01 –Ayush Bujare

Aim: To Perform Detecting and Recognizing Objects/Text

Objective: Object Detection and recognition techniques HOG descriptor TheScale issues The location issue Non-maximum (or non-maxima) suppression vector machine people detection

Theory:

Object detection and recognition Techniques :-

Object recognition is a computer vision technique used to identify, locate, and classify objects in digital images or real-life scenarios. It is an applied artificial intelligence approach that repurposes a computer as an object detector so it can scan an image or video from the real world. It understands the object's features and interprets its purpose just like humans do.

Object recognition combines four techniques: image recognition object localization, object detection, and image segmentation. Object recognition decodes the features and predicts the category or class of image through a classifier, for example, supervised machine learning models like Support Vector Machine (SVM), Adaboost, Boosting, or Decision Tree. Object recognition algorithms are coded in Darknet, an open-source neural network framework written in C, Cuda, or Python.

HOG descriptors

HOG is a feature descriptor, so it belongs to the same family of algorithms as scale invariant feature transform (SIFT), speeded-up robust features (SURF), and Oriented FAST and rotated BRIEF (ORB). . Like other feature descriptors, HOG is capable of delivering the type of information that is vital for feature matching, as well as for object detection and recognition.Most commonly, HOG is used for object detection. The algorithm – and, in particular, its use as a people detector – was popularized by Navneet Dalal and Bill Triggs in their paper Histograms of Oriented Gradients for Human Detection (INRIA, 2005). HOG's internal mechanism is really clever; an image is divided into cells and a set of gradients is calculated for each cell. Each gradient describes the change in pixel intensities in a given direction. Together, these gradients form a histogram representation of the cell.

The Location issue:-

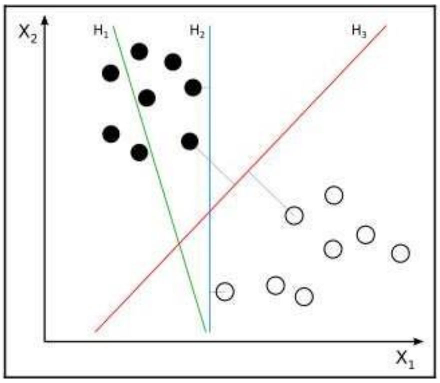
Like other detectors, a HOG-based detector needs to cope with variations in objects' location and scale. The need to search in various locations is addressed by moving a fixed size sliding window across an image. The need to search at various scales is addressed by scaling the image to various sizes, forming a so-called image pyramid Suppose we are using a sliding window to perform people detection on an image. We slide our window in small steps, just a few pixels at a time, so we expect that it will frame any given person multiple times. Assuming that overlapping detections are indeed one person, we do not want to report multiple locations but, rather, only one location that we believe to be correct. In other words, even if a detection at a given location has a good confidence score, we might reject it if an overlapping detection has a better confidence score; thus, from a set of overlapping detections, we would choose the one with the best confidence score.

Non-maximum(or Non-maxima)Suppression:-

A typical implementation of NMS takes the following approach: 1. Construct an image pyramid. 2. Scan each level of the pyramid with the sliding window approach, for object detection. For each window that yields a positive detection (beyond a certain arbitrary confidence threshold), convert the window back to the original image's scale. Add the window and its confidence score to a list of positive detections. 3. Sort the list of positive detections by order of descending confidence score so that the best detections come first in the list. 4. For each window, W, in the list of positive detections, remove all subsequent windows that significantly overlap with W. We are left with a list of positive detections that satisfy the criterion of NMS. Besides NMS, another way to filter the positive detections is to eliminate any subwindows. When we speak of a subwindow (or subregion), we mean a window (or region in an image) that is entirely contained inside another window (or region). To check for subwindows, we simply need to compare the corner coordinates of various window rectangles. We will take this simple approach in our first practical example, in the Detecting people with HOG descriptors section. Optionally, NMS and suppression of subwindows can be combined

Support vector machines

Given labeled training data, an SVM learns to classify the same kind of data by finding an optimal hyperplane, which, in plain English, is the plane that divides differently labeled data by the largest possible margin



Hyperplane H1 (shown as a green line) does not divide the two classes (the black dots versus the white dots). Hyperplanes H2 (shown as a blue line) and H3 (shown as a red line) both divide the classes; however, only hyperplane H3 divides the classes by a maximal margin.

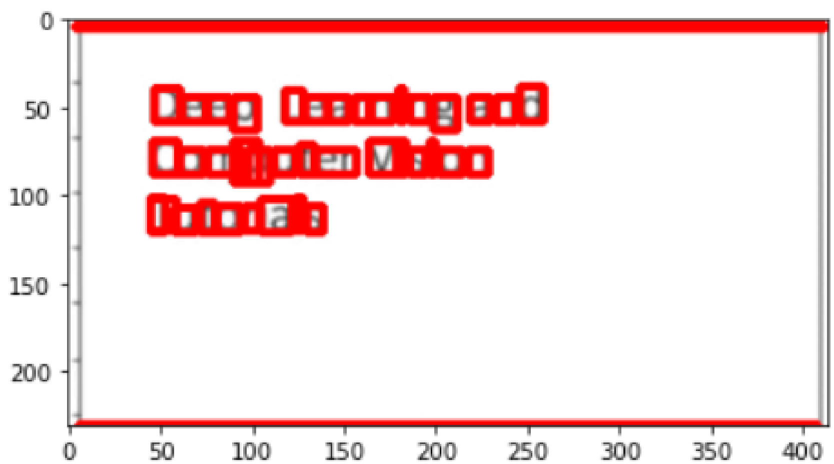
Code :-

```
import cv2
import pytesseract
import numpy as np
import matplotlib.pyplot as plt
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract'
font_scale = 1.5
font = cv2.FONT_HERSHEY_PLAIN
cap = cv2.VideoCapture(1)
 #cap.set(cv2.CAP_PROP_FPS, 170)
#Check if the webcam is opened correctly
if not cap.isOpened():
    cap = cv2.VideoCapture(0)
if not cap.isOpened():
    raise IOError("Cannot open webcam")
cntr = 0;
while True:
    ret ,frame = cap.read()
    cntr= cntr+1;
    if ((cntr%20)==0):
        imgH, imgw,_ = frame.shape
        #eye_cascade = cv2.CascadeClassifier(cv2.data. haarcascades + +"haarcascade_eye.xml")
        x1,y1,w1,h1= 0,0,imgH, imgw
        imgchar = pytesseract.image_to_string(frame)
        imgboxes = pytesseract.image_to_boxes (frame)
        for boxes in imgboxes.splitlines():
            boxes = boxes.split(' ')
            x,y,w,h= int(boxes[1]), int (boxes[2]), int (boxes[3]), int (boxes[4])
            cv2.rectangle(frame, (x,imgH-y), (w, imgH-h), (0,0,255), 3)
    cv2.putText(frame, imgchar, (x1+ int(w1/50),y1+ int(h1/50)), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
(255,0,0), 2)
    font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.imshow('text detection tutorial',frame)
    if cv2.waitKey(2) & 0xff ==ord('q'):
        break
cap.release()
cv2.destroyAllwindows()
```

Output :-

Intput Image:-

Output image:-




```
Deep Learning and
Computer Vision
Tutorials


~ 4 226 412 228 0
D 47 173 61 192 0
e 62 173 75 187 0
e 76 173 88 187 0
p 90 168 103 187 0
L 118 173 128 191 0
e 129 173 141 187 0
a 142 173 153 187 0
r 156 173 164 187 0
n 165 173 177 187 0
i 180 173 183 192 0
n 186 173 197 187 0
g 199 168 212 187 0
a 219 173 230 187 0
n 233 173 244 187 0
d 246 173 259 193 0
C 46 143 60 162 0
o 61 143 74 157 0
m 76 143 95 157 0
p 90 138 104 162 0
u 98 138 110 157 0
t 112 143 124 157 0
e 126 143 134 160 0
r 135 143 157 157 0
V 164 143 179 162 0
i 180 143 184 162 0
s 186 143 195 157 0
i 197 143 200 162 0
o 202 143 215 157 0
n 218 143 229 157 0
T 45 110 53 130 0
u 45 111 59 129 0
t 58 110 70 124 0
```

Conclusion: -

The Histogram of Oriented Gradients (HOG) is a popular feature descriptor technique in computer vision and image processing. It analyzes the distribution of edge orientations within an object to describe its shape and appearance. The HOG method involves computing the gradient magnitude and orientation for each pixel in an image and then dividing the image into small cells. By using the HOG algorithm and SVM a car detection program was written which identifies whether a particular image contains a car or not.