Face Detection & Face Recognition

Artificial Intelligence Final Project Report

Mr. Raymond Kosala / 18 January 2018

Team Members:

➢ Giorgio Fasolini	(1901498082)
➢ Luffandri	(1901498321)
➢ Ramada Atyusa	(1901498763)

Table of Contents

1. Introduction

1.1. Background

- Artificial Intelligence (AI), once only exist in the science fiction, is now implemented in our daily life. GPS, maps that compute the optimal driving routes, recommendations feature, and even email spam filters are rely on a form of AI. In recent years, we are becoming more and more relying on technologies around us which makes our life easier. This lead to our main problem which is to detect and recognize photos that are hard for us to see it clearly.

1.2. Problem Description

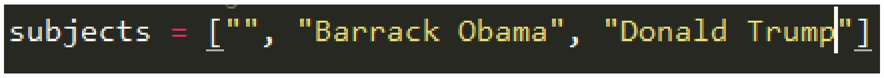
- We found out that memory is volatile, because of that we keep our best memory safe and to ensure that, we sometimes take pictures of it to remind us of that memory. Sometimes pictures could get old and changed the color of the picture. Sometimes, we took photos with low light intensity or contrast. Because of that, it might be hard for us to recognize who is in the picture. It might be you or your sister or your brother who is like you.

1.3. How to Run

1. Open folder “training-data” and make folder s1, s2, …., and so on. And input 1 person to one folder, For example: Images of Barrack Obama in S1 and Donald Trump in S2.

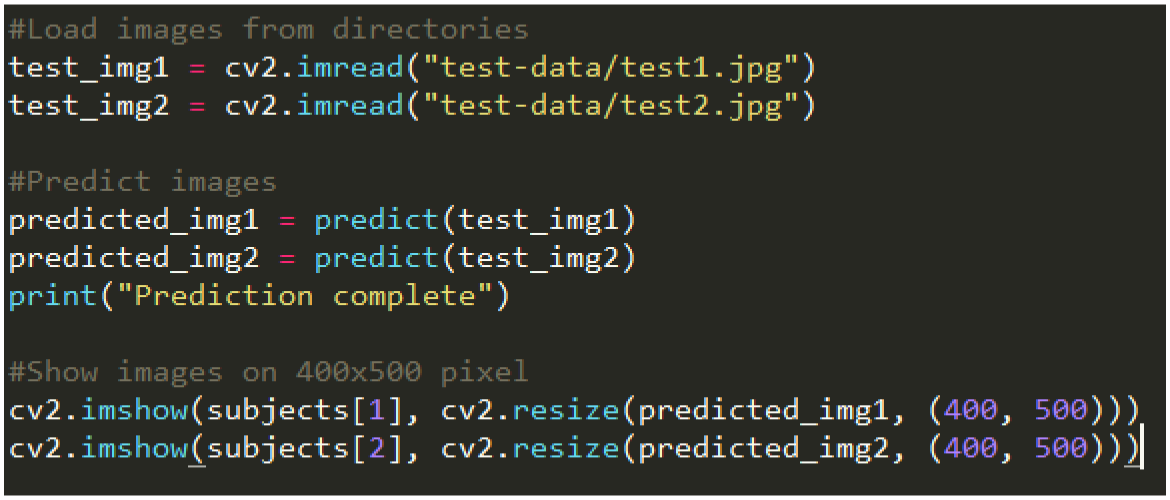
2. Open folder “test-data” and put an image to test the face recognition. For example, form data number 1, S1 is Barrack Obama so in “test-data” folder rename the image barrack Obama become “test1” and Donald Trump become “test2”.

3. Open “finalproject.py” in sublime and add a name to the labels.



```
subjects = ["", "Barrack Obama", "Donald Trump"]
```

4. Declare the training-data, test-data, and show data in the program.



```
#Load images from directories
test_img1 = cv2.imread("test-data/test1.jpg")
test_img2 = cv2.imread("test-data/test2.jpg")

#Predict images
predicted_img1 = predict(test_img1)
predicted_img2 = predict(test_img2)
print("Prediction complete")

#Show images on 400x500 pixel
cv2.imshow(subjects[1], cv2.resize(predicted_img1, (400, 500)))
cv2.imshow(subjects[2], cv2.resize(predicted_img2, (400, 500)))
```

5. Open CMD where the program is there and then type “python finalproject.py” and then enter. Wait the program to train and it will show the result.

2. Technical Specification

2.1. Solution Features

- This program uses two algorithms to recognize pictures of people. First, the Local Binary Pattern (LBP) to detect face in a picture by using algorithm that can detect whether the picture consist of face or other things. Second, the Local Binary Pattern Histograms (LBPH) to recognize face in a picture by training pictures of the same person with many different pictures. Through these algorithms, face recognition and detection can work even if some pictures have different contrast and brightness from the surrounding lights.

- Face detection and recognition can help us to detect and recognize faces in photos even with different contrast or brightness. This could help us to detect and recognize faces that are hard for human to see with their bare eyes.

2.2. Solution Design Architecture

- To build this project using the LBP and LBPH algorithms, we need Open CV. It consists of several algorithms to detect faces such as Haar and recognize faces such as Eigen and Fisher. However, we choose to use the LBP algorithm for our face detection and LBPH algorithm for our face recognition. LBP algorithm detect faces in grayscale image, and create 3x3 windows with 9 pixels. This process makes it faster than Haar algorithm, which is important because we train around 350 photos for each people and the accuracy for LBP algorithm is not that different with Haar algorithm.

LBPH algorithm uses histogram to save each people's face and it will compare the histogram with other histogram to find the best match.

We are training our program with famous people faces because it is easier to find their face datasets in the internet. For each person, we use around 350 photos to train and we pick one different photo to be the test data.

In testing the program, we do the normal training first, then we test other things such as changing the contrast/brightness/hue/saturation for the training data or the test data to see if our program still can detect and recognize faces with different surrounding colors.

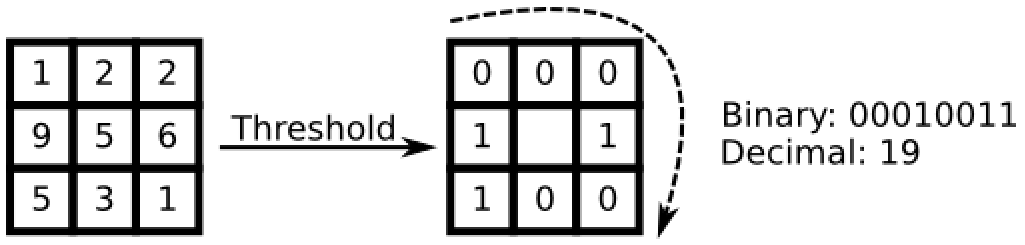
We train our data using Asus ROG with NVidia GeForce GTX 950M and i7 4720HQ@2.6 ghz and it is recommended to use the similar or better GPU and CPU to train faster.

2.3. How the Algorithms Work

- There are 3 libraries involved in this project, they are OpenCV (For Machine Learning), OS (For folder paths directory), and numpy (For Dimensional Array & Matrix). Furthermore, there are 2 more algorithms that are used for Face detection & Face recognition. The details are:

1. Local Binary Pattern (For face detection)

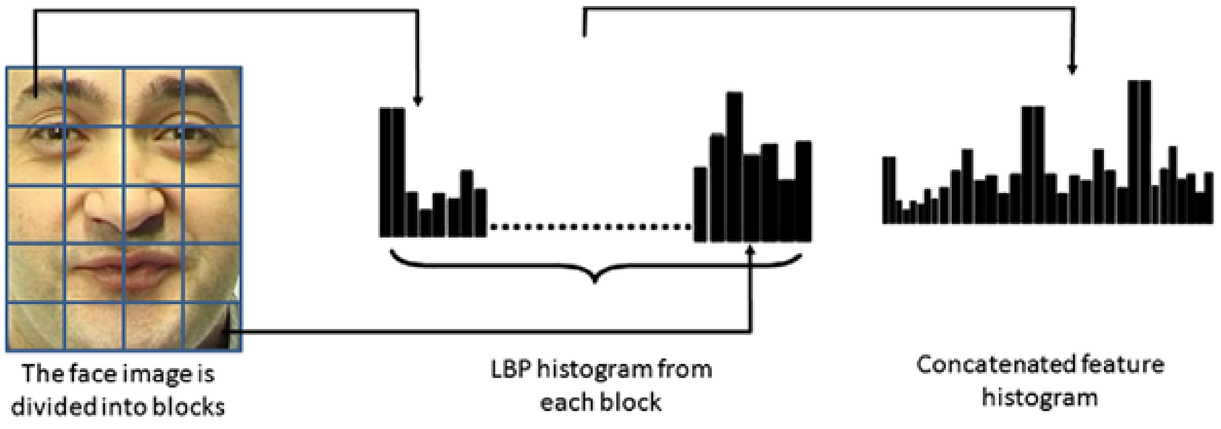
- This algorithm works by comparing images to the library, and converting them into binary code which will later be stored in numpy array.



As can be seen, the images are converted into into dimensional matrix (Left image), with center of the matrix as key. Those which is less than a center is labeled 0 and greater than center is labeled 1. After that the binary number is obtained by looping clockwise, and thus are stored in numpy array

2. Local Binary Pattern Histogram (For face recognition)

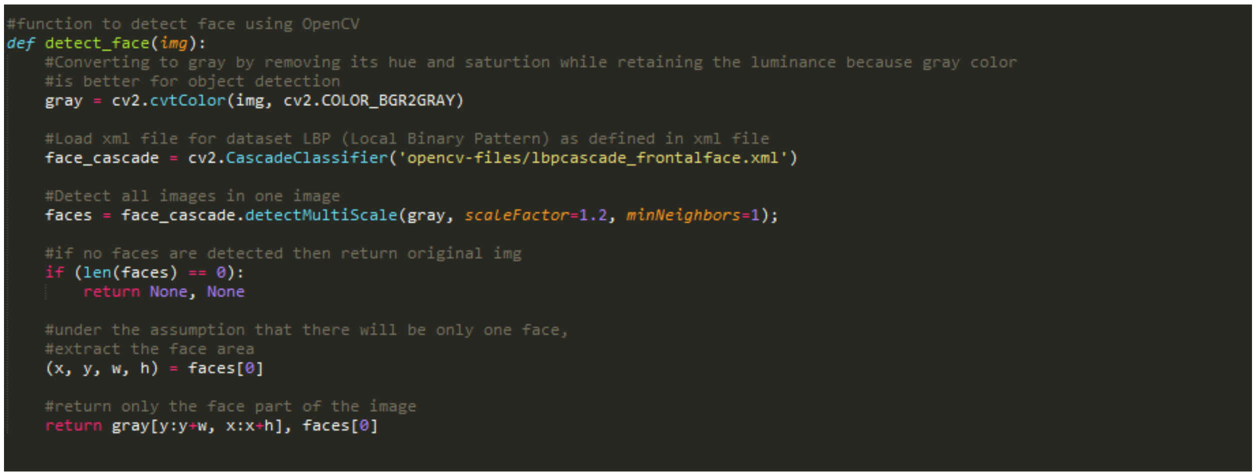
- This algorithm works by dividing images in a few blocks, and then for each block the unique numpy arrays are converted into histogram charts, at the end of the training the closest test data to the image histogram will be selected as the result


The face image is divided into blocks
LBP histogram from each block
Concatenated feature histogram

As can be seen, each block which has unique numpy arrays are converted into histogram chart, which later will be combined as one, therefore one image will have one unique histogram. At the end the closest chart to the test data will be recognized and selected as a result.

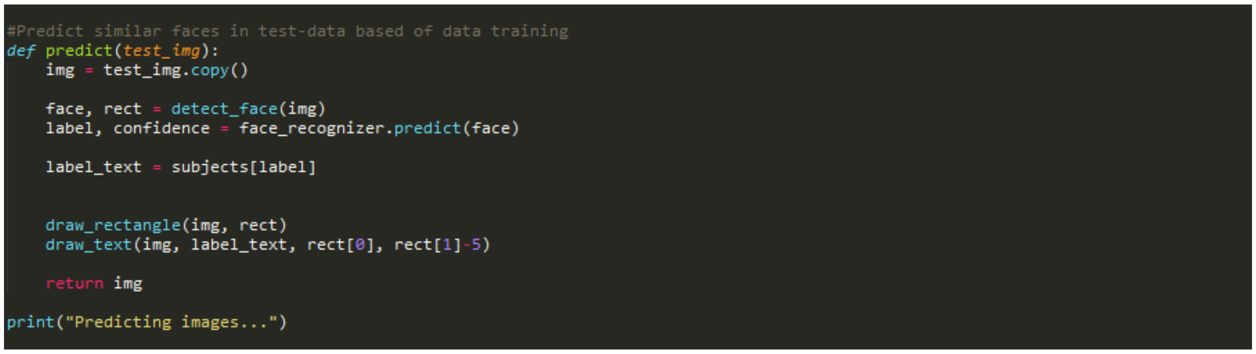
3. Program Manual

3.1. Code



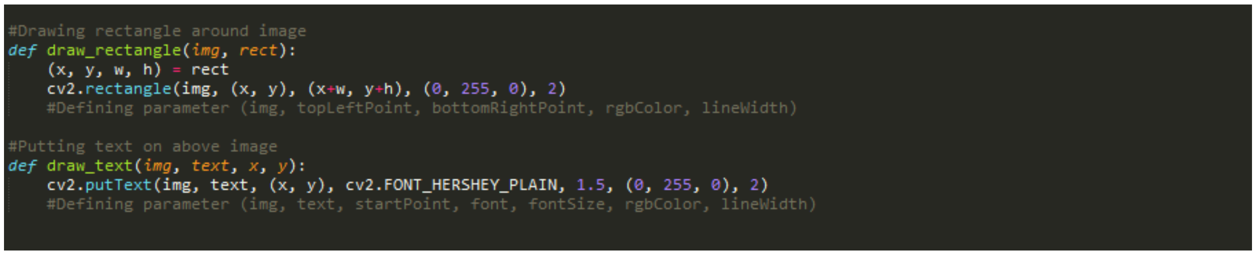
```python
#function to detect face using OpenCV
def detect_face(img):
    #Converting to gray by removing its hue and saturtion while retaining the luminance because gray color
    #is better for object detection
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    #Load xml file for dataset LBP (Local Binary Pattern) as defined in xml file
    face_cascade = cv2.CascadeClassifier('opencv-files/lbpcascade_frontalface.xml')

    #Detect all images in one image
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=1);

    #if no faces are detected then return original img
    if (len(faces) == 0):
        return None, None

    #under the assumption that there will be only one face,
    #extract the face area
    (x, y, w, h) = faces[0]

    #return only the face part of the image
    return gray[y:y+w, x:x+h], faces[0]
```

Face-detection (1)



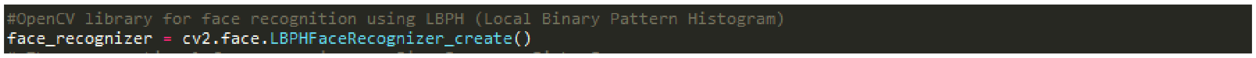
```python
#Predict similar faces in test-data based of data training
def predict(test_img):
    img = test_img.copy()

    face, rect = detect_face(img)
    label, confidence = face_recognizer.predict(face)

    label_text = subjects[label]


    draw_rectangle(img, rect)
    draw_text(img, label_text, rect[0], rect[1]-5)

    return img

print("Predicting images...")
```

Predicting images (2)



```python
#Drawing rectangle around image
def draw_rectangle(img, rect):
    (x, y, w, h) = rect
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
    #Defining parameter (img, topLeftPoint, bottomRightPoint, rgbColor, lineWidth)

#Putting text on above image
def draw_text(img, text, x, y):
    cv2.putText(img, text, (x, y), cv2.FONT_HERSHEY_PLAIN, 1.5, (0, 255, 0), 2)
    #Defining parameter (img, text, startPoint, font, fontSize, rgbColor, lineWidth)
```

Drawing rectangle & text (3)

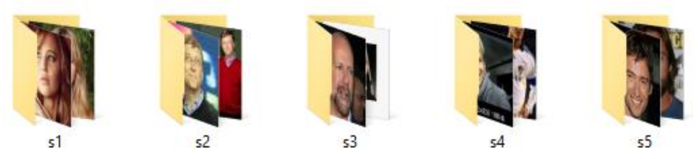


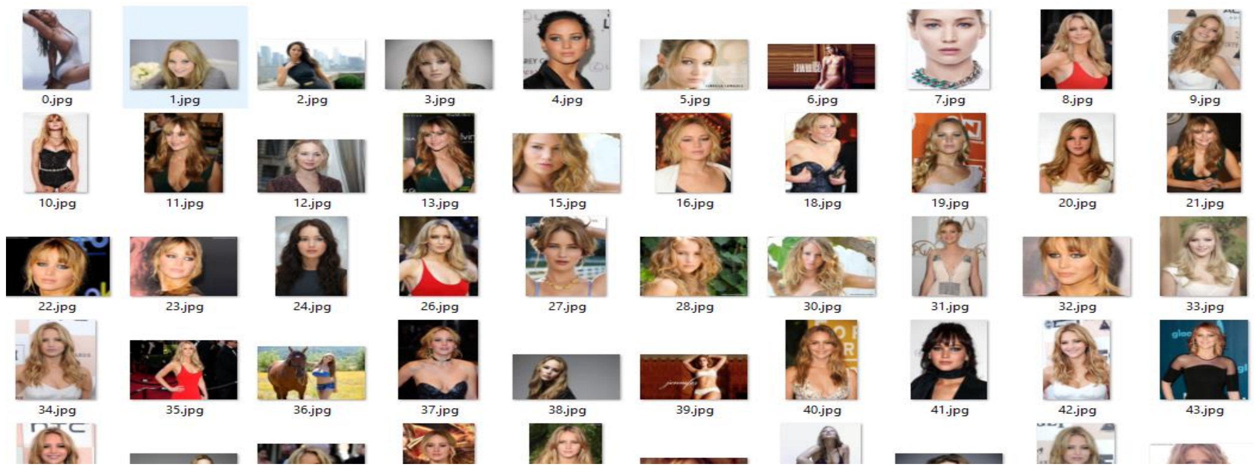
```python
#OpenCV library for face recognition using LBPH (Local Binary Pattern Histogram)
face_recognizer = cv2.face.LBPHFaceRecognizer_create()
```

Face-recognition (4)

3.2. Dataset Training

- We tested this program with 5 labels of celebrity names which are "Jennifer Lawrence", "Bill Gates", "Bruce Willis", "David Beckham", and "Hugh Jackman". Each of them consist of around 350 images of their faces to identify their own face. After we tested several times with different number of datasets, the program can identify the right faces of the celebrities.

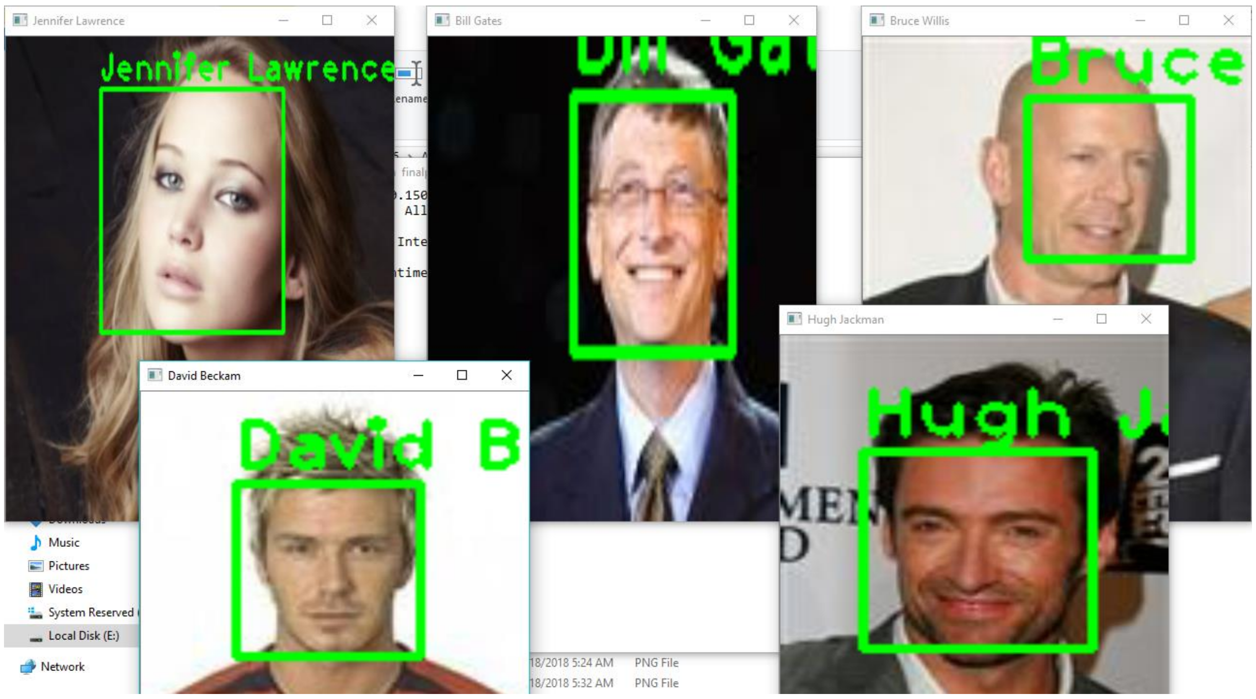




4. Testing

4.1. Testing Result (Normal)

- Here is the result of our training that we present last week on the first presentation. We tested several celebrities and the program can detect and recognize correctly all the celebrities and famous people. We also found out that the program cannot identify the face and label the name correctly if the datasets is insufficient. We tried testing "Jennifer Lawrence" with only five training datasets and the program error.



4.2. Testing Result (with experimental)

4.2.1. Training data

- In this section, we train using default testing data, but with different training data. We changed the properties of the training data (Hue, Brightness, contrast, etc.).

- We also count the accuracy using this formula:
  (Detected Faces / Total Valid Data) * 100

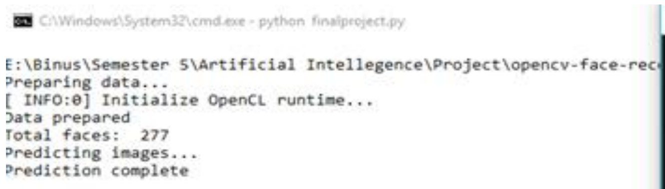
Celebrity face : Jennifer Lawrence

Total face image: 352 Faces (277 valid)

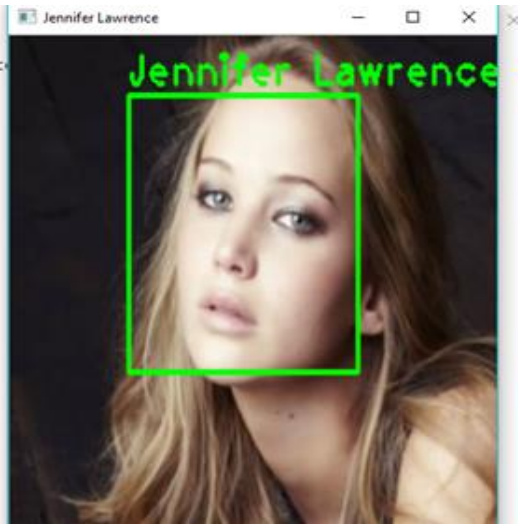
I. Default Train Data

Face Detection = 277 faces, Accuracy = 100%

Face Recognition = ✓



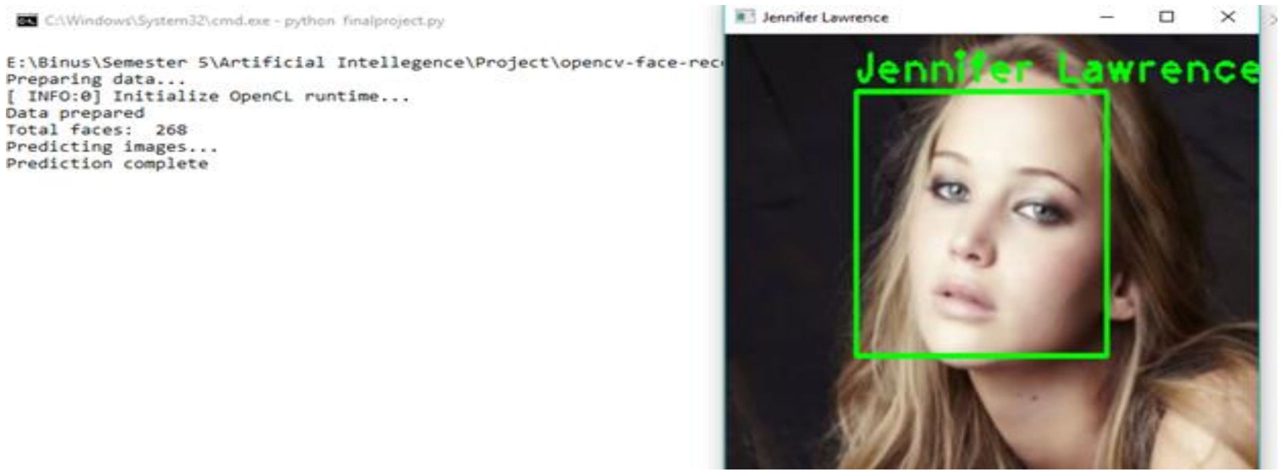
```
C:\Windows\System32\cmd.exe - python finalproject.py

E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-rec
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  277
Predicting images...
Prediction complete
```



II. Brightness +150

Face Detection = 268 Faces, Accuracy = 96,75%

Face Recognition = ✓

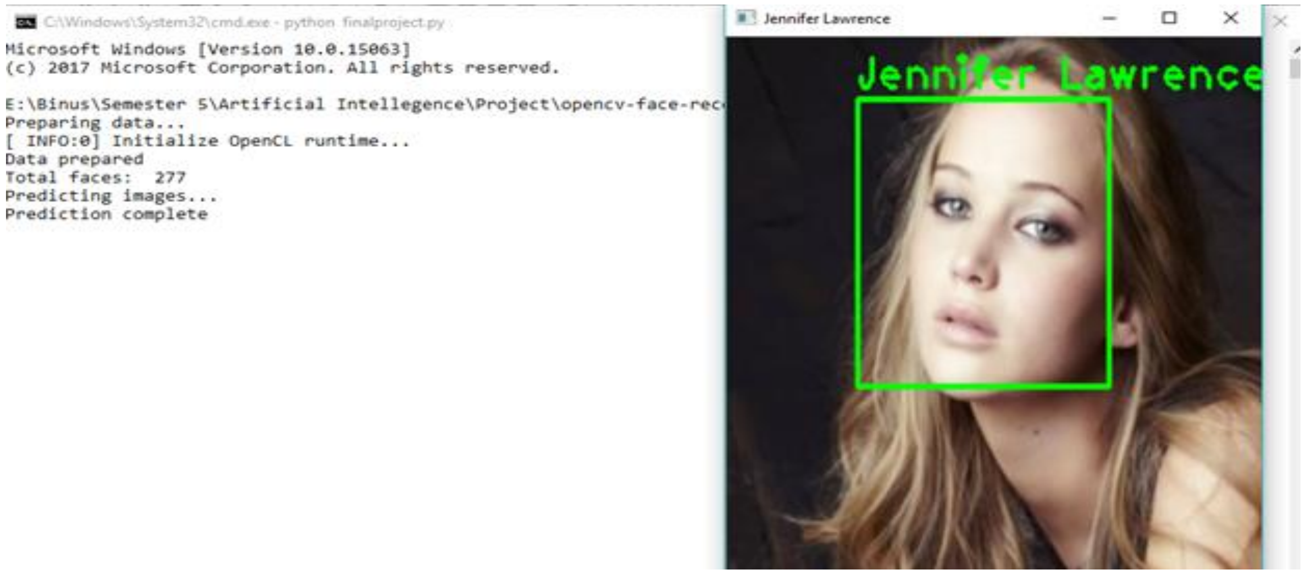


```
C:\Windows\System32\cmd.exe - python finalproject.py

E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-rec
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  268
Predicting images...
Prediction complete
```

III. Brightness = -150

Face Detection = 277 Faces, Accuracy = 100%

Face Recognition = ✓

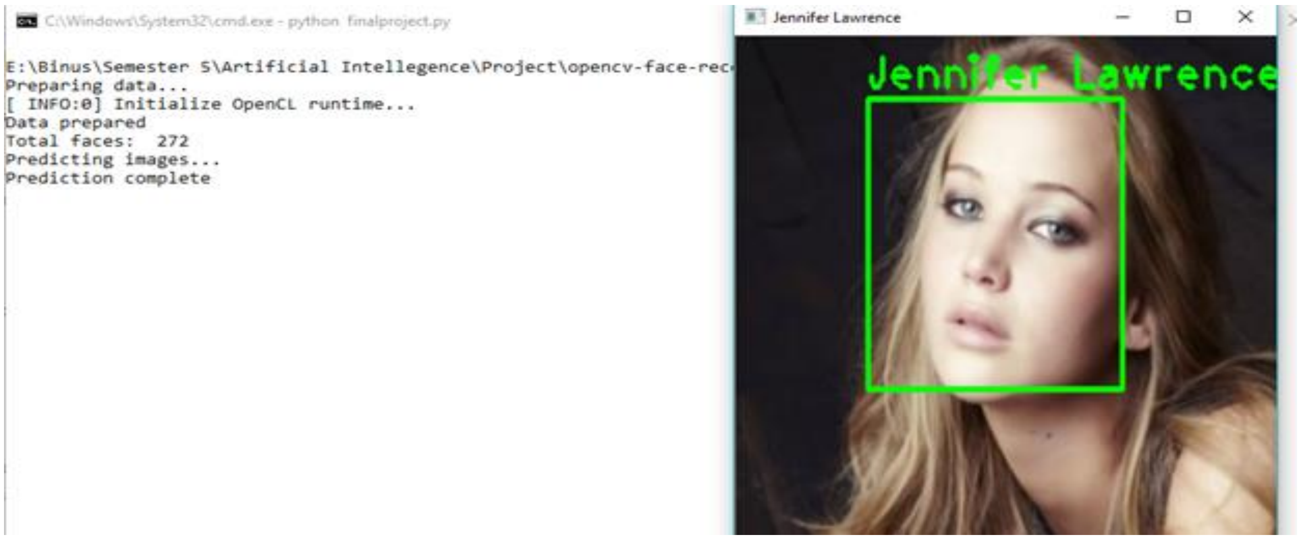


```
C:\Windows\System32\cmd.exe - python finalproject.py
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-rec
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  277
Predicting images...
Prediction complete
```

IV. Hue = +180

- Face Detection = 272 Faces, Accuracy = 98,19%

  Face Recognition = ✓



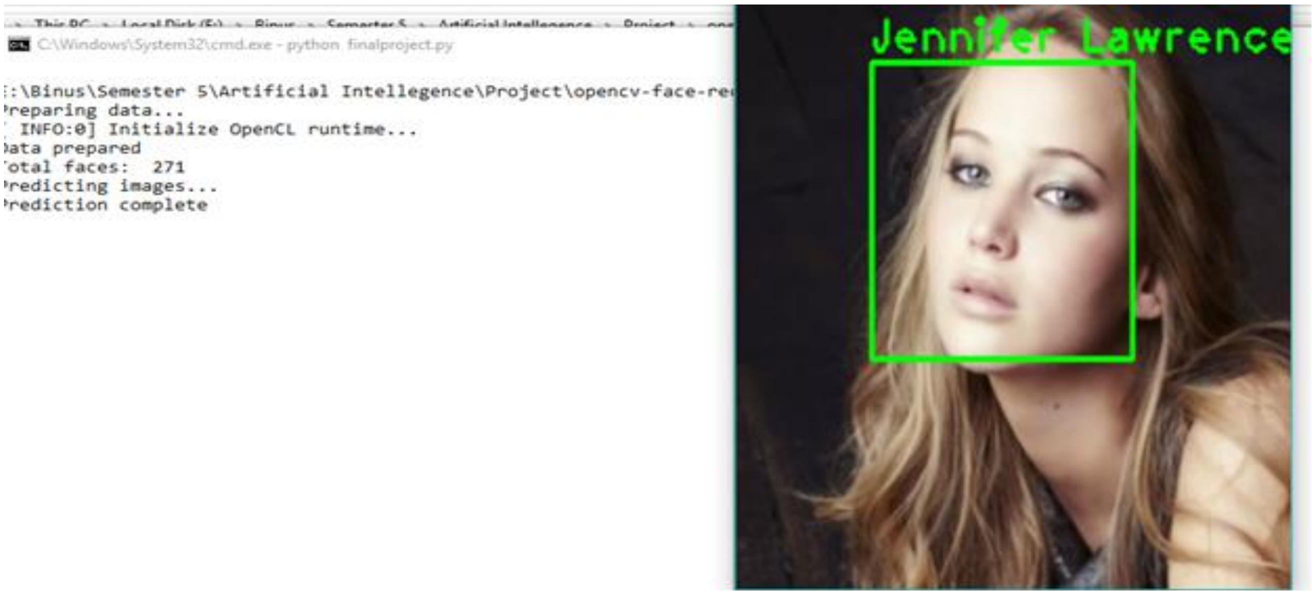
C:\Windows\System32\cmd.exe - python finalproject.py

```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-rec
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  272
Predicting images...
Prediction complete
```

V. Hue = -180

- Face Detection = 272 Faces, Accuracy = 98,19%

  Face Recognition = ✓



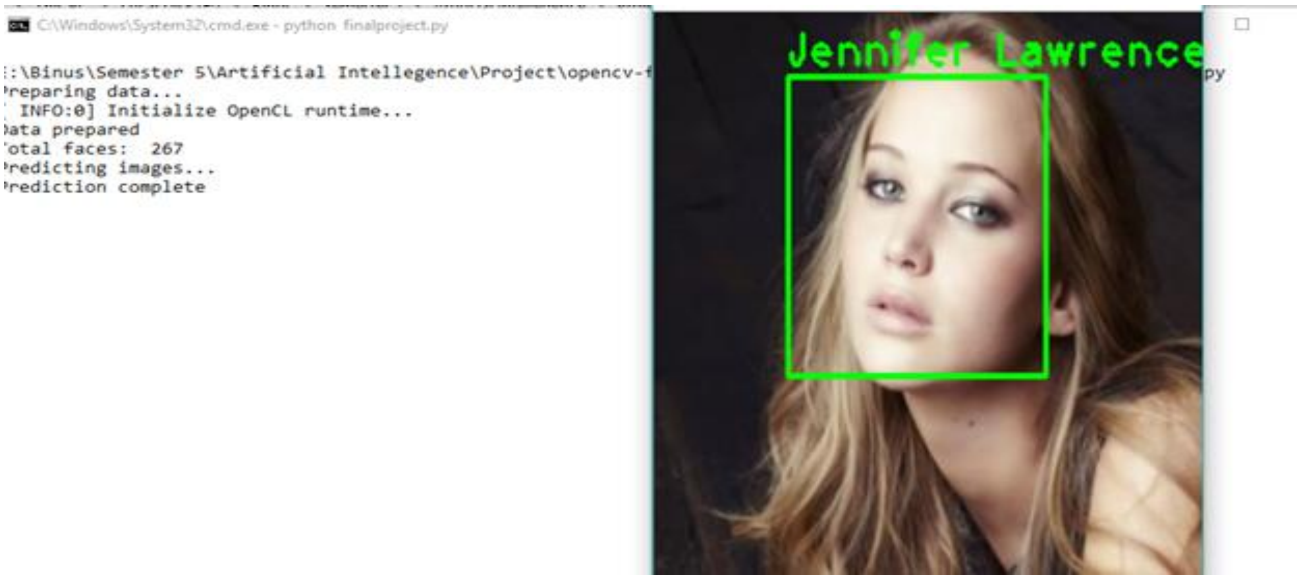
C:\Windows\System32\cmd.exe - python finalproject.py

```
:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-re
reparing data...
 INFO:0] Initialize OpenCL runtime...
ata prepared
otal faces:  271
redicting images...
rediction complete
```

VI. Saturation = +100

- Face Detection = 267 Faces, Accuracy = 96,38%

  Face Recognition = ✓

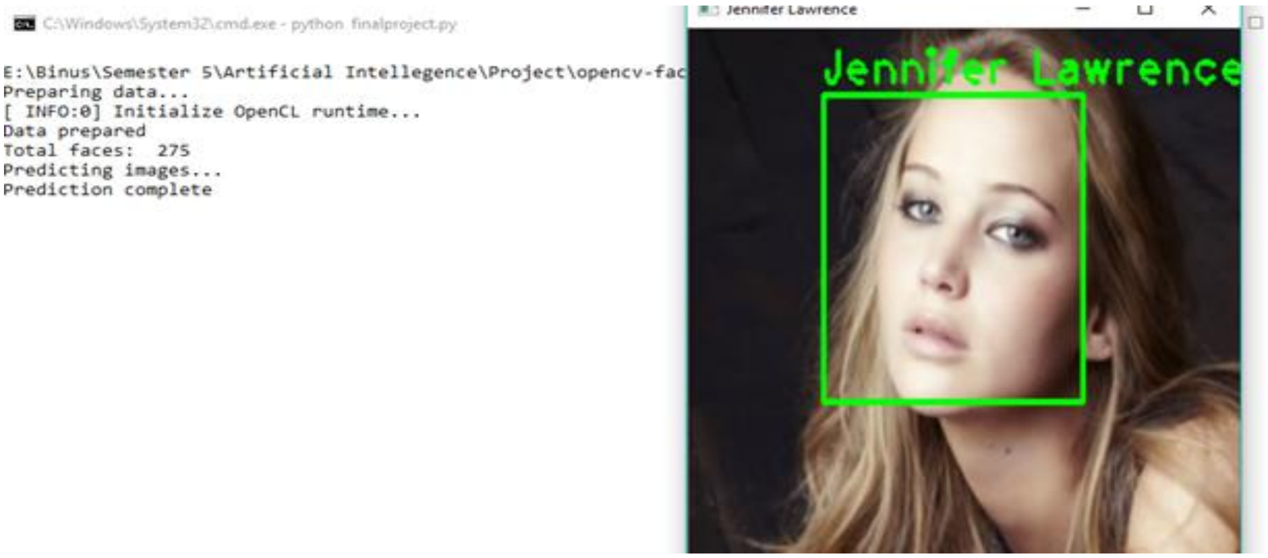


```
C:\Windows\System32\cmd.exe - python finalproject.py

:\Binus\Semester 5\Artificial Intellegence\Project\opencv-f
reparing data...
 INFO:0] Initialize OpenCL runtime...
ata prepared
otal faces:  267
redicting images...
rediction complete
```

VII. Saturation = -100

- Face Detection = 275 Faces, Accuracy = 99,27%

  Face Recognition = ✓

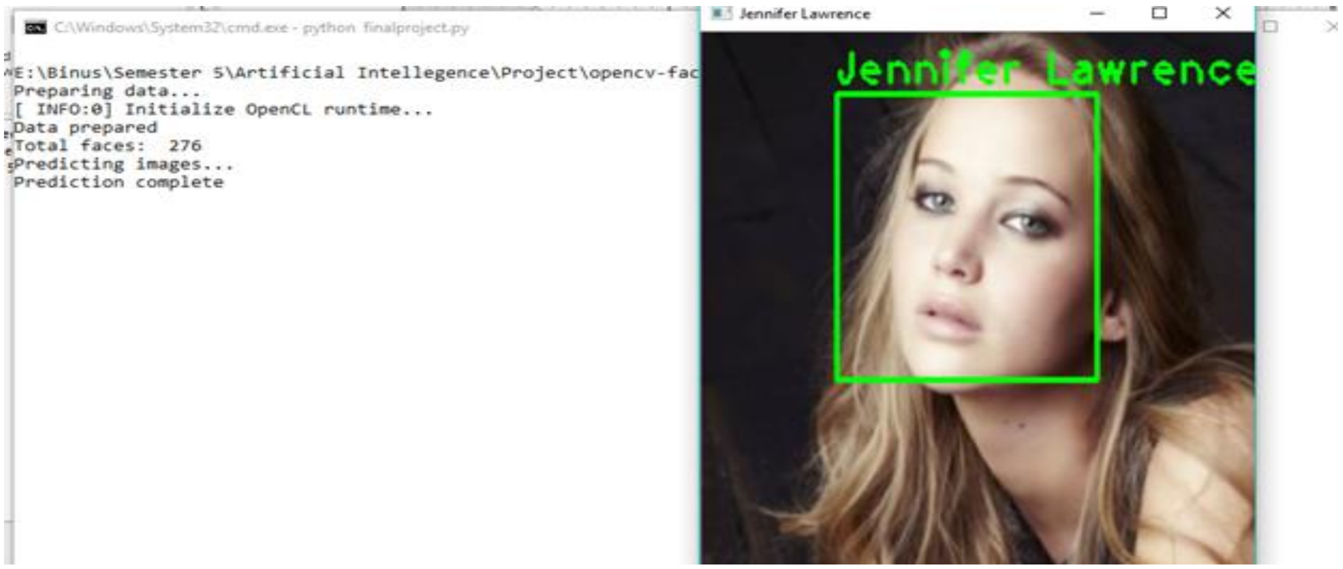


```
C:\Windows\System32\cmd.exe - python finalproject.py

E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-fac
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  275
Predicting images...
Prediction complete
```

VIII. Lightness = +80

- Face Detection = 276 Faces, Accuracy = 99,63%

  Face Recognition = ✓



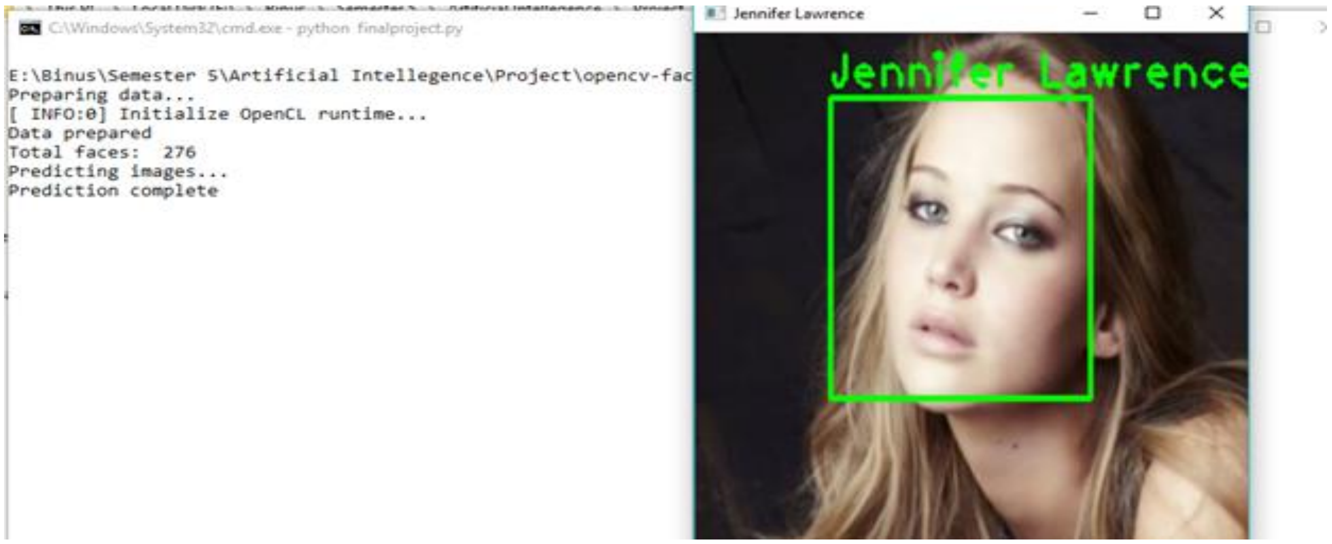
C:\Windows\System32\cmd.exe - python finalproject.py

```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-fac
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  276
Predicting images...
Prediction complete
```

IX. Lightness = -80

- Face Detection = 276 Faces, Accuracy 99,63%

  Face Recognition = ✓



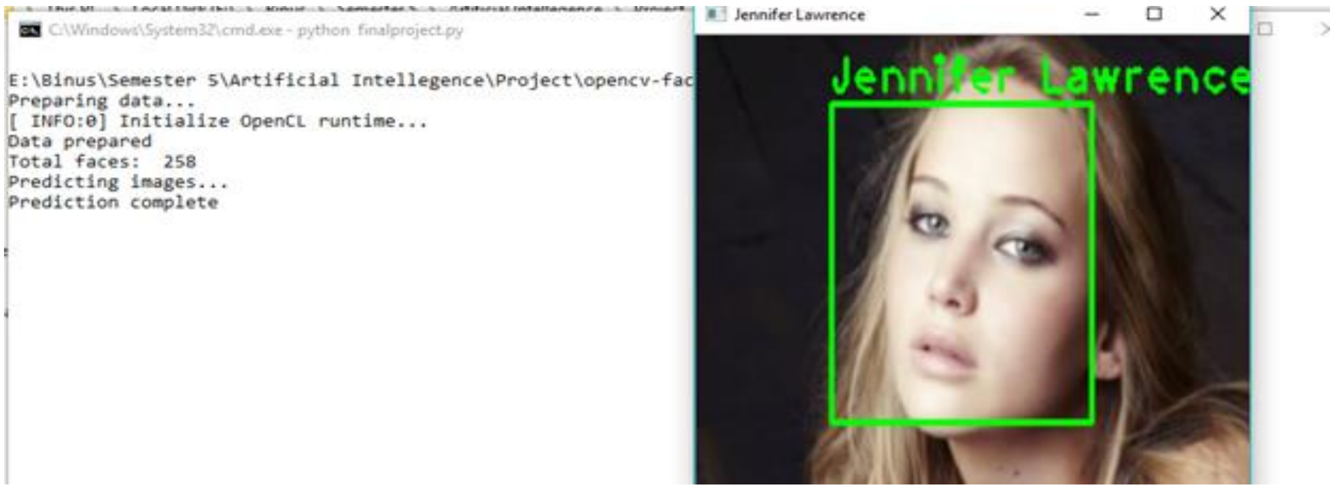
C:\Windows\System32\cmd.exe - python finalproject.py

```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-fac
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  276
Predicting images...
Prediction complete
```

X. Lightness = +74, Hue = -180, Saturation = +91

Face Detection = 258 Faces, Accuracy = 93,14%

Face Recognition = ✓



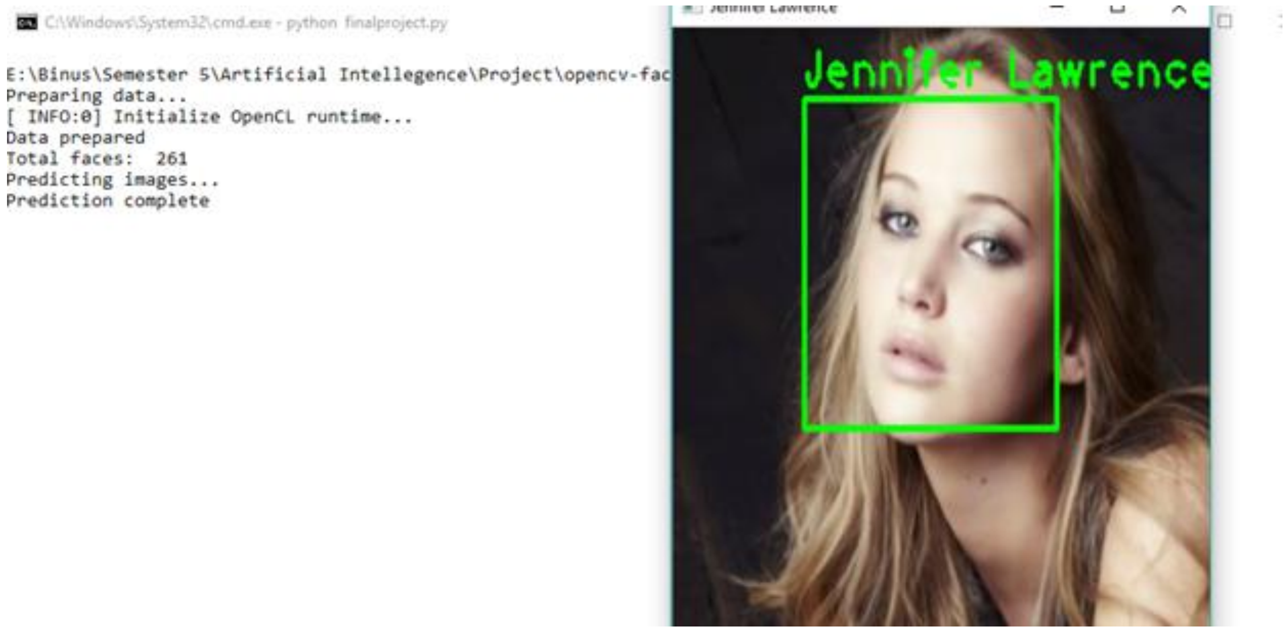
C:\Windows\System32\cmd.exe - python finalproject.py

```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-fac
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  258
Predicting images...
Prediction complete
```

XI. RGB: 100, -100, -100

Face Detection = 261 Faces, Accuracy = 94,22%

Face Recognition = ✓



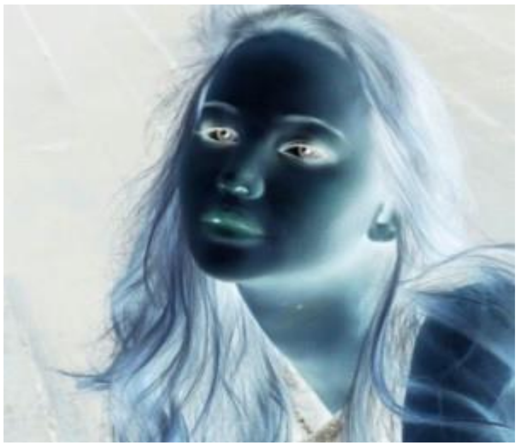
C:\Windows\System32\cmd.exe - python finalproject.py

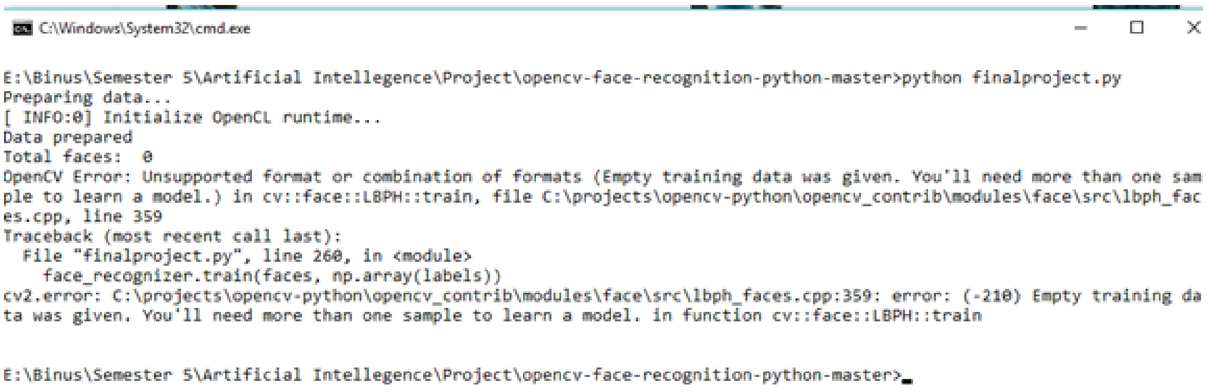
```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-fac
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  261
Predicting images...
Prediction complete
```

XII. Invert

Face Detection = 0 Faces, Accuracy = 0%

Face Recognition = ✗





```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-recognition-python-master>python finalproject.py
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  0
OpenCV Error: Unsupported format or combination of formats (Empty training data was given. You'll need more than one sam
ple to learn a model.) in cv::face::LBPH::train, file C:\projects\opencv-python\opencv_contrib\modules\face\src\lbph_fac
es.cpp, line 359
Traceback (most recent call last):
  File "finalproject.py", line 260, in <module>
    face_recognizer.train(faces, np.array(labels))
cv2.error: C:\projects\opencv-python\opencv_contrib\modules\face\src\lbph_faces.cpp:359: error: (-210) Empty training da
ta was given. You'll need more than one sample to learn a model. in function cv::face::LBPH::train


E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-recognition-python-master>
```

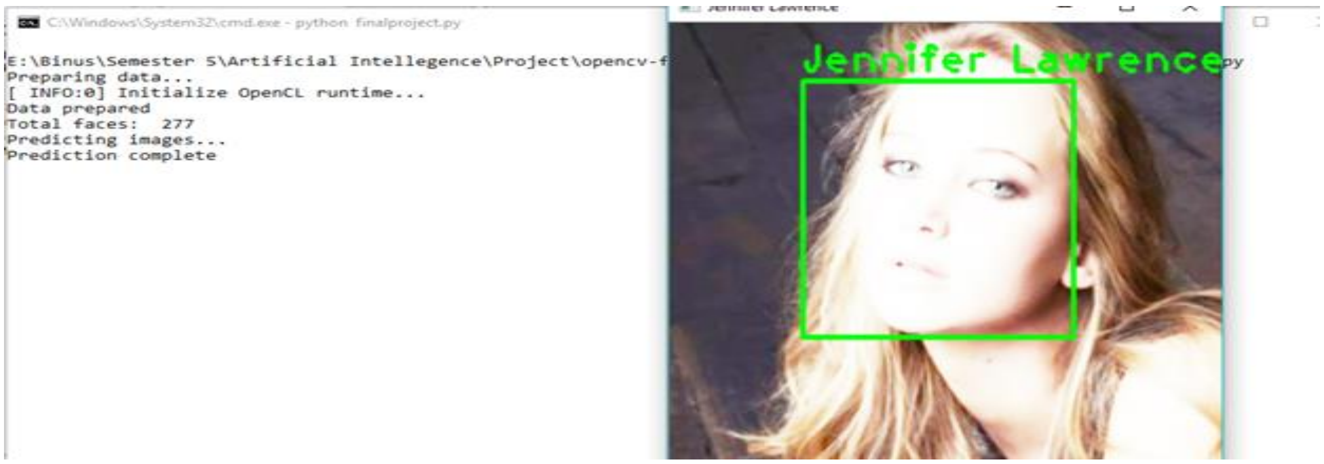
4.2.2. Test Data

- We will change the saturation, contrast, brightness, hue, lightness, invert, RGB, and using filters on the test dataset.

I. Brightness = +150

Face Detection = 277 Faces, Accuracy = 100%

Face Recognition = ✓

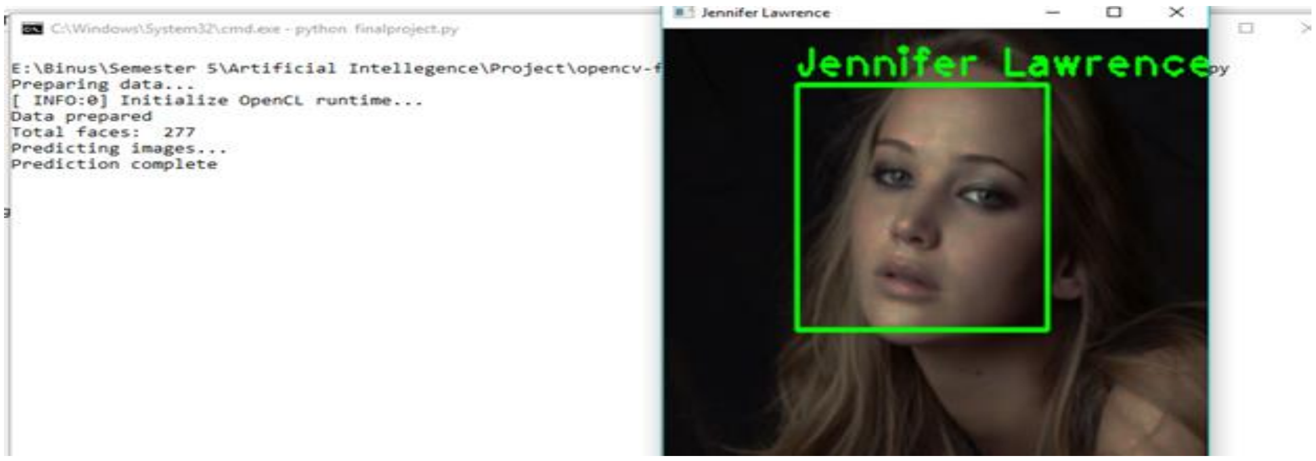


```
C:\Windows\System32\cmd.exe - python finalproject.py

E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-f
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  277
Predicting images...
Prediction complete
```

II. Brightness = -150

Face Detection = 277 Faces, Accuracy = 100%

Face Recognition = ✓

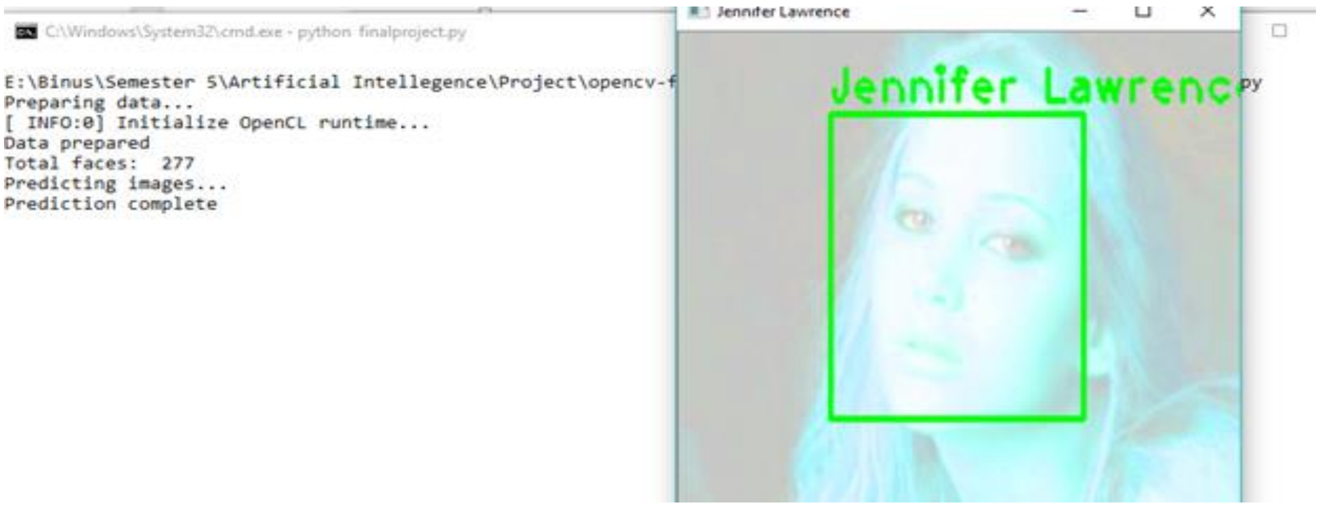


```
C:\Windows\System32\cmd.exe - python finalproject.py

E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-f
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  277
Predicting images...
Prediction complete
```

III. Hue: +162, Saturation: +86, lightness: +6

Face Detection = 277 Faces, Accuracy = 100%

Face Recognition = ✓

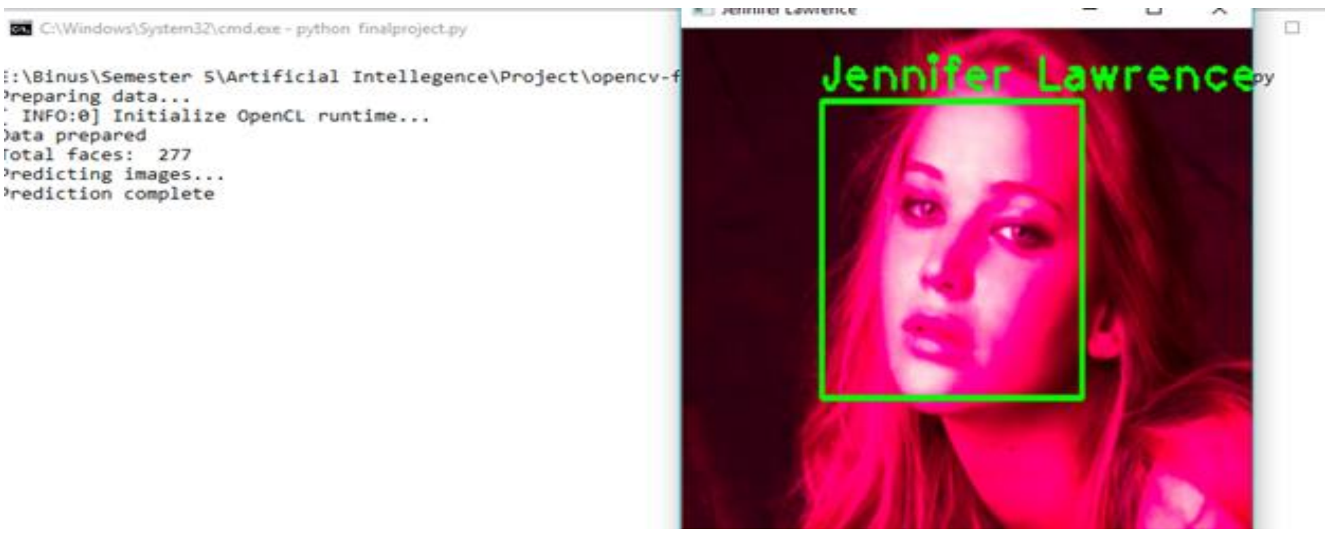


```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-f
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  277
Predicting images...
Prediction complete
```

IV. Color Balance 66, 100, 20

Face Detection = 277 Faces, Accuracy = 100%

Face Recognition = ✓

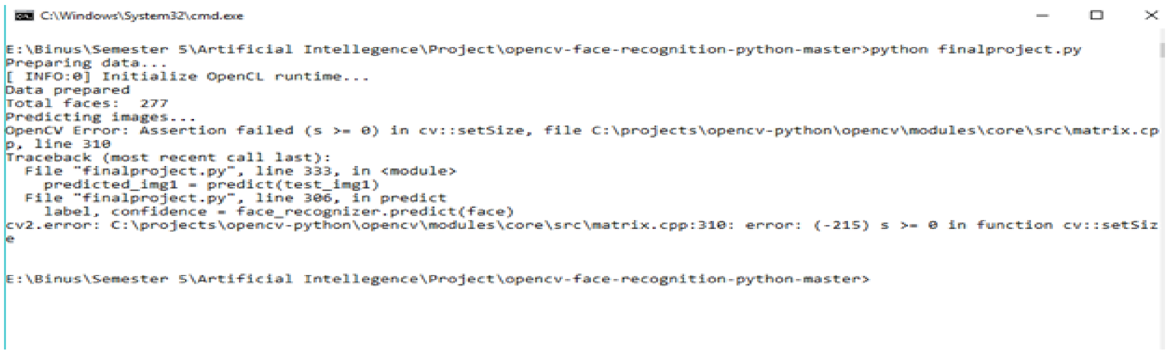


```
:\Binus\Semester 5\Artificial Intellegence\Project\opencv-f
reparing data...
 INFO:0] Initialize OpenCL runtime...
ata prepared
otal faces:  277
redicting images...
rediction complete
```

V. Invert

Face Detection = 0 Faces, Accuracy = 0%

Face Recognition = ✗



```
E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-recognition-python-master>python finalproject.py
Preparing data...
[ INFO:0] Initialize OpenCL runtime...
Data prepared
Total faces:  277
Predicting images...
OpenCV Error: Assertion failed (s >= 0) in cv::setSize, file C:\projects\opencv-python\opencv\modules\core\src\matrix.cp
p, line 310
Traceback (most recent call last):
  File "finalproject.py", line 333, in <module>
    predicted_img1 = predict(test_img1)
  File "finalproject.py", line 306, in predict
    label, confidence = face_recognizer.predict(face)
cv2.error: C:\projects\opencv-python\opencv\modules\core\src\matrix.cpp:310: error: (-215) s >= 0 in function cv::setSiz
e


E:\Binus\Semester 5\Artificial Intellegence\Project\opencv-face-recognition-python-master>
```

4.3. Result

We did two kinds of testing, the first is the normal test which is to detect and recognize several celebrities correctly. Second, we tested with modificated training datasets or test datasets to know whether this program still can detect and recognize their faces.

In the second test, we found out that by changing the properties of the test data, it will not affect the accuracy. However, when we changed the properties of the training data, some will affect the accuracy of the program. Moreover, this program cannot detect invert photos either in test datasets or training datasets.

4.4. Limitations

1. Cannot read image more than 1000x1000 pixels. But you can put image below 1000 pixels with random pixels as long under 1000 pixels
2. Cannot read unclear image such as poster or mask especially draw image
3. Detecting non-plain face such as a person from not having beard then in test-data have beard require more training
4. Detecting through twin May be inaccurate because sometimes the twins are very similar, and it is hard for the program to detect. It could detect it, but it must have many training-data
5. Inverted Image data will return an error. In this section we do not know why the program return error when we test inverted color to the image. We inverted the color of the image using Adobe Photoshop
6. The program must learn again after the program is closed which mean the data that already be trained cannot be memorized.

5. Conclusion

Face Recognition AI is like a child, if the child wants to do anything you must train him/her until he can. The more hours of train the more child can do. Same as Face Recognition AI, the face recognition needs training to recognize faces that he sees either it is A or it is B. We can only give him data to the AI, and then the AI will learn itself. The more data we gave, the smarter Face recognizer will be. The face recognition could know the exact person either he is changed because he now has beard, wear glasses or even twin. For our face recognition, we made it learn thousands data sets of five famous celebrities with estimated time of 30s to 60s depends on the specification of the PC. Also, from our testing result, we found out that the algorithms we use, LBP and LBPH, can detect and recognize faces even if the photos, either the training dataset or the test data, are not in normal condition. It still can detect and recognize faces after the photos are being edit except for the invert.

6. References

- https://www.solarianprogrammer.com/2016/09/17/install-opencv-3-with-python-3-on-windows/
- https://www.lfd.uci.edu/~gohlke/pythonlibs/
- https://stackoverflow.com/questions/44633378/attributeerror-module-cv2-cv2-has-no-attribute-createlberecognizer
- https://github.com/opencv/opencphfacv_contrib/tree/master/modules/face
- https://anaconda.org/anaconda/python
- http://www.numpy.org/
- https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html
- https://docs.opencv.org/3.0-beta/modules/face/doc/facerec/index.html
- https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html
- https://github.com/informramiz/opencv-face-recognition-python/blob/master/OpenCV-Face-Recognition-Python.py
- https://www.superdatascience.com/opencv-face-recognition/
- https://www.superdatascience.com/opencv-face-detection/

Link to our program: https://github.com/giorfasolini/FinalprojectAI

Link to our video: https://www.youtube.com/watch?v=IH_mN2Y5Vyg&t=15s