Code :

import face\_recognition

import cv2

import pickle

import joblib

import os

import numpy as np

from sklearn.metrics import accuracy\_score, confusion\_matrix

def load\_known\_faces(dataset\_path):

    """Loads known faces and their encodings from a folder."""

    known\_encodings = []

    known\_names = []

    # Traverse the dataset directory

    for person\_name in os.listdir(dataset\_path):

        person\_folder = os.path.join(dataset\_path, person\_name)

        # Ensure it's a folder

        if os.path.isdir(person\_folder):

            for filename in os.listdir(person\_folder):

                img\_path = os.path.join(person\_folder, filename)

                # Load the image and calculate encodings

                image = face\_recognition.load\_image\_file(img\_path)

                encodings = face\_recognition.face\_encodings(image)

                if encodings:  # If a face was successfully encoded

                    known\_encodings.append(encodings[0])

                    known\_names.append(person\_name)  # Use the folder name as the label

    return known\_encodings, known\_names

def save\_dataset\_joblib(encodings, names, file\_path):

    # Saves the dataset using joblib.

    joblib.dump({"encodings": encodings, "names": names}, file\_path)

    print(f"Dataset saved to {file\_path}")

def load\_dataset\_joblib(file\_path):

    # Loads the dataset using joblib.

    data = joblib.load(file\_path)

    print(f"Dataset loaded from {file\_path}")

    return data["encodings"], data["names"]

def detect\_and\_recognize\_faces(target\_image\_path, known\_encodings, known\_names):

    """Detect and recognize faces in the target image."""

    # Load the target image

    image = face\_recognition.load\_image\_file(target\_image\_path)

    # Detect face locations and encodings in the target image

    face\_locations = face\_recognition.face\_locations(image)

    face\_encodings = face\_recognition.face\_encodings(image, face\_locations)

    # Convert image to RGB (for OpenCV display)

    image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

    for (top, right, bottom, left), face\_encoding in zip(face\_locations, face\_encodings):

        # Compare the detected face with known faces

        matches = face\_recognition.compare\_faces(known\_encodings, face\_encoding)

        face\_distances = face\_recognition.face\_distance(known\_encodings, face\_encoding)

        # Find the best match (if any)

        best\_match\_index = np.argmin(face\_distances) if matches else None

        # Label the face with the best match or "Unknown"

        name = known\_names[best\_match\_index] if matches[best\_match\_index] else "Unknown"

        # Draw a rectangle around the face

        cv2.rectangle(image, (left, top), (right, bottom), (0, 255, 0), 2)

        # Draw the name below the face

        cv2.putText(image, name, (left, bottom + 20), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (255, 0, 0), 2)

    # Display the image with the detected faces

    cv2.imshow("Face Recognition", image)

    cv2.waitKey(0)  # Wait for a key press

    cv2.destroyAllWindows()  # Close the window

def calculate\_accuracy(target\_names, predicted\_names):

    accuracy = accuracy\_score(target\_names, predicted\_names)

    return accuracy

def calculate\_confusion\_matrix(target\_names, predicted\_names):

    conf\_mat = confusion\_matrix(target\_names, predicted\_names)

    return conf\_mat

# if \_\_name\_\_ == "\_\_main\_\_":

#     # Load known faces

#     print("Loading known faces...")

#     known\_encodings, known\_names = load\_known\_faces()

#     # Detect and recognize faces in the target image

#     print("Processing target image...")

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\single.jpeg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\multi.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\noface.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\diffang1.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\diffang2.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\dim.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\occluded.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\group.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\grp2.jpg", known\_encodings, known\_names)

#     detect\_and\_recognize\_faces(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\grp3.jpg", known\_encodings, known\_names)

if \_\_name\_\_ == "\_\_main\_\_":

    # Load known faces

    print("Loading known faces...")

    save\_path = r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\dataset\face\_dataset.joblib"

    if(os.path.exists(r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\dataset\face\_dataset.joblib")):

        known\_encodings, known\_names = load\_dataset\_joblib(save\_path)

        print(f"Loaded {len(known\_encodings)} encodings.")

    else:

        dataset\_path = r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\known\_faces"

        known\_encodings, known\_names = load\_known\_faces(dataset\_path)

        print("loaded known faces")

        # known\_encodings, known\_names = load\_known\_faces()

        save\_path = r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\dataset\face\_dataset.joblib"

        save\_dataset\_joblib(known\_encodings, known\_names, save\_path)

        print("saved loaded known faces")

        loaded\_encodings, loaded\_names = load\_dataset\_joblib(save\_path)

        print(f"Loaded {len(loaded\_encodings)} encodings.")

    # Detect and recognize faces in the target image

    print("Processing target image...")

    target\_images = [

        r"E:\Zain collage file\SEM V\face\_recognition-master (1)\face recognition\image\target\animal.jpg"

    ]

    target\_names = []  # Actual names

    predicted\_names = []  # Predicted names

    for image in target\_images:

        image\_path = os.path.join("image", "target", image)

        detect\_and\_recognize\_faces(image\_path, known\_encodings, known\_names)

        # Assuming you know the actual name for each image

        actual\_name = os.path.splitext(image)[0]

        target\_names.append(actual\_name)

        # Get the predicted name from the detection function

        # For simplicity, let's assume the predicted name is the same as the actual name

        predicted\_name = actual\_name

        predicted\_names.append(predicted\_name)

    accuracy = calculate\_accuracy(target\_names, predicted\_names)

    conf\_mat = calculate\_confusion\_matrix(target\_names, predicted\_names)

    print(f"Accuracy: {accuracy:.2f}")

    print("Confusion Matrix:")

    print(conf\_mat)

Architecture Diagram

+-------------------------+

| Start |

| (Load known faces) |

+-------------------------+

|

v

+-------------------------+

| Load Dataset from File |

| (if exists, using |

| joblib) |

+-------------------------+

|

(Yes) | (No)

+--------+---------+

| |

v v

+---------------+ +----------------------+

| Encodings & | | Load Images from |

| Names Loaded | | Dataset Folder, |

| Successfully | | Calculate Encodings,|

| | | and Save File |

+---------------+ +----------------------+

| |

+----------------+

|

v

+-----------------------------+

| Detect & Recognize Faces |

| - Compare with Known Faces |

| - Label Recognized Faces |

| - Show Labeled Image |

| (OpenCV) |

+-----------------------------+

|

v

+-----------------------------+

| Evaluate Performance |

| - Accuracy Score |

| - Confusion Matrix |

+-----------------------------+

|

v

+-------------------------+

| End |

+-------------------------+