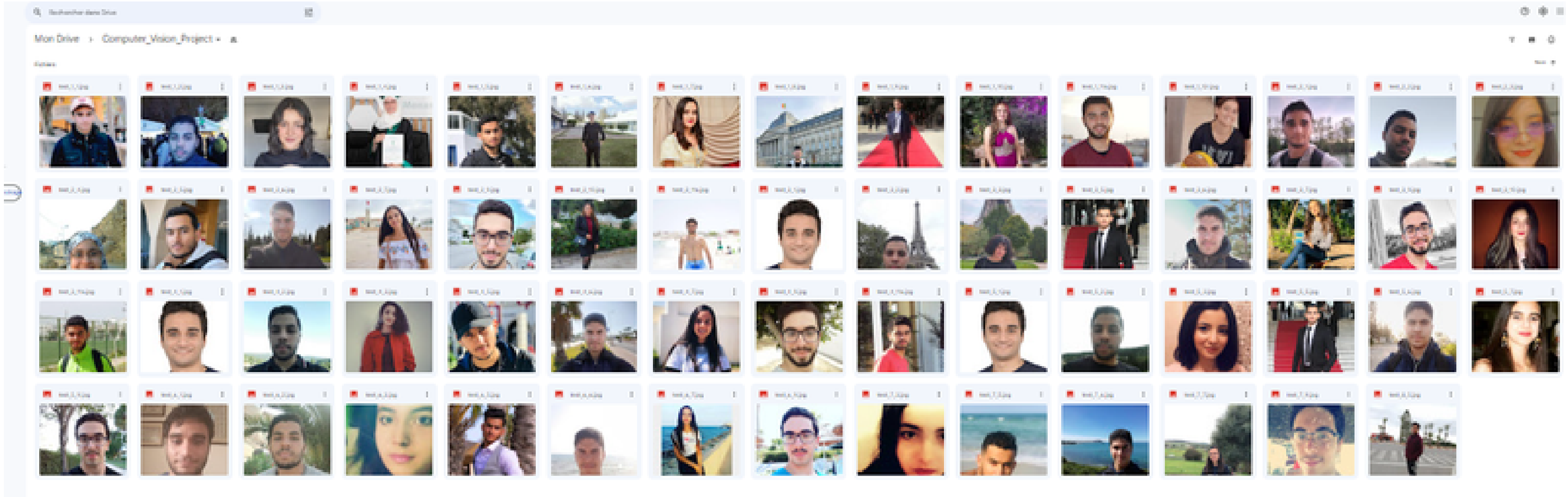
Computer Vision

Reconnaissance Faciale

Ahmed LOUHICHI
Mohamed Aymen BOUYAHIA

«Telle une rose éclose, chaque image partagée sur les réseaux sociaux expose un pétale de notre intimité, risquant d'être cueilli par des mains malveillantes.»

Collecte de Données



Importation des bibliothèques

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

import os
from google.colab import drive
from google.colab.patches import cv2_imshow

from skimage.feature import hog
from skimage import exposure

from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
```

Création d'une liste d'images et de labels

```python
drive.mount('/content/drive')

# Directory path in your Google Drive where the images are stored
directory = '/content/drive/MyDrive/Computer_Vision_Project'

# List to store the images
images = []
labels =[]
images_path = []
# Iterate through all the files in the directory
for filename in os.listdir(directory):
    # Check the file extension (adjust accordingly)
    if filename.endswith('.jpg') or filename.endswith('.png'):
        # Load the image
        image_path = os.path.join(directory, filename)
        images_path.append(image_path)
        print(image_path)
        start_index = image_path.index("test_") + 7
        end_index = image_path.index(".jpg")
        expression = image_path[start_index:end_index]
        labels.append(expression)
        image = cv2.imread(image_path)

        # Add the image to the list
        images.append(image)
```

```
/content/drive/MyDrive/Computer_Vision_Project/test_1_11a.jpg
/content/drive/MyDrive/Computer_Vision_Project/test_1_12r.jpg
/content/drive/MyDrive/Computer_Vision_Project/test_2_11a.jpg
/content/drive/MyDrive/Computer_Vision_Project/test_3_11a.jpg
/content/drive/MyDrive/Computer_Vision_Project/test_4_11a.jpg
/content/drive/MyDrive/Computer_Vision_Project/test_3_1.jpg
Number of loaded images: 59
['1', '1', '1', '1', '2', '2', '2', '2', '2', '2', '3', '3', '3', '3',
```

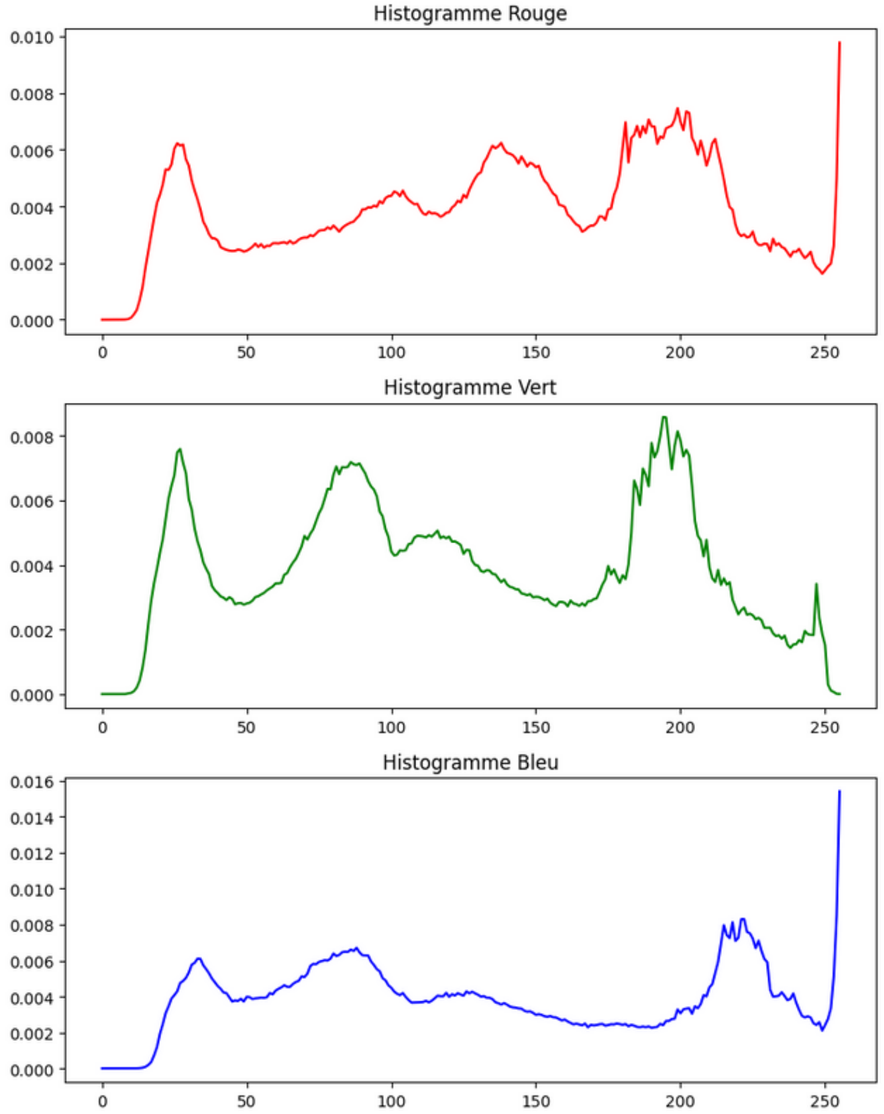
```python
# Fonction pour extraire l'histogramme d'une image en couleur
def extract_color_histogram(image):
    # Séparation des canaux de couleur
    b, g, r = cv2.split(image)

    # Calcul des histogrammes pour chaque canal de couleur
    hist_r = cv2.calcHist([r], [0], None, [256], [0, 256])
    hist_g = cv2.calcHist([g], [0], None, [256], [0, 256])
    hist_b = cv2.calcHist([b], [0], None, [256], [0, 256])

    # Normalisation des histogrammes
    hist_r /= hist_r.sum()
    hist_g /= hist_g.sum()
    hist_b /= hist_b.sum()

    histogram = np.concatenate((hist_r, hist_g, hist_b)).flatten()
    return histogram
```

Une approche bien naive!!!



Modèle KNN en utilisant les histogrammes de couleurs comme features

```
histograms = []
for image in images:
    histogram = extract_color_histogram(image)
    histograms.append(histogram)
histograms = np.array(histograms)

labels = np.array(labels)

# Split the dataset into training and testing subsets
train_data, test_data, train_labels, test_labels = train_test_split(
    histograms, labels, test_size=0.2, random_state=42
)

# Création du classificateur KNN
k = 3 # Nombre de voisins à considérer
knn = KNeighborsClassifier(n_neighbors=k)

# Entraînement du modèle KNN sur les données d'entraînement
knn.fit(train_data, train_labels)

# Prédiction des étiquettes pour les données de test
predicted_labels = knn.predict(test_data)

# Calcul du score
score = knn.score(test_data, test_labels)
```

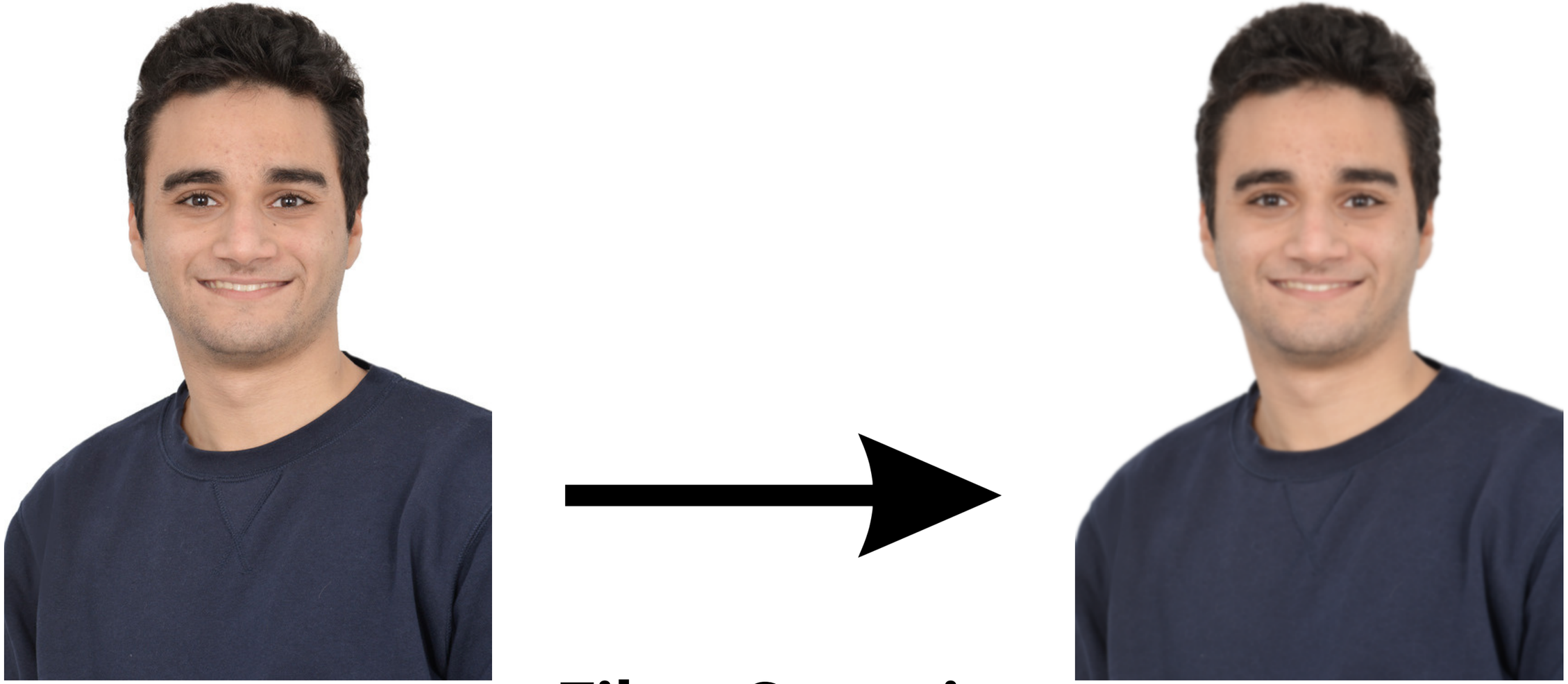
machine learning

train_test_split

Un score de 0.08!!!

```
Predicted labels: ['1' '1' '1' '10' '1' '10' '1' '1' '2' '5' '2' '6']
True labels: ['1' '2' '6' '3' '9' '11a' '11a' '5' '9' '3' '9' '1']
Score: 0.08333333333333333
```

Il faut adopter une nouvelle approche

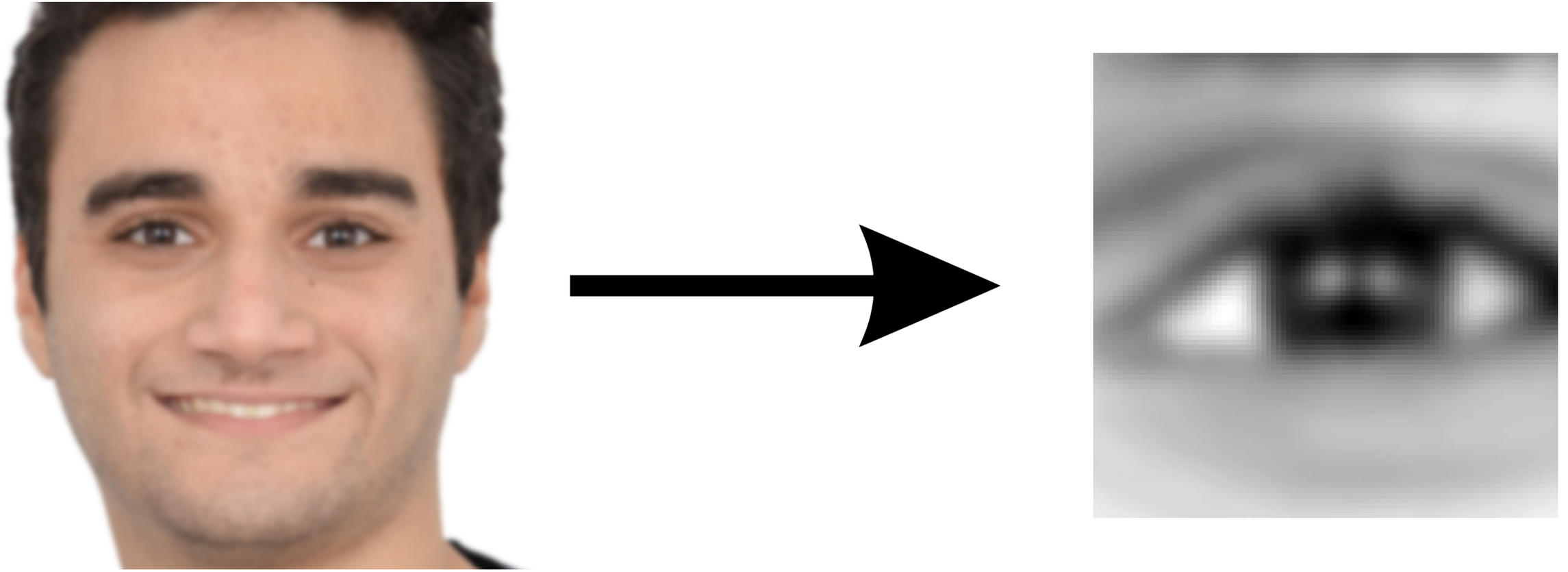


Filtre Gaussien

Il faut adopter une nouvelle approche

Extraction de visages

Il faut adopter une nouvelle approche

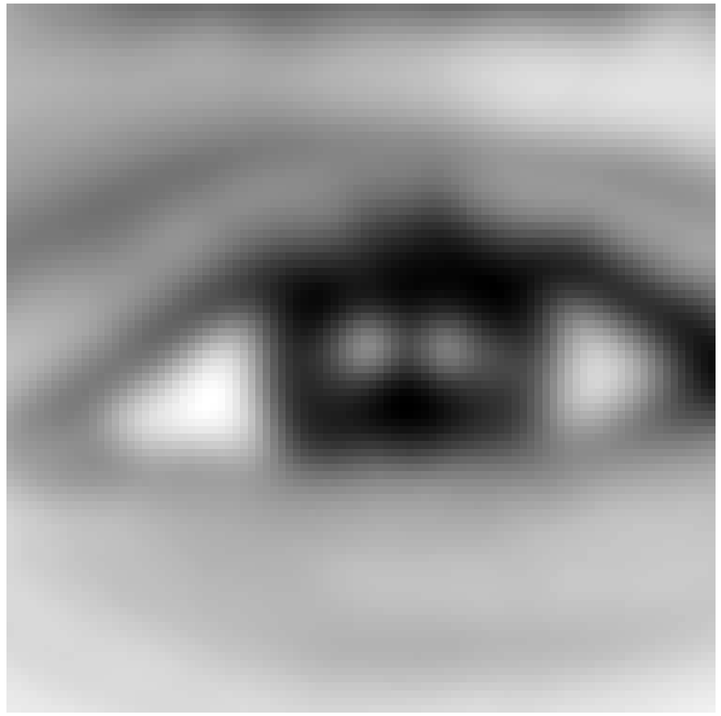


Extraction des yeux

Que doit-on faire maintenant?

On extrait le HOG des images des yeux

Eye ROI



HOG Features - Eye ROI

Modèle KNN

Score: 0.2307692307692307 8

Modèle arbre de décision

Score: 0.1538461538461538 5

Cependant, l'arbre de décsions était plus facile à implémenter

Vers le Deep Learning

Utilisation du modèle pré entrainé Face_recognition

```
!pip3 install face_recognition
```

```
# Mount Google Drive
drive.mount('/content/drive')

# Directory path in your Google Drive where the images are stored
directory = '/content/drive/MyDrive/Computer_vision_api_data'

recognized_names = recognize_faces(directory+"/aymen.jpg", directory + "/wassim.jpg", directory + "/asma.jpg", directory +
                                   directory + "/anas.jpg", directory + "/adam.jpg", directory + "/firas.jpg", directory +
                                   directory + "/rania.jpg", directory + "/sirine.jpg", directory + "/sadok.jpg", "test.j
print(recognized_names)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=T
['Aymen']
```

Merci de Votre attention

Lien pour la base de donnée utilisée

https://drive.google.com/drive/folders/14ssGTAO8wnCcEsNcfzlSHYLX6bndx2Tx?usp=sharing