Small Data Training for Medical Images

Team : 機械學習機器學習

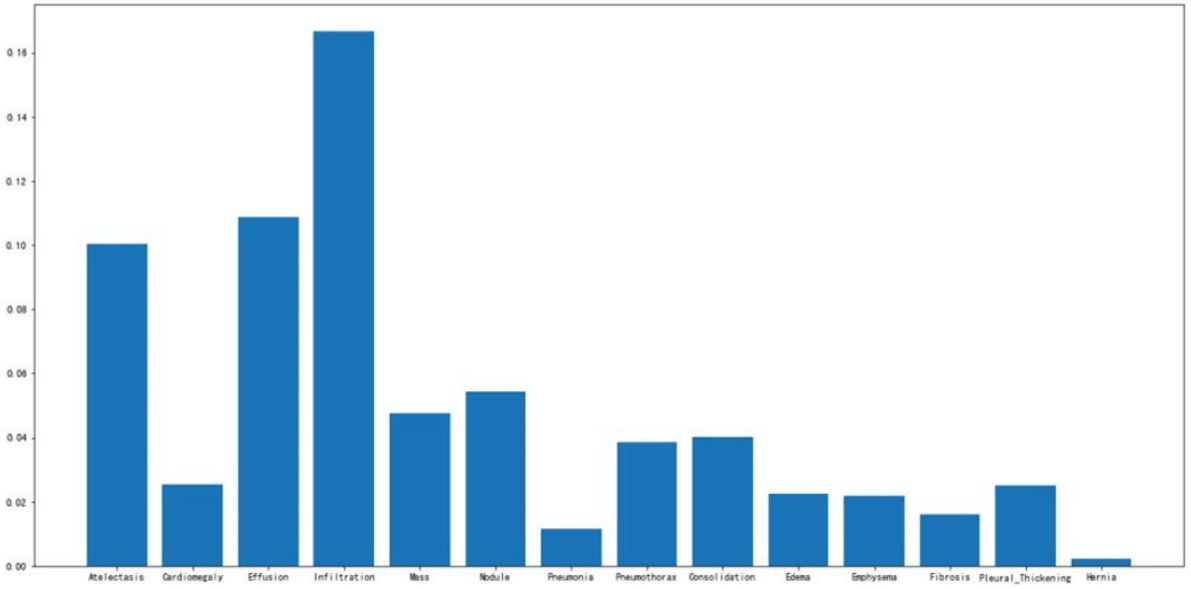
Team member :司福民 陳俊翰 陳振豪 許晉豪

1. Introduction & Motivation

現今在醫療圖片應用上，儘管我們可以很容易的得到大量的診療圖片。但是在診療圖片上的分類、標記成本卻是非常高昂。所以通常都只有少數的圖片是有被標記的。然而如果使用CNN去對很少數標記的圖片去做訓練，往往會造成over-fitting。信運的是，以目前Deep Learning的技術，可以將CNN network分成兩部分:feature extractor和classifier,分開去進行訓練,來達到較佳的結果。因此我們的目的就是去找到較佳的方法去訓練feature extractor和classifier。

2. Data Preprocessing & Feature Engineering

Exploratory Data Analysis ：

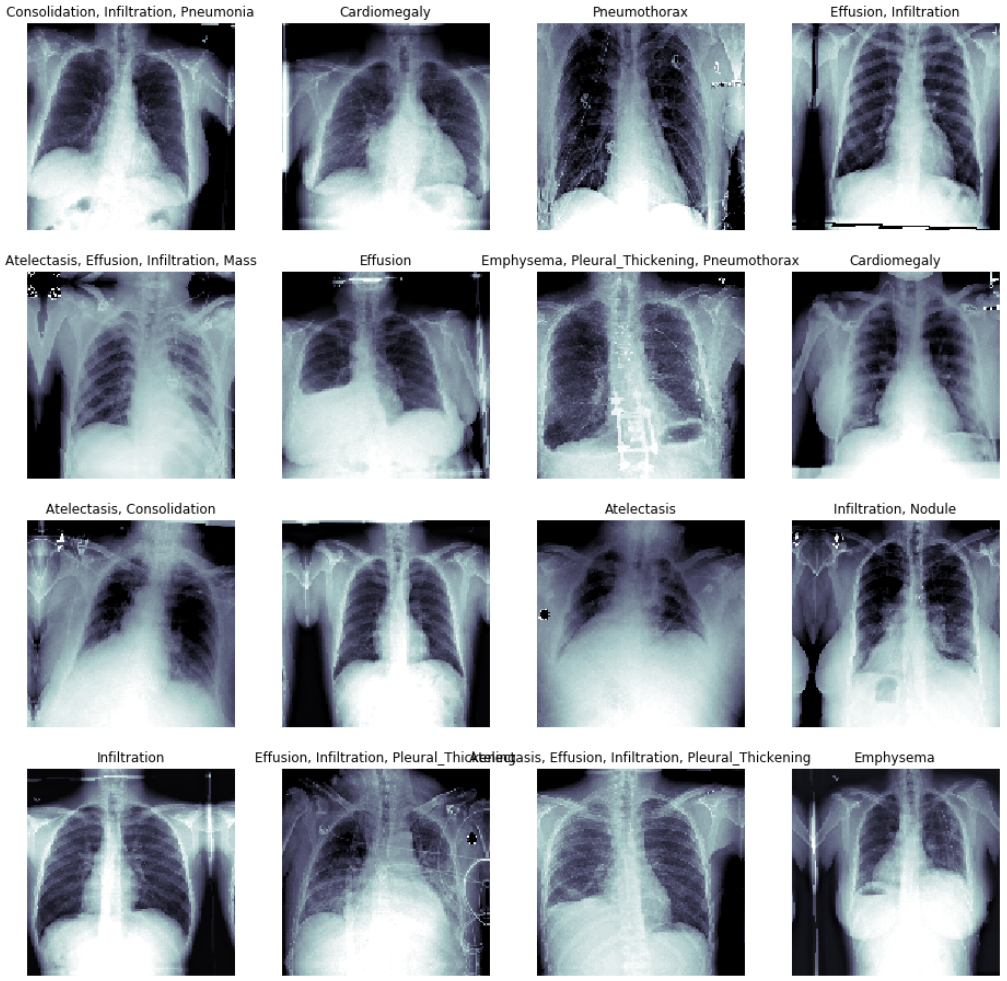


疾病分布

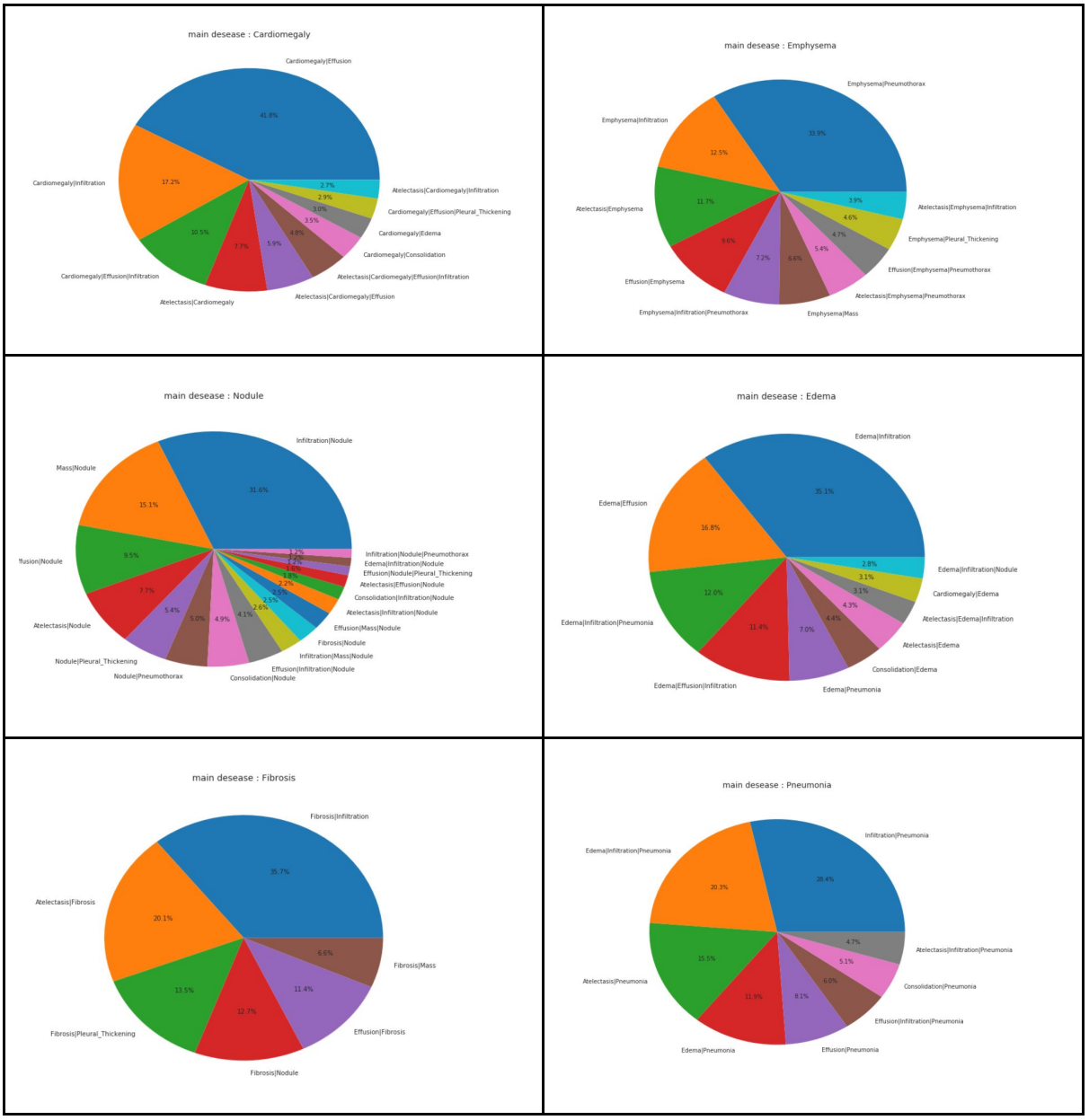
由上圖可以看出疾病的分布非常的不平均，可能會產生的問題是有些數量很少的疾病如 : Hernia，會在其他病還沒收斂的時候就overfit，經實測結果Hernia在第一個epoch還沒結束就會overfit，因此需要我們在第四部分提出一些特別的方法解決這個問題。

Domain Knowledge (疾病概述及觀察):

Atelectasis 肺擴張不全：肺腔小小的
Cardiomegaly 心臟肥大：心臟大大的
Effusion 胸積水：會有水平線！！！認真？
Infiltration 肺浸潤 ：consolidation的餘集？白白霧霧？無輪廓！不能太白
Mass >30mm清晰結節：大大的白點點（病？）
Nodule <30mm清晰結節：小小的白點點
Pneumonia 肺炎：有點像Consolidation（病！）
Pneumothorax 氣胸：滿滿空氣一片黑！！！
Consolidation 肺實變：跟mass很接近(影像特徵白白的)？有輪廓
Edema 肺水腫：大片霧霧的？
Emphysema 肺氣腫：肺泡壞掉，漏氣弱版氣胸(沒那麼黑，沒紋路)
Fibrosis 肺纖維化：白白像蜘蛛網？沒有一大塊白白的？
Pleural Thickening 肋膜肥厚：整個「框框」厚厚的
Hernia 疝氣 ：有不明物體鼓起！！



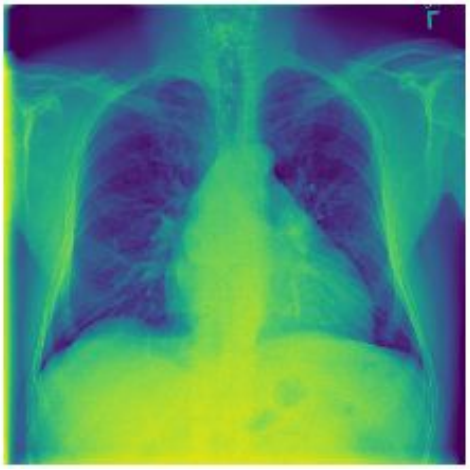
疾病條件機率分布:



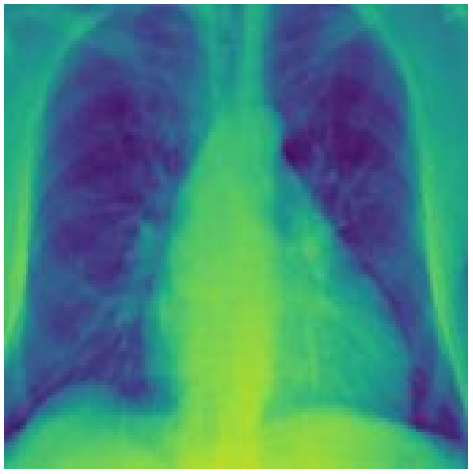
我們從醫學系同學說明、X光片的觀察、疾病的條件機率分布找出一些重要的資訊，例如: Infiltration和Consolidation相似度非常高，實際看X光片不是專業人士很難分辨出來且兩個病的關聯性也很高約18.3%(consolidation出現，Infiltraion出現的機率)，另外從疾病的條件機率分布也觀察到有些病有更高度的關聯(3x%)如: Endemma和Infiltration、Fibriosis和Infiltration、Nodule和Infiltraion，我們發現Infiltraion資料量多且和很多病都常常一起出現。

本次題目所提供的醫療影像大小為1024*1024，我們在使用densenet121作為訓練模型的情況下，batch_size最大只能到2。這個batch_size對單category的分類問題或許還可

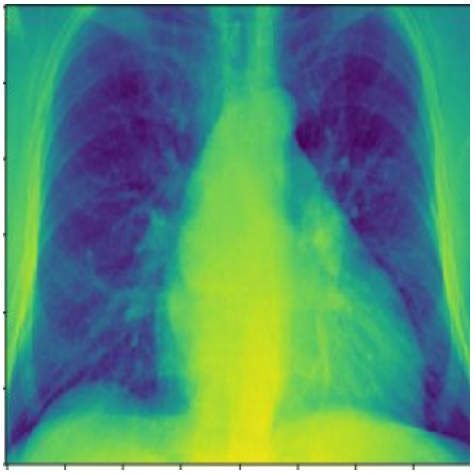
行，但對多category同時資料又具有複數label的分類問題，這個batch_size將使得各batch差異過大，導致模型難以收斂。
我們觀察到這次的label是肺部疾病，其最具代表性的特徵應出現在胸部X光的中央區域，外圍部分是可被捨棄的，因此，我們選定了3種圖像大小進行cropping，分別是600*800、300*400、100*134，而其相應的batch_size最大可到4、12、120。



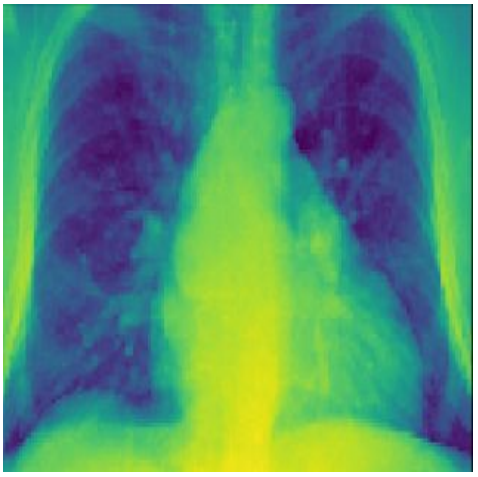
1024*1024



600*800



300*400



100*134

實際訓練發現，600*800的cropping參數搭配相應的batch_size，具有最好的表現。
另一方面，訓練資料包含一萬筆label data和數萬筆unlabel data，如何使用有限的label data，生成更多具有品質的pseudo-label data，是提高分數的關鍵之一。以下分項介紹本組嘗試過的圖像生成方法：

(1) GAN

我們參考部落客Yi-Hsiang Kao的GAN程式碼教學[2]，修改其模型架構，輸入unlabel data，嘗試生成更多具品質的unlabel data，增加資料庫深度。
然而，由於醫療影像之大小過大，連帶影響模型參數量飆升，超出了常規桌機記憶體的負荷範圍，我們嘗試將原始圖片縮小，同時降低模型的複雜度，卻發現模型生成的圖片走樣，不但解析度不足，且多項特徵無法還原。最終我們只能放棄使用GAN。

(2) Data augmentation

我們使用Keras.preprocessing.image提供的ImageDataGenerator，對label data進行微幅修飾，增加資料量。在設定augmentation的參數部分，我們分別觀察各項參數的合理上限，並檢驗所有參數同時使用是否合理，最後選定的參數項目與上限如下：

rotation_range=10.0,
width_shift_range=0.05,
height_shift_range=0.05,
shear_range=0.04,
zoom_range=0.04,
horizontal_flip=True,
vertical_flip=False

我們將修飾後的圖片與原始圖片放於同目錄下，賦予它們與原圖同樣的label，並將資料輸出成csv檔，以利後續訓練使用。

最後在訓練過程中，由於單筆資料較大，不會一次將所有訓練資料讀入，而是在需要使用到該筆資料時才將其讀入，以確保記憶體(RAM)不會因資料暫存而被消耗。此一功能可以使用Keras.utils.Sequence，搭配Keras.models.fit_generator來實現。本組為節省開發時間，採用博主chestnut提供的class DataGenerator作為初始架構[1]，再根據我們自己的需求修改後完成。

3. Model Description(two models)

在初期時，我們使用VGG16的model並將最後一層的output layer改成經由sigmoid然後產生14個0~1的輸出值。跟新函數的方法為Adam優化器，參數則是Keras預設值(lr=0.001, beta_1=0.9, beta_2=0.999)。我們將epoch設為50並記錄其中有最高的validation accuracy的model。然而這個方法的結果在Kaggle上的成績卻不盡理想，在Public Leaderboard上只拿到0.60266的分數。

為了加強我們的model，之後我們選擇DenseNet121。DenseNet可以減輕梯度消失的問題，並且可以加強圖片特徵上的傳遞。另外跟ResNet相比較起來，DenseNet能使用到較少的參數、計算量且可以得到較佳的結果。在訓練圖片前，我們先使用ImageNet 的pre-trained model。之後一樣將最後一層的output layer改成經由sigmoid然後產生14個輸出。跟新函數的方法為Adam優化器，參數則是(lr=0.00001, beta_1=0.9, beta_2=0.999)。因為訓練時間較長，我們將epoch設為20並記錄其中有最高的validation

accuracy的model。低一個版本就在Kaggle的Public Leaderboard上也取得0.76左右的成績。所以我們就選取DenseNet作為我們的model。

4. Experiment & Discussion

底下為我們嘗試過的各種方法 :

1. Ensemble 14 個單病model

方法 : 將14種病分別輸入14個的model，最後將14個病的預測結果整合在一起，每種病的模型都是DenseNet，只做一些參數上面的微調。
結果 : 效果沒有預期的好，且因為訓練時間會是原本的14倍，沒有足夠的時間調整參數最佳化模型，而在前面也有提過14種病之間不是完全獨立的，有些病之間是有關聯性，例如 : Infiltration 肺浸潤和Pneumonia 肺炎就有非常強的關聯，普通人可能根本分辨不出來，所以這個方法沒有達到預期效果。

2. Seed & Proportional data

為了確保每次epoch訓練的data能有一定比例的資料，解決資料不均衡的問題，且為了解決有些稀少的疾病會很快就overfit的問題，我們設計了"Seed"，在每個epoch取出一定數量的data來做訓練，且讓資料比例是我們設計的情況。

```
def randd(seed):
    global k
    k=np.random.choice(len(TesIdList),seed,replace=False)
    return k
def randd2(seed):
    global k2
    k2=np.random.choice(len(TrainListall),seed,replace=False)
    return k2

    training_generator = DataGenerator(TrainIdList[randd2(para['train_seed'])],MultiHotLabel[k2],
        TrainingPara['batch'],Data['width'],Data['channel'],Label['category'])
    validation_generator = DataGenerator(TesIdList[randd(para['val_seed'])],MultiHotLabeltes[k],
        TrainingPara['batch'],Data['width'],Data['channel'],Label['category'])
```

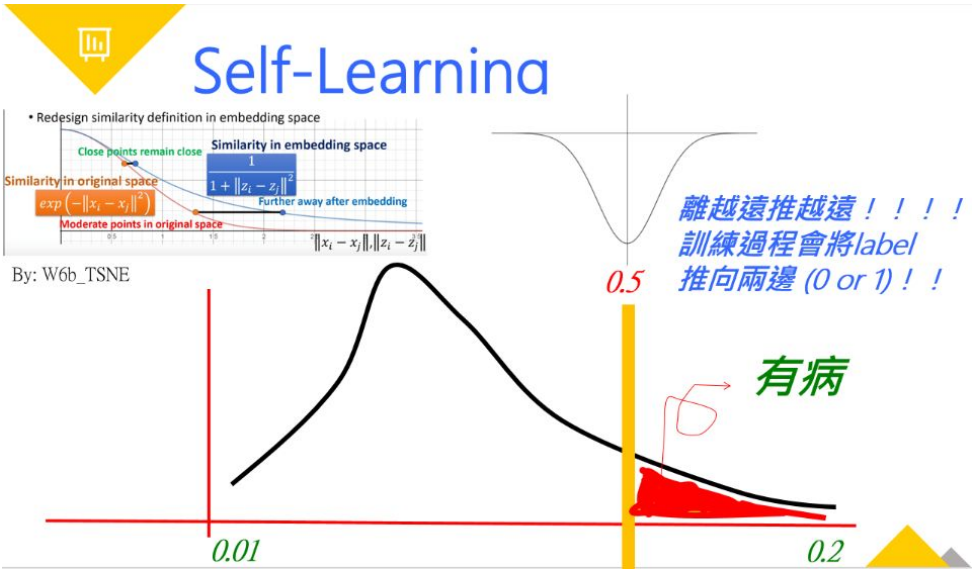
3. Auto-Encoder (Unsupervised Learning)

方法:將DenseNet的model當作encoder，然後透過covolution、upsampling組成decoder。最後將整個model合併在一起。訓練方式為，encoder將unlabeled的圖片壓縮為特徵向量，decoder在將壓縮後的結果還原成原來的圖片。然後再將訓練完的encoder的model來當DenseNet的pre-trained model。
結果:使用auto-encoder生成的pre-trained model並不會比使用ImageNet上來的好。可能是因為auto-encoder並沒有將圖片還原得很完美，導致部分重要的病理特徵被模糊掉了。所以這個方法也未能達成我們的要求。

4. Self-learing

除了label data外，unlabel data的正確使用也是決勝點之一。我們嘗試使用Semi-supervise中常見的self-trianing method，將unlabel data輸入至使用label data訓練好的可靠模型進行predict，並對預測結果最靠近1和0的5%進行pseudo-label，再丟入模型進行訓練。

然而，初步測試時的結果並不理想。我們分出部分的label data，丟入剩餘label data訓練出的模型進行predict、取pseudo-label，卻發現正確率不高。最終我們放棄使用self-trianing。

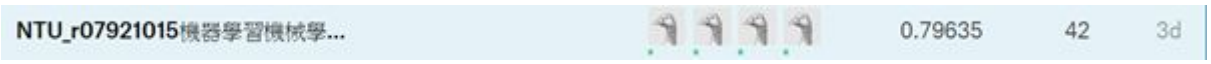


我們將原圖片10002張加上augmentation出來的100020張，總共110022張。一起在DenseNet121的模型下訓練,並使用Seed 和Proportional data方式來防止over-fitting。底下為我們訓練出來的結果在validation上的表現:

病名	Atelectasis	Cardiomegaly	Effusion	Infiltration	Mass	Nodule	Pneumonia
AUROC	0.80205	0.85530	0.83842	0.70864	0.74315	0.76621	0.75377

病名	Pneumothorax	Consolidation	Edema	Emphysema	Fibrosis	Pleural Thickening	Hernia
AUROC	0.85510	0.78410	0.83370	0.77520	0.72993	0.74148	0.79342

在Kaggle的Public Leaderboard上的表現:



5. Conclusion

從這次的挑戰，我們發現模型沒辦法解決所有問題，這次專案剛開始我們參考CheXNet的DenseNet121模型，且利用ImageNet的Pretrain Model，卻成效不彰，而參數調整和資料處理才是提升成效的有效方法，經過trial-and-error，調整ADAM的learning rate到 10^{-4}，資料處理方面，我們將圖片裁減只看胸腔部分，且利用Data Augmentation解決資料過少會產生overfittig的問題，再加上前面所提到的"Seed and Proportional Data"解決資料不均衡的問題。

目前我們的訓練的資料，主要還是在使用label data。所以我們之後的目標會繼續研究在如何更有效的利用unlabeled data，像是提高auto-encoder的準確性或是self-training的參數調整，另外因這次花很多時間在調整參數方面心很累，希望之後可以"Genetic Algorithm"來自動最佳化訓練參數(learning rate 、weight decay等等)，希望未來能達到更好的結果。

6. Reference

1. https://zhuanlan.zhihu.com/p/35005794?fbclid=IwAR0u2KAsJYdNGp6MXnK-Ui19V2HJyfeCnV3UBVwy_F7nv-7RYY8Iqe4Eqys
2. https://chtseng.wordpress.com/2017/11/11/data-augmentation-%E8%B3%87%E6%96%99%E5%A2%9E%E5%BC%B7/
3. https://blog.csdn.net/m0_37477175/article/details/79716312
4. http://medium.com/@gau820827/教電腦畫畫-初心者的生成式對抗網路-gan-入門筆記-tensorflow-python3-dfad716629
5. https://www.kaggle.com/sbernadac/lung-deseases-data-analysis
6. Yosinski, Jason, et al. "How transferable are features in deep neural networks?." *Advances in neural information processing systems*. 2014.
7. Tajbakhsh, Nima, et al. "Convolutional neural networks for medical image analysis: Full training or fine tuning?." *IEEE transactions on medical imaging* 35.5 (2016): 1299-1312.
8. Huh, Minyoung, Pulkit Agrawal, and Alexei A. Efros. "What makes ImageNet good for transfer learning?." *arXiv preprint arXiv:1608.08614* (2016).
9. Tran, Toan, et al. "A bayesian data augmentation approach for learning deep models." *Advances in Neural Information Processing Systems*. 2017.
10. Antoniou, Antreas, Amos Storkey, and Harrison Edwards. "Data augmentation generative adversarial networks." *arXiv preprint arXiv:1711.04340* (2017).
11. Suggested Source Code for Chest X-Ray Dataset CheXNet implementation in PyTorch (https://github.com/zoogzog/chexnet)