ML final project report – Medical Image Detection

Team : NTU_b06902028_Απόλλων
Member : b06902028 林柏劭
b06902030 邱譯
b06902058 吳崇維

1. Introduction & Motivation

在這次的機器學習實戰演練，在新聞檢索、影像除霧以及肺炎判斷中，由於在之前的作業中已經有大量使用 CNN 的經驗，因此我們選擇了第三個主題作為我們的題目。

在這個題目，我們會拿到兩萬多張圖片，以及標示其 bbox 的位置(肺炎所在處)，還有一些關於這些圖片的基本資訊 (性別…)。然後我們要將給定的 test picture 標示出其是否有肺炎，若有的話其 bounding box 的位置會在哪。

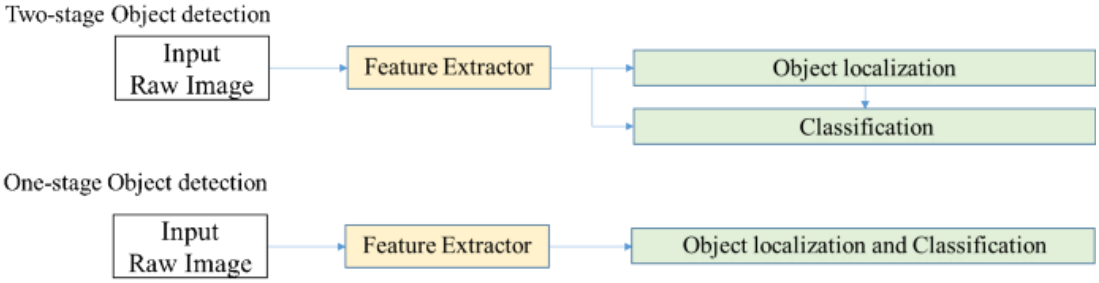
這是一個以單一 object 為目標的 object detection，雖與 CNN 有些許相關，但對於我們依舊是一個新的主題，因此我們打算閱讀相關論文後，以網路上前人的相關套件為基礎，去改進以實作這次的題目。

2. Data Preprocessing/Feature Engineering

這次所給的資料圖片皆為 1024*1024，由於手邊機器效能不足、以及圖片大 performance 不一定會好，我們將圖片以 openCV 的套件去做 resize，並以 resnet101 去做 feature extract，之後再依方法的不同去接上不同的 neural network。至於 resize 完後的大小將在後面 Experiment and Discussion 中再做討論

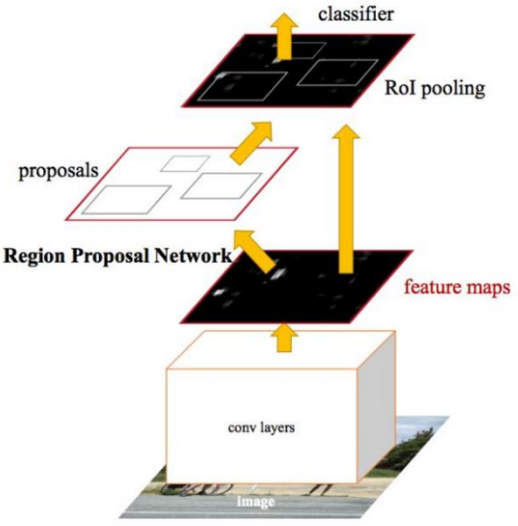
另外，我們有去更改 bounding box 的表示方式以方便我們實作，並將全部 21764 張圖片手動切出 20998 當作 train data，剩餘當作 valiation data。

3. Methods



在 object detection 中，主要分成兩種做法 : two stage detector 以及 one stage detector，兩者皆是先使用 feature extractor 去取出 feature (我們使用以 imagenet pretrained 的 resnet101)，再依 object localization, classification 的做法去做分別 ： 前者分開，後者則以單一的 neural network 實作。
在這兩種做法中，前者我們採用了 faster-RCNN 的方法去實作，後者我們則採用了 Retinanet，其結合了 Feature Pyramid Network 以及 focal loss 的作法。

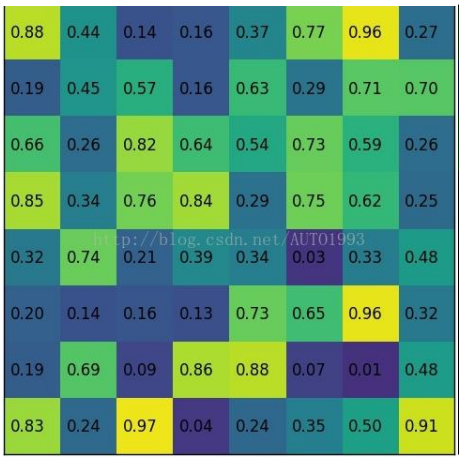
1. **faster-RCNN**



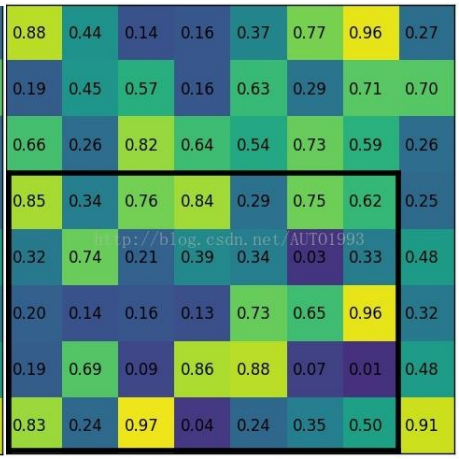
首先先使用 RPN (Region Proposal Network) ，RPN 為一個 CNN，它會在之前 feature extractor 所取出的 feature map 上取 sliding window，每個 sliding window 的中心點稱之為 anchor point，然後將事先準備好的 k 個不同尺寸比例的 box 以同一個 anchor point 去計算可能包含物體的機率(score)，取機率最高的 box。這 k 個 box 稱之為 anchor box。所以每個 anchor point 會得到 2k 個 score 以及 4k 個座標位置 (box 的左上座標，以及長寬)。
藉由 RPN，我們可以得到一些 bounding box 以及這些含有 object 的機率(score)，利用這些 score 塞選出特定數量的 Region of Interest (RoI)出來，但由於每個 RoI 的大小不一樣，因此會再進行 RoI pooling 後再針對每個 RoI 獨立去進行 classification 的動作。

RoI pooling 過程如下 ：

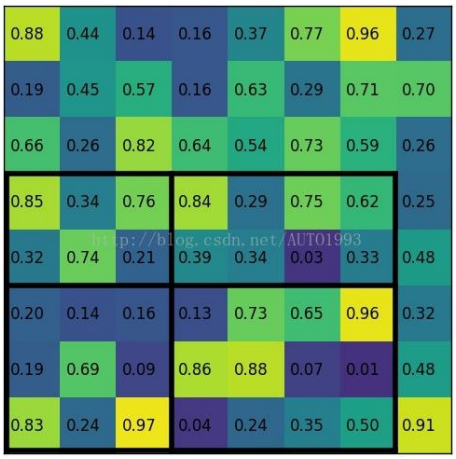
在 8*8 的 feature map 上將粗黑框的 RoI 輸出成 2*2



Feature map



RoI 範圍



切割成 2*2 的區域

在各區域進行 maxpooling

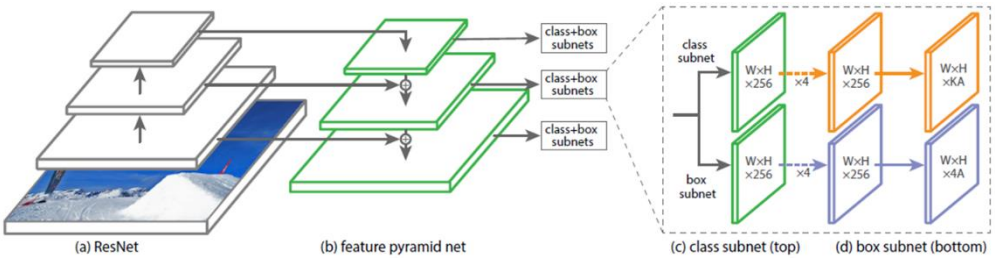
2. Retinanet

在一般的 one stage detector 中，表現通常會比 two stage detector 來的差，原因是因為「類別不平衡」：一張圖片中 object 通常不多，大多都是 background，因此就算機器將所有的 bounding box 判斷為 background，accuracy 也不會太差。因此有人便提出了 focal loss 的想法

$$\mathrm{FL}(p_t) = -(1-p_t)^{\gamma}\log(p_t)$$

使用這新的 loss function，可以使得 background 對於 loss 的影響變少，object 對於 loss 的影響提高，使機器能往較為正確的方向去做優化。

使用 focal loss 的 loss function 在搭配上如下圖的 Feature Pyramid Network，此便是 Retinanet



4. Experiment and Discussion

1. **faster-RCNN**

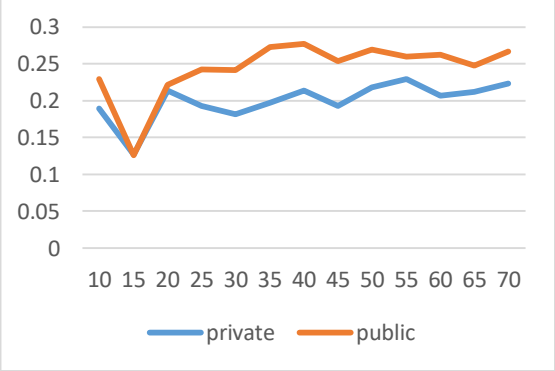
   我們使用網路上 faster-RCNN 的套件，並稍作修改以符合 data 的形式，以 resnet101 作為 feature extractor train 了 12 個 epoch，得到如下圖的結果:

   submission.csv 0.06329 0.08230
   a month ago by b06902030_資工重返一級
   add submission details

   此結果離 simple baseline 還有一段距離，若加大 epoch 數，推測應能達成不錯的 performance，但由於我們手邊的機器資源有限，使用此方法所耗費的時間又相當的多(1 個 epoch 大要 train 2 個小時)，因此我們便捨棄此方法，將心力投注在 Retinanet 上。

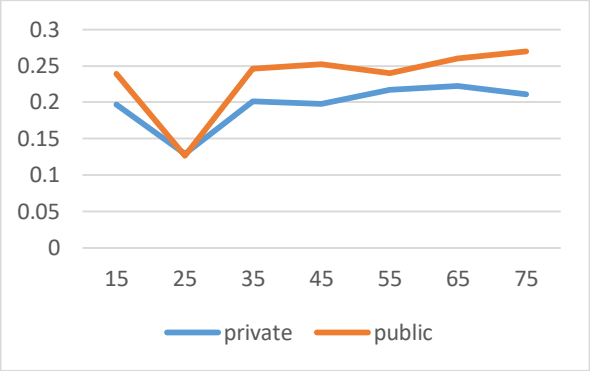
2. **Retinanet**

   我們使用網路上以 keras 實作 Retinanet 的套件，並參考之前 RSNA 比賽第三名的作法，將 Retinanet 出來的結果以 Non-Maximum Suppression 的方法去找出最佳的 bounding box。我們先藉由 validation loss 去找到大致要 train 的 epoch 數，再以全部的 data 去 train。以下即為把 img 縮小到 412*412 的結果:

   overall.csv 0.22911 0.25994
   2 hours ago by b06902058_Wayne
   step=500 batch=8 epoch = 55 val 0

但要小心的是 img_size 的部分，若縮到太小，縮然跑的速度比較，但 performance 會下降許多:



overall.csv an hour ago by b06902058_Wayne step=500 batch=8 epoch =55 val 0 img_size=256	0.21708	0.24005

5. Conclusion

在這次的題目中，我們實做了 faster-RCNN 與 Retinanet，這確實讓我們理解到了 two stage detector 與 one stage detector 的差別 ：

1. two stage detector 的確十分耗時。
2. one stage detector 運用 focal loss 的 loss function 以及將 object localization, classification 中間的架構複雜化，能以較少的時間達到或超越 two stage detector 的 performance。

不過，我們認為依舊有許多可以改進 ：

1. 可使用別的 data set 去 pretrain resnet，例如 NIH-Chest X-ray，其效果應會比使用 imagenet 來的好。
2. 我們目前是將 resnet101 的參數 fix，但若與 Retinanet 一起 train，應能獲得比較好的結果，但相對的也需要較多的運算資源。
3. 若能想出一個結合各個 bounding box 的 score 去 ensemble 的方法，應能大幅提升 performance。

6. Reference

https://github.com/kbardool/keras-frcnn/tree/master/keras_frcnn
https://medium.com/@syshen/%E7%89%A9%E9%AB%94%E5%81%B5%E6%B8%AC-object-detection-740096ec4540
https://arxiv.org/abs/1708.02002

https://github.com/pmcheng/rsna-pneumonia
https://github.com/fizyr/keras-retinanet
https://towardsdatascience.com/review-retinanet-focal-loss-object-detection-38fba6afabe4
https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e