ML final project: Medical Image Detection

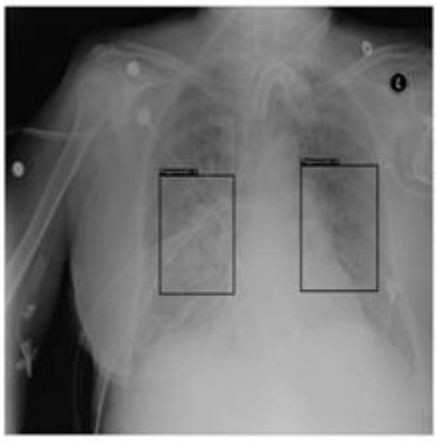
隊名: NTU_r07942091_test
隊員: 1.許博閔 r07942091 2.李利元 b04501073
3.陳鈞廷 b04201040 4.白佳灝 r07942092

Introduction & Motivation

Introduction

本次期末專案題目為檢測胸部 X 光影像中的肺炎病變 -- Chest X-ray Pneumonia Detection。對於沒有醫學背景的一般人來說，辨別是否有肺炎異常是非常困難的，此外，標記 X 光影像中的肺炎區域對放射科醫師也是一項耗時的工作，因此使用演算法自動識別以及標記肺炎區域是個具有挑戰與價值的研究。這次的資料集中訓練集的圖片數量為 21764 張，而測試集的數量 4998 張，資料同時包含了正常 (non-pneumonia) 以及有肺炎 (diseased) 的影像 (請參考下圖)，因此演算法必須要有辨別是否有肺炎病變的能力。在 Kaggle 的 RSNA Pneumonia-Detection-Challenge 比賽中，前幾名的模型架構有 RetinaNet 、 Mask-RCNN 和 Deformable R-FCN。

Normal



Diseased

Motivation

整題問題我們可以歸類成 objective detection，而許多過去的文獻中，已對這方面有些著墨，像是 RCNN、Fast RCNN、YOLO、RetinaNet 等。其中又分為 one-stage detector 和 two-stage detector 兩類，而 one stage detectors (EX: yolo)會直接將一張圖片分成許多網格，並利用回歸的方法預測 box 的位置，速度通常較快但相對精準度也比較低，而 two stage detectors(EX: RCNN) 會預先選出許多可能的 bounding box 位置，再一一檢測這些 box 是否含有 object，雖然會因此比較精準但速度相對較慢。而此次問題，考量速度的關係，我們專注於使用 one stage detectors 的方法，分別是 YOLO、RetinaNet，並且在後續作詳細的討論對於不同的實驗方法和參數調整。

Data Preprocessing/Feature Engineering

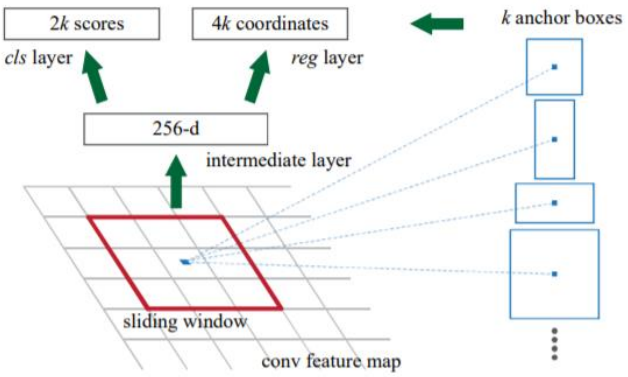
本次的 X 光圖片，和常見的 object detection dataset 如 VOC 和 coco 相比，比較特別的是有很多的 negative sample(完全沒有 bounding box 的圖片)，因此需要處理這些 negative sample，以及讓每張圖片的 label 轉換成 deep learning model 可以預測的形式。因原本 X 光圖片的 label 包含 box 左上角點的座標位置和 box 的寬與高，首先會把這四個數值轉換成 box 的 xmin, ymin, xmax, ymax，而原始圖片大小是 1024*1024，需要將圖片和 box 等比例縮小讓 GPU 的記憶體夠用。為了增加 training data 以及避免 overfitting，我們使用 data augmentation 的方法對圖片作水平翻轉和平移，因 feature extractor 都使用 ImageNet 的 pretrained model 來 fine tune，所以用和 ImageNet 相同的數值對圖片 normalize。而接下來的處理則因不同的 model 有不同的處理方法。

YOLOv1:

YOLOv1 會將圖片分成 7*7 個網格，每個網格預測 2 個 box 各自的中心座標、寬、高和信心分數，因此 yolov1 的 output 是一個維度 7*7*11 的 tensor。轉換的時候會將沒有 box 的網格的 ground true label 都設成 0，有 box 的網格則輸出 normalize 後的正確位置，信心分數和 label 都設為 1，而 negative sample 因為完全沒有 box，ground true 的值全部都是 0。

Retinanet:

Retinanet 因為使用了 anchor box 的技術，在處理上會比較複雜。Anchor box 擁有固定的形狀，以 VOC dataset 為例，因這個 dataset 有很多的行人和轎車，這些人和車所形成的 box 的長寬比例通常很接近，因此在預測的時候可以先產生很多長寬比固定，大小、位置不同的 anchor box，再判斷這些 anchor box 是否含有 object，評斷標準則是計算這些 anchor box 和 ground truth box 的重疊比例(IOU)，超過 0.5 則這個 anchor box 就被認定為有 object，而實作的時候我們並沒有特別去改 anchor box 的數量和比例，和[4]所使用的方法相同。



Anchor box 出自[5]

Method

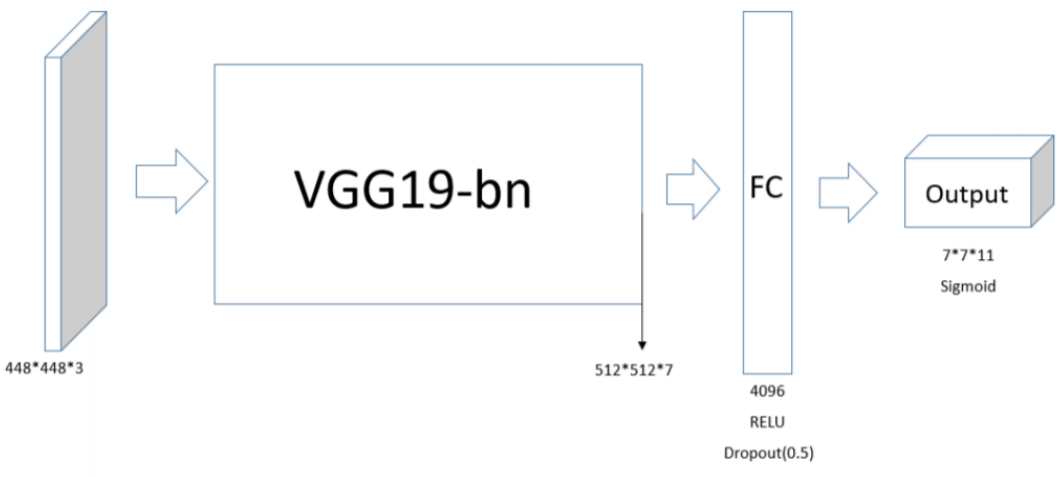
YOLOv1:

我們使用[1],[3]來實作，並通過 kaggle baeline，YOLOv1 主要方法如下:

1. 將輸入圖片分成 7*7 的網格，每個網格只負責預測一個 object，每個網格預測兩個 bounding box，因此總共會有 7*7*2=98 個 bounding box。預測時圖片會通過常見的 CNN model 如 VGG、ResNet 抽出 feature，再通過全連接層來預測結果。
2. 輸出形式:預測每個 bounding box 的中心位置座標，長、寬和 confidence score，和 one hot encoding 的 class label，因此 YOLOv1 的輸出維度是 7*7*(10+N)，N 代表有幾種 class，以本次 final 來說，N=1。
3. Loss function : location loss + confidence score loss + classification loss，其中 location loss 會乘上較大的權重，而沒有 object 的 confidence loss 會乘上一個較小的權重。
4. Non-maximum suppression(NMS) : 用來決定最後那些 bounding box 該留下來的方法，首先會保留 confidence score 最高的 box，並計算剩餘的 box 和分數最高 box 的 IOU(兩個 box 重疊的比例)，IOU 超過閥值的 box 就會被放棄， NMS 會一直重複這個動作直到沒有 bounding box 為止。

Model 架構:

Feature extractor 的部分有嘗試 VGG19_bn、ResNet50、DenseNet121



訓練參數:

將圖片 resize 到 448*448

Epoch : 20

Batch size = 16

Optimizer : SGD, momentum = 0.9, weight decay = 5e-4
Learning rate : 前 10 個 epoch = 0.001，後 10 個 epoch = 0.0001
Data augmentation : 水平翻轉、縮放、平移
Output confidence score threshold = **0.1**
NMS IOU threshold = 0.5

Retinanet:

我們用[2][4]來實作，並得到比 YOLOv1 更好的結果，Retinanet 的主要方法如下:

1. 使用 Feature Pyramid Network(FPN): FPN 會將 feature extractor 中不同層的輸出疊在一起，再分別通過 classification subet 與 box regression subnet 來預測 label 和 box 的精確位置，可以想像成不同層的輸出是圖片在不同尺度下得到的結果，因此前幾層的輸出有助於預測小的 box，而後面幾層可以預測大的 box。
2. Anchors : 運用 anchor box 的技術，就不會像 YOLOv1 受限於最多只能預測 98 個 box，且可以自己決定要用多少個 anchor box，因此可以在準確率和時間上作出取捨。
3. Focal loss: 在 one stage object detection 中，會遇到 class imbalance 的問題，只有少數的 box 是有 object 的，大多數的 box 沒有 object 且很好判斷，這會使 training 的時候 loss 被這些沒有 object 且好判斷的 box 佔據，使得最後得到的 model 無法和 two stage object detection 有相同的準確率，因此[2]提出 focal loss 來解決此問題。

   Focal loss 是在 cross entropy loss 上作出改變，alpha 可以改善正負樣本不平衡的問題，gamma 則可以改善各個 box 難易度不同的問題。

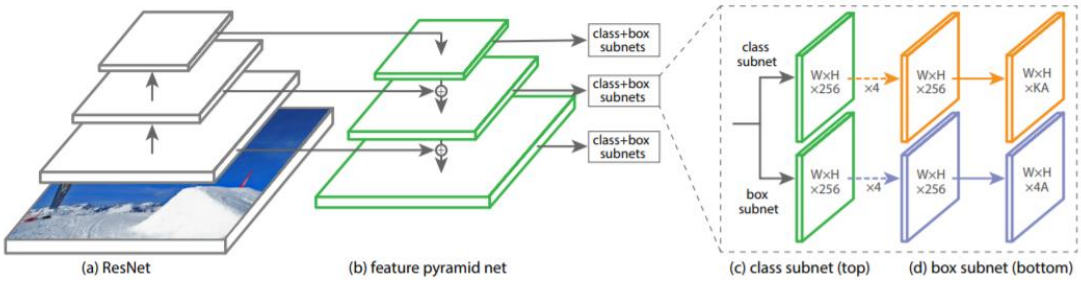
$$\mathrm{CE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{otherwise.} \end{cases}$$

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise,} \end{cases} \qquad \alpha_t = \begin{cases} \alpha & \text{if } \quad y = 1 \\ 1-\alpha & \text{otherwise} \end{cases}$$

   Focal loss 最終可以寫成:

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

   Model 架構:
   出自[2]



訓練參數:

將圖片 resize 到 608*608, 使用 ResNet50

Epoch : 20

Batch size = 8

Optimizer : Adam, Learning rate = 0.00001

Data augmentation : 水平翻轉

Output confidence score threshold = **0.5**

NMS IOU threshold = 0.5

Experiment and Discussion

1. YOLOv1 使用不同的 CNN fearture extractor:
   現在已經有許多的 CNN model，皆在 ImageNet 上取得了很好的成果，不同的 model 或許能得到不同的結果，結果如下:

	VGG19_bn	ResNet50	DenseNet121
Kaggle public result	0.17791	0.15898	0.15157

   從結果上來，比較讓人意外的是在 ImageNet classification 表現較好的 ResNet 和 DenseNet 在這個 task 上表現得比較差，我們覺得可能原因有兩點。第一是 VGG 的參數量比較多，比較有機會找出一組適合本次 task 的參數，另一個原因猜測是 ResNet 和 DenseNet 最後會做 average pooling，說不定會讓一些精確的資訊消失，而 YOLOv1 又只用最後一層的結果來預測 box，導致預測結果比 VGG 還差。

2. Yolov1 和 Retinanet 的比較，結果如下:

	YOLOv1	Retinanet
Kaggle public result	0.17791	0.18847
Kaggle private result	0.16962	0.18670

   我們認為這是合理的結果，因為 Retinanet 是改進過的方法，運用 FPN、anchor box、focal loss 等技術，因此會有較好的結果。

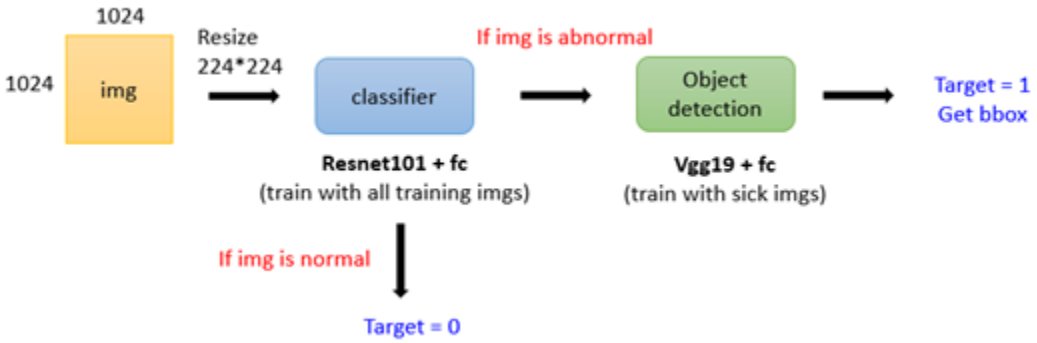
3. Box shrinking :經過 final presentation 聽同學說到而有做嘗試，以下是比較結果:

	原始大小		縮小成 0.9		縮小成 0.85	
YOLOv1	0.17791	0.16962	0.21330	0.17911	0.21632	0.17911
Retinanet	0.18847	0.18670	0.23415	**0.20063**	**0.24567**	0.20000

上面表格左邊是 kaggle public 的分數，右邊是 kaggle privte 的分數。
Public 最好的結果為 0.24567，private 則是 0.20063

可以看到將 box 的大小縮小在分數上有很大的提升，無論是 YOLOv1 或是 Retinanet 都可以突破 strong baseline。我們認為是在訓練的時候要先對圖片做 resize 才會造成此種結果，因為把圖片變小的過程中，可能會失去一些重要的資訊，而 model 預測出來的 box 又必須變回原始圖片的大小，這樣一來一回的過程中可能使 model 傾向於預測較大的 box，因此將 model 預測出的 box 等比例縮小，能夠在分數上有很大的提升。

實驗一
之前的實驗顯示，如果把所有的 training data 直接丟入 YOLOv1(VGG19+fc)，效果可能不會到那麼好，原因有可能是因為，大部分我們拿到的 training data 都是屬於沒有生病的影像(沒有 box)，因此如果先 train 一個 classifier 做 2 元分類(生病/沒生病)，再將那些有生病的圖丟到 model，不知道會不會得到比較好的效果，新的 model 如下圖。



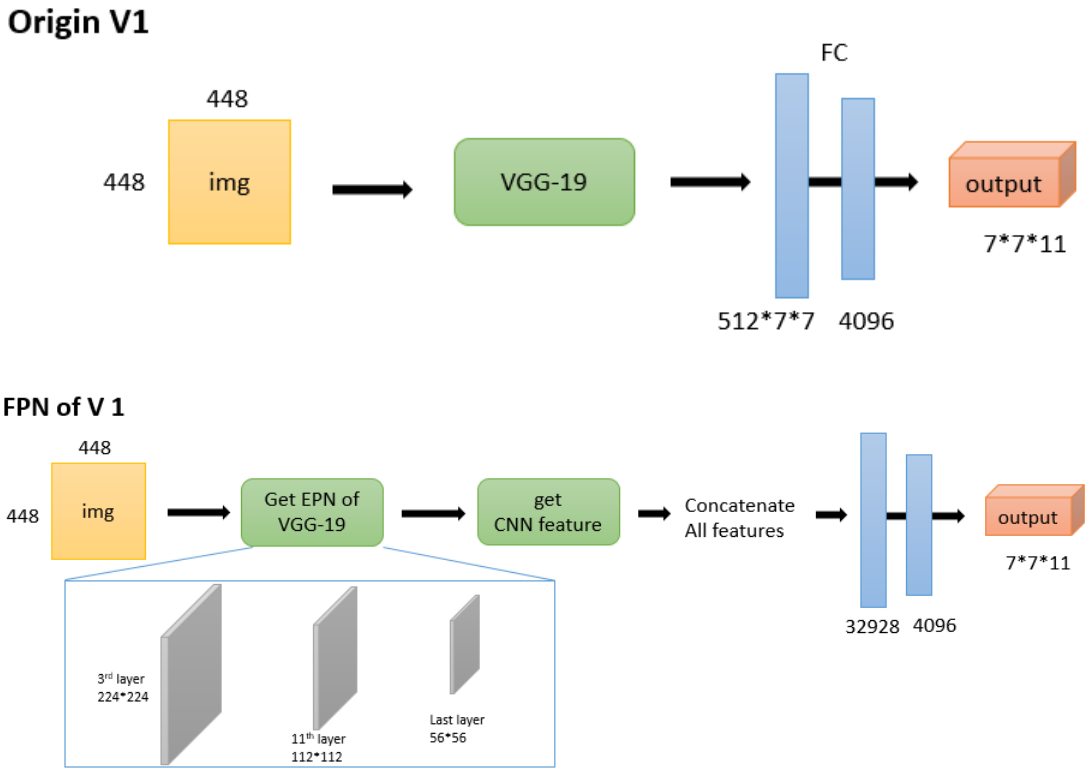
首先，我們會將所有的 training data 圖片都 resize 成 224*224，然後放置 resnet101 抽取 feature，訓練二元分類的 classifier。另外，我們會將所有生病的圖片用來訓練此 model，而架構和先前一樣。如果有一張 testing 圖片，則他會先經過 classifier 做分類，如果有問題才會給 object detection model 去做 bounding box。

但是結果似乎不太理想，雖然我們可以將 classifier 訓練的還不錯(validation 大概可以答對 8 成)，但是因為 data imbalance 的問題，亦即沒生病的圖太多，所以分類器看到新的圖片大可全部猜沒生病，最終的 validation 也會很高，即

便我們調整不同 resize 的大小(224、256、512)去 train，但分類器似乎還是不太能學到東西，如果把 data 弄得更 balance 一點(好壞各 5000 張)，結果 validation 只會更低(50%左右)，可能是這樣的訓練資料真的有點少而學不好。而 object detection 的 model 因為看到的圖片相當少(只有生病的圖，5603 張)，所以不太能 train 起來，最終上傳至 kaggle 的分數比先前的還低，只有 0.12 左右，因為 classifier 認為所有 testing 的圖片都是沒生病的，因此這邊的關鍵點還是有沒有把分類器訓練至很好。

實驗二

另外，我們嘗試使用 FPN(Feature Pyramid Networks)的架構用在 yolo 上，看看是否因此因為參考比較多層的 feature，而有比較好的結果，而在這邊，我們選取 VGG-19 中前中後三層的 layer 作為最後送進 fully connected 的 feature，而這三層的 feature image 使從原本位在第 3、第 11 以及最後一層的架構中抽出來的，而這些 feature image 還會再經過我們自接的 convolution 層做一些資訊的整理，可以參考下圖，圖二為原本的 yolo v1，圖三為後來使用 FPN 在 yolo 架構上設計的 model



但是結過似乎不太理想，上傳到 kaggle 的分數大概也是落在 0.1265 左右，猜測原因可能為，選擇的 layer 可能不是具有代表性的 feature，再加上又多了 convolution 重新去整理資訊，使得參數變得相當的多，但是資料卻還是原本的量，其實從 learning 的過程中我們也發現到，一開始的 training loss 比之前純

粹使用 yolo v1 的 loss 高了一些，validation 相對也是這樣，而最終用來 testing 的 model 所訓練的 loss 還是比之前的高，所以 model 可能沒辦法 learn 得很好，如果有更多的資料(含有 box 的)，推測訓練的結果或許會更好。

4. 提高 performance 的方法:
    a. 我們認為 Retinanet 的結果應該可以做到更好，比如說將 feature extractor 換成更深的 ResNet101，或是改用收斂速度比較慢，但比較能得到更好結果的 SGD optimizer 等，以及 Retinanet 中 focal loss alpha 和 gamma 的值，改動這些參數都有機會提升結果，但因為時間因素，沒有多做嘗試。
    b. 使用 ensemble 的方法，我們本次都用訓練資料的前 20000 個 case 當作訓練資料，剩下的來 validation，因此在訓練時並沒有用到所有的資料，從過去課堂作業的結果顯示，ensemble 的方法能有效的提高 kaggle 的分數，因此多訓練幾個 model 一起預測應該也能得好的結果。

Conclusion

本次 final project 做了醫療影像檢測，我們運用 object detection 中著名的方法 YOLOv1 和 Retinanet 來檢測胸部 X 光影像沒有有生病，並嘗試精確的框出生病的位置，我們在 YOLOv1 和 Retinanaet 上都有取得不錯的結果，並得到以下幾點結論:

1. YOLOv1 使用不同的 CNN model 會得到不同的結果，不同的 task 適合不同的 model，並非在 ImageNet 做得比較好的 model 就一定能有較好的結果。
2. Retinanet 中使用的 FPN，anchor box, focal loss 等方法確實有助於提升預測的準確率。
3. 因為原始影像偏大，對影像做縮放時後會失去一些重要的資訊，導致 model 傾向於預測較大的 box，將原始結果等比例縮小可以讓分數大幅提升。
4. 因為 data imbalance 的關係，先訓練一個二元分類器判斷影像是否生病，再只用有生病的影像來訓練 YOLOv1，但結果變的更差，應該是因為訓練資料過少導致，因此即使是完全沒有 box 的 negative sample，仍有助於訓練 YOLOv1 的參數。
5. 雖然 deep learning 的方法確實能有效預測 X 光影像中的生病位置，但如何改善 data imbalance 和原始影像過大等問題，仍有待大家深入研究，找出更好的解決方法。

Reference

[1] You Only Look Once: Unified, Real-Time Object Detection https://arxiv.org/pdf/1506.02640.pdf
[2] Focal Loss for Dense Object Detection https://arxiv.org/pdf/1708.02002.pdf
[3] Yolov1 code https://github.com/xiongzihua/pytorch-YOLO-v1
[4] Retinanet code https://github.com/yhenon/pytorch-retinanet
[5] Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks https://arxiv.org/pdf/1506.01497.pdf

[6]Object Detection On Aerial Imagery Using RetinaNet https://towardsdatascience.com/object-detection-on-aerial-imagery-using-retinanet-626130ba2203

[7]RSNA Pneumonia Detection Challenge(2nd place solution) https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/discussion/70427