East West University

Department of CSE

Project Report

CSE 438

Digital Image Processing

Submitted To:

Md. Mahir Ashhab

Lecturer

Department of Computer Science and Engineering

Submitted By:

Adri Saha

ID: 2019-1-60-024

Submission Date: 18 September 2022

Project title: **Face detection and recognition**

Introduction

Face recognition/detection is a digital technique based on artificial intelligence (AI) that finds and recognizes human faces from images. Face images are converted into numerical expressions using computer-generated filters used in facial recognition, which may then be compared to determine how similar they are. A person can be recognized by this system only by looking at them. It identifies, collects, stores, and evaluates facial features using machine learning algorithms so that they may be compared to images of people in a database.
Moreover, face recognition and detection are not same. There are some differences like:

Face Detection: Its objective is to find, and measure faces in an image with the aim of extracting them for the face recognition algorithm to use.
Face Recognition: The face recognition algorithm is responsible for determining the attributes that best define the image once the facial images have been retrieved, cropped, scaled, and generally converted to grayscale.

Uses:

Facial recognition is used for issuing identity documents, most frequently in combination with other biometric technology like fingerprints (preventing ID fraud and identity theft). Also, it

- Helps find missing people.
- Protects businesses against theft.
- Face match is used at border checks to compare the portrait on a digitized biometric passport with the holder's face.

Methods/algorithm which applied:

- The OpenCV approach is a popular one for face recognition. By removing the facial Haar features from the image, it first extracts the feature images from a sizable sample set.
- Then, a Face-Recognition method called LBPH (Local Binary Pattern Histogram) is utilized to identify a person's face. One of the simplest face recognition algorithms is this one. It is renowned for its effectiveness in identifying a person's face from both the front and side faces, which can represent specific local features in the photos. According to LBPH, a data set is produced by capturing images with a camera or by saving images, labeling a name or other unique identifier to the person in the image, and then adding the images to a database.

So, in this project, our task is to detect faces from several datasets of images and recognize them by their name or student ID.

Objectives

The main purpose of this study is:

- First, we must detect the face from an image using haarcascade classifier.
- Then we must train our model dataset by labeling their individual name or ID.
- After training done using LBPH algorithm, start matching with the input image with trained model.
- And then finally recognize the face.

Theoretical background

OpenCV: OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library used for computer vision in artificial intelligence, machine learning, face recognition, etc.

Haar cascade: Haar cascade is an object detection algorithm that is used to find faces in still images or moving videos. The classifier is trained using many both positive and negative images in the Haar Cascade technique, which is based on machine learning.

Positive images - These images include the images that we want our classifier to be able to recognize. **Negative images**- these are images of everything else that don't include the thing we're trying to find.

LBPH: The Face-Recognition algorithm LBPH (Local Binary Pattern Histogram) is used to identify a person's face. It is renowned for its effectiveness and for being able to identify a person's face from both the front and the side.

With an accuracy rate of 89.1%, the system can identify the required person. The accuracy will rise along with the quantity of datasets as we add more.

Confidence interval: The confidence interval estimate (CI), for both continuous and discrete variables, is a range of likely values for the population parameter based on the point estimate. The desired degree of confidence level can be between 0 and 100%.

Npy file: The Python software package with the NumPy library installed produces NPY files, which are NumPy array files. It includes an array that has been saved in NumPy (NPY) file format. All the data, including dtype and shape information, that is necessary to recreate an array on any computer is stored in NPY files.

Edge detection: Edge detection is an image-processing technique that helps to identify the edges of objects or regions inside an image. One of the most important aspects of images is their edges.

Reasons for converting grayscale image: Instead of using color images as input, several image processing and CV algorithms use grayscale images. Grayscale conversion separates the luminance plane from the chrominance planes, which is one significant factor. Also, luminance is more crucial for identifying specific visual elements in an image. All color information is removed, leaving only various shades of gray, with white being the lightest and black being the darkest. Since less information must be provided for each pixel, these images differ from all other types of color images.

Data Collection

Dataset:

1. **Foreign celebrity dataset:** Our respective faculty "Md. Mahir Ashhab" sir has collected a full image dataset for foreign actor-actresses and share with us on google drive. On that shared drive there were 2 directories for training "train" and testing "val". Each directory has 5 folders of 5 different celebrities. Their name is: "Ben Afflek"," Elton John"," Jerry Seinfield"," Madonna" and "Mindy Kaling". Similarly, on folder "val" there are also 5 folders of these 5 celebrities. So, I trained total 93 images of different celebrities dividing into 5 folders. 93 images for train+ 25 images for testing= total 118 images for foreign celebrities.

2. **Classmate dataset:** On next week, we all students have asked to share at least 25 pictures on google drive and recognize faces by their student ID. So, 5 students including me clicked pictures from different angles and shared images on google drive for data collection. We worked on that by dividing into 2 parts "train" and "val". On train there are different folders for different classmates. Each folder represents with each classmates ID. As there were 5 classmates including me who had given their own 25+ images. I split 15-20 images for train and rest images for testing on "val" folder. On "val" folder also there have 5 folders of my 5 classmates where I put classmates name in folder name. So, I worked with collection of almost 130+ images of classmates in this dataset.

3. **Bangladeshi celebrity dataset:** Students who could not upload their images for religious or personal issues they uploaded 25 images for a particular Bangladeshi celebrity like Bidya Sinha Mim, Arfan Nisho etc instead of them. And on a same way, I divided pictures for train and test. And that is how I collected 300+ images from this dataset.

Directory: I made 3 different directories inside my project folder for 3 kinds of dataset which are:

1. **ForeignCelebrities:** For foreign actor-actress dataset which is given by our faculty.
2. **Classmates**: My classmates who willingly shared their different angled images.
3. **BangladeshiCelebrities**: Classmates who shared different Bangladeshi celebrity's images.

So, in total I worked with almost 118+135+300= 553 images.

Methodology

For foreign celebrities,

1. We took 93 images of 5 different celebrities in 5 folders for train images. 93 training images generated by the algorithm using "haarcascade_frontile_default.xml" from Haar Cascades, so got 93 histograms.
2. Crops the images to pixel values and reduce the RGB images to grayscale.
3. During testing, the algorithm again creates histogram from the test images, and it compares that with the training histograms to try to get a match.
4. We took 25 images of those 5 different celebrities for testing dataset.

   Same thing did for other two datasets as well.

Implementation details

Face detection: For face detection on detection.py:

- First, I imported an image by OpenCV library or module.
- The converted it from BGR image to Grayscale image to eliminate color information and leaves only gray shades. It helps to separates the luminance plane from chrominance plane also.
- Then used haarcascade classifier to identify faces from the image. Here, I used a haarcascade.xml file from internet.

In this project, I used this haarcascade.xml file:

https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_frontalface_default.xml

- Then created a rectangle on the picture where the face will be detected. We can change the color, size of rectangle using parameters inside rectangle.
- Next, run a loop on detected_face_coordinates and fill the list of array with the cropped faces of different images.
- Lastly, returned the cropped faces and detected face coordinates.

Parameters I used: The parameters to detect Multiscale are given below. From that, we can detect coordinates of detected faces.

1. **minNeighbors:**

The parameter minNeighbors defines the minimum number of neighbors that each candidate rectangle must have to be maintained. In other words, the quality of the faces that are detected will be impacted by this attribute. A greater value results in fewer detections but of higher quality.

In this case, I fixed min neighbors = 10. So, I put the minNeighbor value higher for better quality result. If we set minNeighbors less value like 2, it shows detection on other places too.

2. **ScaleFactor:**

The scale factor is a way to compare figures with similar appearances but differing scales or measurements. The image size reduction for each scale is indicated by the scaleFactor parameter. The scale factor for the image is specified by this parameter. The image gets reduced by 5% if this value is 1.05. The image is scaled down by 10% if this value is 1.10. So here, I put scale factor 1.1 for better scaling of our image.

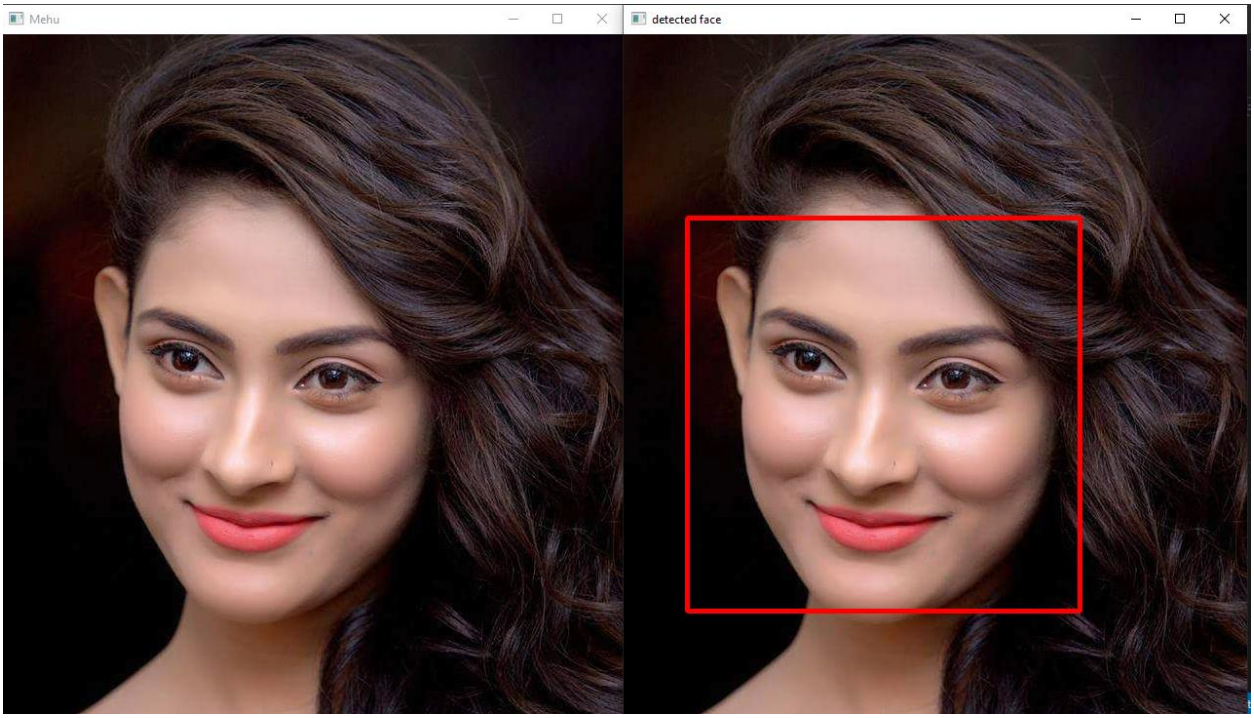

Figure 1: Original image and detected image

Training dataset: To train datasets I created training.py file.

Module:

OS: The OS module in Python has functions for adding and deleting folders, retrieving their contents, changing the directory, locating the current directory, and much more. In this case, to create directory or go through the path folders I imported this OS module.

OpenCV: OpenCV is a library for processing video and images that is used for a variety of image and video analysis tasks, including facial recognition, reading license plates, photo retouching, and advanced robotic vision.

NumPy: I used this module to create NumPy array for features and labels. It is for save the features of the object and save the labels. It is important to train the recognizer on the feature list and label list.

Implementation:

- First, I defined a list where I set those strings which folders I want to train for recognizing.
- Then read the directory 'train' where the training images are kept on some folder inside and then kept 'train' folder on 'DIR'.
- Create haar_cascade to open haarcascade.xml file.
- Then define 2 list of arrays named 'object' where we can find the features of a face like eye, nose, ear. And the other is labeling where I set the individual label for each similar face.
- Now define a function named 'training'. Here, for faces in person run a loop for path and labels.
- Check the name of folders in 'train' folder and list of labels using index.
- If the directory name match, train all the images by labeling the directory names.
- Convert images on grey to remove color information.
- Using haar_cascade detect the gray image faces on a rectangle.
- Then on rectangle_detect run a loop to fill objects and labels of grey faces.
- Finally, trained the images successfully.
- Now create another 2 numpy array 'feature_obj' and 'new_label' to save features and labels on a numpy file.
- Call LBPHFaceRecognizer algorithm to recognize faces. Pass the features and labels on it.

Recognition: To recognize images created a recognition.py file.

Here used module is also OpenCV and NumPy. Using Haar cascade classifier and LBPH algorithm images has been recognized.

Implementation:

- First, used haar_cascade.xml classifier and kept it on haar_cascade variable.
- Then define a list by filling names of labeling which we want to see by image recognition.
- Create face recognizer by calling LBPHFaceRecognizer_create() library function.
- Read trained_faces.yml by lbph_recognizer.
- Then read an image which we want to be recognized by our system.
- Then resized the image using resize function and setting parameter (500,500).
- Convert that image on grayscale.
- Then to detect the face in the image,
- send that gray_image as parameter of detectMultiScale on haar_cascade classifier and kept it on rectangle_detect.
- Run a loop on rectangle_detect and start predicting faces by lbph_recognizer.
- By that when it found the similarity of trained image put the label which write on the top and predict confidence level which shows the similarity percentages of both images.
- putText and rectangle are the command of OpenCV library from where we can set the co-ordination, font, color, thickness of text and rectangle.

- Finally show the detected image with recognized labeling.

Parameters:

Method name	Parameters	Recognition	Detection	Training
detectMultiScale	scaleFactor	1.1	1.3	1.1
	minNeighbors	4	10	10
	minSize		(30,30)	
putText	org(Coordinates of the bottom-left corner of the thext string)	(50,50)		
	font(font style)	FONT_HERSHEY_SIMPLEX		
	fontScale	1.0		
	color	(0,155,255) BGR (155,0,255) BGR		
	thickness	2		
rectangle	start_point	(x,y)		
	end_point	(x+w, y+h)		
	color	(155,0,255) BGR		
	thickness	2		
resize	Dimension pixel	(500,500)		

Table 01: Table for showing parameters I used on different methods

Output Results

Rightly recognized: Almost my machine has recognized 70% images correctly. From these, I put some of the image recognition output which gave accurate name or ID.

For foreign celebrities:

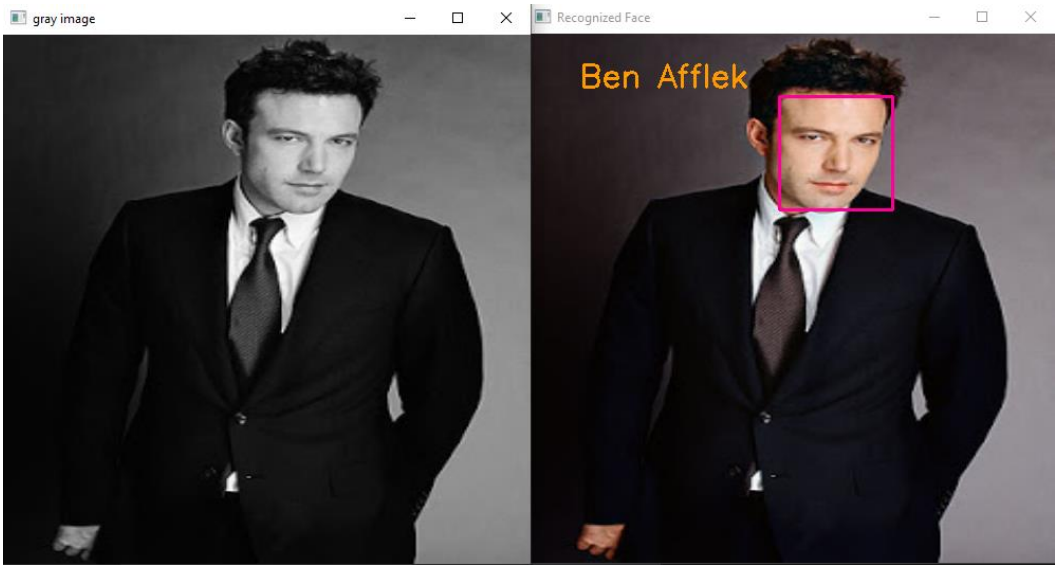

Figure 2: The person is Ben Afflek and the confidence level is: 77.59464368193228

Figure 3: The person is Jerry Seinfield and the confidence level is: 77.68801222357698

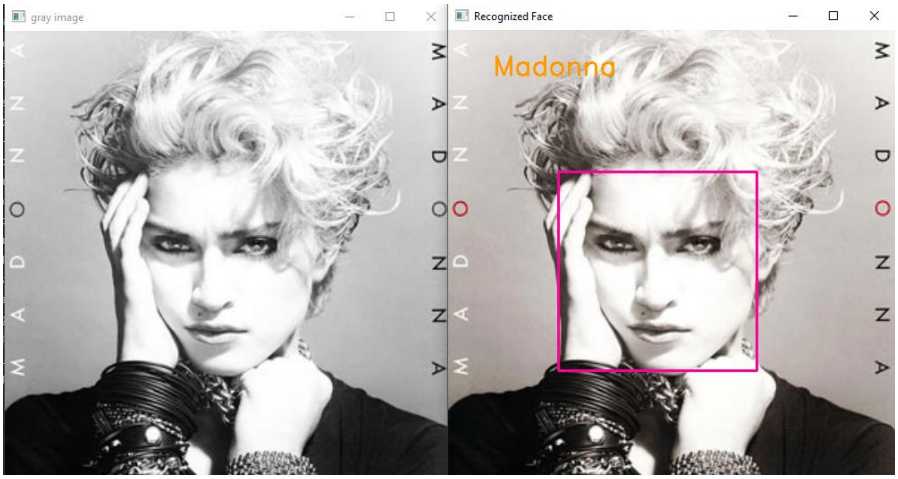

Figure 4: The person is Madonna, and the confidence level is: 87.61233864933608

Bangladeshi celebrities:

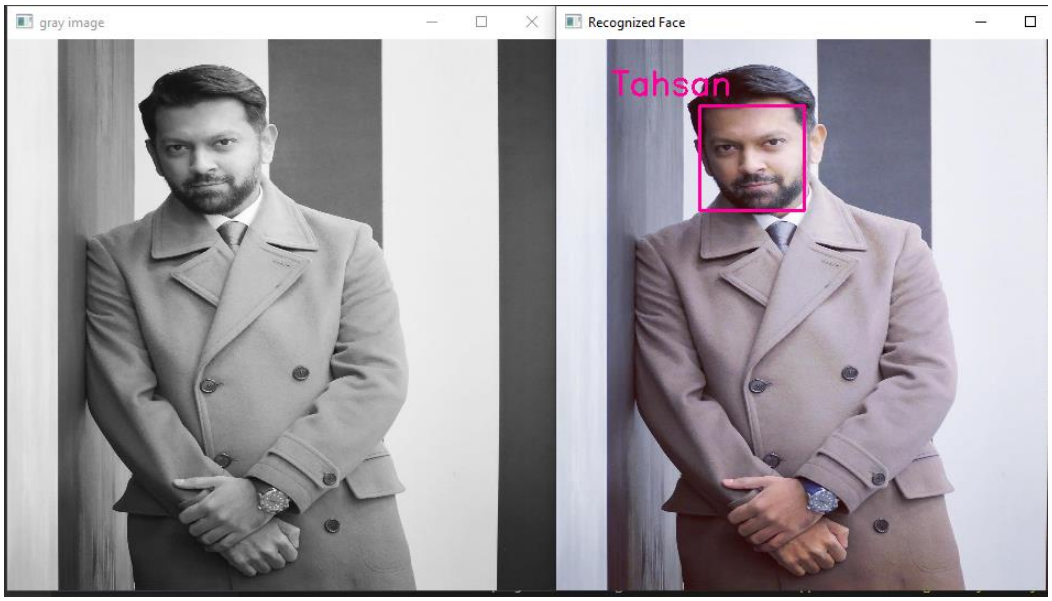

Figure 10: The person is Tahsan and the confidence level is: 90.8770396530287. So here we see the machine learned correctly and recognized Tahsan correctly.

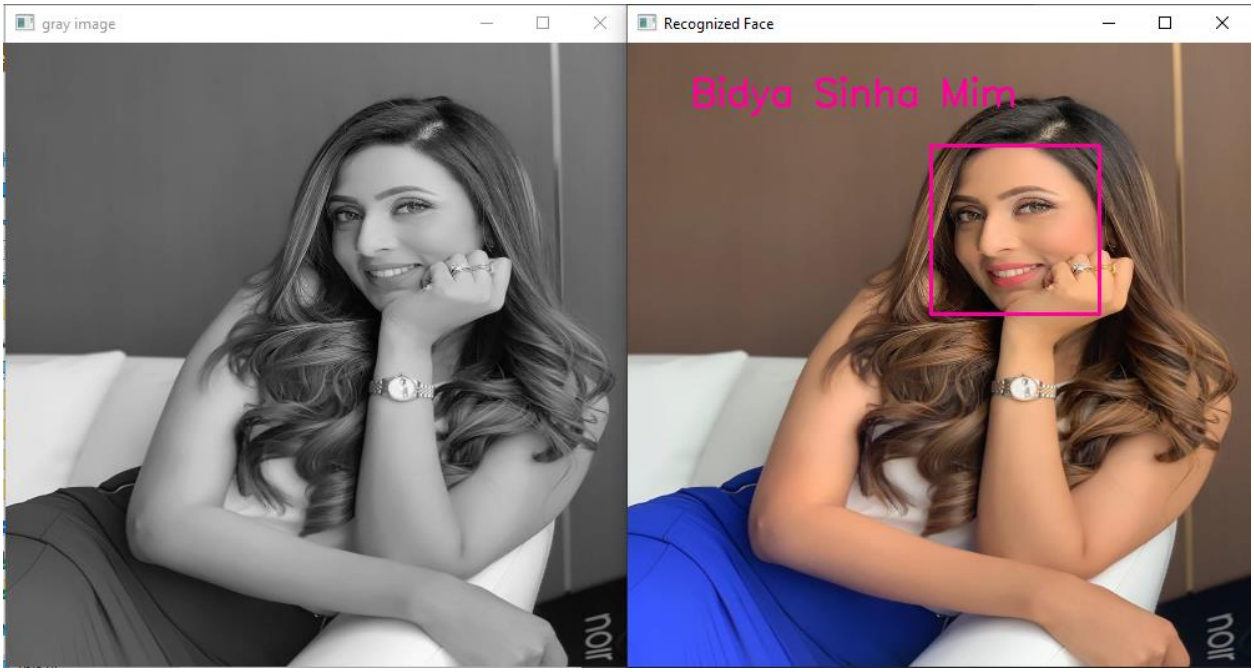

Figure 11: The person is Bidya Sinha Mim and the confidence level is: 78.94289607334734

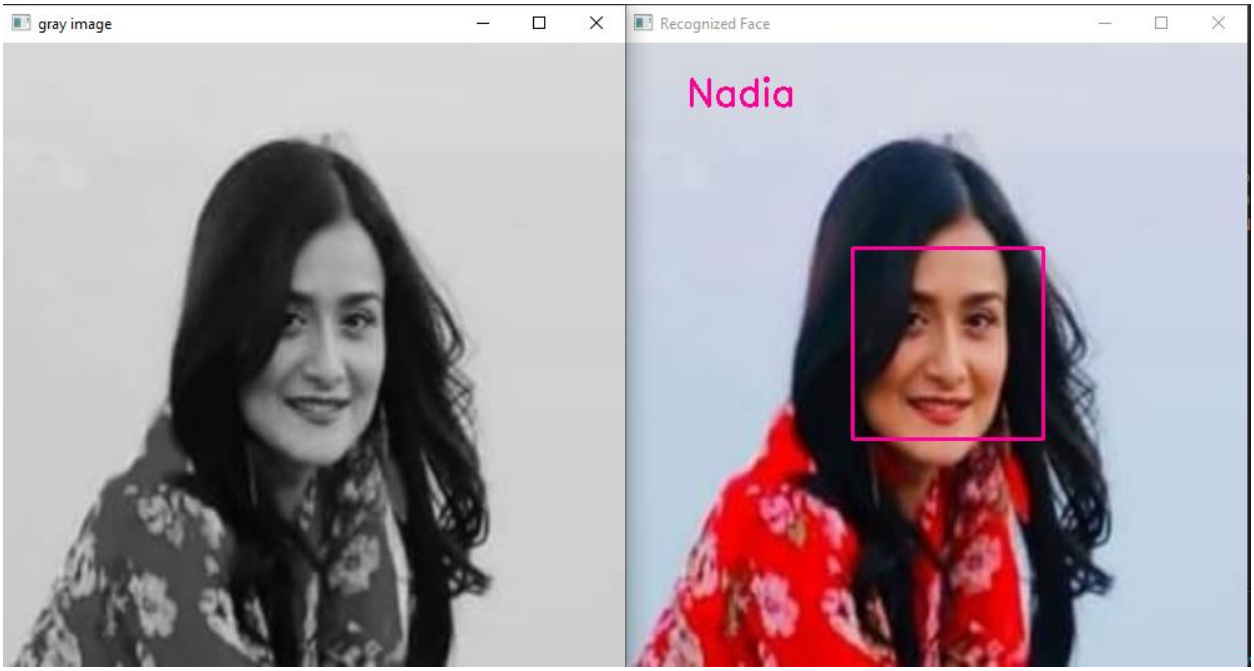

Figure 12: The person is Nadia, and the confidence level is: 83.60919635448803

Wrong recognized

For foreign celebrities:

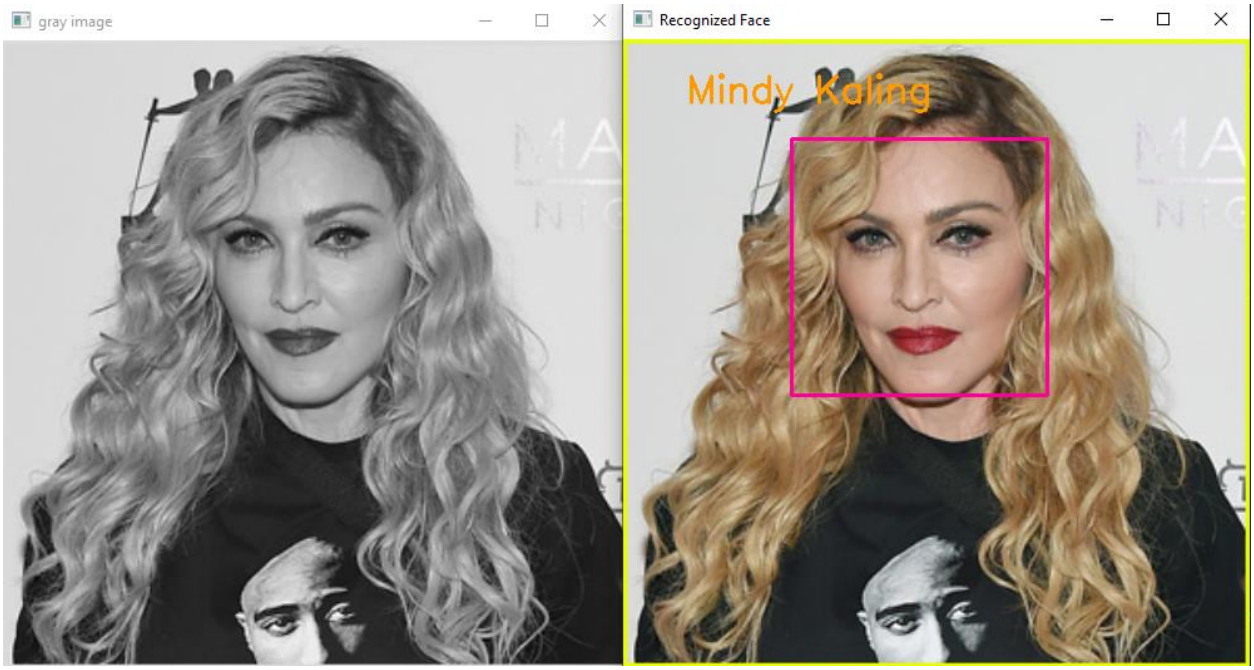

Figure 13: The person is Mindy Kaling, and the confidence level is: 66.24513926215973 which is wrong. She is Madonna. For similarities in histogram of Madonna and Mindy Kaling images it showed this wrong labeling.

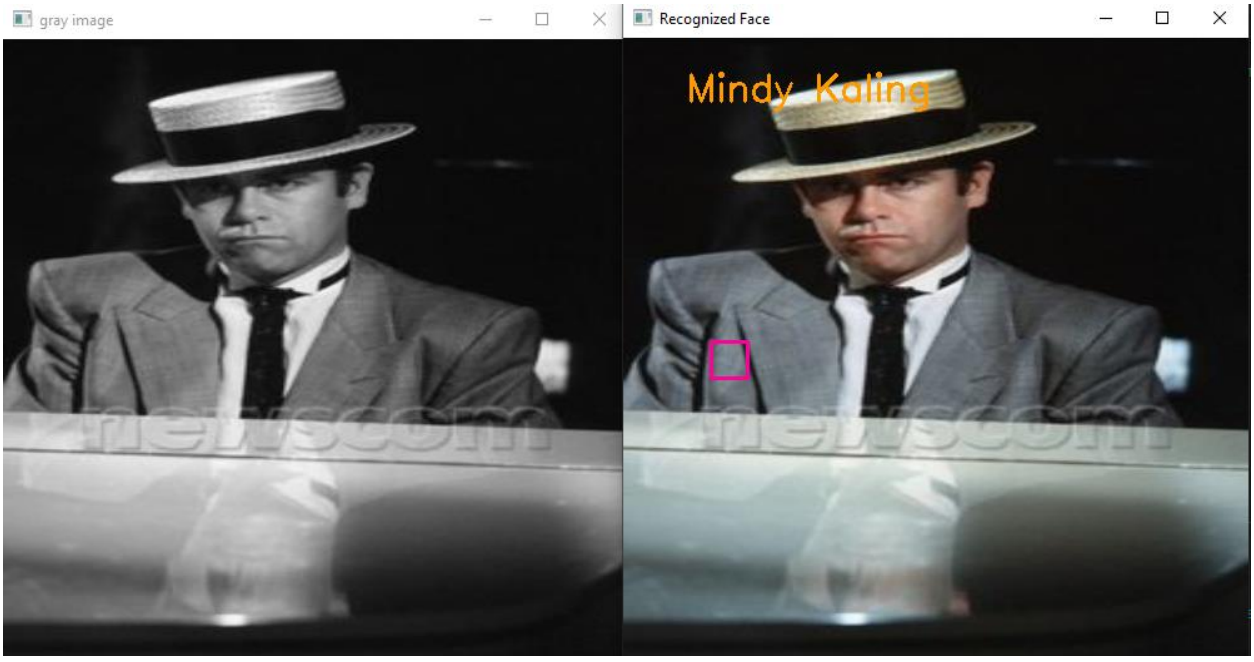

Figure 14: The person is Mindy Kaling, and the confidence level is: 182.04633330287555. Here the person is Elton john, but the system is showing Mindy Kaling. We also see the CI is 182 which is also not possible. So, there is an error.

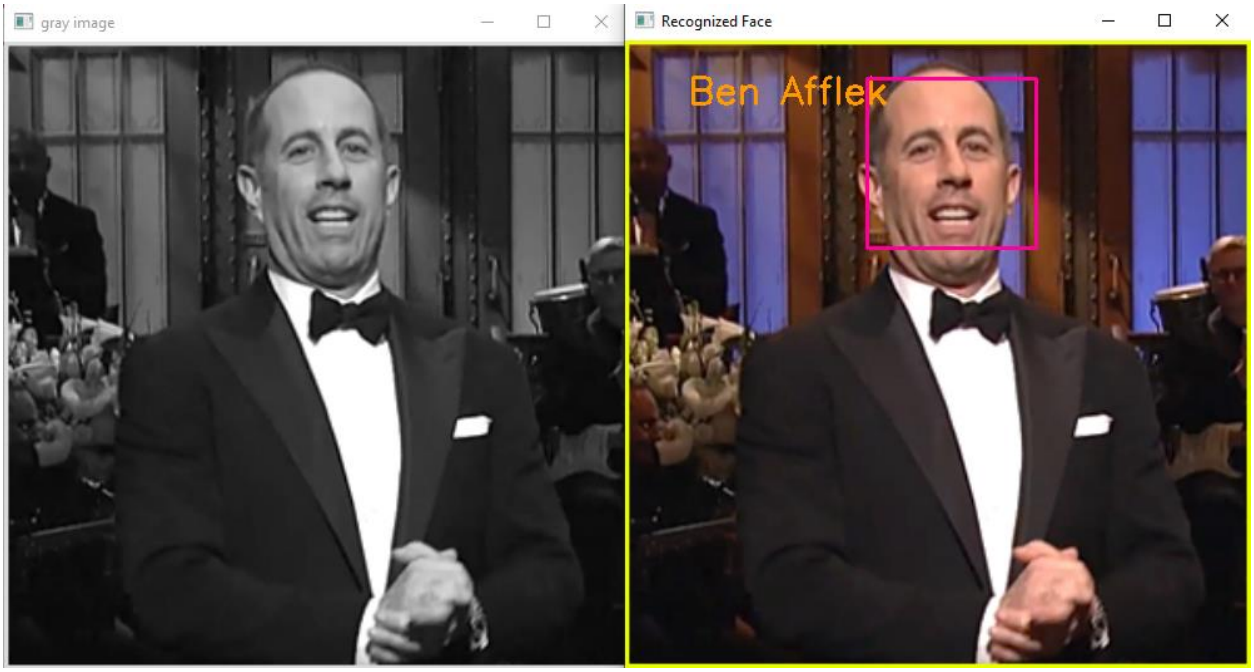

Figure 15: Output shows, “The person is Ben Afflek and the confidence level is: 83.12533254601661” But the person is Jerry Seinfeld.

For Bangladeshi celebrities:

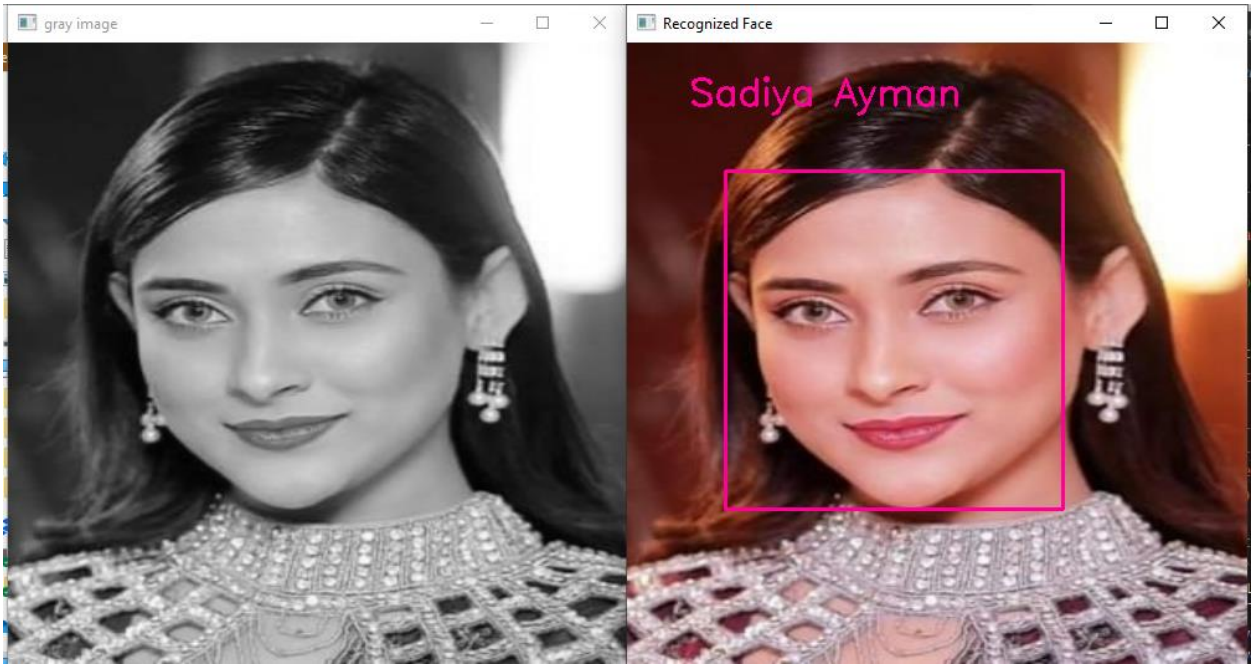

Figure 19: Output shows, “The person is Sadiya Ayman and the confidence level is: 55.92895148914356.” But she is Mehazabien Chowdhury. So, gives us incorrect result.

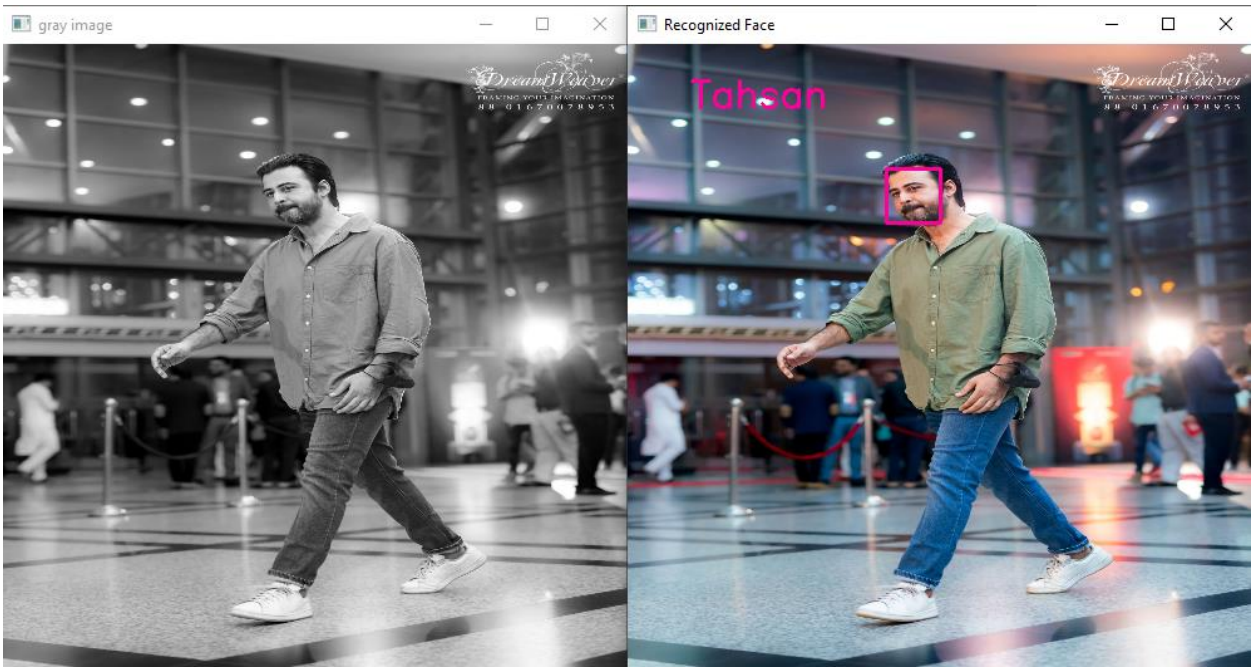

Figure 20: Output shows The person is Tahsan and the confidence level is: 151.6655601376805. But this is wrong. He is Afran Nisho. We can also see the confidence level is much higher than 100. And as the angled position is not very clarified on this image, machine recognized wrong person.

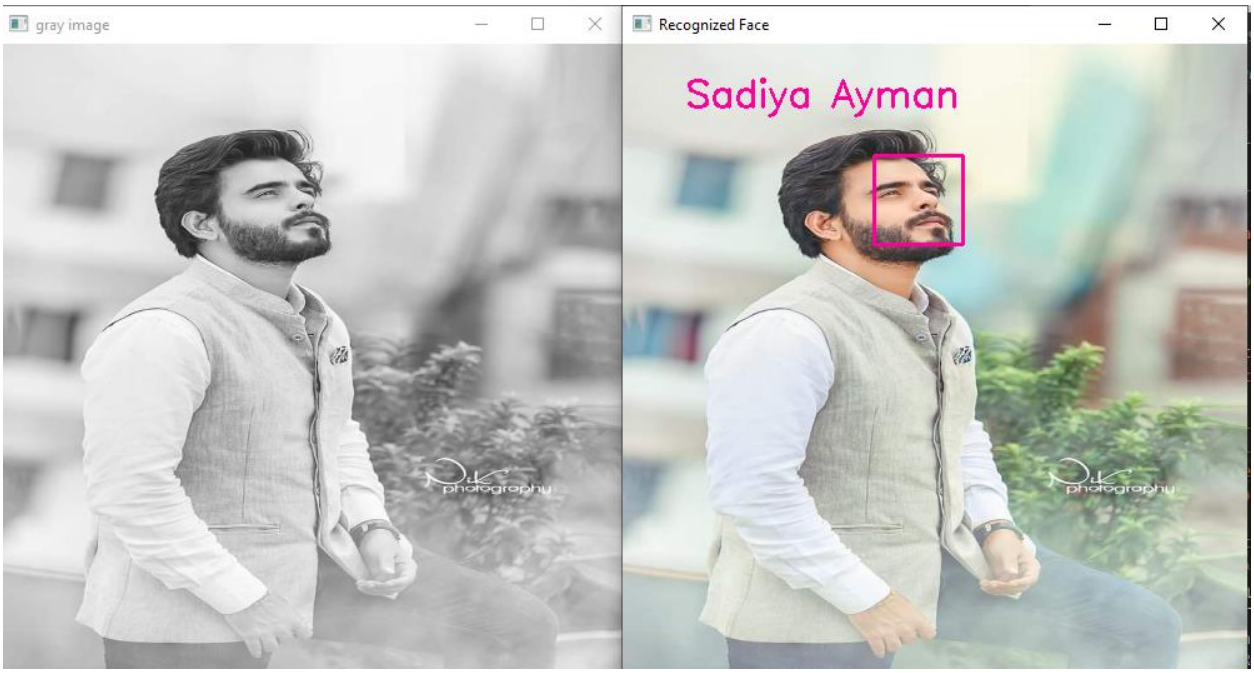

Figure 21: Output shows the person is Sadiya Ayman and the confidence level is: 130.6335089183289. But he is Siam Ahmed. Confidence interval is also greater than 100 which isn't possible. So, system has mismatched between them.

Analysis

Some experimental results for some of the images are given below:

Confidence Interval		Rightly recognized	Wrongly recognized
	Foreign celebrities	Ben Afflek with CI 77.59%	Mismatched with Madonna and Mindy kaling, CI: 66.245%
		Jerry Seinfield with CI 77.68%	Mismatched with Elton John and Mindy kaling, CI: 182.04%
		Madonna with CI 87.612%	Mismatched with Jerry Seinfeld. and Ben Afflek. CI: 83.125%
	Bangladeshi celebrities	Tahsan with 90.87%	Mismatched with Mehazabien Chowdhury and Sadiya Aman. CI: 55.92%
		Bidya Sinha Mim with 78.94%	Mismatched Afran Nisho with Tahsan. CI: 151.66%
		Nadia with CI 83.61%	Mismatched Siam Ahmed with Sadiya Ayman. CI: 130.63%

Table 02: Showing some of the correct and incorrect recognition output with their confidence interval

I took almost all the images as input and check is it recognizing correctly or not. Almost 70% images were recognized successfully correctly. There were several mismatched on Bangladeshi celebrities. This could be happened for image quality, similarity on histogram, angle/pose similarity, confidence level.

Discussion

On these images, we see there are different types of images. Like, from different angle images, different size images, multiple profile images, dark image, light images. In detection, sometimes the rectangle appears in wrong places. By increasing the number of minNeighbors it was solves. Moreover, in results, we see some wrongly recognized output. So, our machine couldn't recognize 100% accurately. Reasons for showing wrong image: Predicting wrong face because of image histogram analysis. Images which have similarities on histogram and grayscale color these are mixing up their labels and so predict wrong name.

Conclusion

First, I want to thank to our honorable faculty MD. Mahir Ashrab Sir to give us this very important project topic of face recognition. Facial recognition is a technology that has the potential to assist society by increasing safety and security, lowering crime, and eliminating human interaction. Face recognition is important in many fields like security, caught thief, mobile unlock, biometric registration etc. It has become more important in our daily life. By doing this project we see some false values for some images. Because of histogram analysis after grayscale conversion is similar with other person images, my machine could not detect accurate results for all images. We see some of these cases confidence intervals are giving 100+ value which is ignoring the rule. As we know interval of confidence is 0 to 100. So, on these cases, images are wrongly recognized. It may be solved by using more higher algorithms and changing the value of parameters. Except some images otherwise, my machine is working well.