Báo cáo số 5 - Project 2

April 26, 2020

1 Báo cáo số 5

- Họ và tên: Nguyễn Văn Chức
- Mssv: 20170044
- Nội dung tìm hiểu: Object Recogition trong Image processing

1.1 Object Detection and Object Recognition

Như đã trình bày trong báo cáo số 4, các thư viện trong Open CV cung cấp một số công cụ cho phép nhận diện và theo dõi vật thể dựa vào các đặc trưng. Tuy nhiên, các thuật toán dùng cho thư viện OpenCV này không thể đặt được hiệu suất làm việc cần thiết trong một số điều kiện nhất định.

1.2 Áp dụng Deep Learning trong Object Reconition

Một số thuật toán Deep Learning cho phép nhận diện vật thể với hiệu suất cao hơn. Các thuật toán có thể kể đến như CNN(Convolutional Neural Network), R-CNN, Fast-RCNN, Faster-RCNN, RetinaNet và YOLO.

1.2.1 Thuật toán CNN

Chúng ta có thể nhận thấy rằng với một phép tính Convolutional cho phép khảo sat một thuộc tính của bức ảnh. Một Neural NetWork bao gồm các layer được chia vào 3 loại:

- Convolution layer
- Pooling layer
- Fully connected layer

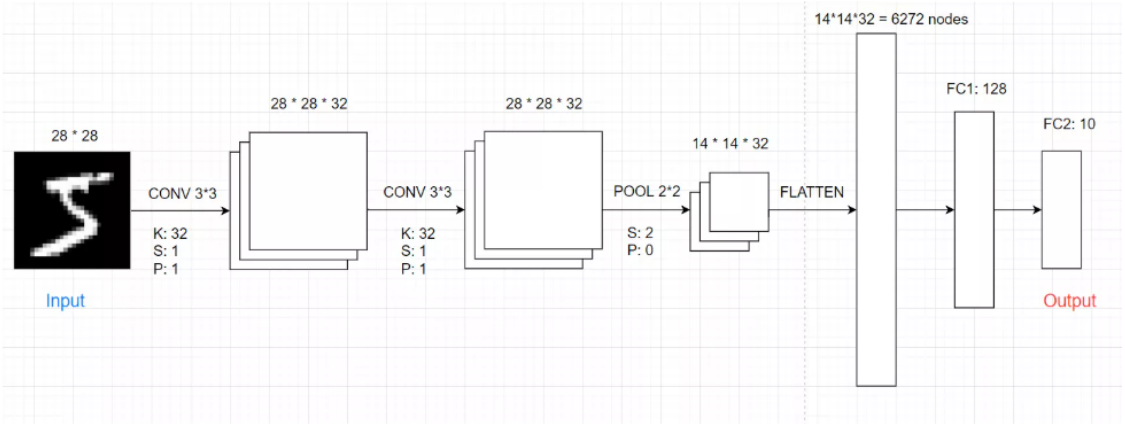
Với Convolution Layer, một bức ảnh màu được khảo sát dưới dạng một bức ảnh RGB sẽ bao gồm 3 ma trận chứa thông tin ứng với 3 màu cơ bản do đó khi áp dụng thì kernel của phép tính Convolution sẽ là 1 tensor 3 chiều kích thước kxkx3.

Trong một phép tính convolution, giá trị sau tính toán với kernel kxk tương ứng với từng màu chúng được cộng lại và thêm vào một số bias.

Tuy nhiên, một kernel áp dụng chỉ cho phép cho ra 1 đầu ra. Chúng ta sẽ áp dụng K kernel để được đầu ra nhiều hơn. Đồng nghĩa với việc khảo sát nhiều thuộc tính qua một layer của mạng

Các Pooling Layer được đặt xen kẽ giữa các Convolutional Layer nhằm giảm khối lương tính toán những vẫn đảm bảo giữ được đặc trưng cơ bản của bức ảnh. Một phép Pooling cũng có một kernel kích thước kxk và phép tính thay vì nhận các hệ số và công lại như trong convolution thì chọn giá trị max hoặc tính trung bình của các ô trong kernel. Đôi khi người ta sử dụng Convolutional Layer với hệ số stride > 1 để giảm kích thước tính toán thay cho việc sử dụng Pooling layer.

Sau khi được Sau khi ảnh được truyền qua nhiều convolutional layer và pooling layer thì model đã học được tương đối các đặc điểm của ảnh thì tensor của output của layer cuối cùng, sẽ được chuyển về dưới dạng vector 1 chiều. Sau đó ta dùng các fully connected layer để kết hợp các đặc điểm của ảnh để ra được output của model.



Các thuật toán RCNN, Fast-RCNN, Faster-RCNN về cơ bản cũng là mô hình của CNN và kết hợp thêm các thuật toán khác để nâng cao hiệu quả thuật toán.

1.2.2 Thuật toán YOLO

Yolo là một mô hình mạng CNN cho việc phát hiện, nhận dạng, phân loại đối tượng. Yolo được tạo ra từ việc kết hợp giữa các convolutional layers và connected layers.Trong đóp các convolutional layers sẽ trích xuất ra các feature của ảnh, còn full-connected layers sẽ dự đoán ra xác suất đó và tọa độ của đối tượng.

Hình ảnh đầu vào được chia thành S x S các ô. Đối với mỗi object trong hình ảnh, mỗi ô cần xác định xác suất vật thể có nằm trong ô đó hay không. Nói theo cách đó là ô mà tâm của vật thể rơi vào hay không.

Môi ô dự đoán B bounding bõ và C lớp xác suất. Mỗi bounding box dự đoán tâm của box tương đối trong ô chứa nó cũng như kích thước của nó đối với toàn bộ bức ảnh, và một giá trị confident score tính bằng tích của xác suất vật thể xuất hiện trong ô và giá trị IOU(pred,truth). Với IOU được tính bằng diện tích phần giao trên diện tích phần hợp của predicted bounding box và ground truth bounding box.

Mạng nơ ron áp dụng có cấu trúc giống như CNN với max pooling layer và 2 fully connected layer ở cuối.

2 Thực hành

Các thư viện ImageAI Cung cấp các công cụ thực thi YOLO và RetinaNet với model được tạo sẵn.

[ ]:
```
!pip install imageai --upgrade
```

[ ]:
```
!pip install tensorflow==1.14.0
```

[ ]:
```
from imageai.Detection import ObjectDetection
# import tensorflow as tf

detector= ObjectDetection()
# đặt chế độ làm việc với RetinaNet
# có thể dùng chế độ làm việc với YOLO bằng cách dùng hàm
#detector.setModelTypeAsYOLOv3()
detector.setModelTypeAsRetinaNet()
#load đặt đường dẫn đến model
detector.setModelPath("/content/drive/My Drive/Python code/Project 2 Practice/
 ↪objectDetection/resnet50_coco_best_v2.0.1.h5")
#laod model
detector.loadModel()
#detect object
detections= detector.detectCustomObjectsFromImage(input_image="/content/drive/
 ↪My Drive/Python code/Project 2 Practice/objectDetection/animal.jpg",\
                                                  output_image_path="/content/
 ↪drive/My Drive/Python code/Project 2 Practice/objectDetection/newanimal.jpg")
#in ra kết quả
for eachObject in detections:
    print(eachObject["name"] , " : ", eachObject["percentage_probability"], " :␣
 ↪", eachObject["box_points"] )
    print("--------------------------------")
```

Bức ảnh trước khi biến đổi

Bức ảnh kết quả:

Bức ảnh mới nhận diện được 4 chú mèo, còn chú mèo thứ 5 thì vẫn chưa thể nhận diện được (chú mèo ở bên phải ngoài cùng)

Hoặc chúng ta có thể sử dụng thư viện của OpenCV

[ ]:
```
import cv2
import numpy as np

# Load Yolo
net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")
classes = []
with open("coco.names", "r") as f:
    classes = [line.strip() for line in f.readlines()]
layer_names = net.getLayerNames()
output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
colors = np.random.uniform(0, 255, size=(len(classes), 3))

# Loading image
img = cv2.imread("room_ser.jpg")
img = cv2.resize(img, None, fx=0.4, fy=0.4)
height, width, channels = img.shape

# Detecting objects
blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True,␣
 ↪crop=False)
net.setInput(blob)
outs = net.forward(output_layers)

# Showing informations on the screen
class_ids = []
confidences = []
boxes = []
for out in outs:
    for detection in out:
        scores = detection[5:]
        class_id = np.argmax(scores)
        confidence = scores[class_id]
        if confidence > 0.5:
            # Object detected
            center_x = int(detection[0] * width)
            center_y = int(detection[1] * height)
            w = int(detection[2] * width)
```

```
            h = int(detection[3] * height)
            # Rectangle coordinates
            x = int(center_x - w / 2)
            y = int(center_y - h / 2)
            boxes.append([x, y, w, h])
            confidences.append(float(confidence))
            class_ids.append(class_id)

indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

font = cv2.FONT_HERSHEY_PLAIN
for i in range(len(boxes)):
    if i in indexes:
        x, y, w, h = boxes[i]
        label = str(classes[class_ids[i]])
        color = colors[i]
        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
        cv2.putText(img, label, (x, y + 30), font, 3, color, 3)
cv2.imshow("room_detect_result", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Bức ảnh đầu vào: