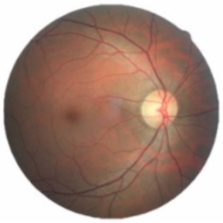
DLP Lab3 309551067 吳子涵

1. Introduction

在這次的 lab 中，需要透過分析視網膜的圖片，來預測糖尿病視網膜病變的程度，嚴重程度由輕到重分別用 0 到 4 表示。首先，要用 Pytorch 的框架建立自訂義的 Dataloader；接下來，使用 ResNet18 與 ResNet50 以及各別先經過 pretrain 過的網絡來進行病變嚴重程度的預測；最後，計算 confusion matrix 來評估 4 個 network 的成效。左下圖為資料集的圖片範例，右下圖為 5 個類別所代表的意義。



Class
0 - No DR
1 - Mild
2 - Moderate
3 - Severe
4 - Proliferative DR

2. Experimental setups

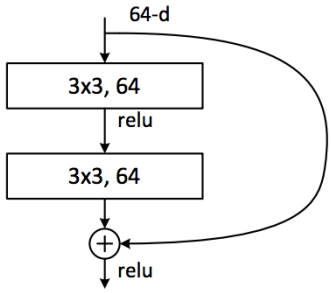
A. The detail of your model (ResNet)

下表為 ResNet 的結構圖。表格中一個中括號就代表一個 basic block 或 bottlenet，在 ResNet18 中使用的是 basic block，而在 ResNet50 中使用的是 bottlenet。建立完 2 種基本的 block 後，再根據以下 18-layer 和 50-layer 的架構與參數來建構 ResNet 的網絡。

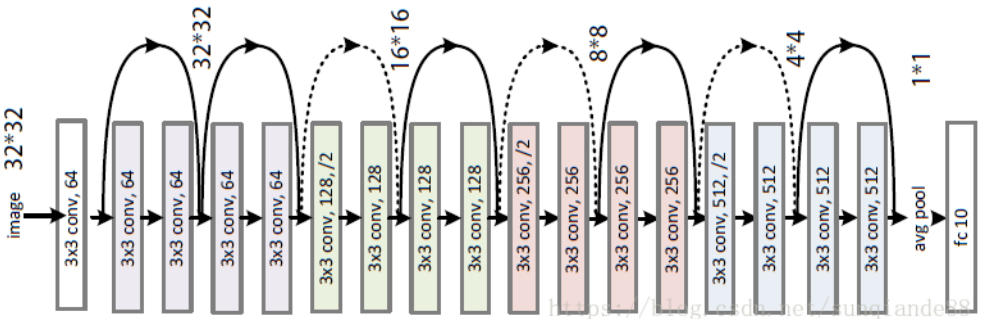
layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

(1) ResNet18

一個 basic block 包含 2 層 kernel size 為 3x3 的 convolution layer，搭配 batch normalization 和 ReLU，最後把輸出的結果加上 input 的值，就完成一個 basic block 的輸出。下圖為 basic block 的示意圖。

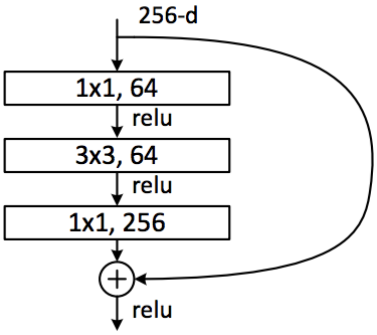


ResNet18 就是由許多的 basic block 堆疊起來，詳細架構如下圖所示。其中，紫色部分代表由 2 個 channel=64 的 basic block 堆疊；綠色部分代表由 2 個 channel=128 的 basic block 堆疊，依此類推橘色及藍色部分的架構。最後經過 fully connected layer 即可得到預測的類別。



(2) ResNet50

一個 bottlenet 包含一層 kernel size 為 1x1 的 convolution layer、一層 kernel size 為 3x3 的 convolution layer、一層 kernel size 為 1x1 的 convolution layer，並搭配 batch normalization 和 ReLU，最後把輸出的結果加上 input 的值，就完成一個 bottlenet 的輸出。下圖為 bottlenet 的示意圖。



ResNet50 就是由許多的 bottlenet 堆疊起來，其中，有 channel 數分別為 64、128、256、512 的區塊，而相對應的 bottlenet 數量分別為 3、4、6、3。最後經過 fully connected layer 即可得到預測的類別。

B. The details of your Dataloader

Dataloader 共有 3 個 function，分別為 init, len, getitem。

(1) init function: 初始化參數

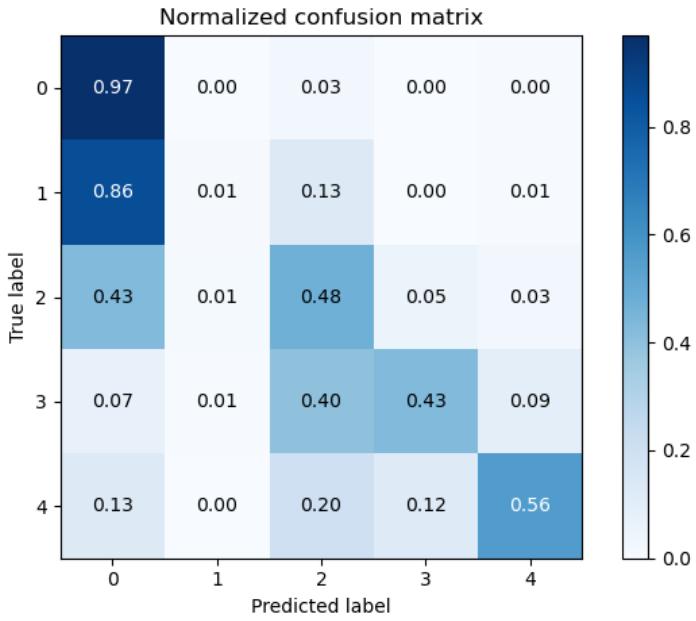
(2) len function: 回傳 dataset 的大小

(3) getitem function: 指定如何取得數據，並將數據轉換成 tensor 形式。同時對資料進行格式轉換：將圖片的 pixel value 由 [0, 255] 轉換成 [0, 1]，再將圖片的格式由 [H, W, C] 轉換成 [C, H, W]。

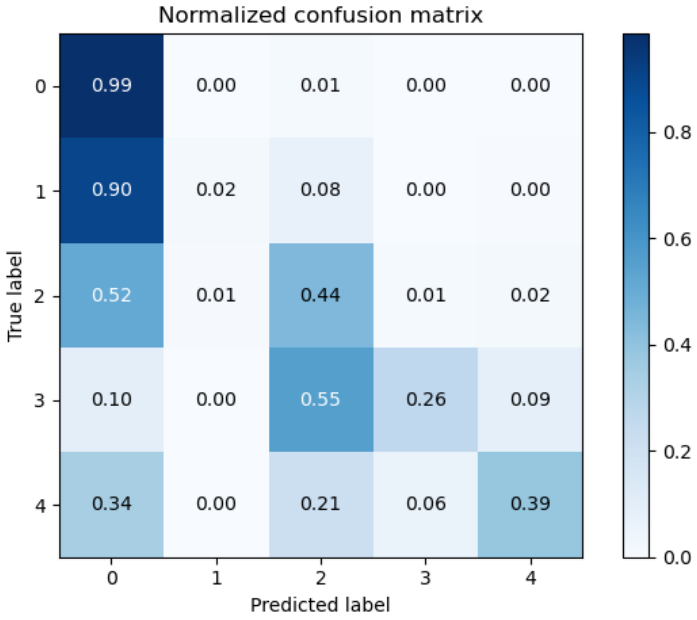
C. Describing your evaluation through the confusion matrix

Confusion matrix（混淆矩陣）是一種判斷模型好壞的指標，x 軸是經由 model 預測出來的 label，y 軸是真正的 label，也就是當資料落在 x 軸和 y 軸的 label 相同時，即代表預測正確。因此，當左上到右下的斜對角顏色越深，就表示 model 的準確率越高。下圖分別為有 pretrain 過的 ResNet18 以及 ResNet50。

(1) ResNet18 with pretraining



(2) ResNet50 with pretraining



由於此資料集的 label 大部分是 0，所以在 label=0 的資料中預測的準確率較高，而 label 為 1 的圖片可能因為和 label 為 0 的圖片差異不大，又因為資料量少，所以大多會被預測成 0，使得準確率低。而 label 越大的圖片，與 label 為 0 的圖片差異較大，因此準確率較 label=1 的資料高。

3. Experimental results

A. The highest testing accuracy

- Screenshot

  ResNet 18 with pretraining

  ```
  ResNet18 with Pretraining Testing Accuracy: 0.8054092526435852
  ```

  ResNet 50 with pretraining

  ```
  ResNet50 with Pretraining Testing Accuracy: 0.8078292012214661
  ```

- Anything you want to present

  一開始訓練沒有 pretrain 的 ResNet18 與 ResNet50 時，分類效果很差，且因資料大多為 label=0 的類型，model 幾乎會把所有資料預測成第 0 類。而使用 pretrain 過的 ResNet18 與 ResNet50 後，model 的分類能力就有慢慢提升，較有能力分辨 label=1~4 的圖片類型。

  ResNet18 與 ResNet50 的準確率差不多，沒有經過 pretrain 的網絡準確率都在 73%左右，有 pretrain 過的網絡準確率都在 80%左右。

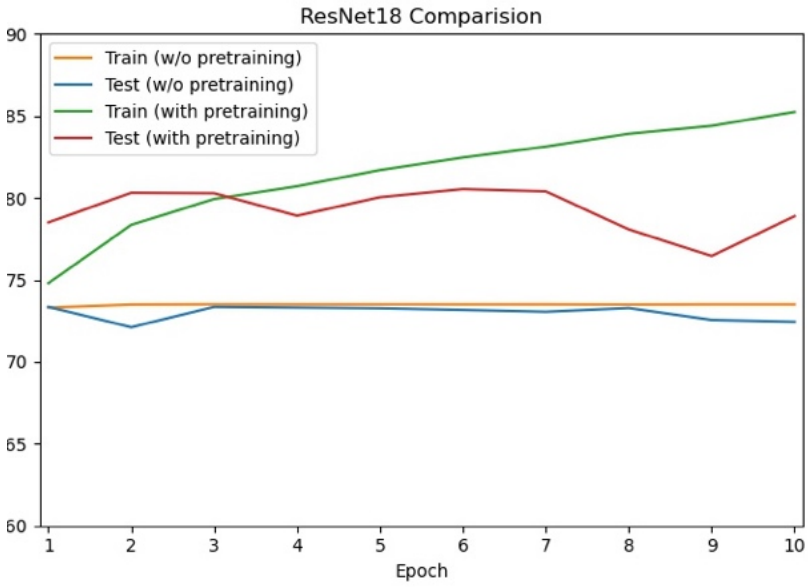
  另外，這個資料集較不好分析，我想有一部分原因是因為資料不平均的關係。我有試著經由複製 label=1~4 的資料，並將這些圖片水平、垂直翻轉來增加資料量，希望 model 能更好的學習 label=1~4 的圖片，不過準確率沒有太大的提升，最高的準確率仍為 80%左右。

B. Comparison figures

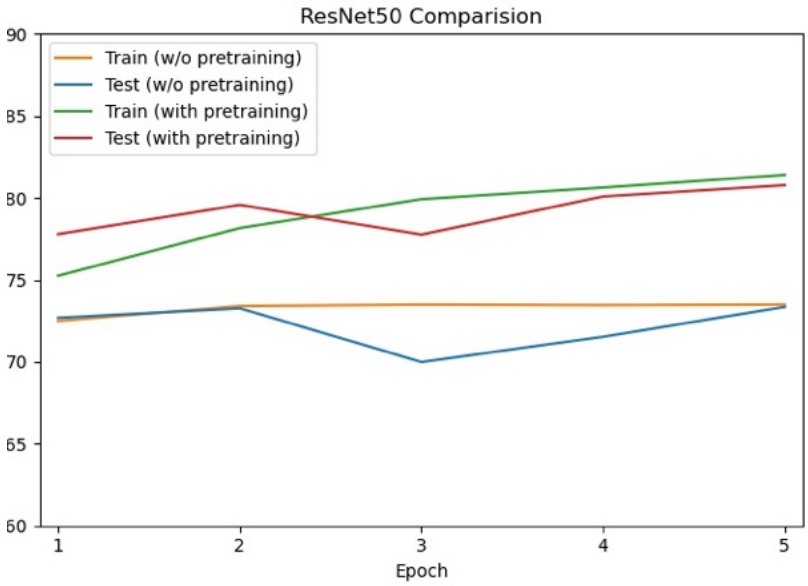
- Plotting the comparison figures
  (ResNet18/50, with/without pretraining)

  不論是 ResNet18 或是 ResNet50，經過 pretrain 都能提升準確率。沒有經過 pretrain 的網絡準確率都只能達到 73%左右，無法提升，且預測結果很差，大部分的預測結果都是 label=0。有經過 pretrain 的網絡準確率都能達到 80%左右。

(1) ResNet18



(2) ResNet50



4. Discussion

A. Anything you want to share

這次的 model 一個 epoch 就要跑很久，要完整跑完 ResNet18 的 10 個 epoch 與 ResNet50 的 5 個 epoch，與他們 pretrain 的版本，需要花 10 小時左右。但是，我有查 gpu 的使用率，在 training 時 gpu 使用率一直都維持在 99%，感覺沒辦法從這部分加速了。