1.Introduction

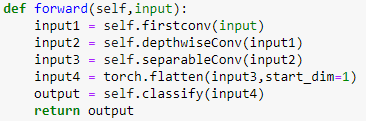
這次的作業就是經由給定data，透過EEEGNET和DeepConvNet來做binary classification。而這次作業主要的目標就是希望學生們能讀懂網路的規格表，實作EEEGNET和DeepConvNet，並且學習如何調hyper parameter。

2.Experiment setup

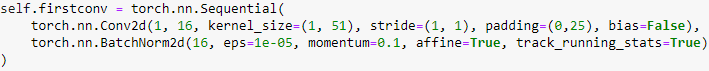
1. The detail of your model

EEGNet:

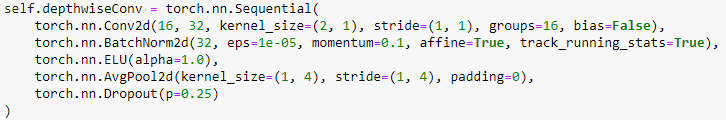
其實就照著reference刻，總共可分為firstconv，depthwiseConv，separableConv，flatten和classify共五層。



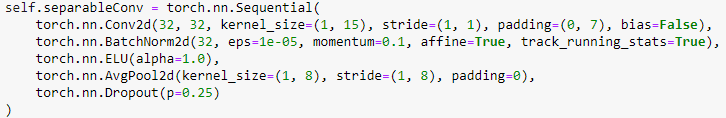
firstconv這層就是做一次conv和batchNorm。



depthwiseConv這層就是做一次conv，batchNorm2d，ELU，AvgPool和Dropout。



separableConv這層基本上和depthwiseConv差不多，只是Conv和AvgPool的大小不太一樣。



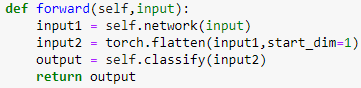
然後因為此時輸出還是4dim矩陣，而為了要binary classification，所以我們要透過flatten將4dim矩陣轉成2dim。

最後再做classification，也就單純將channel數從736變成2。

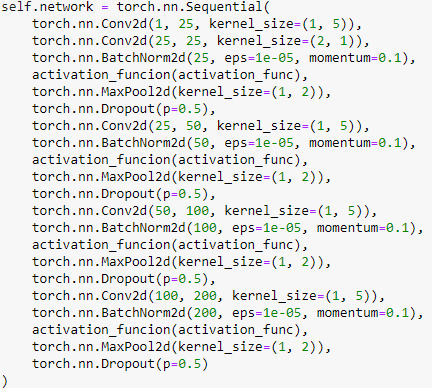


DeepConvNet:

主要也是照抄，分成network，flatten和classify。



network的部分，一開始先疊一個conv，後面再疊四次conv->batchnorm->activation\_func->maxpool->dropout，然後conv的out\_channel也是用後面越大。

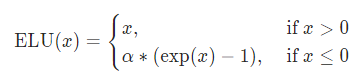


然後因為此時輸出還是4dim矩陣，而為了要binary classification，所以我們要透過flatten將4dim矩陣轉成2dim。

最後再做classification，也就單純將channel數從8600變成2。

1. Explain the activation function (ReLU, Leaky ReLU, ELU)

ELU:



他的偏微分是

if x>=0:

1

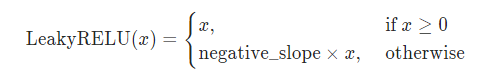
if x<0:



理論上，ELU具有RELU的所有優點，而且不會有dead\_relu的問題。

缺點是計算成本有一點高。

Leaky ReLU:



偏微分則是:

if x>=0:

1

else:

negative\_slope

理論上，Leaky ReLU具有RELU的所有優點，而且不會有dead\_relu的問題。

ReLU:



偏微分則是:

if x>=0:

1

else:

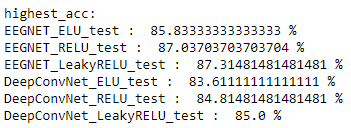
0

優點是計算成本小，不存在梯度飽合問題。

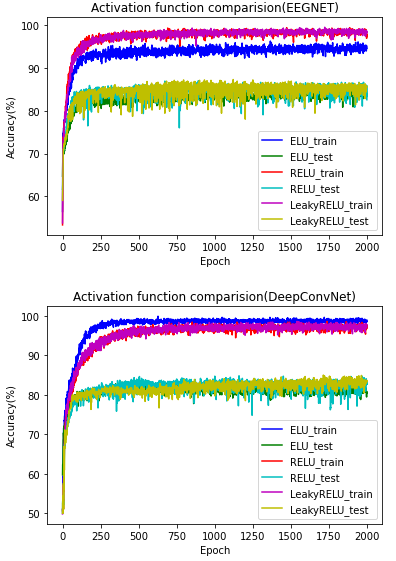
缺點是可能發生dead\_relu，也就是當x<0，就完全不更新。

3.Experimental results

1. The highest testing accuracy



1. Comparison figures



4.Discussion

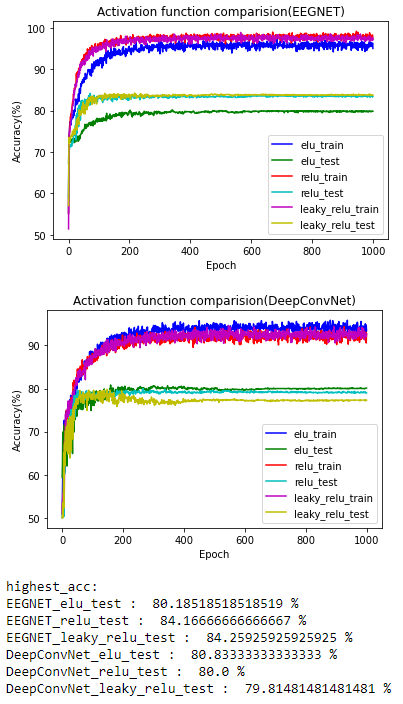
1. Anything you want to share

1.

從圖表(3)可以觀察到，DeepConvNet的test\_acc一直比EEGNET的test\_acc還來的差，這是因為DeepConvNet比較適合用在dataset較大的情況。

值得注意的是，train\_acc和test\_acc其實也不一定有正相關，像ELU的train\_acc整體來說應該是六種裡面最差的，但是他的test\_acc在六種裡面是第三好的。

2.我曾經試過加入scheduler，發現其實sheduler對於結果來說影響並不大，如下圖，我對助教給的參數們加入scheduler=stepLR(0.99)，結果和沒加scheduler差不多，所以最後我就拿掉了。



3.寫這份作業時，我有和其他人討論hyperparameter的設置，後來發現就算程式碼和hyperparameter都相同，使用不同gpu好像也會對test\_acc影響到2~3%，所以hyperparameter的設置我覺得好像沒這麼絕對。像我為了改良test\_rate的acc，我的batch\_size用到了400，雖然一般人都用batch\_size=32或64，然後跑出了87%，但用我RTX2060跑，最高好像只能到84%。

至於其他hyperparameter，我設置epoch=2000，learning\_rate=1e^-3，optimizer=adam()，weight\_decay=0.01，這個過程純粹通靈。