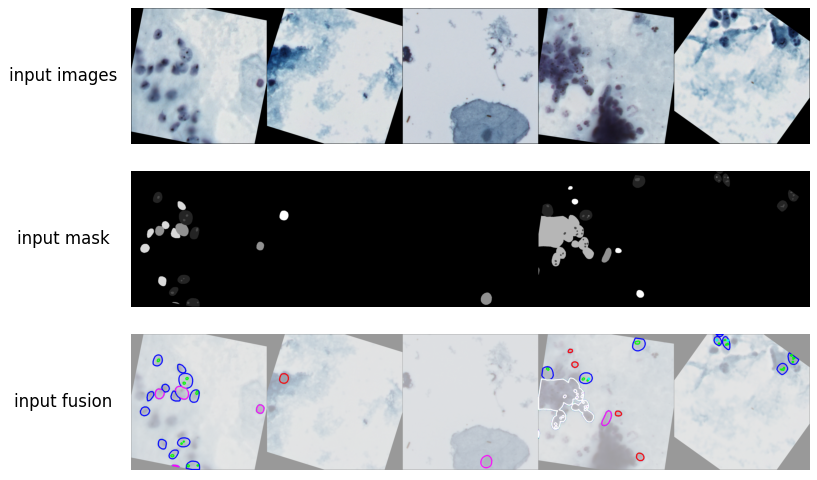
DataLoader :

Dataloader的部分，我針對A~O的class每個class都分成7:3，分別加進trainset, testset，確保dataset裡每個class平均分配，再對trainset, testset做shuffle確保每次training拿資料的順序不同，每個iter根據檔案名稱去取用每張IMG。此外還發現如果用os.listdir去取用檔案名稱的話會cpu沒辦法快速處理完，這樣會拖累GPU的使用效率。

Preprocess :

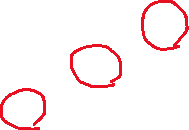
資料前處理的部分，我使用了albumentations函式庫，他可以同時對兩個不同的data做”一模一樣”的transform，這樣可以使IMG跟MASK做完transform不會切在不同的位置。

其中又發現如果今天圖片做完前處理後，若batch裡剛好取到很多個mask裡面沒有感興趣的地方(EX:全黑)的圖片，在訓練的時候會造成前後batch出來的結果差很多，導致\最後loss變得異常，所以在做transform的時候會檢查MASK裡是否有感興趣位置，若沒有的話會重新transform直到MASK裡有感興趣位置。模型架構(一)

UNet :

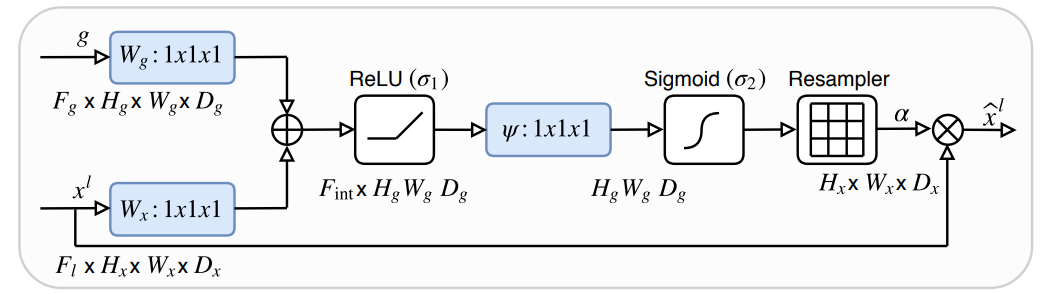
一張含有 文字 的圖片

自動產生的描述 模型架構的部分，我使用了pytorch-UNet的模型，並參考了attention-Unet的架構(如圖二)自己改良成這次的model，在每一次concate前加上了mask-Attention-gate，這大大的提升了模型的performance。



模型架構(二)





attention Gate (圖三)

Attention gate:

DownSmple layer()，Upsample layer ()，與分別會先各自經過一個CNN(維度,img size不變)，兩者相加之後原本MASK有值的地方會變得更大，經過一個Relu layer後會過濾掉負值，經過一個CNN和一個sigmoid layer可以得到一個針對每個pixel的attention score，最後再將這個attention score乘回原本的mask，這個動作相當於利用attention score去加強我們所感興趣的部分。實際上的implement如圖四。

一張含有 文字 的圖片

自動產生的描述

Attenrion gate implement (圖四)

Show Batch

圖片show的部分參考了<https://github.com/tumeng0302/lab5_utlities.git>的程式碼

Loss

Loss的部分除了用cross entropy，我還加了dice loss，加了這個之後loss下降變慢許多，performance也變好許多。

一張含有 文字 的圖片

自動產生的描述

DICE loss (圖五)

Result:

由於電腦壞掉所以只有少少的訓練epoch和result，來不及做matrix比較。

圖片裡gt vs pred如果pred的結果跟gt有重疊(預測正確)就顯示白色，若沒有重疊(預測錯誤)就顯示紅色。

最後我只跑了50個epoch，其中發現一開始avg loss降得很快(第一個epoch就跑到0.03左右)，後來都卡在這附近且降得很緩慢，但是performance卻有越來越好的趨勢，後來我認為是avg loss是看所有的batch，但每一個batch算出的loss還是有大有小，模型還是會根據大的loss去更新，所以還是會越來越好。

其中也發現，訓練到越後面模型才開始比較能學會對class分類正確，訓練不多的情況下他只會學到segment。

