Aim: To 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 detection and recognition are essential tasks in computer vision and image processing. They involve identifying and localizing objects of interest within images or videos. Several techniques and algorithms have been developed to achieve these objectives. In this experiment, we will explore some of these techniques:

1. HOG Descriptors (Histogram of Oriented Gradients):

- HOG descriptors are a widely used feature extraction method for object detection. They capture the local gradient information of an image by dividing it into small cells and computing gradient histograms for each cell. These histograms represent the intensity and orientation of edges in the image.

- HOG descriptors are particularly useful for detecting objects with distinctive texture or shape patterns, such as pedestrians and vehicles.

2. The Scale Issue:

- Object detection often involves handling objects at different scales. Objects can appear larger or smaller in images depending on their distance from the camera or variations in perspective. Dealing with scale variations is a critical challenge in object detection.

3. The Location Issue:

- Accurately determining the location of objects within an image is another key challenge. Objects can be partially occluded, rotated, or in complex backgrounds. Detecting their precise position is crucial for various applications, including tracking and augmented reality.

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

- After detecting potential object candidates in an image, there may be multiple detections for the same object or region. Non-maximum suppression is a post-processing step that eliminates redundant or overlapping detections, retaining only the most confident ones.

5. Support Vector Machines (SVM):

- SVM is a machine learning algorithm used for classification and regression tasks. In object detection, SVM can be employed to classify object candidates as either objects of interest or background. It helps in making decisions about whether a given region contains an object or not.

Code:

Now, let's discuss the code for implementing object detection and recognition using HOG descriptors and SVM. This code will focus on detecting people in images.

code:

```
import matplotlib.pyplot as plt
```

```
%matplotlib inline
import tensorflow as tf
import numpy as np
img = plt.imread('catdog.jpg')

def box_corner_to_center(boxes):
    """Convert from (upper-left, lower-right) to (center, width, height)."""
    x1, y1, x2, y2 = boxes[:, 0], boxes[:, 1], boxes[:, 2], boxes[:, 3]
    cx = (x1 + x2) / 2
    cy = (y1 + y2) / 2
    w = x2 - x1
    h = y2 - y1
    boxes = tf.stack((cx, cy, w, h), axis=-1)
    return boxes

def box_center_to_corner(boxes):
    """Convert from (center, width, height) to (upper-left, lower-right)."""
    cx, cy, w, h = boxes[:, 0], boxes[:, 1], boxes[:, 2], boxes[:, 3]
    x1 = cx - 0.5 * w
    y1 = cy - 0.5 * h
    x2 = cx + 0.5 * w
    y2 = cy + 0.5 * h
```

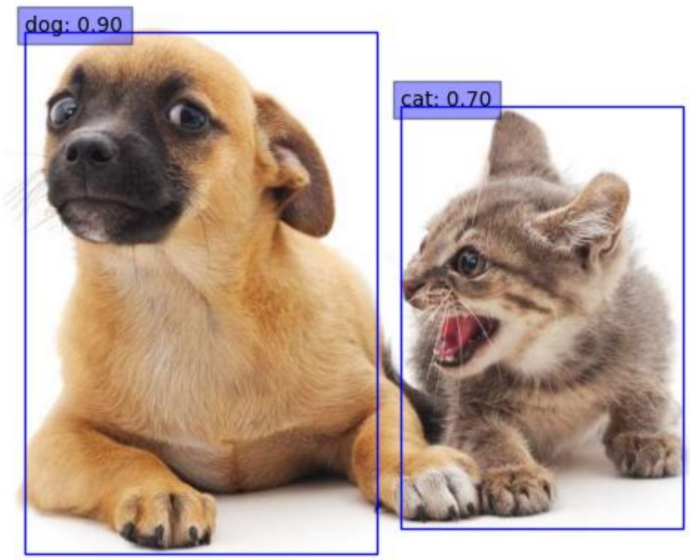
```
    boxes = tf.stack((x1, y1, x2, y2), axis=-1)
    return boxes
dog_bbox, cat_bbox = [60.0, 45.0, 378.0, 516.0], [400.0, 112.0, 655.0, 493.0]
boxes = tf.constant((dog_bbox, cat_bbox))
box_center_to_corner(box_corner_to_center(boxes)) == boxes
boxes = tf.constant((dog_bbox, cat_bbox))
box_center_to_corner(box_corner_to_center(boxes)) == boxes

def bbox_to_rect(bbox, color):
    """Convert bounding box to matplotlib format."""
    return plt.Rectangle(xy=(bbox[0], bbox[1]), width=bbox[2] - bbox[0],
                         height=bbox[3] - bbox[1], fill=False,
                         edgecolor=color, linewidth=2)

def visualize_detections(
    image, boxes, classes, scores, figsize=(7, 7), linewidth=1, color=[0, 0, 1]
):
    """Visualize Detections"""
    image = np.array(image, dtype=np.uint8)
    plt.figure(figsize=figsize)
    plt.axis("off")
    plt.imshow(image)
```

```
    ax = plt.gca()
    for box, _cls, score in zip(boxes, classes, scores):
        text = "{}: {:.2f}".format(_cls, score)
        x1, y1, x2, y2 = box
        w, h = x2 - x1, y2 - y1
        patch = plt.Rectangle(
            [x1, y1], w, h, fill=False, edgecolor=color, linewidth=linewidth
        )
        ax.add_patch(patch)
        ax.text(
            x1,
            y1,
            text,
            bbox={"facecolor": color, "alpha": 0.4},
            clip_box=ax.clipbox,
            clip_on=True,
        )
    plt.show()
    return ax
visualize_detections(img,[dog_bbox, cat_bbox],['dog', 'cat'], [0.9, 0.7])
```

output:


dog: 0.90
cat: 0.70

Conclusion:

The combination of HOG descriptors and Support Vector Machines (SVM) is a strong method for object recognition, emphasizing shape and texture. Challenges include scale variation and precise location. Techniques such as image pyramids and non-maximum suppression address scale issues. SVMs classify objects effectively. Achieving accurate object detection hinges on parameter tuning and training the SVM with annotated data.