucial biometrics as its unique features play a crucial role in providing human identity and emotions. Varying situations cause different moods which result in showing various emotions and eventually change in facial expressions.

5. Low resolution

The minimum resolution for any standard image should be 16*16. The picture with the resolution less than 16*16 is called the low resolution image. cameras can capture a small part of the human face area and as the camera is not very close to face, they can only capture the face region of less than 16*16. Such a low resolution image doesn't provide much information as most of them are lost. It can be a big challenge in the process of recognizing the face.

6. Ageing

Face appearance/texture changes over a period of time and reflect as ageing, which is yet another challenge in facial recognition system.

7. Model complexity

Existing state-of-the-art facial recognition methods rely on ‘too-deep’ Convolutional Neural Network (CNN) architecture which is very complex and unsuitable for real-time performance on embedded devic

Limitations

Face recognition is not perfect and struggles to perform under certain conditions. Ralph Gross, a researcher at the Carnegie Mellon Robotics Institute, describes one obstacle related to the viewing angle of the face: “Face recognition has been getting pretty good at full frontal faces and 20 degrees off, but as soon as you go towards profile, there’ve been problems.” Other conditions where face recognition does not work well include poor lighting, sunglasses, long hair, or other objects partially covering the subject’s face, and low-resolution images. Another serious disadvantage is that many systems are less effective if facial expressions vary. Even a big smile can render the system less effective. For instance: Canada now allows only neutral facial expressions in passport photos. There is also inconstancy in the datasets used by researchers. Researchers may use anywhere from several subjects to scores of subjects and a few hundred images to thousands of images. It is important for researchers to make available the datasets they used to each other, or have at least a standard dataset. On 18 January 2013 Japanese researchers created a privacy visor that uses nearly infrared light to make the face underneath it unrecognizable to facial recognition software.

Future Scope of the Project

In future direction of study shall focus on,

* This work can be extended to surveillance applications and monitoring the patent health conditions.
* The mining rate of video frames that match with the query text (hidden in the sliced frame) is improved by using a thread based mining. The improvement in speed is obtained as a measure.
* The mining of video frames that match with the query text (hidden in the sliced frame) needs to be improved by combining a thread based mining.
* The fall detection device can ported as hand held portable device with Wi-Fi facility. It may helps to the patient when they are in movement.

CHAPTER 5

CONCLUSION

As the necessity for higher levels of security rises, technology is bound to swell to fulfill these needs. Any new creation, enterprise, or development should be uncomplicated and acceptable for end users in order to spread worldwide. This strong demand for user friendly systems which can secure our assets and protect our privacy without losing our identity in as a of numbers, grabbed the attention and studies of scientists toward what's called biometrics.

Biometrics is the emerging area of bioengineering; it is the automated method of recognizing person based on a physiological or behavioural characteristic. There exist several biometric systems such as signature, finger prints, voice, iris, retina, hand geometry, ear geometry, and face. Among these systems, facial recognition appears to be one of the most universal, collectable, and accessible systems. Biometric face recognition, otherwise known as Automatic Face Recognition (AFR), is a particularly attractive biometric approach, since it focuses on the same identifier that humans use primarily to distinguish one person from another: their "faces". One of its main goals is the understanding of the complex human visual system and the knowledge of how humans represent faces in order to discriminate different identities with high accuracy.

REFERENCES

1. En.wikipedia.org. (2020). [online] Available at: https://en.wikipedia.org/wiki/Systems_design [Accessed 14 December2021].
2. www.softwaretestinghelp.com (2020).Available at: https://www.softwaretestinghelp.com/software-development-tools/ [Accessed 14 December2021].

3. www.incapp.learn.in

   https://learn.incapp.in/learn/home

   [Accessed 14 December2021].