import cv2

# Load pre-trained face detection model

face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')

# Load pre-trained face recognition model (e.g., using OpenCV DNN module or custom model)

face\_recognition\_model = ...

# Function to detect and recognize faces in an image

def detect\_and\_recognize\_faces(image):

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

faces = face\_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))

for (x, y, w, h) in faces:

# Crop face region from the image

face = image[y:y+h, x:x+w]

# Preprocess face (e.g., resize, normalize pixel values)

# Perform face recognition (extract features and compare with known faces)

recognized\_identity = face\_recognition\_model.predict(face)

# Draw bounding box and label on the image

cv2.rectangle(image, (x, y), (x+w, y+h), (0, 255, 0), 2)

cv2.putText(image, recognized\_identity, (x, y-10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9, (0, 255, 0), 2)

return image

# Example usage for processing images

image = cv2.imread('image.jpg')

output\_image = detect\_and\_recognize\_faces(image)

cv2.imshow('Output', output\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

# Example usage for processing video stream

video\_capture = cv2.VideoCapture(0)

while True:

ret, frame = video\_capture.read()

if ret:

output\_frame = detect\_and\_recognize\_faces(frame)

cv2.imshow('Video', output\_frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

video\_capture.release()

cv2.destroyAllWindows()