網媒所碩一 R10944028 蔡易儒

DLMI HW report

1. Task introduction

影像分割是許多視覺化系統很重要的一個部分，其目的為將影像或影片的畫格切割出物體的輪廓或邊界，此任務在許多領域都是一個需要解決的問題，醫學影像也不例外，如找尋腫瘤的邊界，因此一路發展下來，有了許多對於此問題的演算法，而此次作業的目標則是使用深度學習的方法來進行影像分割。

另外，影像的分類也是一塊有許多應用的領域，也會拿同一份 dataset 來進行醫學影像的分類，一個將影像做二分類，判斷腦部是否有腫瘤，另一則是做三分類，分成正常、良性腫瘤與惡性腫瘤。

最後一部分則是試著使用兩階段的訓練，先藉由訓練好的分類模型判斷一張影像是否有腫瘤，按照分類模型的預測在訓練的過程進行某種程度的後處理，期望可以達到 knowledge transfer 的效果。

2. Dataset

(1) Brain MRI Images for Brain Tumor Detection
(https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection)

此資料來源為 google images，圖片皆為腦部的磁共振成像，label 為是否有腦瘤，但因為此 dataset 並沒有附上 mask，無法做 segmentation，因此在此 dataset 我們只實作了 tumor detection 的部分。

(2) Breast Ultrasound Images Dataset
(https://www.kaggle.com/datasets/aryashah2k/breast-ultrasound-images-dataset)

此 dataset 為胸腔的超音波檢測，目的在檢查是否有胸部的腫瘤，共分成三個種類，分別是良性腫瘤、惡性腫瘤及正常(無腫瘤)。另外還附有 mask 當作 segmentation 的標準答案，讓我們能夠分別進行監督式與非監督式的機器學習。

此 dataset 的資料來源為為 25 至 75 歲的女性，共蒐集了 600 位女性的胸部超音波圖，於 2018 年蒐集完成。

DLMI HW report

3. Model

(1) custom CNN

分類的 model 都使用自己設計的 CNN-based model，先藉由第一部份的 Convolution layers 進行特徵萃取，最後將萃取出來的特徵交給第二部分的全連階層進行訓練已得到最後的預測機率。輸入為一個 channel 的 512*512 影像，輸出則是看 label 的種類數量決定維度，以 Brain MRI Images for Brain Tumor Detection 來說則為 2，Breast Ultrasound Images Dataset 則為 3。

第一部分的 Convolution layers 由 3 層的 convolution layer 組成，其 kernel 的大小分別為 5,5,5，stride 的大小分別為 2,2,2，padding 的大小分別為 2,2,2，channel 則從 1 -> 64 ->128 -> 256，而每層之間都有使用 BatchNorm、activation function (ReLU)與 Maxpool(2*2)，將所有 neural 壓平後其維度為 256*8*8。

第二部分則為 Fully connected layers，總共有三層分別為 256,64,2 or 3。每層之間僅有使用 activation function ReLU。

整體 model structure 如下圖所示。

```
Classifier(
  (cnn_layers): Sequential(
    (0): Conv2d(1, 64, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2))
    (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU()
    (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (4): Conv2d(64, 128, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2))
    (5): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (6): ReLU()
    (7): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (8): Conv2d(128, 256, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2))
    (9): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (10): ReLU()
    (11): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (fc_layers): Sequential(
    (0): Linear(in_features=16384, out_features=256, bias=True)
    (1): ReLU()
    (2): Linear(in_features=256, out_features=64, bias=True)
    (3): ReLU()
    (4): Linear(in_features=64, out_features=3, bias=True)
  )
)
```

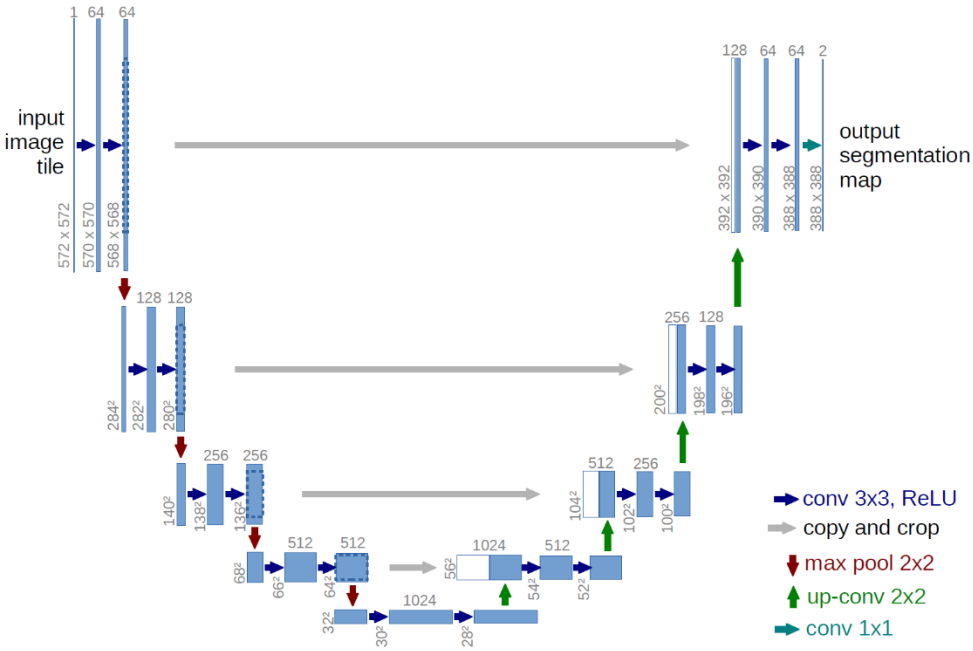
```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 256, 256]           1,664
       BatchNorm2d-2         [-1, 64, 256, 256]             128
              ReLU-3         [-1, 64, 256, 256]               0
         MaxPool2d-4         [-1, 64, 128, 128]               0
            Conv2d-5         [-1, 128, 64, 64]         204,928
       BatchNorm2d-6         [-1, 128, 64, 64]             256
              ReLU-7         [-1, 128, 64, 64]               0
         MaxPool2d-8         [-1, 128, 32, 32]               0
            Conv2d-9         [-1, 256, 16, 16]         819,456
      BatchNorm2d-10         [-1, 256, 16, 16]             512
             ReLU-11         [-1, 256, 16, 16]               0
        MaxPool2d-12           [-1, 256, 8, 8]               0
           Linear-13                 [-1, 256]       4,194,560
             ReLU-14                 [-1, 256]               0
           Linear-15                  [-1, 64]          16,448
             ReLU-16                  [-1, 64]               0
           Linear-17                   [-1, 3]             195
================================================================
Total params: 5,238,147
Trainable params: 5,238,147
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 1.00
Forward/backward pass size (MB): 118.63
Params size (MB): 19.98
Estimated Total Size (MB): 139.61
----------------------------------------------------------------
```

DLMI HW report

(2) UNet

整體來說，UNet 主要由 Convolution layers 所組成，架構某種程度上與 autoencoder 十分相似，前半部都是利用 Convolution layer 與，並不斷減少使用的變數量，將原本的影像，利用高維度但大小較小的特徵向量來進行表示，與 autoencoder 中的 encoder 十分類似，最後也是透過 Convolution layer 將影像從高維特徵向量還原回原本的影像大小，與 autoencoder 中的 decoder 十分類似，而對稱的 encoder 與 decoder 之間也會使用 residual connection 連接，以避免深層網路的梯度消失問題。其模型優點為，複雜度低，因此在醫學影像這種 dataset 比較小的狀況相比更複雜的 model，比較不會遇到嚴重的 over-fitting 問題，大部分影像的結構也較固定，影像的語意也較簡易理解，因此像 UNet 較簡單的模型表現反而能夠更好。

整體 model structure 如下圖，由於版面關係，就不放上實際的 torchsummary，而是以概念圖表示。詳細的參數可以搭配 code 來查閱，或是直接參考 UNet 的參數。



DLMI HW report

4. Experiments

(1) Methodology

實驗主要分為三個部分，預處理的部分都是先將圖片轉成黑白的，並縮放到 512*512 的大小，接著轉換成 0-1 之間的 tensor。

第一是單純的分類問題，在 Brain MRI Images for Brain Tumor Detection 上做簡單的二分類問題，預測是否有腦瘤，而在 Breast Ultrasound Images Dataset 做三分類問題，預測是良性腫瘤、惡性腫瘤還是沒有腫瘤。使用的 loss function 為 Cross Entropy，optimizer 則是使用廣用性較佳的 AdamW，learning rate 為了實驗方便固定為 0.001，L2 norm regularization 設為 1e-5，且有另外設定將 model weight parameter prune 為 5，避免有過大的 weight dominate 整個 model。

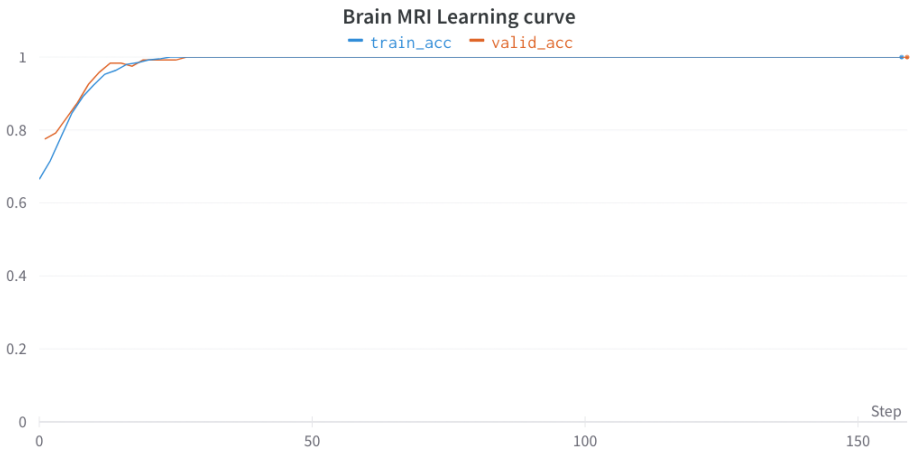
第二部分則是單純的 segmentation 問題，由於 Brain MRI Images for Brain Tumor Detection 並沒有 mask 的資料，因此只在 Breast Ultrasound Images Dataset 上進行此實驗，分別使用 Dice loss, Binary Cross Entropy 及 Mean Squared error 三種 loss function 進行訓練，由於三種 loss function 都有各自的特性，因此決定三種都進行訓練再來觀察其訓練的結果，optimizer 使用廣用性較佳的 Adam，learning rate 設為 0.001，L2 norm regularization 設為 0，也有另外設定將 model weight parameter prune 為 5，避免有過大的 weight dominate 整個 model。

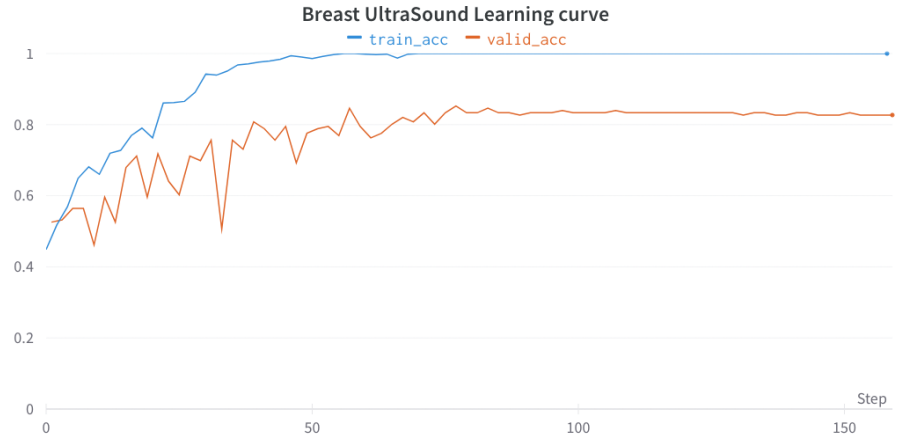
第三部分則是前兩部分的結合與延伸，首先利用第一部分的分類模型，經過訓練後使其進行 label 的預測，接著在第二部分的 UNet 訓練時，參考分類模型的預測結果，如果預測的結果認為此影像屬於沒有腫瘤的話，就直接將 UNet 的輸出結果改為空的 mask，以避免有些應該是正常的影像，偵測出一些小雜訊，藉由此方法就能有效直接將雜訊移除掉，而如果是無腫瘤的影像被預測成有腫瘤，則會保持原樣，唯獨有腫瘤影像被誤測成無腫瘤時，此方法才會使表現變差。與第二部分一樣只在 Breast Ultrasound Images Dataset 上進行此實驗，分別使用 Dice loss, Binary Cross Entropy 及 Mean Squared error 三種 loss function 進行訓練，由於三種 loss function 都有各自的特性，因此決定三種都進行訓練再來觀察其訓練的結果，optimizer 使用廣用性較佳的 Adam，learning rate 設為 0.001，L2 norm regularization 設為 0，也有另外設定將 model weight parameter prune 為 5，避免有過大的 weight dominate 整個 model。

DLMI HW report

(2) Results and Analysis

第一部分：





上圖是第一部分分別在兩個 dataset 的 learning curve，可以明顯發現第一個 dataset 做二分類的任務算是輕而易舉，訓練沒多久就能夠很快的在 train 與 test 兩者都達到 1 的正確率，而觀察了一下 train 與 test 的資料就會發現這個結果可能並不太意外，因為許多 test 的資料在 train 裡面都能找到人眼都能覺得類似的圖片，因此在此 dataset 能做到全對算是在意料之內。

而用同樣的模型與超參數，在 Breast UltraSound 的資料集上就顯得有些吃力了，即使 train accuracy 也能夠達到全對的表現，但也可以發現此模型對於此資料集會有 overfitting 的問題，在 test data 上其分類（良性、惡性、無）的正確率只有 0.859，而分辨有無腫瘤的正確率，則是能夠到 0.904。而查看此資料集後，可以明顯感受到此資料集的影像較 Brain MRI 的影像複雜許多，因此在此資料集的表現較差也是在預期之內。

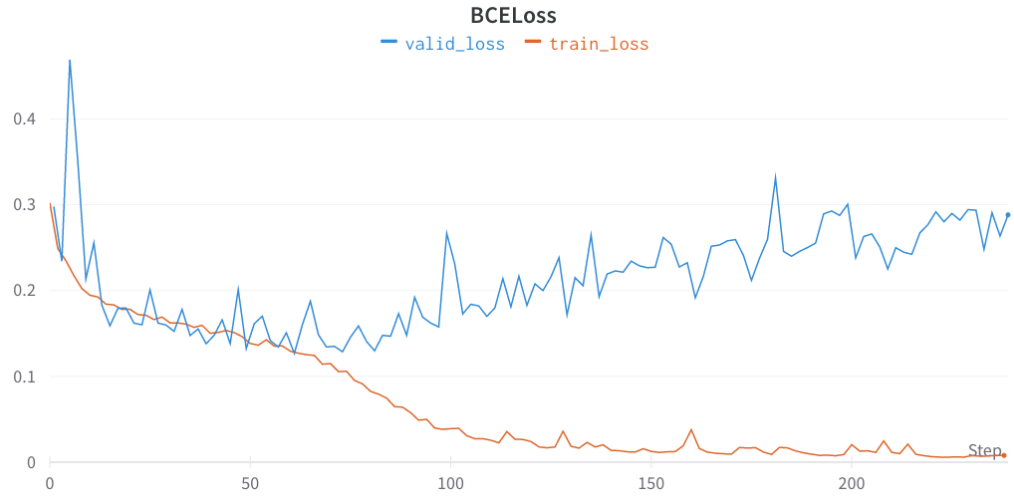
下表為同一模型在兩 dataset 之最後的表現。

	Train accuracy	Test accuracy	Test detect accuracy
Brain MRI	1.0	1.0	1.0
Breast UltraSound	1.0	0.8526	0.904

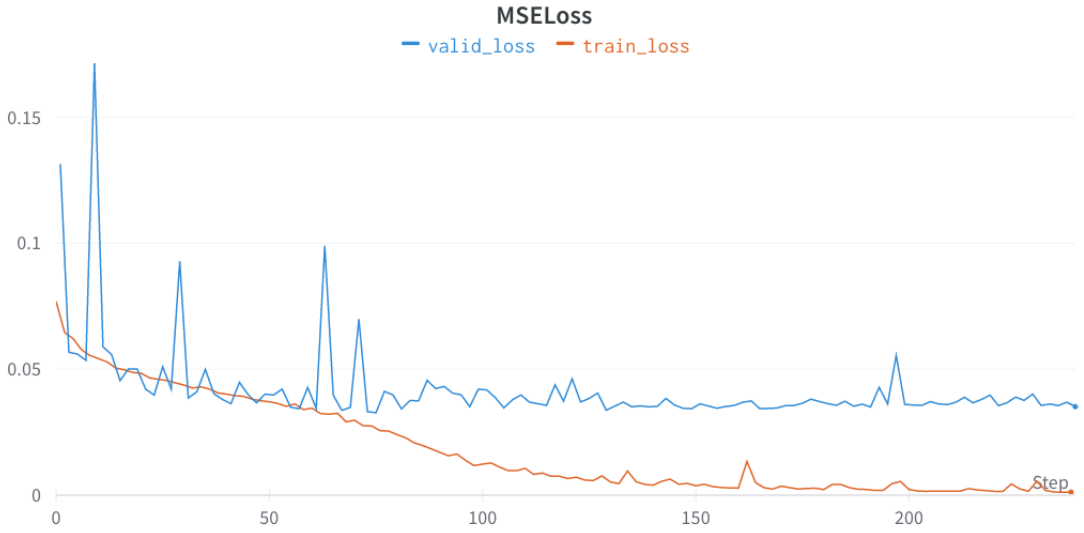
DLMI HW report

第二部分：

首先是 BCELoss 的部分，可以明顯發現 BCELoss 可能較不適合作為 loss function 來進行訓練，雖然 training loss 可以隨著訓練的推進慢慢降低，但是 valid 的部分卻沒有訓練起來的感覺，valid loss 最低的時刻是訓練的前期，猜想有可能是因為對於同樣的輸入 BCELoss 考慮的是 pixel wise 的 0 與 1 的機率分佈，若有很類似的輸入但輸出卻不一致的話，容易使訓練的目標不一致，進而導致模型並沒有真正的學到東西，而只是將看過的資料記憶下來而已。



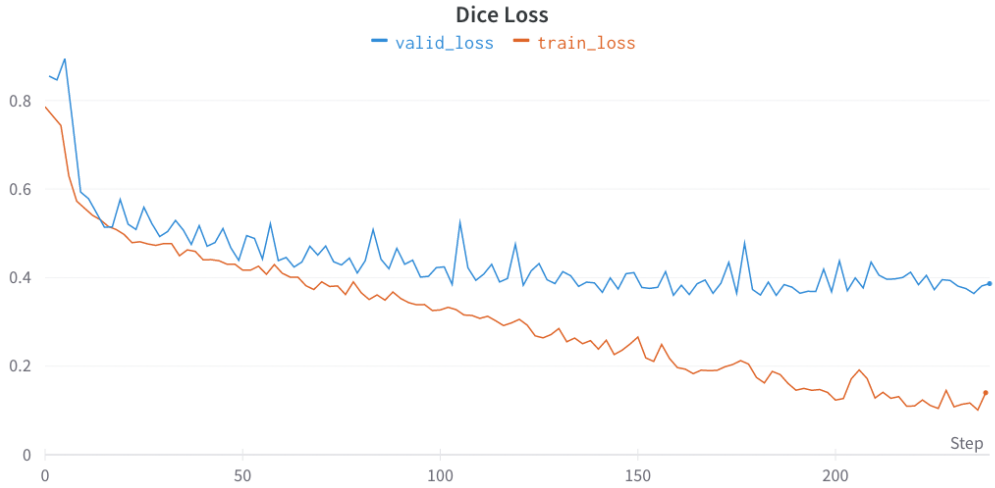
接著是 MSELoss 的部分，可以明顯發現其 learning curve 就比 BCELoss 較好，雖然也有明顯的 overfitting 問題，但是到訓練後段的的 loss 並沒有升高回來。但是一樣會遇到與 BCELoss 一樣的問題就是輸入一樣但正確的 target 可能會不一樣，在某些情況看到的 pixel 一樣但是 target 的 mask 不一致，導致模型的學習方向不一致。



最後是 Dice Loss 的部分，可以明顯發現雖然還是有 overfitting 但此 criterion 的 overfitting 的情形已經比 MSELoss 與 BCELoss 減輕許多，valid loss 有逐漸降低的趨勢。且 Dice Loss 是從整體的表現來算 loss，而不是單純的 pixel

DLMI HW report

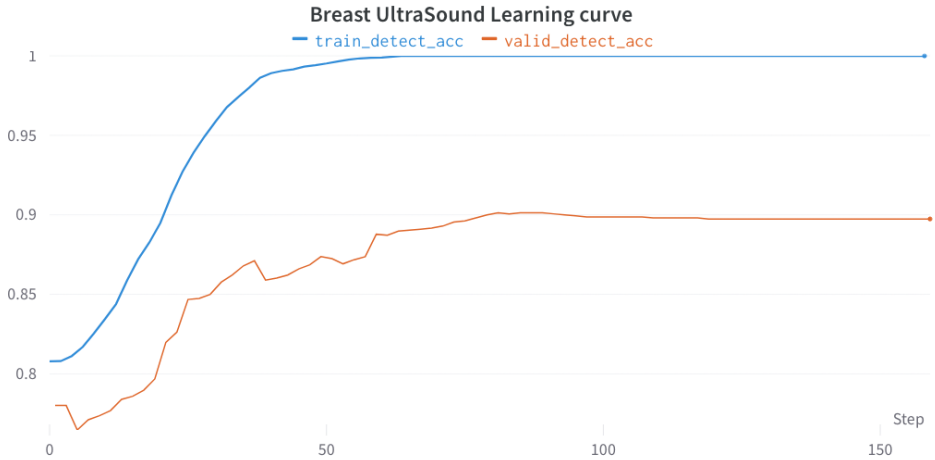
wise 的比較，在評價影像時，有較高的視野來決定預測的 mask 好壞。需要注意的是，這裡的 Dice Loss 為了符合 loss 降低的設計而改成 1-Dice Loss，且添加了 L1 norm 的 smooth，可某種程度上的幫助模型學習。



下表示三種 Loss function 所最後產出的 mask 與正確 mask 所算出的 mIOU。

Loss function	Train	Test
BCELoss	0.54753	0.48947
MSELoss	0.65306	0.56055
Dice Loss	0.71729	0.54007

第三部分：

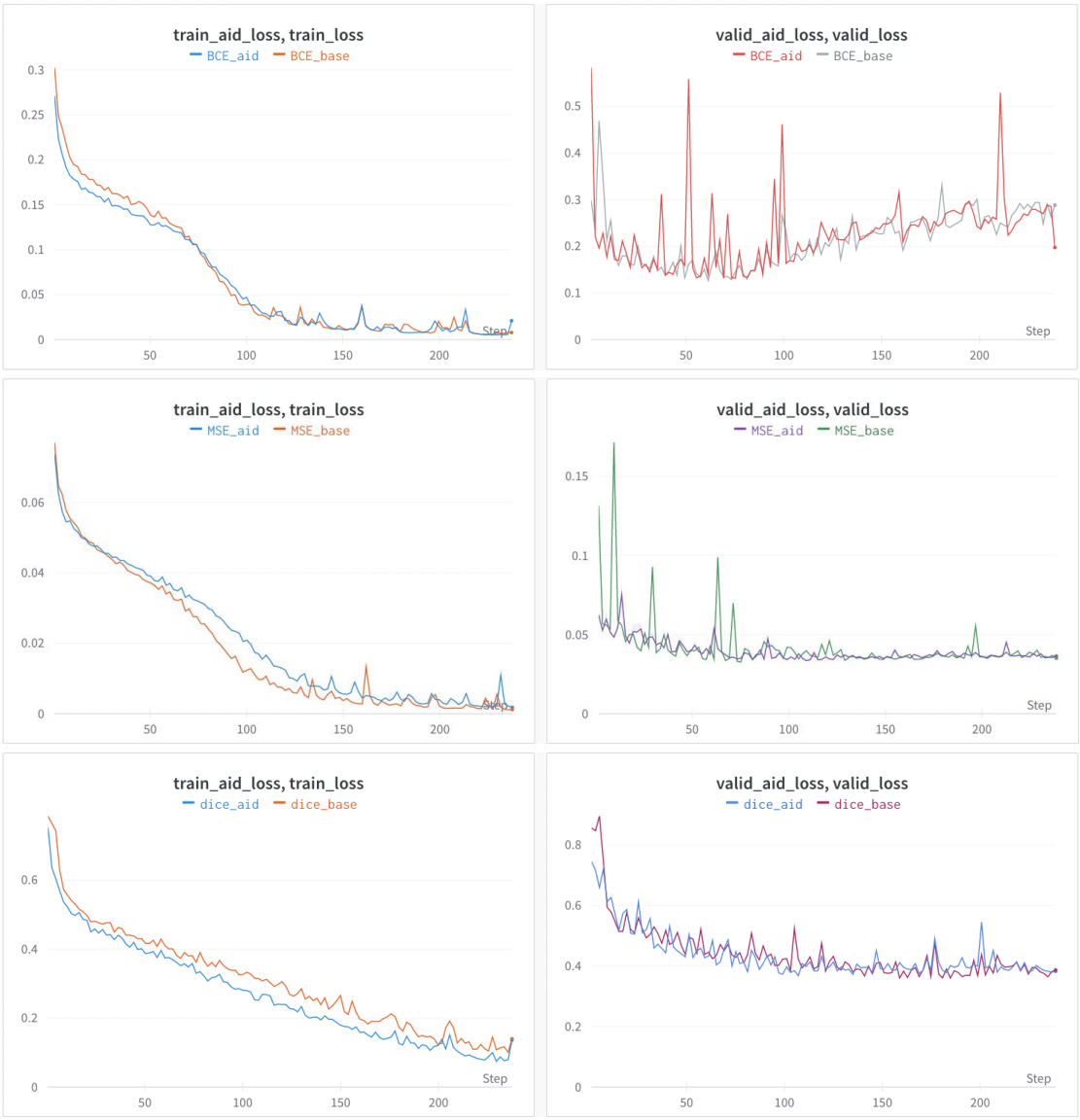


首先我們先使用第一部分訓練好的模型預測一張影像內是否含有腫瘤，而在第一部分的時候可以查看到對於此二分類問題的準確率可以達到九成以上的正確率。將此模型的預測一起傳入第二部分的模型，當預測為沒有腫瘤的時候，將其原本預測的 mask 改成空的 mask，如此一來，即使原本 segmentation 有預測出腫瘤，也會因為第一部分模型的關係，而直接移除掉，如此一來這種資料的 gradient descent 時不會計算到他的 loss，也就是 segmentation 的 model 不需要去考慮這種資料，也就有更高的可能性不會 overfitting 在這種資料上，

DLMI HW report

以得到更佳的表現。

下圖分別為使用 BCELoss、Dice Loss、MSELoss 作為 loss function 的 learning curve。Train_aid_loss 代表的是有使用第一部分模型的預測訓練出來的分數，Train_loss 則是與第二部分的實驗一樣。可以明顯發現在 BCELoss 與 Dice Loss，有搭配第一部分分類模型的 Train_aid_loss 都較 Train_loss 來得低，而 valid_loss 的部分則是不管哪個模型的有使用分類模型的表現都較沒有使用來的更好。



最後將全部的模型都生出其對 test data 的預測，使用 mIOU 進行模型表現的評估，可以發現不管哪個 loss function、不管是 train data 還是 test data，有使用分類模型的表現都明顯較佳，因此此方法應該對於 segmentation 的任務的確有所幫助。

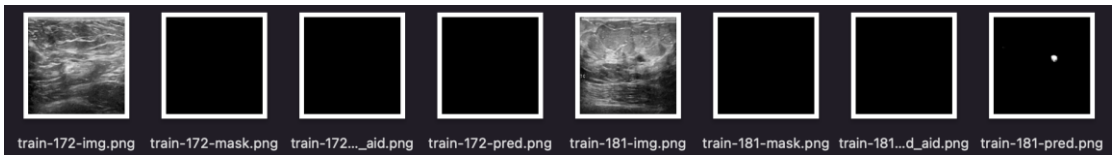
Loss function	Train	Test	Train_aid	Test_aid
BCELoss	0.54753	0.48947	0.62097	0.51996
MSELoss	0.65306	0.56055	0.77512	0.56304
Dice Loss	0.71729	0.54007	0.7584	0.57291

網媒所碩一 R10944028 蔡易儒

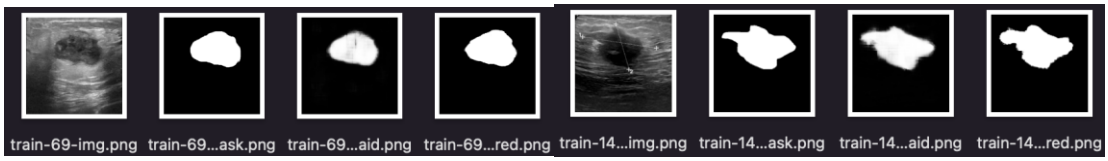
DLMI HW report

最後簡單的看一下實際產出的影像，為了避免佔用過多版面，這裡只放了使用 Dice Loss 的版本。若觀察 Normal 的，則可以很明顯發現若有 classifier 輔助，在這個分類底下能夠很好的幫助他直接預測出一個乾淨的 mask

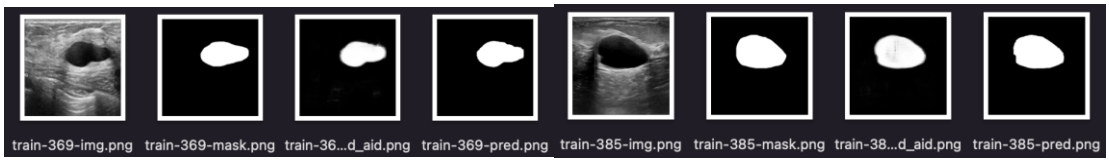
Train Normal:



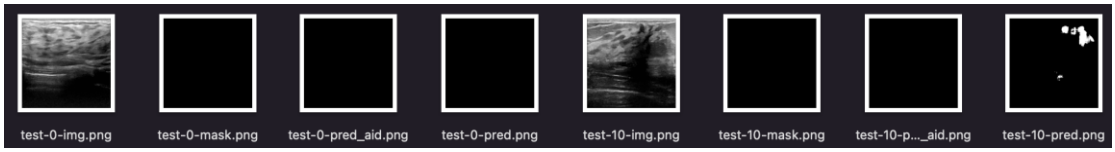
Train Malignant:



Train Benign:

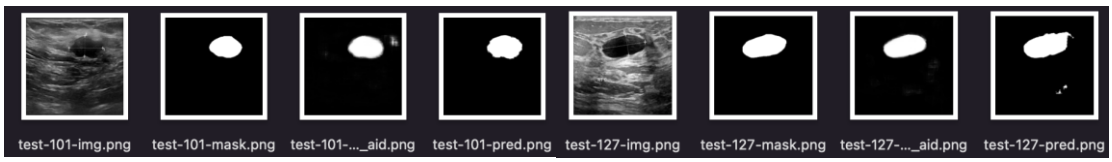


Test Normal:



Test Malignant:

Test Benign:

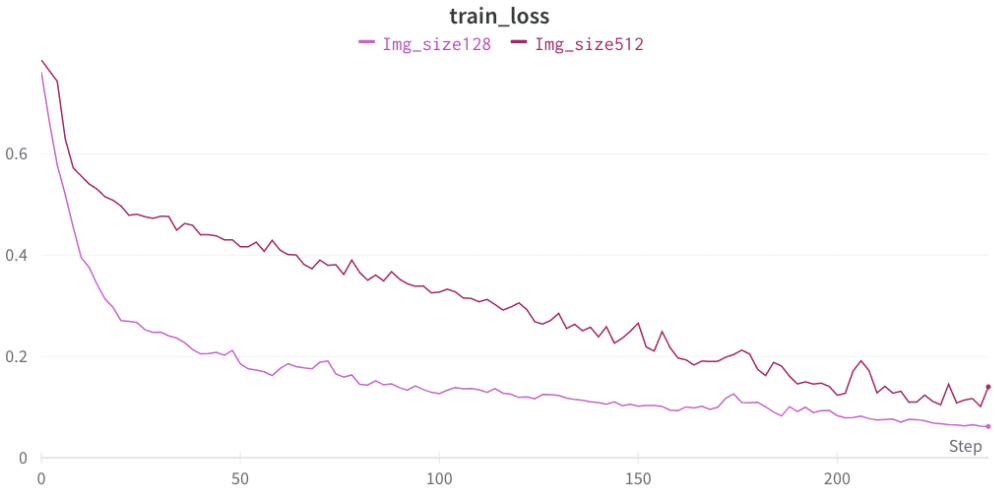


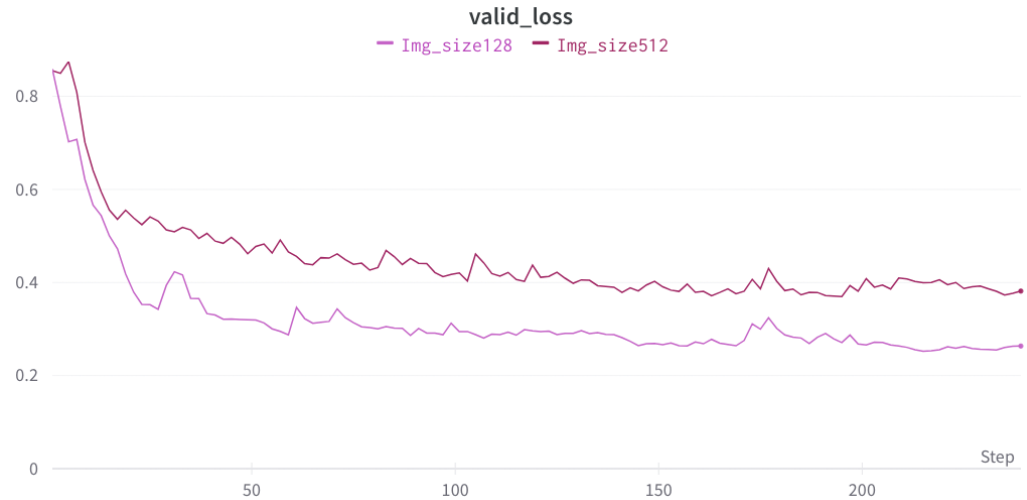
DLMI HW report

5. Conclusion and future work

參閱了許多網路上的程式與訓練結果，本作業 Brain MRI 的成績已經是最高的情況了，而網路上大多看到的成績大多是 85~90%，而有些超過 90%的則是將測試資料有混入到訓練資料內，因此在分類上的分數，此作業的確有一些 improvement。而在 Breast UltraSound 上則是找不太到別人的分類分數，也比較少人在此 dataset 上做分類問題，但是 85%的分數對於三分類問題已經不算太低了，在二分類上更是能達到 90.3 的準確率。

而 Segmentation 的部分，雖然網路上有些人的 Dice Loss 分數能夠比我低很多，我也參考了他們的 code，發現許多人在計算 loss 都有一些明顯的錯誤，如 dice loss 一個 batch 計算完得到的分數已經是 batch 的平均分數，但有些程式仍會將其相加最後除以整個 dataset 的長度，如此一來他們得到的分數就會是原本正確分數的 1/batch size 倍，另外許多人都會將影像大小縮小以達到提高表現分數的目的，而我的影像仍維持在 512*512 的大小，而不同影像大小之間也不太能比較，因為影像大小會大大的影響到任務的難易度，小的影像在做 segmentation 時比較簡單。而我將影像大小調小後也可以發現整體的表現馬上提升了不少。





DLMI HW report

有了期中作業的經驗後，期末會參加肺腺癌病理切片影像之腫瘤氣道擴散偵測競賽，除了第三部分的方法外，未來想要再做 cropping，固定模型的大小改成在預處理時，若影像大小超過模型的輸入大小，則會將影像做切割的動作，最後預測完再拼接回來，根據以前的經驗，若使用此方法，可能可以再提高模型的表現。

6. Reference

[1] Brain MRI Images for Brain Tumor Detection (https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection)

[2] Breast Ultrasound Images Dataset (https://www.kaggle.com/datasets/aryashah2k/breast-ultrasound-images-dataset)

[3] Pytorch-UNet-github (https://github.com/milesial/Pytorch-UNet)

[4] U-Net: Convolutional Networks for Biomedical Image Segmentation (https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/)

[5] 肺腺癌病理切片影像之腫瘤氣道擴散偵測競賽 (https://tbrain.trendmicro.com.tw/Competitions/Details/22)