

HW2_report

姓名:陳鎧勳

學號:0716314

0. GitHub link of my code:

https://github.com/KaivinC/NCTU_CV_HW2

1. Reference:

Efficientdet:

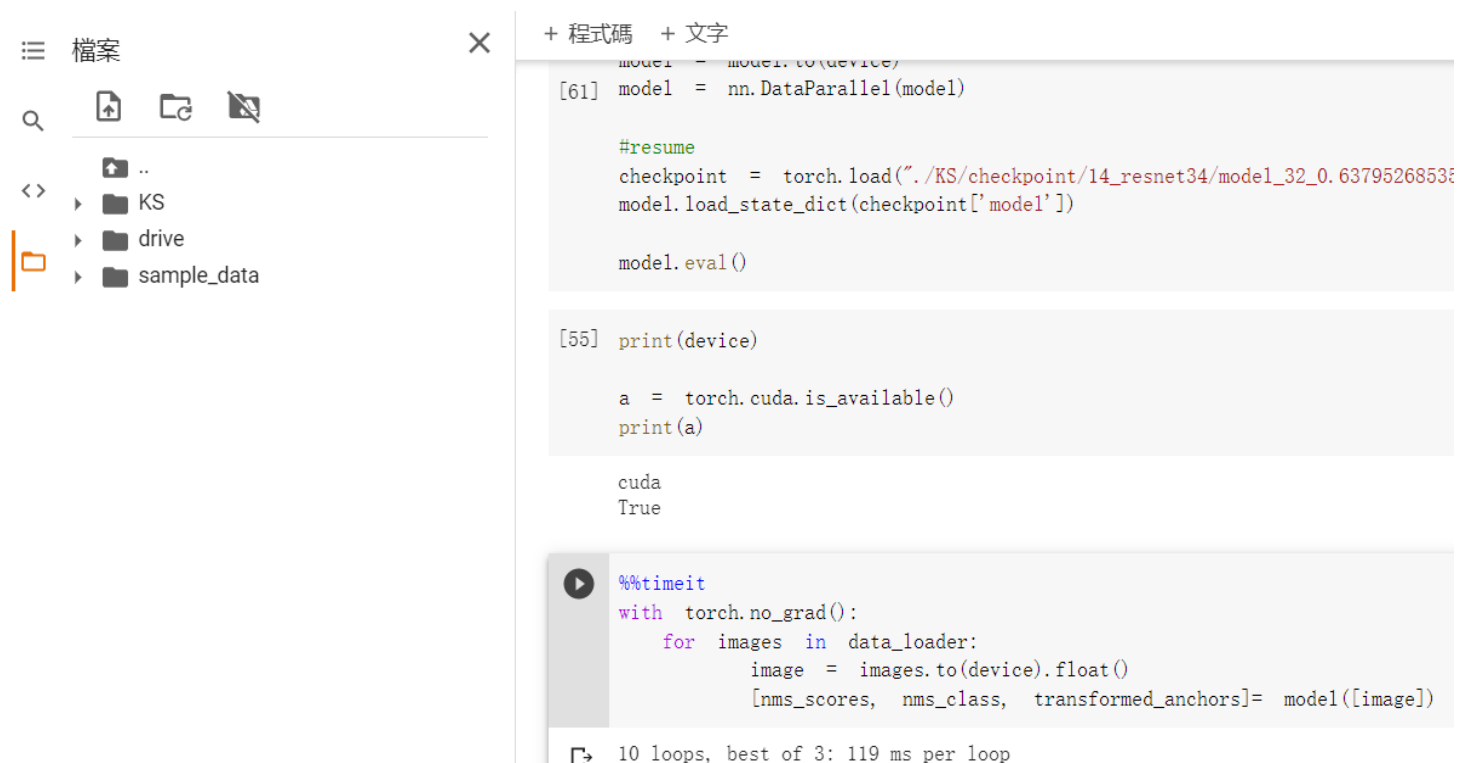
<https://arxiv.org/pdf/1911.09070.pdf>

<https://github.com/signatrix/efficientdet>

Process mat file:

<https://github.com/nate-parrott/jupyter-notebooks/blob/master/digitStruct.py>

2. Speed benchmark:



```
model = model.to(device)
[61] model = nn.DataParallel(model)

#resume
checkpoint = torch.load("./KS/checkpoint/14_resnet34/model_32_0.63795268538")
model.load_state_dict(checkpoint['model'])

model.eval()

[55] print(device)

a = torch.cuda.is_available()
print(a)

cuda
True

%%timeit
with torch.no_grad():
    for images in data_loader:
        image = images.to(device).float()
        [nms_scores, nms_class, transformed_anchors]= model([image])

10 loops, best of 3: 119 ms per loop
```



3. Brief introduction:

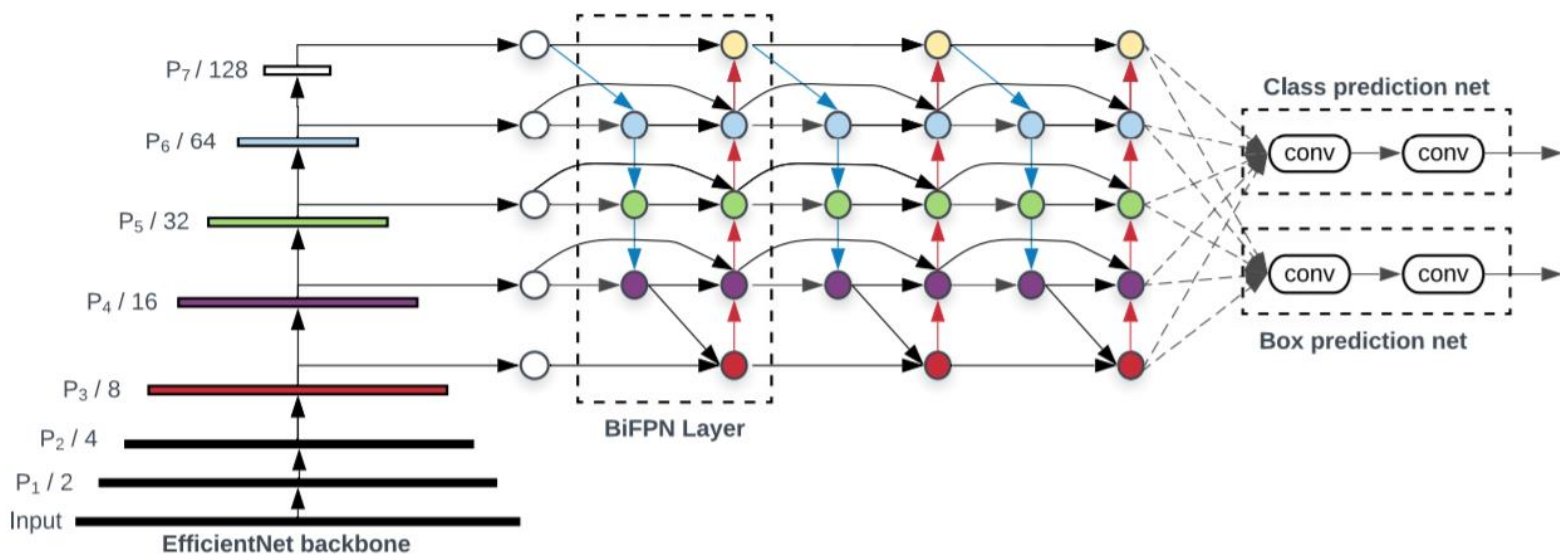
這個作業我所使用的技術是參考了 **EfficientDet: Scalable and Efficient Object Detection** 這篇論文裡的方法，裡面所提到 Bi-FPN 取代了舊有的 FPN，而且他可以當作一個區塊，要重複幾次區塊完全取決於使用者的設備資源，可以說是非常有彈性的模型。

4. Methodology:

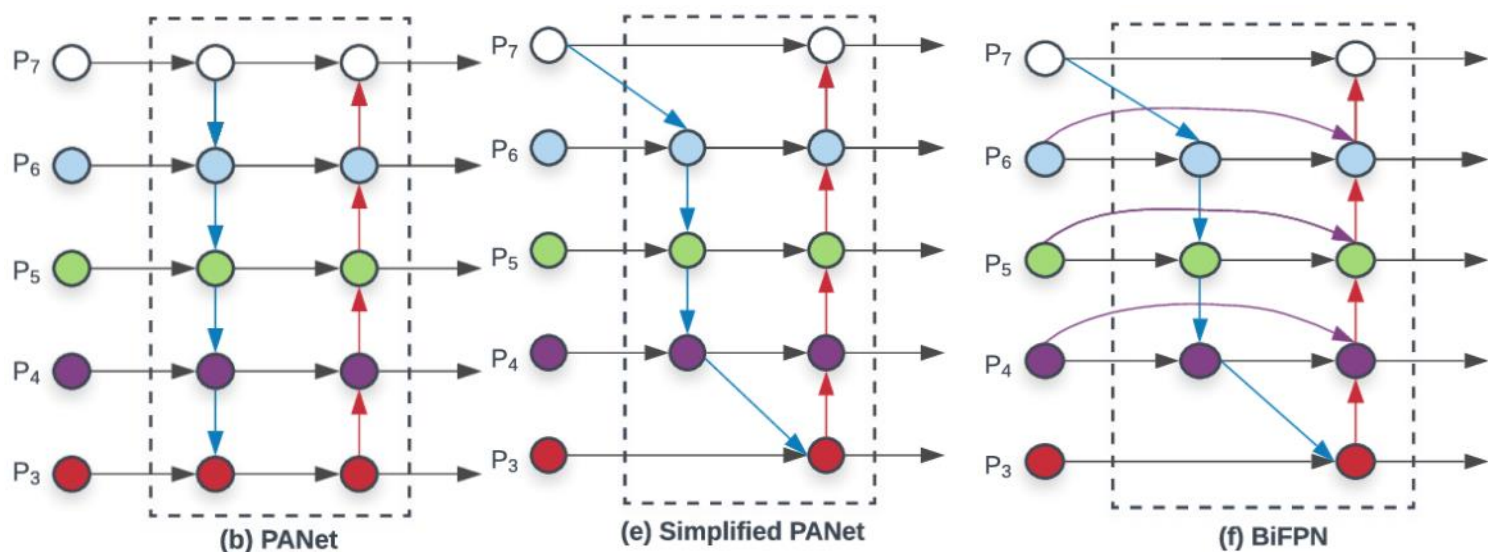
1. Data pre-process:

一開始拿到的資料是一個.mat 檔，透過助教提供的檔案進行轉檔成 h5 檔，但是後來發現一個問題，就是我如果一次把 h5 檔案讀近來會導致系統記憶體被我撐爆，然後整個程式被 pid kill 掉，所以我後來又去 github 找了一個是分別轉成 npy 的檔案，每次 dataloader 要 load 的時候在讀進來。

5. Model architecture:



這個 backbone 主要分為三個部分，分別是 BiFPN layer、Box prediction net 和 Class prediction net。



BiFPN 的發想可以從 PANet(上圖 b)講起，Tan et al. 發現 PANet 的效率比 FPN 還要好，只是計算量更大，於是他就移除掉只有一個輸入的節點(因為他認為相比其他節點，一個輸入的節點比較不重要)，於是得到了 Simplified PANet(上圖 e)，而作者又另外在相同 level 的輸入和輸出節點連了一條線，來融合更多的特徵。而 BiFPN 的好處就是可以當作一個區塊，然後重複數次，完全取決於使用者設備資源。

而 Box prediction net 和 Class prediction net 就相對比較單純，就是用 conv layer 組合起來的，而他的 input feature 則是各層 BiFPN 做 concatenate 的結果。

3. Hyperparameter:

本次作業我的 optimizer 主要有兩個，分別是 Adamw 和 SGD(momentum 則是 0.9)，我先利用 Adamw 把 loss 快速降低到一個程度，之後改用 SGD，而 learning rate 則是利用 cosine annealing 做一個 $1e-3 \sim 1e-10$ 的周期性切換，epoch 總數為 120。

Data augmentation 則是把短邊 resize 成 128，接著 center crop 成 128×128 (因為觀察到圖片的數字幾乎都在正中間)，接著經過自己對整個 dataset 所算出來的 mean 和 std 做標準化。

4. Summary:

這次作業我一開始是使用 FasterRCNN 但效果非常的不好，於是換了個模

型，也就是現在的 **EfficientDet** 後才終於過了 **baseline**，每次做這些作業都讓我對如何架構一個 **model** 有了夠進一步的了解，這個過程就好像拼拼圖一樣，把自己不會的知識一片一片的補上。