object-detection-checkpoint

May 24, 2024

1 Prashant Priyadarshi

1.1 Object Detection with Detectron

[1]:
```
pip install torchvision
```

```
Requirement already satisfied: torchvision in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (0.15.1)
Requirement already satisfied: numpy in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torchvision) (1.24.2)
Requirement already satisfied: requests in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torchvision) (2.31.0)
Requirement already satisfied: torch==2.0.0 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torchvision) (2.0.0)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torchvision) (10.0.0)
Requirement already satisfied: filelock in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torch==2.0.0->torchvision) (3.12.2)
Requirement already satisfied: typing-extensions in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torch==2.0.0->torchvision) (4.6.2)
Requirement already satisfied: sympy in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torch==2.0.0->torchvision) (1.12)
Requirement already satisfied: networkx in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torch==2.0.0->torchvision) (3.1)
Requirement already satisfied: jinja2 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
torch==2.0.0->torchvision) (3.0.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
requests->torchvision) (3.2.0)
```

```
Requirement already satisfied: idna<4,>=2.5 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
requests->torchvision) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
requests->torchvision) (2.0.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
requests->torchvision) (2023.5.7)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
jinja2->torch==2.0.0->torchvision) (2.0.1)
Requirement already satisfied: mpmath>=0.19 in c:\users\prashant
priyadarshi\appdata\local\programs\python\python310\lib\site-packages (from
sympy->torch==2.0.0->torchvision) (1.3.0)
Note: you may need to restart the kernel to use updated packages.
```

2 importing the libraries

[2]:
```
import torch
import torchvision.transforms as T
from torchvision.models.detection import fasterrcnn_resnet50_fpn
from PIL import Image
import numpy as np
import cv2
import matplotlib.pyplot as plt
```

3 pretrained model

[8]:
```
# Load the pretrained model
model = fasterrcnn_resnet50_fpn(pretrained=True)
model.eval()

# Class labels from official PyTorch documentation for the pretrained model
COCO_INSTANCE_CATEGORY_NAMES = [
    '__background__', 'person', 'bicycle', 'car', 'motorcycle', 'airplane',␣
 ↪'Bus','Tractor',
    'truck','Auto','Tempo Traveller','van','boat', 'traffic light', 'fire␣
 ↪hydrant', 'stop sign',
    'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'cow',
    'elephant', 'bear', 'zebra', 'giraffe', 'backpack', 'umbrella', 'N/A',
    'N/A', 'handbag', 'tie', 'suitcase', 'frisbee', 'skis', 'snowboard',␣
 ↪'sports ball',
    'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surfboard',␣
 ↪'tennis racket',
```

```
    'bottle', 'N/A', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl',
    'banana', 'apple', 'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog',␣
 ↪'pizza',
    'donut', 'cake', 'chair', 'couch', 'potted plant', 'bed', 'dining table',
    'N/A', 'toilet', 'N/A', 'tv', 'laptop', 'mouse', 'remote', 'keyboard',
    'cell phone', 'microwave', 'oven', 'toaster', 'sink', 'refrigerator',
    'book', 'clock', 'vase', 'scissors', 'teddy bear', 'hair drier',
    'toothbrush'
]
```

4 Obtaining the predictions

[ ]:
```
def get_prediction(img_path, threshold):
    # Open the image
    img = Image.open(img_path)
    transform = T.Compose([T.ToTensor()])
    img = transform(img)

    # Get predictions from the model
    with torch.no_grad():
        pred = model([img])

    # Filter predictions based on threshold
    pred_class = [COCO_INSTANCE_CATEGORY_NAMES[i] for i in␣
 ↪list(pred[0]['labels'].numpy())]
    pred_boxes = [[i[0], i[1], i[2], i[3]] for i in list(pred[0]['boxes'].
 ↪detach().numpy())]
    pred_score = list(pred[0]['scores'].detach().numpy())
    pred_t = [pred_score.index(x) for x in pred_score if x > threshold][-1]
    pred_boxes = pred_boxes[:pred_t + 1]
    pred_class = pred_class[:pred_t + 1]
    return pred_boxes, pred_class
```

5 Performing object detection

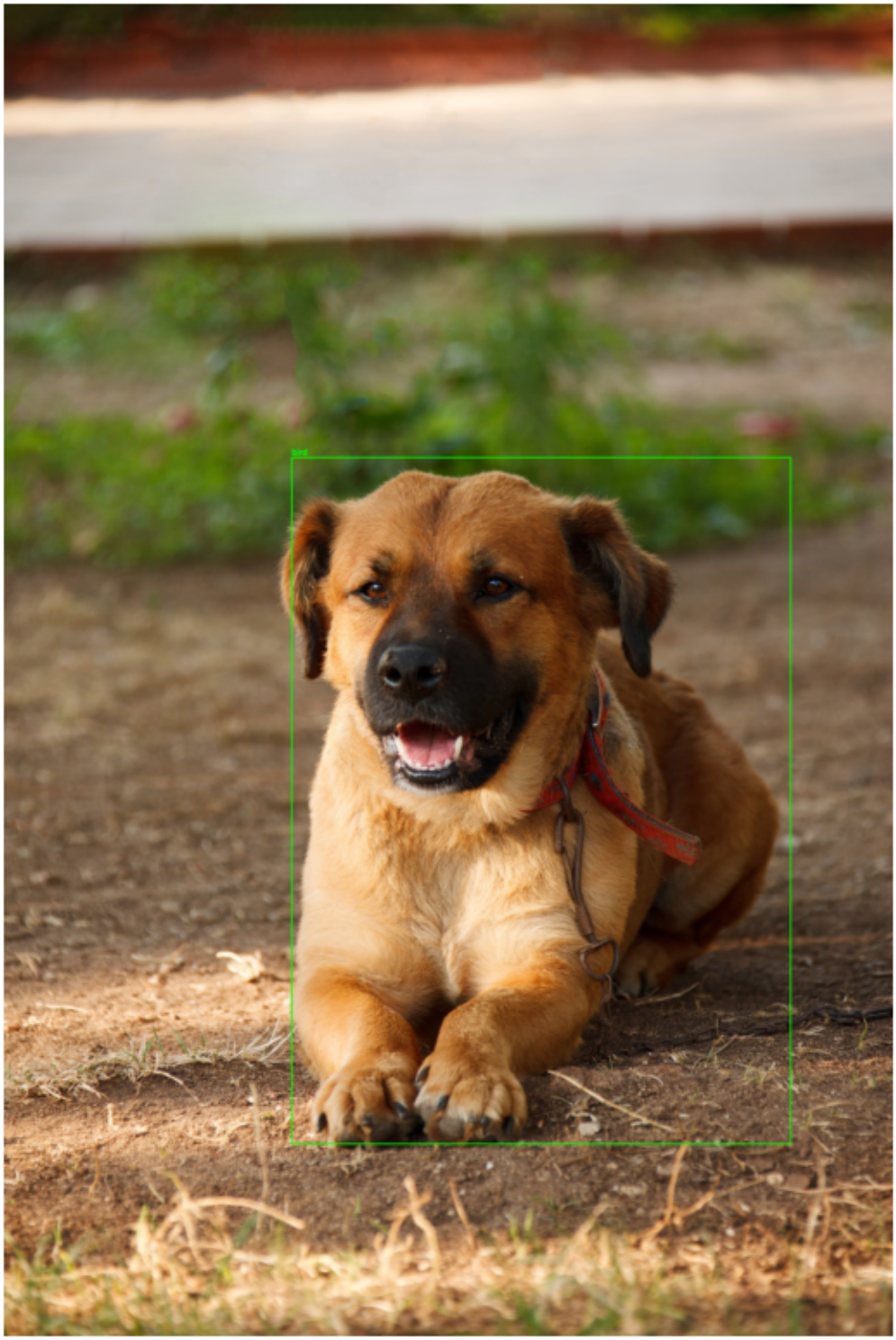
[1]:
```
def object_detection_api(img_path, threshold=0.5, rect_th=3, text_size=1,␣
 ↪text_th=3):
    # Get predictions
    boxes, pred_cls = get_prediction(img_path, threshold)

    # Open the image with OpenCV
    img = cv2.imread(img_path)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    # Loop over the predictions and draw bounding boxes
```

```
    for i in range(len(boxes)):
        box = boxes[i]
        pred_class = pred_cls[i]

        # Extract coordinates for drawing the rectangle and text
        x1, y1, x2, y2 = box

        # Convert the coordinates to integers
        x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)

        # Draw the rectangle
        cv2.rectangle(img, (x1, y1), (x2, y2), color=(0, 255, 0),␣
↪thickness=rect_th)

        # Draw the class label text above the rectangle
        cv2.putText(img, pred_class, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX,␣
↪text_size, (0, 255, 0), thickness=text_th)

    # Display the image
    plt.figure(figsize=(10, 10))
    plt.imshow(img)
    plt.axis('off')
    plt.show()
```

6 Calling the object detection RCNN model

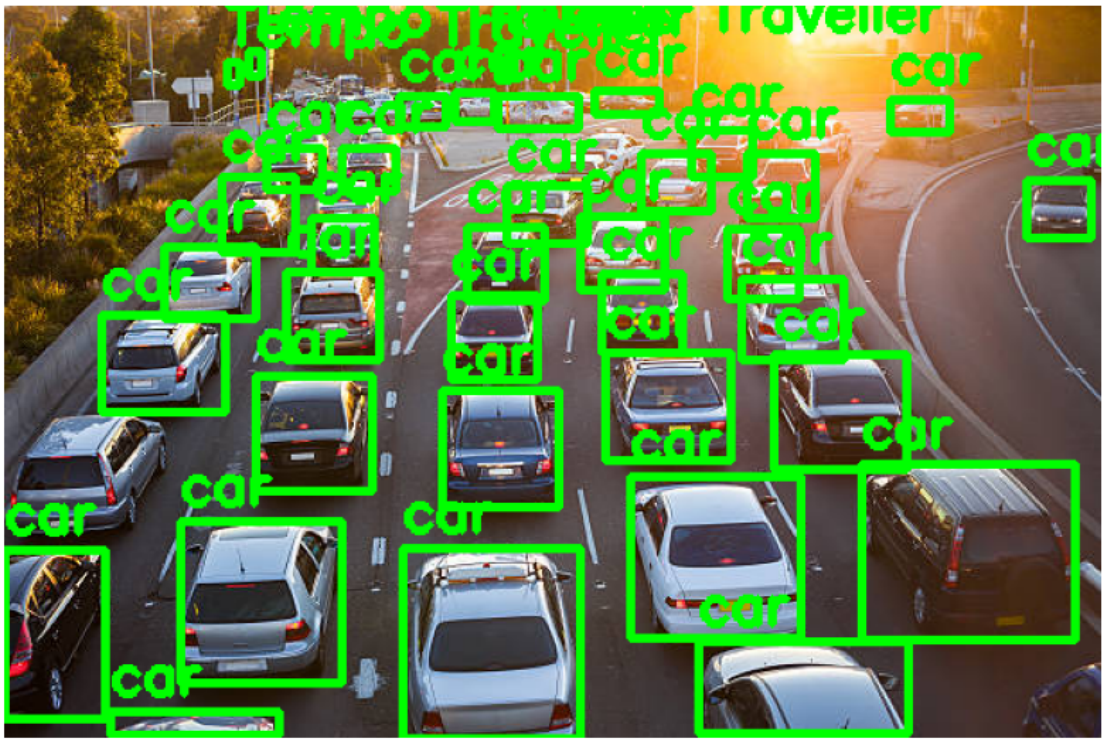
[6]:
```
img_path = "C:\\Users\\prashant␣
↪priyadarshi\\Desktop\\assignment-2\\dataset\\A-Cat.jpg"
object_detection_api(img_path, threshold=0.8)
```

[10]:
```
img_path = "C:\\Users\\prashant␣
 ↪priyadarshi\\Desktop\\assignment-2\\dataset\\dog_lying_193003.jpg"
object_detection_api(img_path, threshold=0.8)
```



[13]:
```
img_path = "C:\\Users\\prashant␣
↪priyadarshi\\Desktop\\assignment-2\\dataset\\traffic jam.jpg"
```

```
object_detection_api(img_path, threshold=0.8)
```



[25]:
```
img_path = "C:\\Users\\prashant␣
 ↪priyadarshi\\Desktop\\assignment-2\\dataset\\pexels-san-fermin-pamplona-1299086.
 ↪jpg"
object_detection_api(img_path, threshold=0.5,text_size=3)
```

[26]:
```
img_path = "C:\\Users\\prashant␣
 ↪priyadarshi\\Desktop\\assignment-2\\dataset\\group of cows.jpg"
object_detection_api(img_path, threshold=0.5,text_size=3)
```

7 Conclusion:

This code implements an object detection API using the Faster R-CNN model. It takes an input image, detects objects in the image, and displays the image with bounding boxes and class labels for the detected objects.

[ ]: