Hw2 Report

0756545 楊秉澄

Github link: https://github.com/BingChengYang/cv_hw2

Reference: https://github.com/facebookresearch/detectron2

Speed Bench Mark:

在 Colab 上辨識一張圖片的時間為 155ms。

```
dict_ans = {'bbox': [], 'score': [], 'label': []}
image = cv2.imread(os.path.join(root+"/test/", '1.png'))
# Make prediction
tStart = time.time()
output = predictor(image)
tEnd = time.time()
print("Prediction time:{}ms".format((tEnd-tStart)*1000))
```

```
/content
Skip loading parameter 'roi_heads.box_predictor.cls_score.weight' to the model due to incompatible shapes: (11, 1024) in the checkpoint but (81, 1024) in the model! You might want to double check if this is expected.
Skip loading parameter 'roi_heads.box_predictor.cls_score.bias' to the model due to incompatible shapes: (11,) in the checkpoint but (81,) in the model! You might want to double check if this is expected.
Skip loading parameter 'roi_heads.box_predictor.bbox_pred.weight' to the model due to incompatible shapes: (40, 1024) in the checkpoint but (320, 1024) in the model! You might want to double check if this is expected.
Skip loading parameter 'roi_heads.box_predictor.bbox_pred.bias' to the model due to incompatible shapes: (40,) in the checkpoint but (320,) in the model! You might want to double check if this is expected.
Metadata(name='dataset_train')
Prediction time:155.4281711578369ms
```

Introduction:

這次作業的 task 為 digit 上的 object detection，所使用的 dataset 為 Street View House Number(SVHN)裡面有 33402 張 training data 和 13068 張 testing data，其中在 training data 裡有 70000 多個 annotations。

這次作業不僅僅要滿足 accuracy 0.36898 的 Baseline 還要滿足 558ms 辨識一張圖片，因此模型的選擇上要可以兼顧上面 2 項條件，我透過 detectron2 這個由 facebook 基於 pytorch 開發的工具分別嘗試 faster rcnn、retinanet 2 種網路架構。

Methodology:

Environment:

- CPU: AMD R5-2600
- RAM: 16G
- GPU: Nvidia RTX-2070 8G

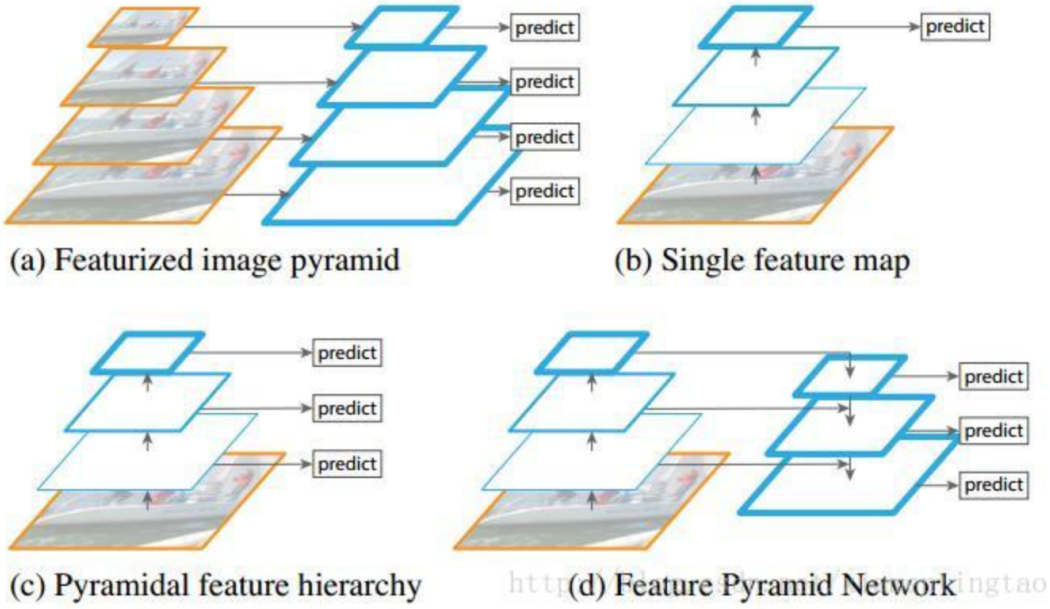
Data Preprocessing:

由於我這次採用 detectron2 來實作 object detection，為了需要將其 annotations 轉換成 coco dataset 的格式，除此之外我沒有特別對 data 做任何處理。

Model Architecture:

Model 我使用 detectron2 內建的 2 種模型，分別是 fpn faster rcnn、retinanet，最

終是選擇 fpn faster rcnn 當作我的模型，我用 resnet50 當作我的 backbone 雖然 resnet101 的效果不差，但為了有更快的辨識速度，因此我還是選擇 resnet50，採用 fpn 是為了避免小偵測目標在最後一層的時候特徵都已經遺失了，所以採用 feature pyramid network 可以有效解決多尺度的檢測問題。



Hyperparameters:

參數分別有 learning rate(lr)、mini_batch_size、Max_iter 需要設置，lr 我試過 0.0001,0.00015,0.001,0.000001，其中 0.001 太大很容易讓 loss 變成 NaN，0.0001 和 0.00015 訓練起來效果差不多，但 0.00015 訓練出來的模型有通過 baseline，因此最後我使用 0.00015 訓練我的模型，0.000001 收斂太慢因此不採用。

Mini_batch_size 嘗試了 32、64、128 其中 3 者訓練起來效果差不多，但 32 訓練出來的 mAP 稍微好一點點，因此用 32 來訓練我最終的模型。

Max_iter 設定為 5000，跑在 1000 以下訓練出來的效果非常不好，因此最少需要設定在 1000 以上。

Summary

這次作業為了設置 detectron2 在 window10 上的運行花費了我很多時間，所幸最後還是有訓練出一個可以通過 baseline 的模型，且在速度上也有不錯的表現，可惜 detectron2 沒有內建 yolo 的使用，因此之後我想嘗試 yolo 並比較他與 faster rcnn 看誰的效果比較佳。