Qassim University
جامعة القصيم

Face Detection Technique

Team Work:

Alaa Alromaih
Ameera Alorf
Reema Alsmaeil
Ruba Almuziraee
Wesam Alqaadan

SUPERVISOR:

Dr. Waleed Albattah

Outlines :

01 A Quick Recap

02 Implementation And Testing

03 Results And Discussions

04 Conclusion and Future Work

01

A Quick Recap_

Our main goal

Develop System To Face Detection by using Machine Learning (ML) that can determine if human faces appear in images , we aimed for high accuracy in detecting faces based on pre-trained classifiers algorthim and a machine learning software library.

Why

Machine Learning (ML) ?

Machine learning is a branch of artificial intelligence. It is a process of teaching computers to learn from data and involves developing algorithms that can automatically detect patterns in data and then make predictions based on those patterns.

In Machine Learning (ML)

It uses **Haar Cascade algorithm** to analyze and separate faces from all the other features that may be present in an image.

Motivation & Problem Statement

The main problem that caused concern for sensitive systems that need high accuracy in the face detection process .

Why it's important to solve these problems ?

Provide or develop a system, to be used by sensitive systems to be able to count on performance methods that would detect faces in the image with a focus on high-level accuracy to detection .

Goals & Objectives

01

Display appropriate studies of face detection methods using OpenCV classifiers.

02

A suitable system, keeping in view the requirements that should be accurate.

03

Detect the faces of people by using OpenCV.

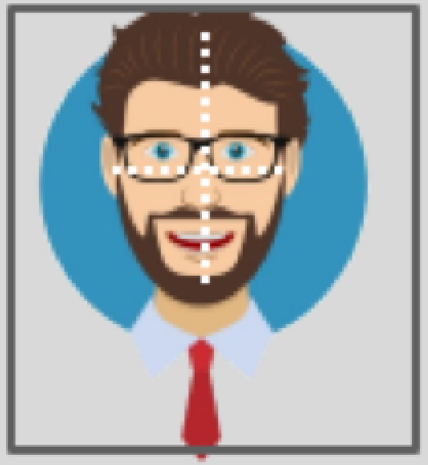
04

Face detection using a pre-trained classifier for faces and eyes called Haar Cascade.

Overview of The Proposed Methodology

Create a cascade classifier. It will contain the feature the features on the face

Input Image



OpenCV

OpenCV will read the input image by convert it into matrix number and the feature file

[[[|2|,|3|,2|3]
[2|3,|43,|76]
[|24.|27.|28]
.....
.....
[|34,|35,|72]
[2|3,|43,|76]
[2|3,|43,|76]]]

Displayed Image

02

Implementation and Testing

- Software Configuration

1. PyCHARM

2. Python

3. Used Libraries(CV2 , Numpy,OS)

- System Coding

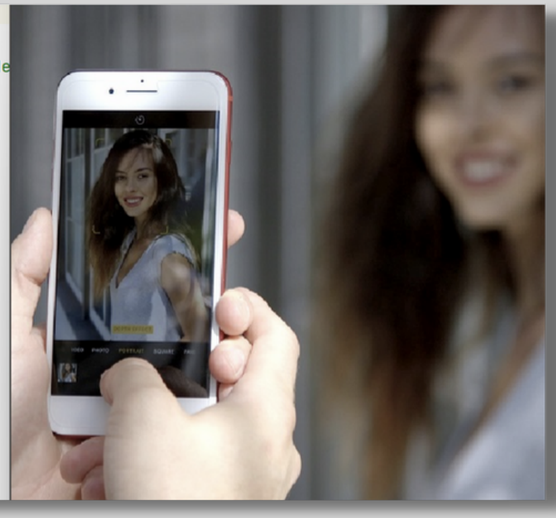
```python
import cv2
#Loading the haarcascade front face file
faceCascade = cv2.CascadeClassifier('haarcascade_frontalface_de
#Reading the face image
#img = cv2.imread('test1.jpeg')
#img = cv2.imread('test3.png')
img = cv2.imread('test3.jpg')
#Converting the image into grayscale image
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#Detecting the face
face = faceCascade.detectMultiScale(gray, 1.1,4)
#Draw rectangle around the face
for(x,y,w,h) in face:
    cv2.rectangle(img, (x,y),(x+w, y+h), (255,0,0),2)
#Display the result with face detected
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

One face

Multi faces

```python
import cv2
#Loading the haarcascade front face file
faceCascade = cv2.CascadeClassifier('haarcascade_frontalface_d
#Reading the face image
#img = cv2.imread('test1.jpeg')
#img = cv2.imread('test3.png')
img = cv2.imread('test3.jpg')
#Converting the image into grayscale image
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#Detecting the face
face = faceCascade.detectMultiScale(gray, 1.1,4)
#Draw rectangle around the face
for(x,y,w,h) in face:
    cv2.rectangle(img, (x,y),(x+w, y+h), (255,0,0),2)
#Display the result with face detected
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```python
import cv2
#Loading the haarcascade front face file
faceCascade = cv2.CascadeClassifier('haarcascade_frontalface_de
#Reading the face image
#img = cv2.imread('test1.jpeg')
#img = cv2.imread('test3.png')
img = cv2.imread('test3.jpg')
#Converting the image into grayscale image
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#Detecting the face
face = faceCascade.detectMultiScale(gray, 1.1,4)
#Draw rectangle around the face
for(x,y,w,h) in face:
    cv2.rectangle(img, (x,y),(x+w, y+h), (255,0,0),2)
#Display the result with face detected
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



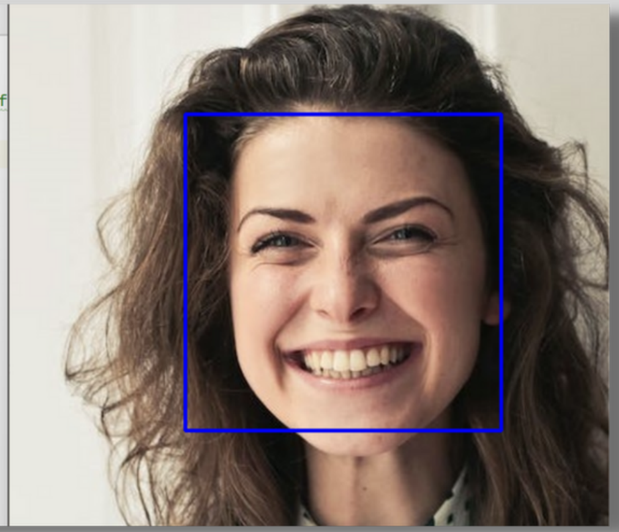
Blurred face

03

Results And Discussions

run.py main.py test3.jpg test1.jpeg README.md

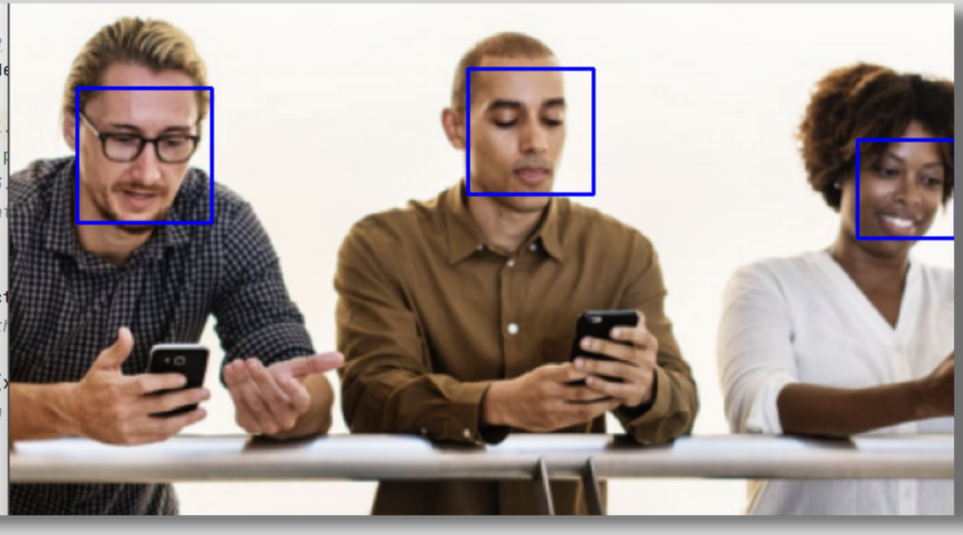
```python
import cv2
#Loading the haarcascade front face file
faceCascade = cv2.CascadeClassifier('haarcascade_frontalf
#Reading the face image
img = cv2.imread('test1.jpeg')
#img = cv2.imread('test3.png')
#img = cv2.imread('test3.jpg')
#Converting the image into grayscale image
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
#Detecting the face
face = faceCascade.detectMultiScale(gray, 1.1,4)
#Draw rectangle around the face
for(x,y,w,h) in face:
    cv2.rectangle(img, (x,y),(x+w, y+h), (255,0,0),2)
#Display the result with face detected
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



One face

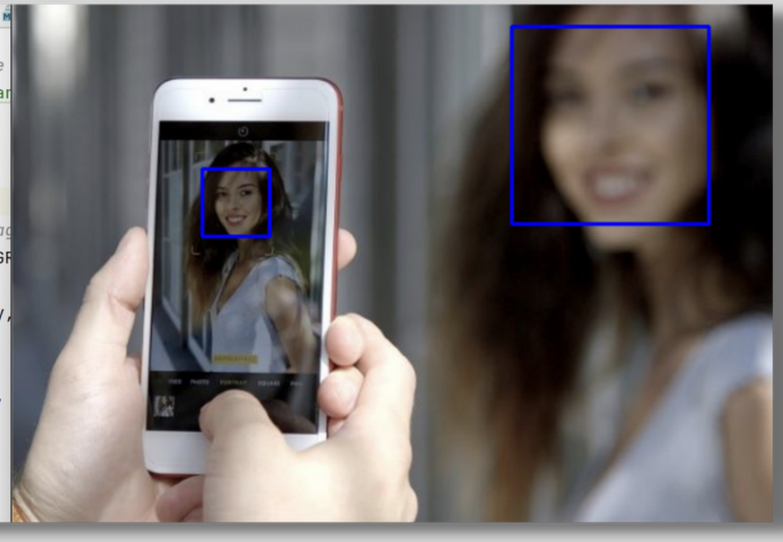
Multi faces

```python
import cv2
#Loading the haarcascade
faceCascade = cv2.Cascade
#Reading the face image
#img = cv2.imread('test1.
img = cv2.imread('test2.p
#img = cv2.imread('test3.
#Converting the image int
gray = cv2.cvtColor(img,
#Detecting the face
face = faceCascade.detect
#Draw rectangle around th
for(x,y,w,h) in face:
    cv2.rectangle(img, (x
#Display the result with
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



run.py main.py test3.jpg test1.jpeg

```python
import cv2
#Loading the haarcascade front face file
faceCascade = cv2.CascadeClassifier('haar
#Reading the face image
#img = cv2.imread('test1.jpeg')
#img = cv2.imread('test2.png')
img = cv2.imread('test3.jpg')
#Converting the image into grayscale imag
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GR
#Detecting the face
face = faceCascade.detectMultiScale(gray,
#Draw rectangle around the face
for(x,y,w,h) in face:
    cv2.rectangle(img, (x,y),(x+w, y+h),
#Display the result with face detected
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



Blurred face

MAJOR FINDING

WE HAVE SUCCESSFULLY ACHIEVED OUR GOALS!!

- Display appropriate studies of face detection methods.
- Searching for a suitable system for the project requirements.
- Choosing OpenCV library for detection.
- Using a pre-trained HAAR cascade classifier for detection.

LIMITATIONS

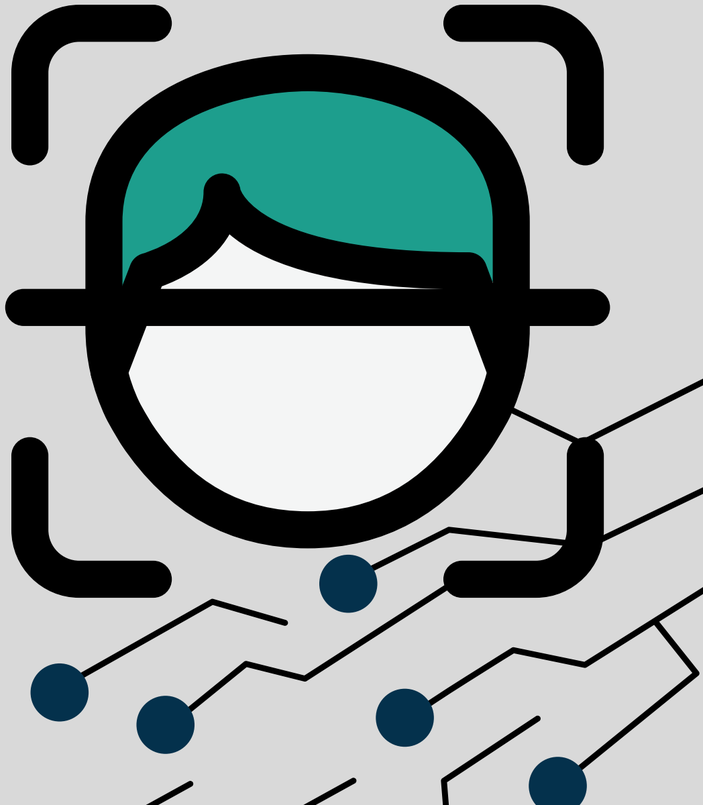
- Slow performance
- Not much improvement
- Can not detect faces for people with masks

04

Conclusion and Future Work

Conclusion

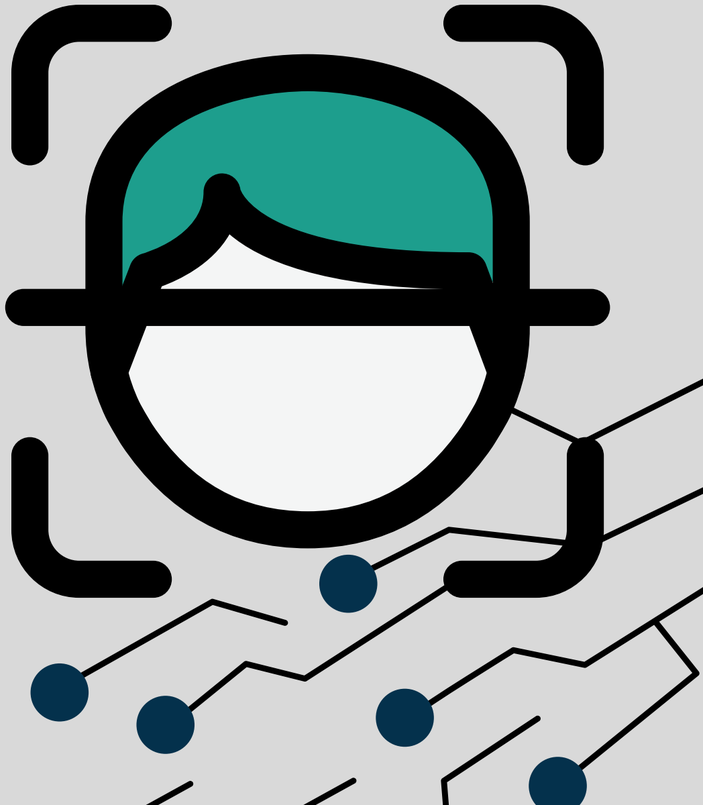
- Detect human faces in photos



Future Work

★ Detect the faces of newborn babies.

★ Detect faces with masks on.



Thank You