Face Recognition Project

Overview

This project implements a face recognition system using Python's `face_recognition` library and OpenCV. The system captures video from a webcam, detects faces in real-time, and recognizes known individuals based on their facial features. It serves as an introduction to computer vision and machine learning applications in Python.

Features

- Real-time face detection and recognition using a webcam.
- Ability to recognize multiple known faces.
- User-friendly interface that displays names of recognized individuals.

Technologies Used

- **Python**: The primary programming language used for the implementation.
- **face_recognition**: A Python library for face recognition built on top of `dlib`. It provides simple APIs to identify faces in images and videos.
- **OpenCV (cv2)**: A powerful library used for image processing and computer vision tasks.

Installation

To run this project, ensure you have Python installed along with the required libraries. You can install the necessary libraries using pip:

```
bash
Copy code
pip install face_recognition opencv-python
```

Code Explanation

1. Importing Libraries

```
python
Copy code
import face_recognition
import cv2
```

- **face_recognition**: Used for facial recognition tasks.
- **cv2**: Used for video capture and image processing.

2. Loading Known Images

The system loads images of known individuals and encodes their faces for recognition:

```
python
Copy code
try:
    known_image1 = face_recognition.load_image_file("path/to/image1.jpg")
    known_encoding1 = face_recognition.face_encodings(known_image1)[0]
except IndexError:
    print("No faces found in the first image.")
    exit()

try:
    known_image2 = face_recognition.load_image_file("path/to/image2.jpg")
    known_encoding2 = face_recognition.face_encodings(known_image2)[0]
except IndexError:
    print("No faces found in the second image.")
    exit()
```

- The code handles the scenario where no faces are found in the provided images by using exception handling (`try` and `except`).

3. Capturing Video from Webcam

```
python
Copy code
video_capture = cv2.VideoCapture(0)
```

- Initializes the webcam for live video capture.

4. Main Loop for Face Detection

In this loop, frames are continuously captured from the webcam:

```
python
Copy code
while True:
    ret, frame = video_capture.read()
```

5. Face Detection and Recognition

The code detects faces and compares them with known encodings:

```
python
Copy code
face_locations = face_recognition.face_locations(frame)
face_encodings = face_recognition.face_encodings(frame, face_locations)
```

- It retrieves the locations of detected faces and computes their encodings.

6. Matching Faces

The code checks if the detected faces match any known individuals:

```
python
Copy code
for (top, right, bottom, left), face_encoding in zip(face_locations,
face_encodings):
```

```
    matches1 = face_recognition.compare_faces([known_encoding1],
face_encoding)
    matches2 = face_recognition.compare_faces([known_encoding2],
face_encoding)

    name = "UNKNOWN"

    if matches1[0]:
        name = "HITESH"
    elif matches2[0]:
        name = "VAIBHAV"
```

- If a match is found, it assigns the corresponding name.

7. Displaying Results

The recognized faces are displayed with rectangles and names:

```
python
Copy code
cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)
font = cv2.FONT_HERSHEY_DUPLEX
cv2.putText(frame, name, (left + 6, bottom - 6), font, 0.5, (255, 255,
255), 1)
```

- The resulting frame is displayed in a window.

8. Cleanup

```
python
Copy code
video_capture.release()
cv2.destroyAllWindows()
```

- Releases the video capture object and closes all OpenCV windows.

Usage

1. Place your known images in the project directory.
2. Update the image paths in the code to match the names of your known images.
3. Run the script to start the face recognition process.
4. Press 'q' to exit the video feed.

Future Enhancements

- Implement the ability to recognize additional faces dynamically.
- Integrate with a database for storing and managing known individuals.
- Optimize performance to handle multiple faces more efficiently.

Conclusion

This face recognition project serves as a practical application of machine learning and computer vision techniques. It demonstrates the basics of real-time face detection and recognition and provides a foundation for further exploration in the field of AI and image processing.