**Face Detection with openCV**

Face detection is a type of application in computer vision where algorithms are developed and trained to properly locate faces or objects (in object detection, a related system), in images. These can be in real time from a video camera or from photographs.

Haar Classifier:

Haar cascade can be used for face detection which is a machine learning algorithm where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

OpenCV already contains many pre-trained Haar classifiers for face, eyes, smile etc.

To apply these pre-trained classifiers, follow the following steps:

1. Load the XML file of required classifiers.
2. Load image in gray-scale mode as OpenCV mostly operates in gray scale.
3. Apply necessary classifiers on the image.

Python code:

#import libraries

import numpy as np

import cv2

#Load required trained XML classifiers by giving their actual path location

face\_cascade = cv2.CascadeClassifier('C:\\anaconda\\mainfile\\lib\\site-packages\\data\\haarcascade\_frontalface\_default.xml');

eye\_cascade = cv2.CascadeClassifier('C:\\anaconda\\mainfile\\lib\\site-packages\\data\\haarcascade\_eye.xml');

#read image

img = cv2.imread('faces.jpg');

#convert the image to gray scale

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY);

#detect faces of all sizes

faces = face\_cascade.detectMultiScale(gray, 1.3, 5);

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

#draw rectangle around the face

cv2.rectangle(img,(x,y),(x+w,y+h),(255,100,100),2);

roi\_gray = gray[y:y+h, x:x+w];

roi\_color = img[y:y+h, x:x+w];

#detect eyes in every face

eyes = eye\_cascade.detectMultiScale(roi\_gray);

for (ex,ey,ew,eh) in eyes:

#draw rectangle around the eyes

cv2.rectangle(roi\_color,(ex,ey),(ex+ew,ey+eh),(100,255,100),2);

cv2.imshow('img',img);

k = cv2.waitKey(0);

if k == 27: # wait for ESC key to exit

cv2.destroyAllWindows();

Output:

