

**COLLECTIVE FACE DETECTION SYSTEM**

**FOR AUTHENTICATION**

*A Project Report Submitted In the partial fulfilment of the requirements for the award of degree of*

ELECTRONICS AND COMMUNICATION ENGINEERING

By

SANJUCTA CHOUDHURY 19BEC1458

***Under the guidance of***

**DR VIJAYAKUMAR P**

**Designation: Associate Professor**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**Section I**

1. **Abstract**

Authentication plays a vital role in improving the security of any organisation. Face recognition system is the fastest way of distinguishing individuals. It mainly consists of two steps one is detection of faces in a photograph or video and second step is recognising the face. The traditional face recognition system dates back to 1980’s to 90 called Eigenfaces. Eigenface is set of eigen vectors in computer vision for face recognition.to recognise a face we figure out the eigenface it is made of. But the major drawback was if pixel positions, lightning conditions or the head positions is changed the algorithm fails.To avoid these issues we implement face recognition using deep learning which takes into account important points in an image and recognising important facial feature points.

For the purpose of our project, we have two steps one is detecting the faces for which we have used HAAR CASCADE algorithm and for recognition we will use LBPH algorithm.In haar cascade The coordinates were used to calculate 20 distances and compare the distances for each photograph, calculate the difference between the distances and return the closed records as a possible match. Whereas Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighbourhood of each pixel and considers the result as a binary number. Both the algorithms are proved to be fast and really efficient in past.The accuracy for the project is 84-90%.

**Keywords**

Computer Vision – **CV**

Artificial Intelligence- **AI**

Machine Learning- **ML**

Local Binary Pattern Histogram, - **LBPH**

Local Binary Pattern - **LBP**

# Introduction

In today’s modern world face recognition have become an integral part of each and every organisation. With increase in online transferring of information it is always important to ensure secure and genuine transfer of information from safe and valid source . Face recognition technology is not new we are using it in our daily lives. Most of us use smart phones nowadays, which often employ face recognition technology to unlock the device.

This technology provides a powerful way to protect personal data and ensure that even if the phone is stolen, sensitive data remains inaccessible by the attacker. The use of face recognition technology is being applied to an ever-expanding set of domains, including safety, security, and payments.it is also seen in social media accounts where verification of profile is done after recognising the face of the owner. Some of the benefits are listed out as below-

* Face recognition makes it easier to track down burglars, thieves and trespassers. The technology is capable of analysing the feed private and public CCTV camera networks.
* The technology is not limited to tracking down criminals. For instance, it could also make it easier to find missing children and seniors.
* Face recognition could make security checkpoints at airports less intrusive to passengers.

1. **Objective**

* The objective of this project is to implement face detection and face recognition
* To analyze the accuracy.
* To test the model on Test images and external images.

# Problem Definition and Proposed Solution

**Problem Statement**

* To perform face detection and recognize the images

**Proposed Solution**

* Using Haar Cascade classifiers to perform face detection which is an Object

Detection Algorithm used to detect faces in an image or a real time video.

* Using LBPH algorithm to perform face recognition
* Implementation using OpenCV and Keras in Python

**Section II**

1. **Related Works**

(M.Jones, 2001) This paper describes a machine learning approach for visual object detection which is capable of processing images extremely rapidly and achieving high detection rates. This work is distinguished by three key contributions. The first is the introduction of a new image representation called the "integral image" which allows the features used by our detector to be computed very quickly. The second is a learning algorithm, based on AdaBoost, which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers. The third contribution is a method for combining increasingly more complex classifiers in a "cascade" which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions.

(] J. CHAO W L, 2015)This paper provides a novel method for facial expression recognition, which distinguishes itself with the following two main contributions. First, an improved facial feature, called the expression-specific local binary pattern (es-LBP), is presented by emphasizing the partial information of human faces on particular fiducially points. Second, to enhance the connection between facial features and expression classes, class-regularized locality preserving projection (cr-LPP) is proposed, which aims at maximizing the class independence and simultaneously preserving the local feature similarity via dimensionality reduction.

**Section III**

1. **Limitations**

The Facial Recognition System is essential nowadays, and it has come a long way. Its use is essential in quite some applications, for example - Photo retrieval, surveillance, authentication/access, control systems etc. But there are a few challenges that have continuously occurred during image or face recognition system. These challenges need to be overcome to create more effective face recognition systems. The Following are the challenges which affect the ability of Facial Recognition System to go that extra mile.

**Illumination**

The illumination plays an essential role during image recognition. If there is a slight change in lighting conditions, it will make major impact on its results. It is the lighting to vary, and then the result may be different for the same object cause of low or high illumination.

**Background**

The background of the object also plays a significant role in Face detection. The result might not the same outdoor as compared to what is produces indoors because the factor - affecting its performance-change as soon as the locations change.

**Pose**

The facial recognition system is highly sensitive to pose variations. The movement of head or different camera positions can cause changes of facial texture and it will generate the wrong result.

**Occlusion**

Occlusion means the face as beard, mustache, accessories (goggles, caps, mask, etc.) also interfere with the estimate of a face recognition system.

**Expressions**

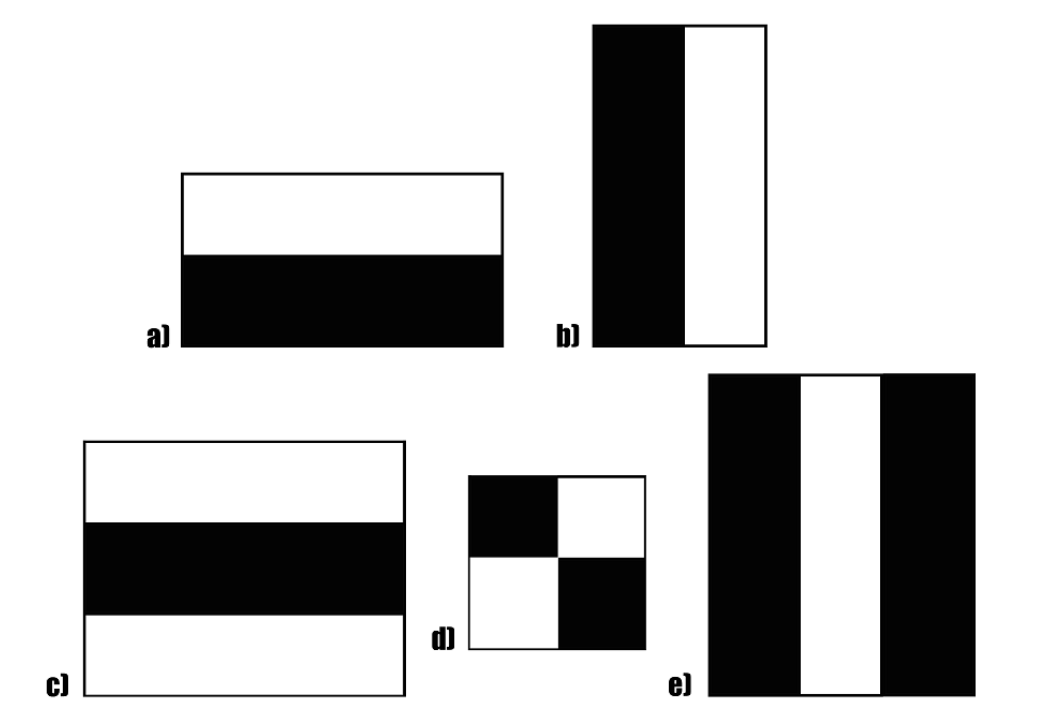
Another important factor that should be kept in mind is the different expression of the same individual. Change in facial expressions may produce a different result for the same individual.

**Section IV**

**6.Proposed algorithm for face detection.**

F**ace Detection**, a widely popular subject with a huge range of applications. Modern day Smartphones and Laptops come with in-built face detection softwares, which can authenticate the identity of the user. There are numerous apps that can capture, detect and process a face in real time, can identify the age and the gender of the user, and also can apply some really cool filters

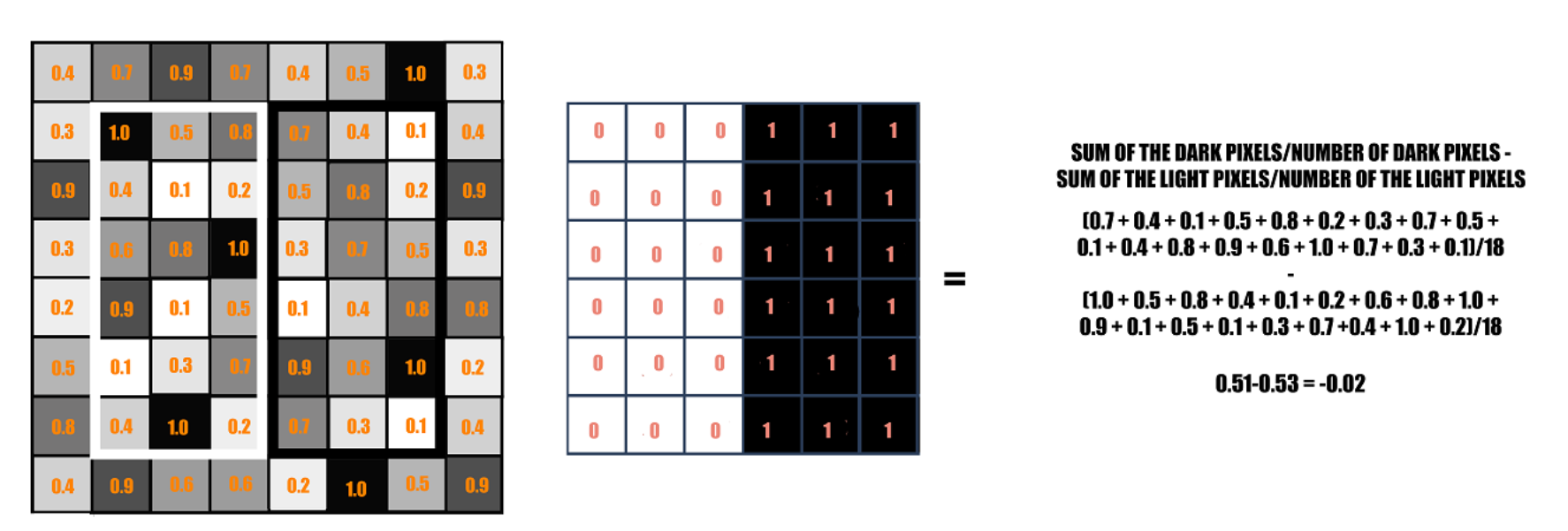
According to the paper published by viola and jones [1] machine learning approach was proposed for detecting the images using features, the method is called Haar Cascade.



**Figure 1:** A example of Haar features

In figure 1, features of different kinds are shown which are used to detect edges and lines in an image called Haar features.

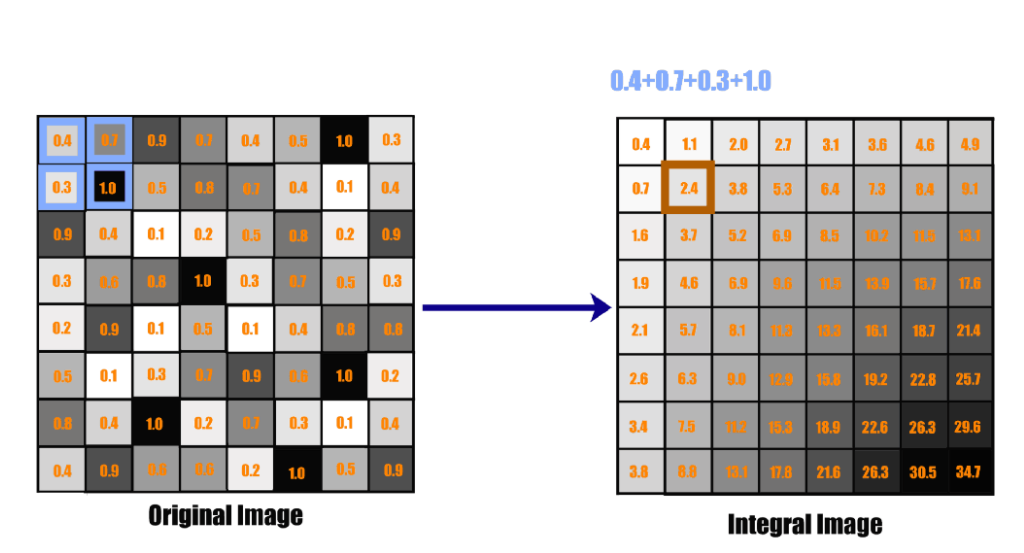
These features on the image makes it easy to find out the edges or the lines in the image, or to pick areas where there is a sudden change in the intensities of the pixels.

**Figure 2 :**The rectangle on the left is a sample representation of an image with pixel values 0.0 to 1.0. The rectangle at the center is a haar kernel which has all the light pixels on the left and all the dark pixels on the right.

In figure 2, the process is demonstrated for finding the edges and main facial features. A feature is masked over an image and sum of pixel intensities in dark and light region is calculated and subtracted from each other if the value is closer to one the image is said to have an edge.

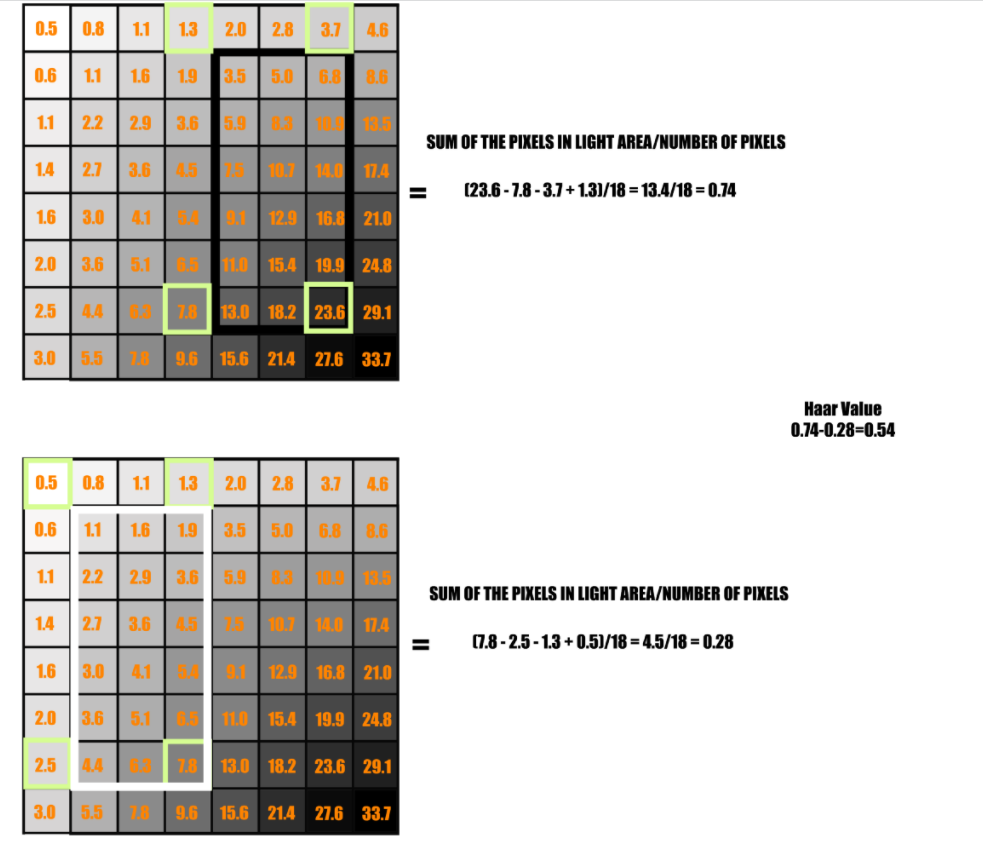
The haar features traversal on an image would involve a lot of mathematical calculations to prevent this from happening the concept of integral image is introduced.

An Integral Image is calculated from the Original Image in such a way that each pixel in this is the sum of all the pixels lying in its left and above in the Original Image. The last pixel at the bottom right corner of the Integral Image will be the sum of all the pixels in the Original Image.This concept is illustrated in figure 3.

**Figure 3:** Each pixel in an Integral image is the sum of all the pixels

in its left and above.

With the Integral Image, only 4 constant value additions are needed each time for any feature size (with respect to the 18 additions earlier). This reduces the time complexity of each addition gradually, as the number of additions does not depend on the number of pixels enclosed anymore. This is shown in figure 4



**Figure 4:** Integral Image is used here to calculate the haar value.

Haar Cascade Detection is one of the oldest yet powerful face detection algorithms invented. It has been there since long, long before Deep Learning became famous. Haar Features were not only used to detect faces, but also for eyes, lips, license number plates etc.

**2.Proposed algorithm for face recognition.**

The algorithm used for the purpose of the project is LBPH.

Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbours.



**Figure 5:** The image below shows the face recognition procedure.

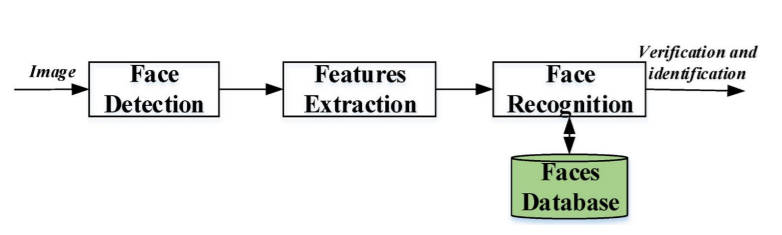
As shown in figure 5, the algorithm is as follows

1. We have a facial image in grayscale.
2. We can get part of this image as a window of 3x3 pixels.
3. It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).here the central value is taken as the threshold.
4. This value will be used to define the new values from the 8 neighbours.
5. For each neighbour of the central value (threshold), we set a new binary value. Set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.
6. Now, the matrix will contain only binary values (ignoring the central value). Concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101). the binary values (e.g. clockwise direction), but the finally we convert this binary value to a decimal value and set it to the central value of the matrix, which is actually a pixel from the original image.

At the end of this procedure (LBP procedure), a new image is obtained which represents the characteristics of the image. Using this image histograms depicted the amount of each pixel intensity is depicted.

This histogram is then compared with image that is fed to the algorithm for identification.

**Block diagram:**



**Figure 7:** The basic workflow of the project

First an image is fed for detection using the haar cascade algorithm after this important facial features are extracted using LBPH computations,the histogram obtained is compared with those in the database for comparison and identification is established.

**Dataset used**

The dataset for the project is created for recognising 2 individual images by feeding 20-30 images of each individual.

Image dataset label0🡪 INDRA NOOYI

Image dataset label1🡪Kangana Ranaut

**CODE:**

from google.colab import drive

drive.mount('/content/drive')

import cv2

import os

import numpy as np

%matplotlib inline

from matplotlib import pyplot as plt # this lets you draw inline pictures in the notebooks

import pylab # this allows you to control figure size

def faceDetection(test\_img):

    gray\_img=cv2.cvtColor(test\_img,cv2.COLOR\_BGR2GRAY)#convert color image to grayscale

    face\_haar\_cascade=cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')#Load haar classifier

    faces=face\_haar\_cascade.detectMultiScale(gray\_img,scaleFactor=1.32,minNeighbors=5)#detectMultiScale returns rectangles

    return faces,gray\_img

#Given a directory below function returns part of gray\_img which is face alongwith its label/ID

def labels\_for\_training\_data(directory):

    faces=[]

    faceID=[]

    for path,subdirnames,filenames in os.walk(directory):

        for filename in filenames:

            if filename.startswith("."):

                print("Skipping system file")#Skipping files that startwith .

                continue

            id=os.path.basename(path)#fetching subdirectory names

            img\_path=os.path.join(path,filename)#fetching image path

            print("img\_path:",img\_path)

            print("id:",id)

            test\_img=cv2.imread(img\_path)#loading each image one by one

            if test\_img is None:

                print("Image not loaded properly")

                continue

            faces\_rect,gray\_img=faceDetection(test\_img)#Calling faceDetection function to return faces detected in particular image

            if len(faces\_rect)!=1:

               continue #Since we are assuming only single person images are being fed to classifier

            (x,y,w,h)=faces\_rect[0]

            roi\_gray=gray\_img[y:y+w,x:x+h]#cropping region of interest i.e. face area from grayscale image

            faces.append(roi\_gray)

            faceID.append(int(id))

    return faces,faceID

#Below function trains haar classifier and takes faces,faceID returned by previous function as its arguments

def train\_classifier(faces,faceID):

    face\_recognizer=cv2.face.LBPHFaceRecognizer\_create()

    face\_recognizer.train(faces,np.array(faceID))

    return face\_recognizer

#Below function draws bounding boxes around detected face in image

def draw\_rect(test\_img,face):

    (x,y,w,h)=face

    cv2.rectangle(test\_img,(x,y),(x+w,y+h),(255,0,0),thickness=5)

#Below function writes name of person for detected label

def put\_text(test\_img,text,x,y):

    cv2.putText(test\_img,text,(x,y),cv2.FONT\_HERSHEY\_DUPLEX,2,(255,0,0),4)

count=0

for path, subdirnames, filenames in os.walk("/content/drive/MyDrive/trainingImages"):

    for filename in filenames:

      if filename.startswith("."):

        print("Skipping File:",filename)#Skipping files that startwith .

        continue

      img\_path=os.path.join(path, filename)#fetching image path

      print("img\_path",img\_path)

      id=os.path.basename(path)#fetching subdirectory names

      img = cv2.imread(img\_path)

      if img is None:

        print("Image not loaded properly")

        continue

      resized\_image = cv2.resize(img, (100, 100))

      new\_path="resizedTrainingImages"+"/"+str(id)

      print("desired path is",os.path.join(new\_path, "frame%d.jpg" % count))#write all images to resizedTrainingImages/id directory

      cv2.imwrite(os.path.join(new\_path, "frame%d.jpg" % count),resized\_image)

      count += 1

#This module takes images  stored in diskand performs face recognition

test\_img=cv2.imread('/content/drive/MyDrive/TestImages/1.jfif')#test\_img path

faces\_detected,gray\_img=faceDetection(test\_img)

print("faces\_detected:",faces\_detected)

test\_img2=cv2.imread('/content/drive/MyDrive/TestImages/Kangana.jpg')#test\_img path

faces\_detected,gray\_img=faceDetection(test\_img2)

print("faces\_detected:",faces\_detected)

#Comment belows lines when running this program second time.Since it saves training.yml file in directory

faces,faceID=labels\_for\_training\_data('/content/drive/MyDrive/trainingImages')

face\_recognizer=train\_classifier(faces,faceID)

face\_recognizer.write('trainingData.yml')

name={0:"indra",1:"Kangana"}#creating dictionary containing names for each label

for face in faces\_detected:

    (x,y,w,h)=face

    roi\_gray=gray\_img[y:y+h,x:x+h]

    label,confidence=face\_recognizer.predict(roi\_gray)#predicting the label of given image

    print("confidence:",confidence)

    print("label:",label)

    draw\_rect(test\_img,face)

    predicted\_name=name[label]

    if(confidence>37):#If confidence more than 37 then don't print predicted face text on screen

        continue

    put\_text(test\_img,predicted\_name,x,y)

resized\_img=cv2.resize(test\_img,(500,500))

from google.colab.patches import cv2\_imshow

cv2\_imshow(resized\_img)

name={0:"indra",1:"Kangana"}#creating dictionary containing names for each label

for face in faces\_detected:

    (x,y,w,h)=face

    roi\_gray=gray\_img[y:y+h,x:x+h]

    label,confidence=face\_recognizer.predict(roi\_gray)#predicting the label of given image

    print("confidence:",confidence)

    print("label:",label)

    draw\_rect(test\_img2,face)

    predicted\_name=name[label]

    if(confidence>37):#If confidence more than 37 then don't print predicted face text on screen

        continue

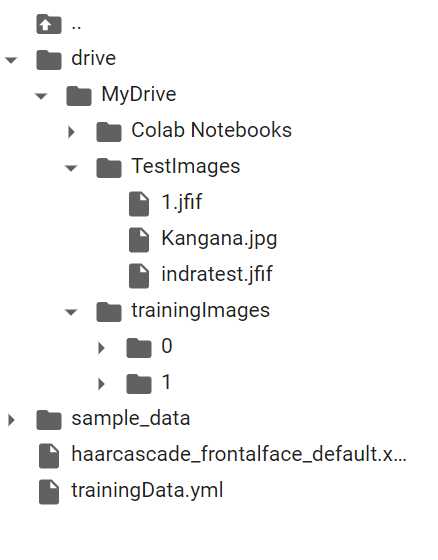
    put\_text(test\_img2,predicted\_name,x,y)

resized\_img=cv2.resize(test\_img2,(500,500))

from google.colab.patches import cv2\_imshow

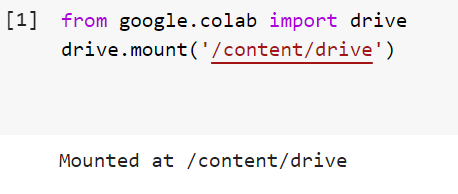
cv2\_imshow(resized\_img)

**OUTPUT SNAPSHOTS**



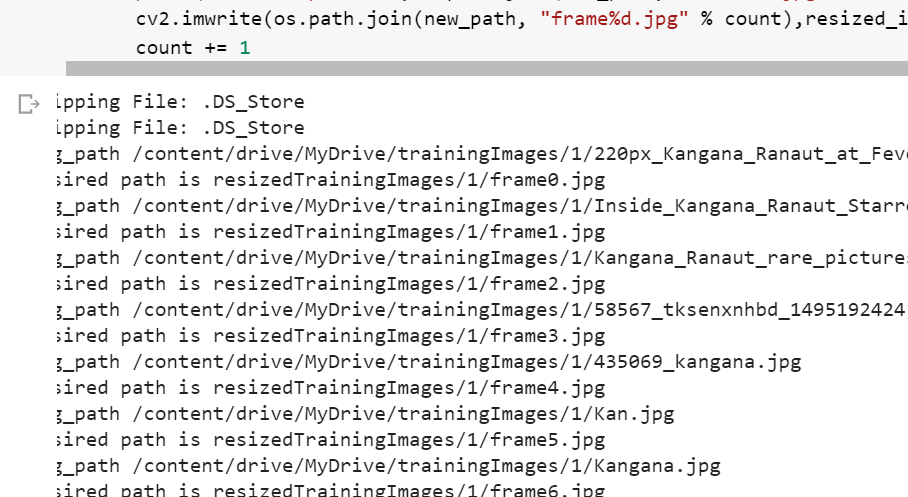
**Figure 8**

**This shows the folder structure of the project where the images in training folder are detected and recognised a model file trainingData.yml is created later the images in test dataset are tested over the model**



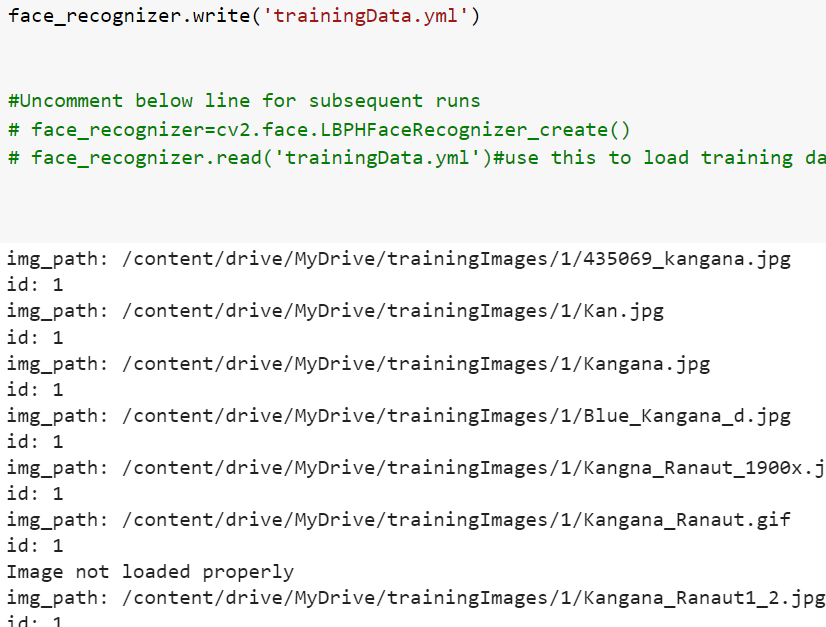
**Figure 9**

**This figure shows the successful connection of the dataset in drive and google colab**



**Figure 10**

**Above output shows the looping of each training image in the training dataset to resize it appropriately for the model**



**Figure 11**

**Looping every resized training image and applying the LBPH algorithm to create a model file trainingData.yml**

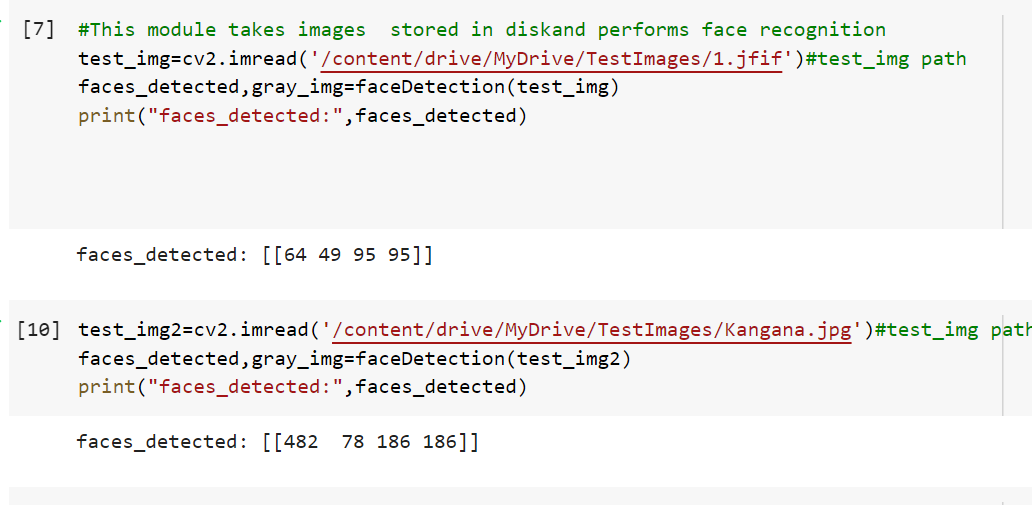


Figure 12

Faces are detected using haar cascades in training images and detected lables are stored in and array for further recognition.



Figure 13

Test image successfully detected with label0🡪indra using the trained model



Figure 13

Test image successfully detected with label1->Kangana using the trained model

# References

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517.

[2] K. Sharma, V. Gupta, S. Verma and S. Avikal, "Study and Implementation of Face Detection Algorithm Using Matlab," 2018 International Conference on Recent Innovations in Electrical, Electronics & Communication Engineering (ICRIEECE), 2018, pp. 2745-2748, doi: 10.1109/ICRIEECE44171.2018.9008990.

[3] A. Ahmed, J. Guo, F. Ali, F. Deeba and A. Ahmed, "LBPH based improved face recognition at low resolution," 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), 2018, pp. 144-147, doi: 10.1109/ICAIBD.2018.8396183.

[4] J. CHAO W L, J J DING and J Z. LIU, "Facial expression recognition based on improved local binary pattern and class-regularized locality preserving projection", Signal Processing, vol. 117, pp. 1-10, 2015.

[5] Paul, K.C. and Aslan, S. (2021) An Improved Real-Time Face Recognition System at Low Resolution Based on Local Binary Pattern Histogram Algorithm and CLAHE. Optics and Photonics Journal, 11, 63-78

[6] Varun Garg and Kritika Garg, "Face Recognition Using Haar Cascade Classifier", Journal of Emerging Technologies and Innovative Research (JETIR), vol. 3, no. 12, December 2016.

[7] Bindushree S , Rakshitha A N,” Face Recognition Using Deep Learning” International Journal of Advanced Scientific Inovation Volume 01 Issue 01, December 2020 ISSN:2582-8436 doi:10.5281/zenodo.4641691