Name : Prerna Kanekar
Roll no : 27
Experiment 08

Aim: To Perform Detecting and Recognizing Objects

Objective: Object Detection and recognition techniques HOG descriptor The Scale 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 scaleinvariant 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 scale issue

For each HOG cell, the histogram contains a number of bins equal to the number of gradients or, in other words, the number of axis directions that HOG considers. After calculating all the cells' histograms, HOG processes groups of histograms to produce higher-level descriptors. Specifically, the cells are grouped into larger regions, called blocks. These blocks can be made of any number of cells, but Dalal and Triggs found that 2x2 cell blocks yielded the best results when performing people detection. A block-wide vector is created so that it can be normalized, compensating for local variations in illumination and shadowing. (A single cell is too small a region to detect such variations.) This normalization improves a HOG-based detector's robustness, with respect to variations in lighting conditions.

The Location issue

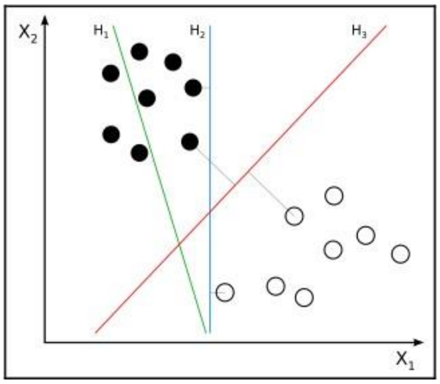
Like other detectors, a HOG-based detector needs to cope with variations in objects' location and scale. The need to search in various locations is addressed by moving a fixedsize sliding window across an image. The need to search at various scales is addressed by scaling the image to various sizes, 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



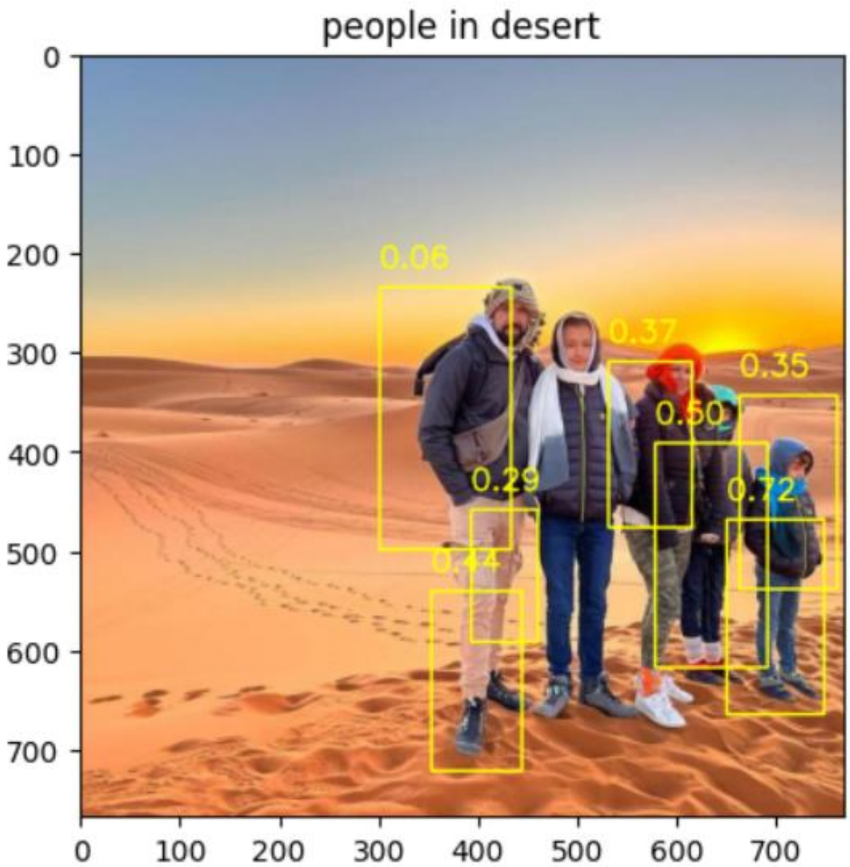
Hyperplane H1 (shown as a green line) does not divide the two classes (the black dots versus the white dots). Hyperplanes H2 (shown as a blue line) and H3 (shown as a red line) both divide the classes; however, only hyperplane H3 divides the classes by a maximal margin.

Code :-

```
#prac8
import cv2
import numpy as np
from matplotlib import pyplot as plt

# Check OpenCV version
OPENCV_MAJOR_VERSION = int(cv2.__version__.split('.')[0])
OPENCV_MINOR_VERSION = int(cv2.__version__.split('.')[1])

def is_inside(i, o):
    ix, iy, iw, ih = i
    ox, oy, ow, oh = o
    return ix > ox and ix + iw < ox + ow and iy > oy and iy + ih < oy +
oh

# Load the image from your local file system in Colab
img = cv2.imread('s6.jpg')

hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

if OPENCV_MAJOR_VERSION >= 5 or (OPENCV_MAJOR_VERSION == 4 and
OPENCV_MINOR_VERSION >= 6):
    # OpenCV 4.6 or a later version is being used.
    found_rects, found_weights = hog.detectMultiScale(
        img, winStride=(4, 4), scale=1.02, groupThreshold=1.9)
else:
    # OpenCV 4.5 or an earlier version is being used.
    # The groupThreshold parameter used to be named finalThreshold.
    found_rects, found_weights = hog.detectMultiScale(
        img, winStride=(4, 4), scale=1.02, finalThreshold=1.9)

found_rects_filtered = []
found_weights_filtered = []

for ri, r in enumerate(found_rects):
    for qi, q in enumerate(found_rects):
        if ri != qi and is_inside(r, q):
            break
    else:
        found_rects_filtered.append(r)
        found_weights_filtered.append(found_weights[ri])

for ri, r in enumerate(found_rects_filtered):
    x, y, w, h = r
```

```
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 255), 2)
    text = '%.2f' % found_weights_filtered[ri]
    cv2.putText(img, text, (x, y - 20), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,
255, 255), 2)

# Display the image
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title('people in desert')
plt.show()

# Save the image
cv2.imwrite('/content/s6.png', img)
```

Output :-



Conclusion: -

Histogram of Oriented Gradients (HOG) is a widely used technique in the fields of computer vision and image processing. It is employed to characterize the shape and visual attributes of objects by analyzing the distribution of edge orientations within them. The HOG approach entails calculating both the magnitude and orientation of gradients for every pixel in an image, followed by partitioning the image into small cells. Leveraging the HOG method, a program was developed for person detection, which determines whether a given image contains a human presence or not.