Face Detection Using Haar Cascade Classifier using OpenCV and Count Total Faces in Image

Code Explanation

#Importing cv2 Package

import cv2

Open CV Stands for Open Source Computer Vision , It is used for Computer Vision and Image Processing , That provides set of tools for various tasks related to computer Vision and Image Analysis , It is written in C++.

#Reading the Image

img = cv2.imread(r'C:\YASWANTH M\KARE-VII SEMESTER\AM\new_img.jpg')

Here, we read an image file named "new_img.jpg" from the specified file path using the cv2.imread() function. The image is stored in the variable img, which represents a NumPy array containing the image data.

#Printing the Image

[[[255 255 249] [255 255 249] [254 254 248] ... [106 120 119] [103 116 118] [102 115 117]] [[247 245 237] [253 252 242] [255 255 247] ... [91 106 108] [90 105 108] [91 106 109]] [[253 252 238] [254 254 238] [253 252 238] ... [95 111 117] [99 115 122] [98 114 121]] ...
...
... [222 220 220] [254 252 252] [252 250 249]]]

The Image Data Will be stored in Matrix form , Each Element Represent Pixel Value , shape indicates Dimensions of Image

#Converting Image to Gray Scale

gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

The Above Line converts RGB Image to Gray Scale , Gray Scale Images have single channel

 # Loading the required haar-cascade xml classifier file

haar_cascade = cv2.CascadeClassifier(r'C:\YASWANTH M\KARE-VII SEMESTER\AM\Haar\Haar-Training\Haar Training\cascade2xml\myfacedetector.xml')

The XML Files is created by Training with Both Positive and Negative Images , We Load XML File using cascade Classifier Function , Haar Cascade Classifier is a

Machine Learning Based Classification Algorithm for deteecting various objects like faces based on Haar Features

The XML file is created by annotation of faces on images and that is stored in a text file , then we run a bat file , direct xml file will be generated

 # Applying the face detection method on the grayscale image
 # Reduced the minNeighbors parameter to 8 to allow more detections
faces_rect = haar_cascade.detectMultiScale(gray_img, scaleFactor=1.1, minNeighbors=8)

Applying Face detection method on grayscale images using haar cascade classifier , detect multi scale detects objects in image , Faces at multiple scale in image , The scale factor and min neighbors control the sensitivity and accuracy of face detection, reducing Min neigbours to 8 allowing model to detect more number of faces , if they are even close together

 # Counting the number of detected faces
num_faces = len(faces_rect)
It counts Number of detected face from image by calculating length of images being identified

 # Draw text on the image to display the count of faces
text = f"Number of faces: {num_faces}"
cv2.putText(img, text, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
Displays Total Number of Images

 # Iterating through rectangles of detected faces
```
for (x, y, w, h) in faces_rect:
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)
cv2.imshow('Detected faces', img)
cv2.waitKey(0)
```

- This Code Iterates through rectangles of detected faces and draws green rectangles around images classified on original Image
- The Image is displayed in window displaying as detected faces
- Wait Key function waits until a key is pressed
- success of face detection largely depends on the accuracy and quality of the Haar cascade classifier used.