Gliwice, 30.06.2022

BIAI PROJECT :

Facial expression recognition

Teacher : **Marcin Wierzchanowski**

Participants : **Yohann Coueraud, Kenan Cosic**

Gliwice, 30.06.2022

Summary

Introduction

Facial recognition is a way of recognizing a human face through technology. A facial recognition system uses biometrics to map facial features from a photograph or video. It compares the information with a database of known faces to find a match. Today, this technology is used in many ways, such as surveillance or face ID recognition on our smartphones, and it is also possible to diagnose diseases such as DiGeorge syndrome.

The purpose of project will consist of creating an AI capable of recognizing human emotions. In construction of the project, we will use python to develop our software. In order to help coding the AI, we will open-cv sources and matplotlib.

To teach our AI, we will use a pre-train model named DeepFace.

Language and libraries used

Python is the most famous language when it comes to machine learning, developing AI and deep learning. It has variety of libraries which help coding AI and it's an open-source.
We used Jupiter Notebook, on Visual Studio Code, an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. This application is really famous to implement AI in Python.

OpenCv stands for open-source computer vision. It is a library commonly used with workings of image processing and performing computer vision tasks.

The aim of computer vision is to know the content of the pictures. It extracts the pictures definition, which can be an object, a content description, three-dimension architecture, etc.

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays. One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc. For us, it will consist to show images with some fonctionnalities added as things progress.

Deepface is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. The open-sourced DeepFace library includes all leading-edge AI models for face recognition and automatically handles all procedures for facial recognition in the background.

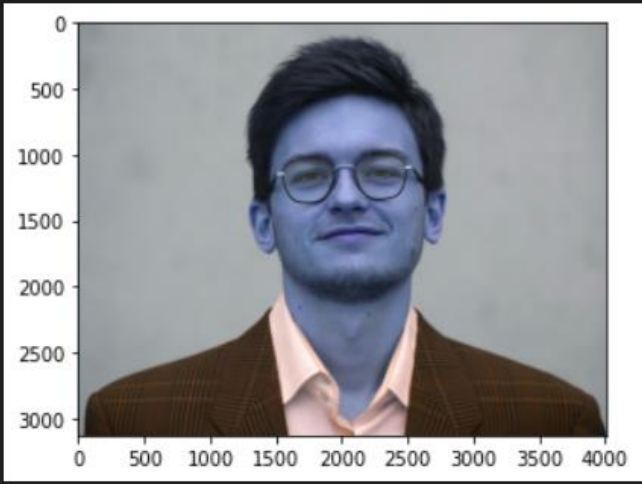
Code explanation

The first step that we decide to take was to implement our IA to recognize the émotions on photos. All the explanations about the functions used are in commentaries on the code.

For images :

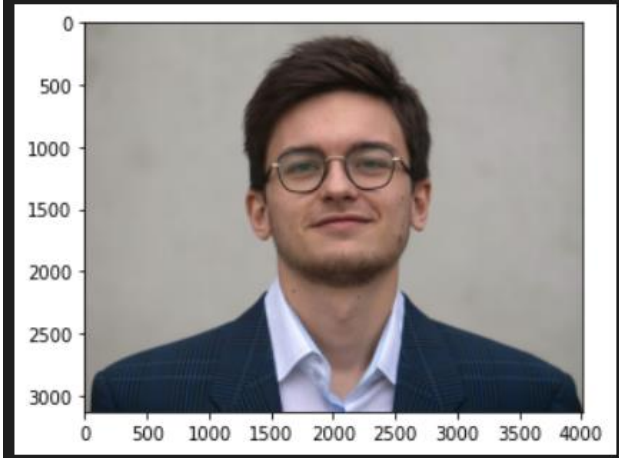
```
import cv2
import matplotlib.pyplot as plt

###loading image using cv2

img = cv2.imread("photo_test.jpg") # fonction for charging the image

### First step generating the raw photo
##showing image using plt in color BGR by default

plt.imshow(img)
plt.show()
```

✓ 1.6s



```python
import cv2
import matplotlib.pyplot as plt

###loading image using cv2

img = cv2.imread("photo_test.jpg") # fonction for charging the image

### We transform the color BGR in RGB to have the real color of the photo
# We use cv2.cvtColor and then showing the image with plt
color_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(color_img)
plt.show()
```

✓ 1.7s




```python
from deepface import DeepFace
prediction = DeepFace.analyze(color_img) #Analyse of the image with the pre-train model Deepface in order to recognize emotions, the race, the age...
```

Python

```
Action: race: 100%|██████████| 4/4 [00:31<00:00,  7.80s/it]
```

```python
prediction #Display the results
```


```
{'emotion': {'angry': 0.00043136865315318573,
  'disgust': 5.352979886609488e-11,
  'fear': 0.0005538270670513157,
  'happy': 98.2509970664978,
  'sad': 0.9149658493697643,
  'surprise': 2.225585582493750e-06,
  'neutral': 0.833056028932333},
 'dominant_emotion': 'happy',
 'region': {'x': 1274, 'y': 521, 'w': 1539, 'h': 1539},
 'age': 26,
 'gender': 'Man',
 'race': {'asian': 1.556966644500335e-06,
  'indian': 1.566382223927576e-06,
  'black': 1.1570382152381598e-08,
  'white': 99.91118311882019,
  'middle eastern': 0.021198853210080415,
  'latino hispanic': 0.06761472905054688},
 'dominant_race': 'white'}
```

```
prediction['dominant_emotion'] #Display of the dominant emotion
```

✓ 0.8s

```
'happy'
```

```
#Creation of a rectangle on the face

#loading our xml file into faceCascade using cv2.CascadeClassifier
# Classifier allow (variable) CascadeClassifier: Any
faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
```

✓ 0.4s

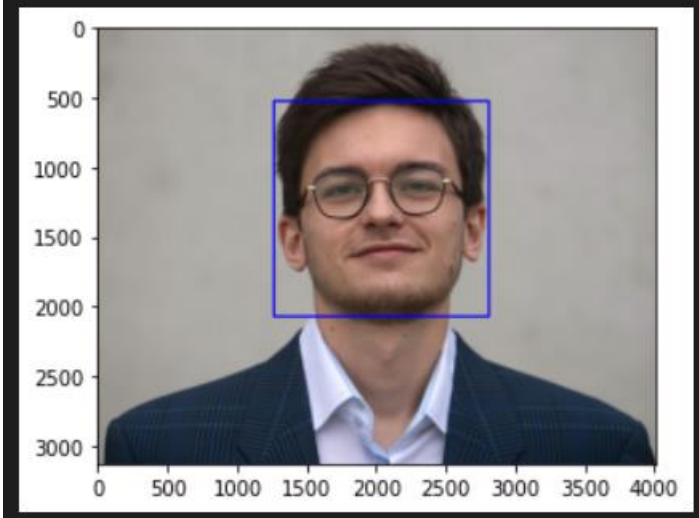
```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Detect object in gray

#detecting face in color_image and getting 4 points(x,y,u,v) around face from the image, and assigning those values to 'faces' variable
faces = faceCascade.detectMultiScale(gray, 1.1, 4)

#using that 4 points to draw a rectangle around face in the image
for (x, y, u, v) in faces:
    cv2.rectangle(color_img, (x,y), (x+u, y+v), (0, 0, 255), 15)
```

✓ 4.7s

```
plt.imshow(color_img)
```

✓ 9.5s

```
<matplotlib.image.AxesImage at 0x2431adfe980>
```



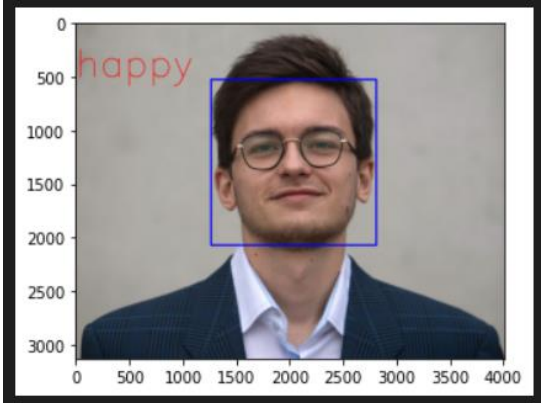

```
font = cv2.FONT_HERSHEY_SIMPLEX #Choice of the font

#Display of the text of the function prediction on color_img
cv2.putText(color_img,prediction['dominant_emotion'],(0,500), font, 12, (255,0,0),8)
```

✓ 0.6s

```
plt.imshow(color_img) #Display the photo
```

✓ 3.8s

```
<matplotlib.image.AxesImage at 0x2431b0ea800>
```



For Videos :

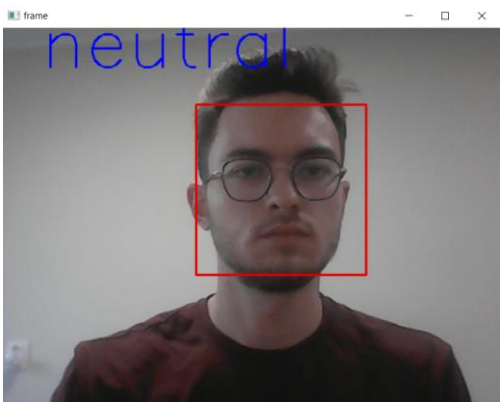
To implement our AI on a video, it's quite the same code unless we read multiple images that create a video instead of just one image.

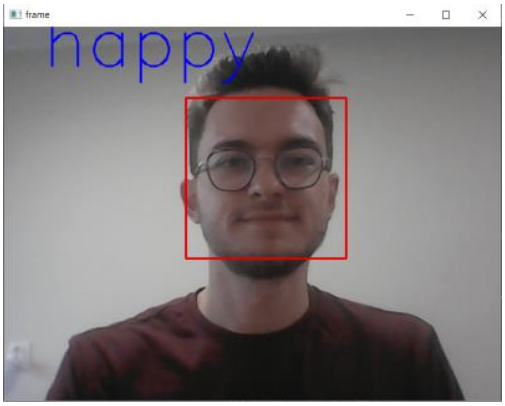
```
import cv2
from deepface import DeepFace

# detect objects and we concentrate on faces
faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades +'haarcascade_frontalface_default.xml')


# define a video capture object. Here it will be the camera of the computer
cam = cv2.VideoCapture(0)

while(True):

    # Capture the video frame
    # by frame
    ret, frame = cam.read() #Reading of the frame

    prediction = DeepFace.analyze(frame, actions = ['emotion']) #Analyse of the emotions from objects in the frame
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(frame, 1.1, 4) #Detection of objects
```

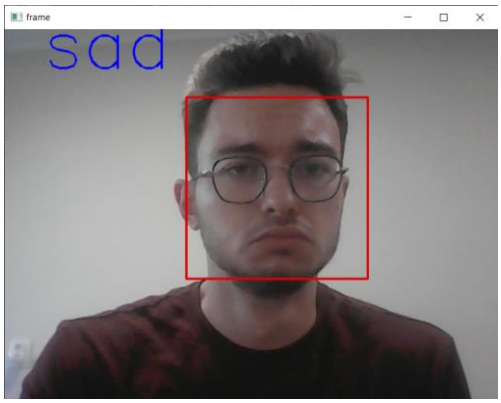
```
    #Creation of a rectangle on the object
    for (x, y, u, v) in faces:
        cv2.rectangle(frame, (x,y), (x+u, y+v), (0, 0, 225), 2)

    font = cv2.FONT_HERSHEY_SIMPLEX

    cv2.putText(frame,prediction['dominant_emotion'],(50,50), font, 3, (255,0,0),2) #Display the emotion on the video

    # Display the resulting frame
    cv2.imshow('frame', frame)

    # the 'q' button is set as the
    # quitting button. We can use an other letter if needed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```

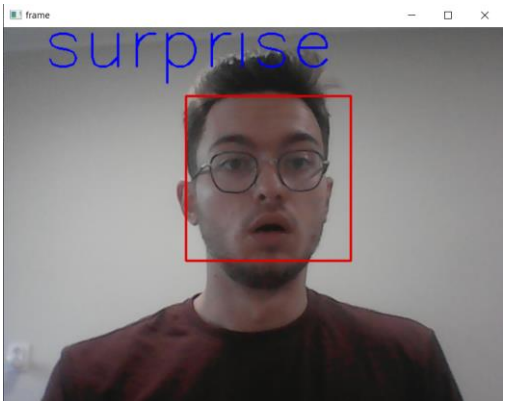
```
# After the loop release the cap object
cam.release()
# Destroy all the windows
cv2.destroyAllWindows()
```

Results :









Conclusion

The task was to develop real-time face recognition AI. In order to solve this task, we researched optimal approach and what could potentially be a solution. In our research, we found out that there was already a library with most optimal up-to-date pre-trained models (deep face library) for emotion recognition. So, with open-CV library (Cascade classifier) which served as a tool to detect faces on the rectangle we manage to create a real-time face recognition AI.

For further knowledge about AI, we can create a neural network with the data set. Make some tests to see the accuracy of the AI and improve it in each test to reach a satisfying accuracy.

References

https://towardsdatascience.com/face-detection-for-beginners-e58e8f21aad9

https://towardsdatascience.com/real-time-face-recognition-an-end-to-end-project-b738bb0f7348

https://www.youtube.com/watch?v=fkgpvkqcoJc

Link project

https://github.com/YohannFrench/BIAI-Project.git