Amity Institute of Information Technology

MINOR PROJECT REPORT – 2022

(Human Face & Eye Detection)

M.Sc. (Data Science)

Submitted To:

……...

Dr. Uddalak Mitra

(Project Guide)

Submitted By:

……

Ms. Dolly Sharma

(A217117721006)

INDEX

ABSTRACT

Face and eye detection is one of the most challenging problems in computer vision area. The face is one of the easiest ways to distinguish the individual identity of each other. Face recognition is a personal identification system that uses personal characteristics of a person to identify the person's identity. The problem is further complicated by differing lighting conditions, image qualities and geometries, as well as the possibility of partial occlusion and disguise. An ideal face detector would therefore be able to detect the presence of any face under any set of lighting conditions, upon any background. The face detection task can be broken down

into two steps. The first step is a classification task that takes some arbitrary image as input and outputs a binary value of yes or no, indicating whether there are any faces present in the image.

The second step is the face localization task that aims to take an image as input and output the location of any face or faces within that image as some bounding box with (x, y, width, height).

INTRODUCTION

Face detection is one of the most challenging problems in disciplines such as image processing,pattern recognition and computer vision. With the improvements in information technology, face detection / recognition has wide usage in applications such as personal identity, video surveillance, witness face reconstruction, computerized aging, control systems (like tracing fatigue of drivers to avoid traffic accidents), HCI (human computer interaction, which provides control of computer based system, with no need of interaction through usage of mouse, keyboard etc.), video game controllers.There are many methods for face detection, but here we will complete this project using **Haar Feature** method.

Face Detection is the first and essential step for face recognition, and it is used to detect faces in the images. It is a part of object detection and can use in many areas such as security, bio-metrics, law enforcement, entertainment, personal safety, etc.

Face detection using Haar cascades is a machine learning based approach where a cascade function is trained with a set of input data. OpenCV is a library of Python bindings designed to solve computer vision problems. OpenCV already contains many pre-trained classifiers for face, eyes, smiles, etc.

HAAR CASSCADES

It is **an Object Detection Algorithm used to identify faces in an image or a real time video**. The algorithm uses edge or line detection features proposed by Viola and Jones in their research paper “Rapid Object Detection using a Boosted Cascade of Simple Features” published in 2001. In their paper, Viola and Jones propose an algorithm that is capable of detecting objects in

images, regardless of their location and scale in an image. Furthermore, this algorithm can run in real-time, making it possible to detect objects in video streams.

Specifically, Viola and Jones focus on detecting faces in images. Still, the framework can be used to train detectors for arbitrary “objects,” such as cars, buildings, kitchen utensils, and even bananas.

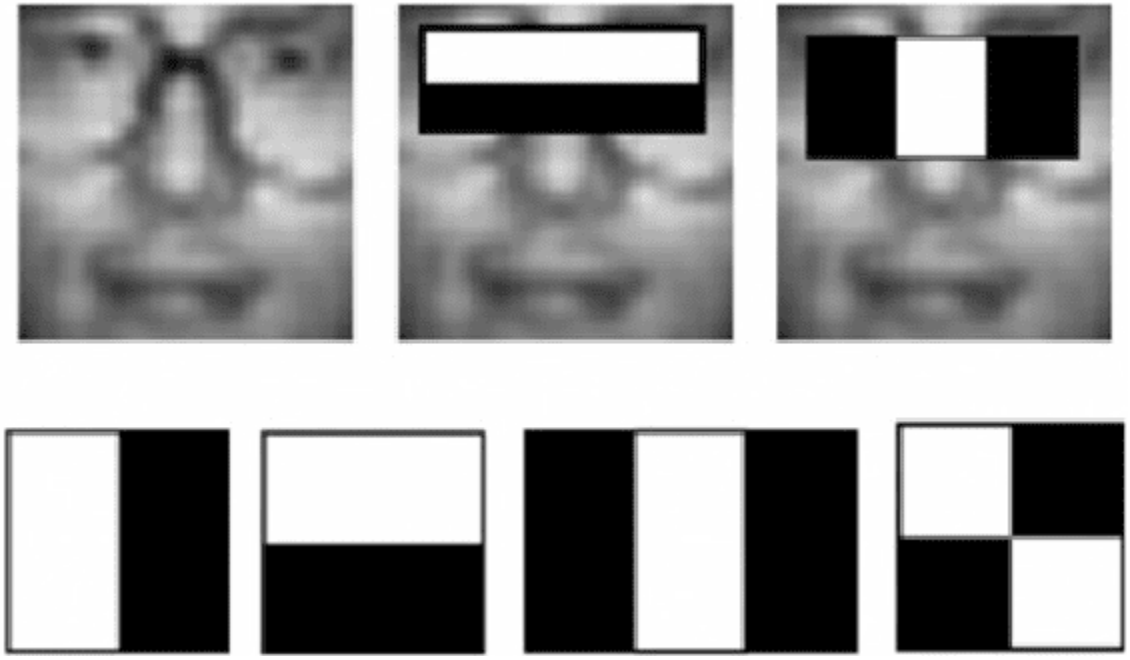


Figure 1: First introduced in 2001, Haar cascades are a class of object detection algorithms

OpenCV

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today’s systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

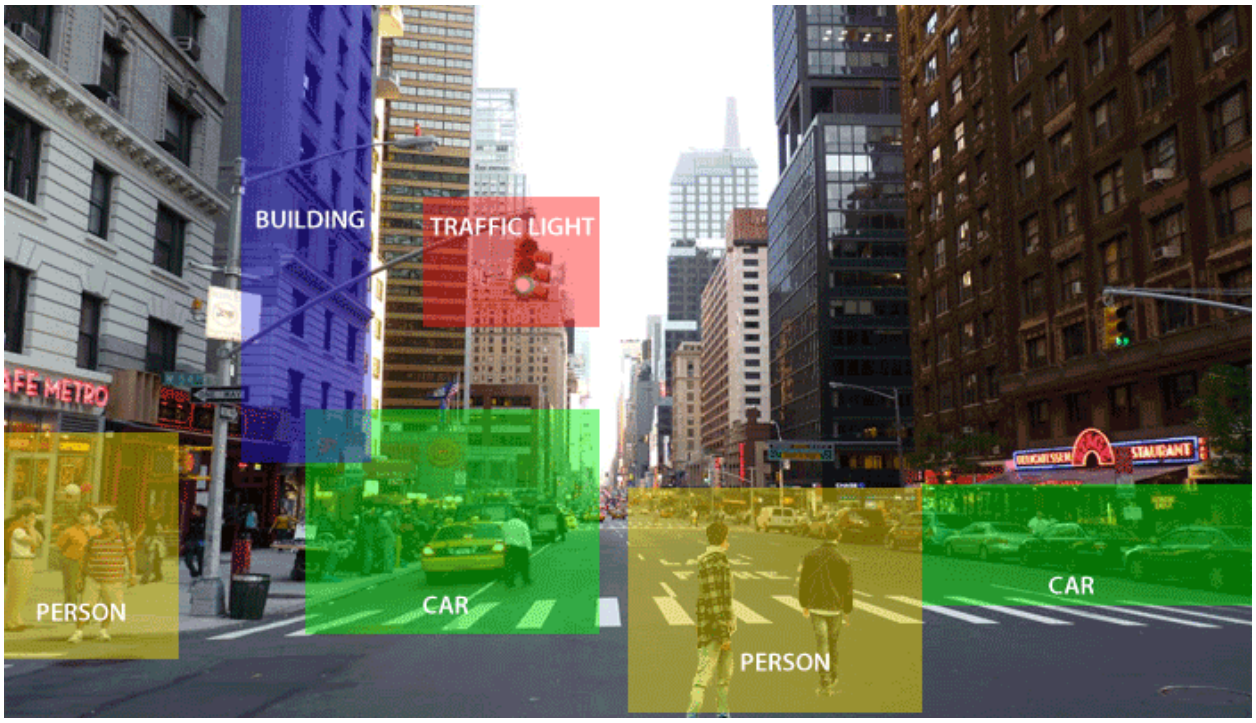


Figure 2 : Object Detection Using OpenCV

Look at the figure2

In the above image a lots of objects like cars, person, building, traffic lights etc are visible so by the help of OpenCV we can get all these types of information from the image.

NEEDS / PROBLEMS

Face Detection is the first and essential step for face recognition, and it is used to detect faces in the images. It is a part of object detection and can use in many areas such as security, bio-metrics, law enforcement, entertainment, personal safety, etc.

It is used to detecting real time for surveillance and tracking of person or objects. It is widely used in cameras to identify multiple appearances in the frame Ex- Mobile cameras and DSLR's. Facebook is also using face detection algorithm to detect faces in the images and recognise them.

Challenges of Facial Recognition As with any technology, there are potential drawbacks to using facial recognition, such as threats to privacy, violations of rights and personal freedoms,

potential data theft and other crimes. There's also the risk of errors due to flaws in the technology.

OBJECTIVES

The primary aim of face & eye detection algorithms is to determine whether there is any human face & eye in an image or not. The objective of face recognition is, from the incoming image, to find a series of data of the same face in a set of training images in a database. The great difficulty is ensuring that this process is carried out in real-time, something that is not available to all biometric facial recognition software providers.

Description of Dataset

For this project we have used 2 harcascade XML files :

1. **haarcascade_frontalface_default :** Detect faces

2. **haarcascade_eye :** Detects the left and right eyes on the face

These XML files contains several types of features and having a lot of negative and positive data points which are used in object detection.

We need to download the trained classifier XML file (haarcascade_frontalface_default.xml), which is available in https://github.com/opencv/opencv/tree/master/data/haarcascades.

```
- <opencv_storage>
  - <cascade type_id="opencv-cascade-classifier">
      <stageType>BOOST</stageType>
      <featureType>HAAR</featureType>
      <height>24</height>
      <width>24</width>
    - <stageParams>
        <maxWeakCount>211</maxWeakCount>
      </stageParams>
    - <featureParams>
        <maxCatCount>0</maxCatCount>
      </featureParams>
      <stageNum>25</stageNum>
    - <stages>
      - <_>
          <maxWeakCount>9</maxWeakCount>
          <stageThreshold>-5.0425500869750977e+00</stageThreshold>
        - <weakClassifiers>
          - <_>
              <internalNodes> 0 -1 0 -3.1511999666690826e-02</internalNodes>
              <leafValues> 2.0875380039215088e+00 -2.2172100543975830e+00</leafValues>
            </_>
          - <_>
              <internalNodes> 0 -1 1 1.2396000325679779e-02</internalNodes>
              <leafValues> -1.8633940219879150e+00 1.3272049427032471e+00</leafValues>
            </_>
          - <_>
              <internalNodes> 0 -1 2 2.1927999332547188e-02</internalNodes>
              <leafValues> -1.5105249881744385e+00 1.0625729560852051e+00</leafValues>
            </_>
          - <_>
              <internalNodes> 0 -1 3 5.7529998011887074e-03</internalNodes>
              <leafValues> -8.7463897466659546e-01 1.1760339736938477e+00</leafValues>
            </_>
          - <_>
              <internalNodes> 0 -1 4 1.5014000236988068e-02</internalNodes>
              <leafValues> -7.7945697307586670e-01 1.2608419656753540e+00</leafValues>
```


PROCEDURES

1. First of all we will install Opencv in Python from " pip install opencv-python" command.
2. Then we are creating two Haarcascade object ("face_detector", "eye_detector") to detect faces and eyes in the frame. From "cv2.CascadeClassifier" command we will create our objects.

```
import cv2
face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_detector = cv2.CascadeClassifier('haarcascade_eye.xml')
```

3. Now we will Creating a VideoCapture object to access the webcam. Argument 0 is passed when we want to use the inbuilt webcam of PC/Laptop, use 1 if you want to use any external camera.

```
cam = cv2.VideoCapture(0)
```

4. Now we will set **ret=True** (just a formality to start the infinite loop).
5. Now we will start the loop,

First we will Convert the image from **BGR** to grayscale because Haar Cascades detect faces in grayscale images efficiently. Then will detect the faces using **detectMultiscale,** Now we have our face coordinates as (x,y,w,h) where (x,y) are the coordinates of the top-left of the rectangle around the face, w is the width and h is the height of the rectangle.

```
while ret:
    ret, frame = cam.read()
    if ret == True:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        face_points = face_detector.detectMultiScale(gray, 1.3, 5)
        for (x, y, w, h) in face_points:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 20), 2)

            face = frame[y:y+h,x:x+w]
            eyes = eye_detector.detectMultiScale(face,1.3,5)
            for (x, y, w, h) in eyes:
                cv2.rectangle(face, (x, y), (x+w, y+h), (155, 0, 120), 2)
```

It gives us an error :

```
while ret:
    ret, frame = cam.read()
    if ret == True:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        face_points = face_detector.detectMultiScale(gray, 1.3, 5)
        for (x, y, w, h) in face_points:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 20), 2)

            face = frame[y:y+h,x:x+w]
            eyes = eye_detector.detectMultiScale(face,1.3,5)
            for (x, y, w, h) in eyes:
                cv2.rectangle(face, (x, y), (x+w, y+h), (155, 0, 120), 2)
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call last)
Input In [11], in <cell line: 1>()
      3 if ret == True:
      4     gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
----> 5     face_points = face_detector.detectMultiScale(gray, 1.3, 5)
      6     for (x, y, w, h) in face_points:
      7         cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 20), 2)

error: OpenCV(4.5.5) D:\a\opencv-python\opencv-python\opencv\modules\objdetect\src\cascadedetect.cpp:1689: error: (-215:Asserti
on failed) !empty() in function 'cv::CascadeClassifier::detectMultiScale'
```

To remove this error we will use **"cv2.data.haarcascades +"** property who is a cv2 property present in Opencv.

From OP

```
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
```

Becomes

```
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_defau
eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_eye.xml')
```

6. After loop we will run "cv2.imshow('Live feed', frame)" command which will gives us result.

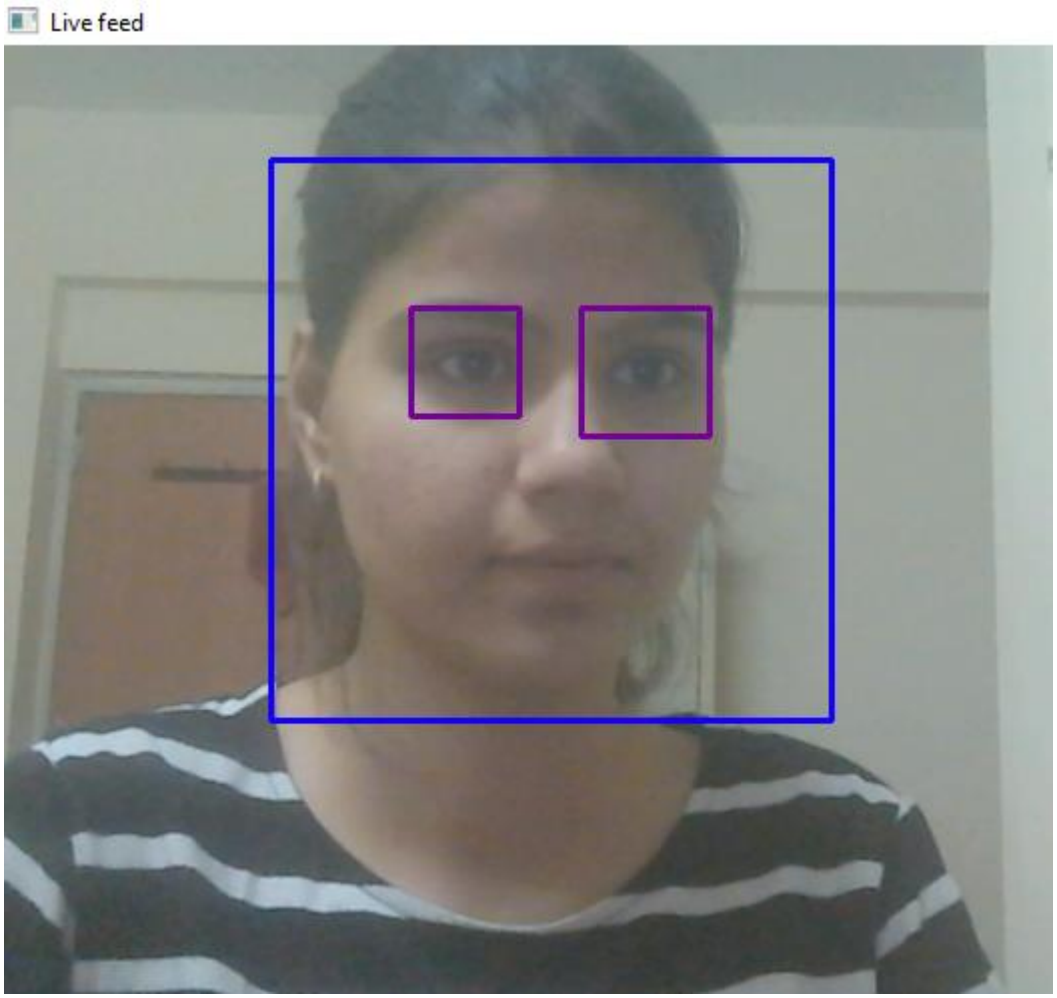
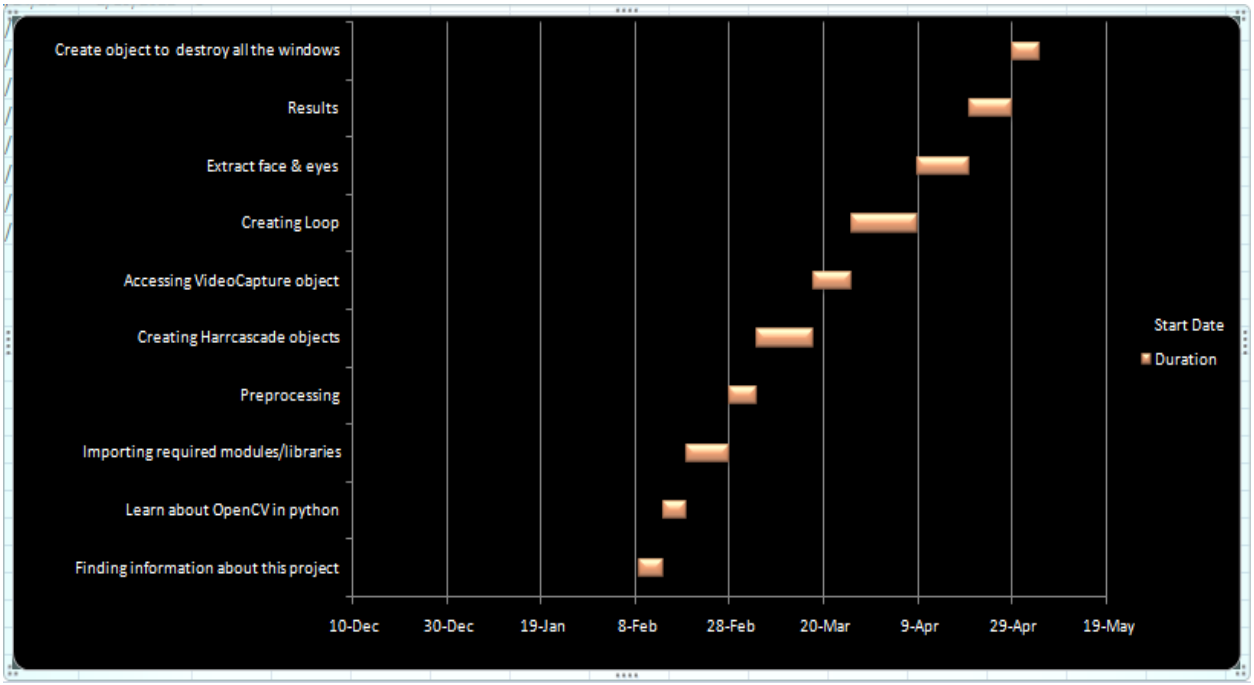


Figure: Final Result Live Face & Eye Detection

In the ent we will give "cam.release()", "cv2.destroyAllWindows()" command for close the all open windows.

GANTT CHART



APPENDIX

Hardware Requirments :

- Netis WF2111 adaptor
- 256 MB RAM
- 1GB hard free drive space

Software Requirments :

- Anaconda
- Jupyter Notebook
- Windows 10 Operating system

ADVANTAGES Of THIS PROJECT

- Face Detection improves surveillance efforts and helps hunt criminals and terrorists.
- Face Detection and Face Recognition Technology is straightforward to integrate, and most solutions are compatible with the bulk of security software.
- Face Detection enables the automation of the identification process, which saves time and improves accuracy.
- Instead of pesky one-time passwords, you'll authorize transactions by watching your smartphone or computer. Biometric online banking is another of the advantages of face Recognition.
- The advantage of Face Recognition in retail is that it could make the shopping experience faster and more convenient. New “Face Pay” technologies can shorten the long checkout lines with slow payments.
- Better Worker Attendance Systems- Every employee has got to pass face-scanning devices to see certain work. From here until checkout, you will be charged for your time. Employees don't have to authenticate their identities or punch in using plastic cards, thus the process is quick.

FUTURE SCOPE

The world is using facial recognition technology and enjoying its benefits. Why should India be left out? There is a huge scope of this technology in India and it can help improve the country in various aspects. The technology and its applications can be applied across different segments in the country.

- Preventing the frauds at ATMs in India. A database of all customers with ATM cards in India can be created and facial recognition systems can be installed. So, whenever user will enter in ATM his photograph will be taken to permit the access after it is being matched with stored photo from the database.
- Reporting duplicate voters in India.
- Passport and visa verification can also be done using this technology.
- Also, driving license verification can be done using the same approach.
- In defence ministry, airports, and all other important places the technology can be used to ensure better surveillance and security.
- It can also be used during examinations such as Civil Services Exam, SSC, IIT, MBBS, and others to identify the candidates.
- For access control verification and identification of authentic users it can also be installed bank lockers and vaults.
- For identification of criminals the system can be used by police force also.

SUMMARY

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. The models are stored on GitHub, and we can access them with OpenCV methods.

Face detection can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of all objects in an image that belong to a given class.

Face-detection algorithms focus on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit. Image matches with the image stores in database. Any facial feature changes in the database will invalidate the matching process.