Poject: Real-time Facial Emotion Recognition using OpenCV and Deepface

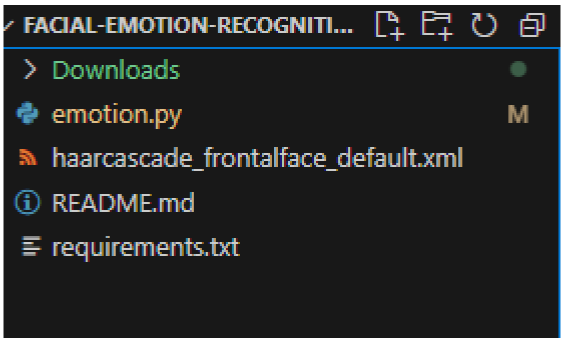
Group: C112-Nami Desai, C113-Vidhie Jhunjhunwala, C130-Vihaal Bhulla

This project demonstrates the implementation of real-time facial emotion recognition using the `deepface` library and OpenCV. The objective is to capture live video from a webcam, identify faces within the video stream, and predict the corresponding emotions for each detected face. The emotions predicted are displayed in real-time on the video frames.

To streamline this process, we've utilized the `deepface` library, a deep learning-based facial analysis tool that employs pre-trained models for accurate emotion detection. TensorFlow is the underlying framework for the deep learning operations. Additionally, we leverage OpenCV, an open-source computer vision library, to facilitate image and video processing.

Code:

```
import cv2

from deepface import DeepFace


# Load face cascade classifier

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')


# Start capturing video

cap = cv2.VideoCapture(0)

while True:

    # Capture frame-by-frame

    ret, frame = cap.read()


    # Convert frame to grayscale

    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)


    # Detect faces in the frame

    faces = face_cascade.detectMultiScale(gray_frame, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))
    for (x, y, w, h) in faces:
        # Extract the face ROI (Region of Interest)
        face_roi = frame[y:y + h, x:x + w]


        # Analyze the face for emotion
    try:
            # Analyze the face using DeepFace
            preds = DeepFace.analyze(face_roi, actions=['emotion'], enforce_detection=False)
            # Extract the dominant emotion
            emotion = preds[0]['dominant_emotion']


            # Draw rectangle around face and label with predicted emotion
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)  # Green rectangle
            cv2.putText(frame, emotion, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2)  #
Green text
    except Exception as e:
            print(f"Error analyzing face: {e}")


    # Display the resulting frame
    cv2.imshow('Real-time Emotion Detection', frame)


    # Press 'q' to exit
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
```


FACIAL-EMOTION-RECOGNITI...

Downloads

emotion.py M

haarcascade_frontalface_default.xml

README.md

requirements.txt

Approach

1. Import Essential Libraries: Import `cv2` for video capture and image processing, as well as `deepface` for the emotion detection model.

2. Load Haar Cascade Classifier: Utilize `cv2.CascadeClassifier()` to load the XML file for face detection.

3. Video Capture Initialization: Employ `cv2.VideoCapture()` to initiate video capture from the default webcam.

4. Frame Processing Loop: Enter a continuous loop to process each video frame.

5. Grayscale Conversion: Transform each frame into grayscale using `cv2.cvtColor()`.

6. Face Detection: Detect faces within the grayscale frame using `face_cascade.detectMultiScale()`.

7. Face Region Extraction: For each detected face, extract the Region of Interest (ROI) containing the face.

8. Preprocessing: Prepare the face image for emotion detection by employing the built-in preprocessing function from the `deepface` library.

9. Emotion Prediction: Utilize the pre-trained emotion detection model provided by the `deepface` library to predict emotions.

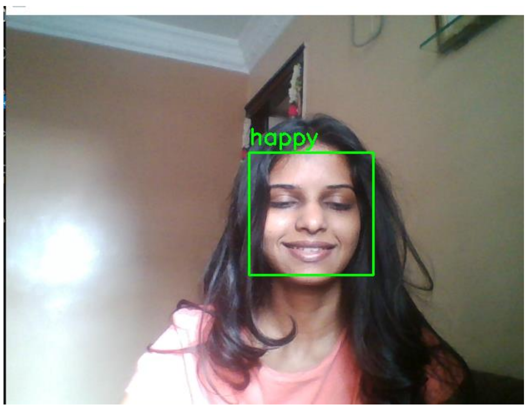
10. Emotion Labeling: Map the predicted emotion index to the corresponding emotion label.

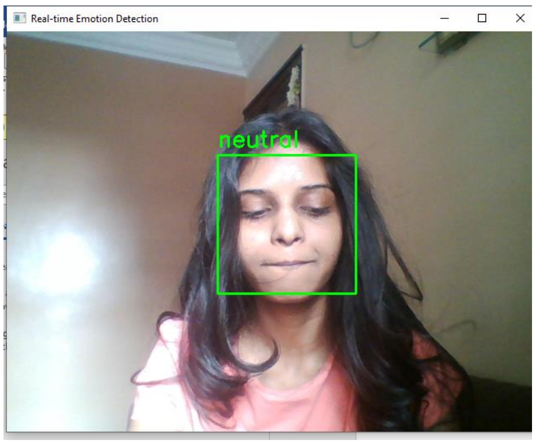
11. Visual Annotation: Draw rectangles around the detected faces and label them with the predicted emotions via `cv2.rectangle()` and `cv2.putText()`.

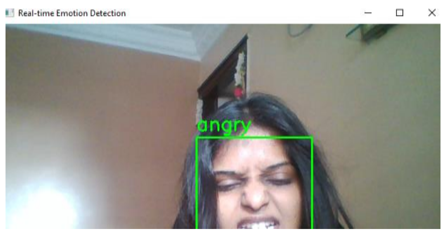
12. Display Output: Present the resulting frame with the labeled emotion using `cv2.imshow()`.

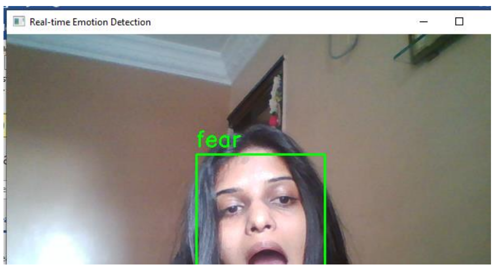
13. Loop Termination: If the 'q' key is pressed, exit the loop.

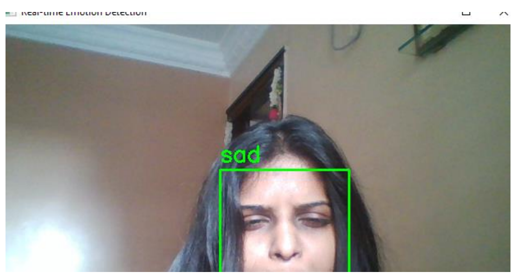
14. Cleanup: Release video capture resources and close all windows with `cap.release()` and `cv2.destroyAllWindows()`.


happy


Real-time Emotion Detection
neutral


Real-time Emotion Detection
angry


Real-time Emotion Detection
fear


Real-time Emotion Detection
sad