Pooja Naskar - 30

Aim:

To study Detecting and Recognizing Faces

Objective:

To Conceptualizing Haar Cascades Getting Haar cascade data Using Open CV to Perform face detections performing face detection on still images

Theory:

Introduction

Discover object detection with the Haar Cascade algorithm using OpenCV. Learn how to employ this classic method for detecting objects in images and videos. Explore the underlying principles, step-by-step implementation, and real-world applications. From facial recognition to vehicle detection, grasp the essence of Haar Cascade and OpenCV's role in revolutionizing computer vision. Whether you're a novice or an expert, this article will equip you with the skills to harness the potential of object detection in your projects.

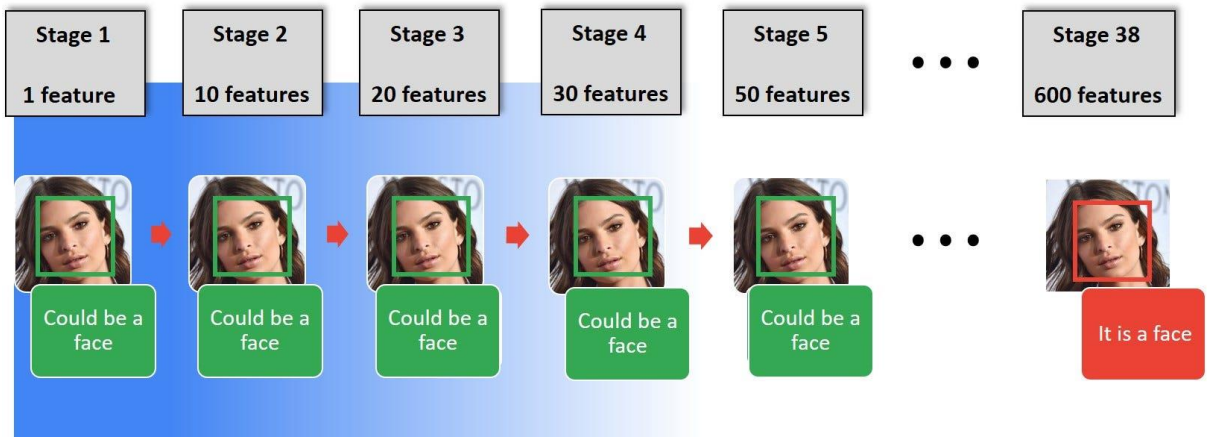


Table of contents

Why Use Haar Cascade Algorithm for Object Detection?

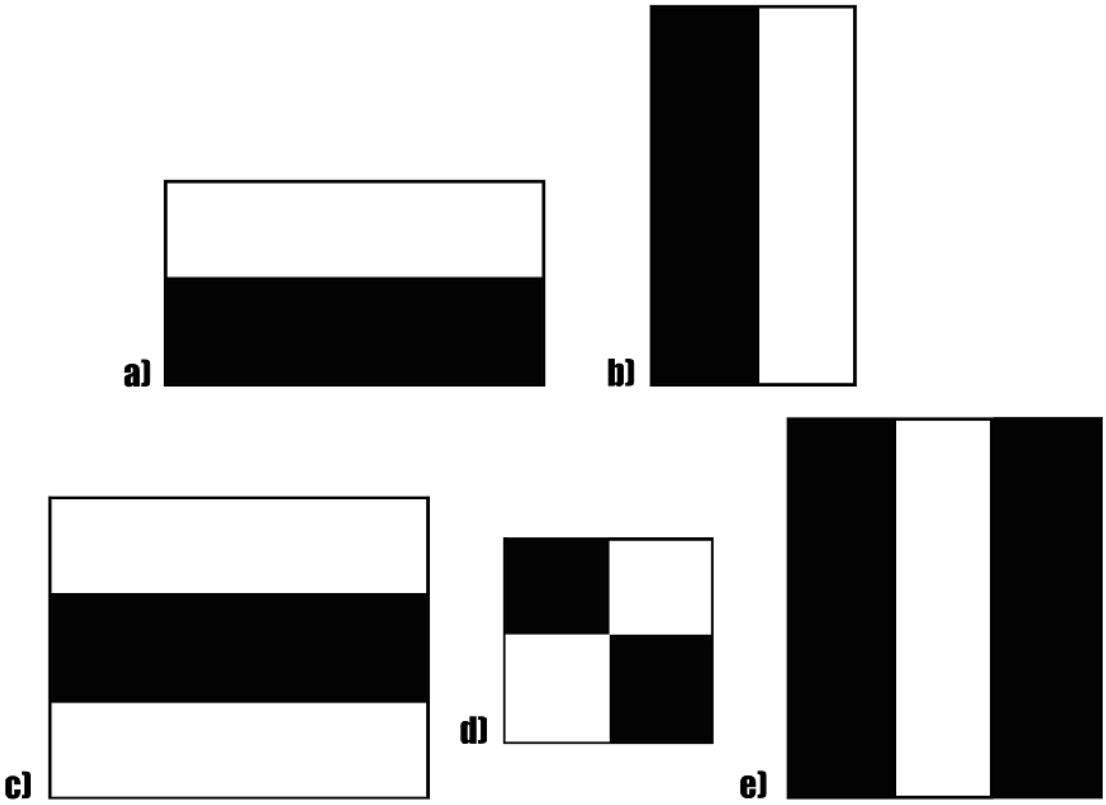
Identifying a custom object in an image is known as object detection. This task can be done using several techniques, but we will use the haar cascade, the simplest method to perform object detection in this article.

What is Haar Cascade Algorithm?

Haar cascade is an algorithm that can detect objects in images, irrespective of their scale in image and location.

This algorithm is not so complex and can run in real-time. We can train a haar-cascade detector to detect various objects like cars, bikes, buildings, fruits, etc.

Haar cascade uses the cascading window, and it tries to compute features in every window and classify whether it could be an object.



Haar cascade works as a classifier. It classifies positive data points → that are part of our detected object and negative data points → that don't contain our object.

- Haar cascades are fast and can work well in real-time.
- Haar cascade is not as accurate as modern object detection techniques are.
- Haar cascade has a downside. It predicts many false positives.
- Simple to implement, less computing power required.

Conceptualizing Haar Cascades:

Pre-trained Haar cascades are kept in a repository by the OpenCV library. The majority of these Haar cascades are either:

➢ Face detection

➢ Eye detection

➢ Mouth detection

➢ Full/partial body detection

A machine learning algorithm called a Haar classifier or a Haar cascade classifier finds objects in images and videos. The first real-time face detection employed Haar classifiers. Similar to other machine learning models, this algorithm needs a lot of positive photographs of faces and negative images of objects that aren't faces to train the classifier.

Essentially, a Haar feature is a detection window position where calculations are done on adjacent rectangular sections. In order to calculate the difference between the sums, the pixel intensities in each region must first be added together.

Getting Haar Cascade Data:

Pre-trained Haar cascades are kept in a repository by the OpenCV library. The majority of these Haar cascades are either:

Using Open CV to perform Face Detection:

Four stages are involved in utilizing Open CV to do face detection using the Haar Cascade.

Haar-feature choice A Haar-like feature is made up of dark and light regions. It produces a single number by measuring the difference between the sum of the intensities of the bright and dark zones. It is necessary to remove valuable components in order to identify an object.

Building Integral Images In the integral image, the left- and right-most neighbors of each pixel are summed to determine its value. The process of extracting Haar-like features, which involves calculating the difference between rectangular bright and dark areas, is substantially sped up using integral images.

AdaBoost Instruction: This algorithm selects the best features out of all the features.It Performing Face detection on a still image:

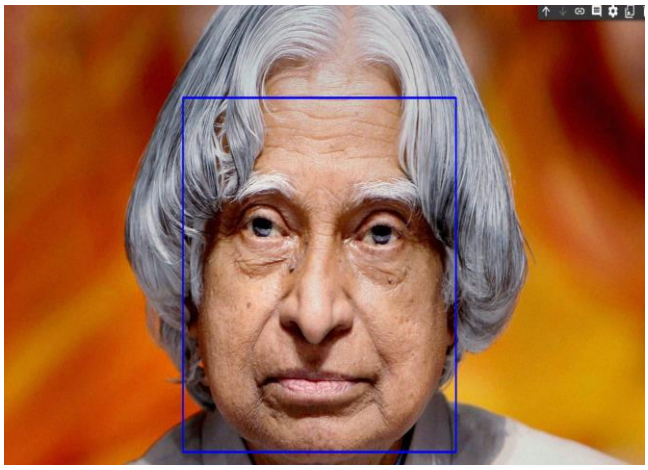
You can perform face detection on a still image by doing the following:

- Adding an image and changing it to grayscale.
- Downloading the necessary XML classifier file for the haar-cascade.
- Utilizing the grayscale image with the face detection technique.
- Iterating across rectangles where faces have been detected

Code:

```
from imutils import paths

import cv2

from google.colab.patches import cv2_imshow

detector = cv2.CascadeClassifier("/content/sample_data/haarcascade_frontalface_default.xml")

img = cv2.imread("/content/sample_data/abdul kalam.jpg")

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

faces = detector.detectMultiScale(gray,1.1,4)

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

  cv2.rectangle(img, (x, y), (x + w, y + h), (255,0, 0), 3)

cv2_imshow(img)
```

Output:



Conclusion :

The simplest technique for object detection is Haar Cascade. No matter where they are in the image or how big they are, objects can be found using the process known as the Haar cascade. It makes an effort to calculate features in each window and determine whether something might be an item.

We successfully used the Haar cascade approach for object detection in this experiment.