Lab 3 Report
309551064 張凱翔

1. Introduction
   實做 EEGNet 和 DeepConvNet 的網路架構來做 BCI dataset 的二元分類問題(左手或右手)。使用 3 種不同的 activation function(ELU, ReLU, Leaky ReLU)來觀察對結果的影響，並調整 model 使 accuracy>=87%。
2. Experiment set up
   A. The detail of your model
      - EEGNet

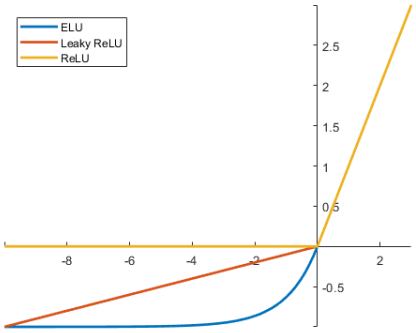
```
EEGNet(
  (firstconv): Sequential(
    (0): Conv2d(1, 16, kernel_size=(1, 51), stride=(1, 1), padding=(0, 25), bias=False)
    (1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (deptwiseConv): Sequential(
    (0): Conv2d(16, 32, kernel_size=(2, 1), stride=(1, 1), groups=16, bias=False)
    (1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.01)
    (3): AvgPool2d(kernel_size=(1, 4), stride=(1, 4), padding=0)
    (4): Dropout(p=0.25, inplace=False)
  )
  (separableConv): Sequential(
    (0): Conv2d(32, 32, kernel_size=(1, 15), stride=(1, 1), padding=(0, 7), bias=False)
    (1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.01)
    (3): AvgPool2d(kernel_size=(1, 8), stride=(1, 8), padding=0)
    (4): Dropout(p=0.25, inplace=False)
  )
  (classify): Sequential(
    (0): Linear(in_features=736, out_features=2, bias=True)
  )
)
```

      - DeepConvNet

```
DeepConvNet(
  (conv0): Conv2d(1, 25, kernel_size=(1, 5), stride=(1, 1))
  (conv1): Sequential(
    (0): Conv2d(25, 25, kernel_size=(2, 1), stride=(1, 1))
    (1): BatchNorm2d(25, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.01)
    (3): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (4): Dropout(p=0.5, inplace=False)
  )
  (conv2): Sequential(
    (0): Conv2d(25, 50, kernel_size=(1, 5), stride=(1, 1))
    (1): BatchNorm2d(50, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.01)
    (3): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (4): Dropout(p=0.5, inplace=False)
  )
  (conv3): Sequential(
    (0): Conv2d(50, 100, kernel_size=(1, 5), stride=(1, 1))
    (1): BatchNorm2d(100, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.01)
    (3): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (4): Dropout(p=0.5, inplace=False)
  )
  (conv4): Sequential(
    (0): Conv2d(100, 200, kernel_size=(1, 5), stride=(1, 1))
    (1): BatchNorm2d(200, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): LeakyReLU(negative_slope=0.01)
    (3): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (4): Dropout(p=0.5, inplace=False)
  )
  (classify): Linear(in_features=8600, out_features=2, bias=True)
)
```

      model 的架構是依照 spec 上提供的架構做出來的
      使用的 hyper parameter 是 spec 上的，也沒有對 model 進行更改，只是在 train 的時候加入 scheduler 對 learning rate 進行調整，結果便可以達到 87%。

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



- ReLU

$$\mathrm{ReLU}(x) = (x)^+ = \max(0, x)$$

在大於 0 的部份 x=y，小於等於 0 的部份 y=0。

- Leaky ReLU

$$\mathrm{LeakyRELU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \text{negative\_slope} \times x, & \text{otherwise} \end{cases}$$

Leaky ReLU 和 ReLU 的差別在於小於 0 的部份，LeakyReLU 會將值定為 x 乘上一個小於 1 的數字。

- ELU

$$\mathrm{ELU}(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha * (\exp(x) - 1), & \text{if } x \leq 0 \end{cases}$$

ELU 在 x>0 時跟 ReLU 一樣 x=y，小於等於 0 的部份則是 exp 後-1 乘上一個 alpha 值，成為一條趨近於-alpha 的曲線。

在這次作業中使用的 activation function 都是使用 pytorch 的 default 值。

3. Experimental results

A. The highest testing accuracy

- Screenshot with two models

EEGNet:

```
elu 0.8361111111111111
relu 0.8722222222222222
leakyrelu 0.8814814814814815
```

DeepConvNet:

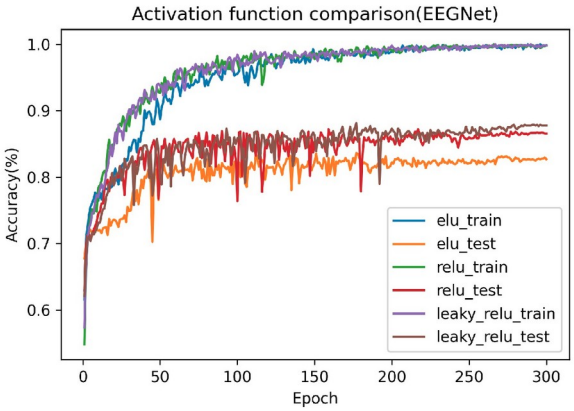
```
elu 0.8138888888888889
relu 0.8194444444444444
leakyrelu 0.8305555555555556
```

	ELU	ReLU	LeakyReLU
EEGNet	83.61%	**87.22%**	**88.15%**
DeepConvNet	81.39%	81.94%	83.06%

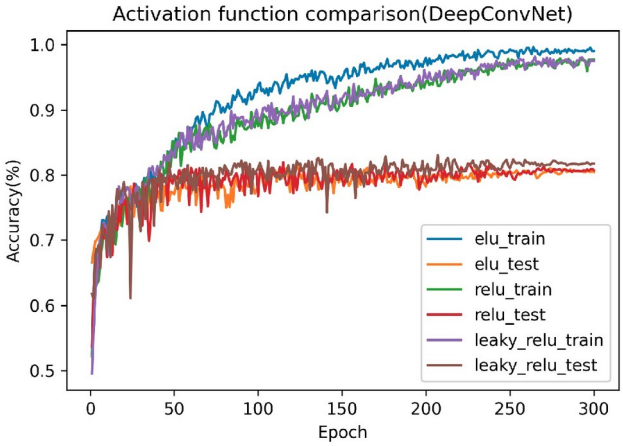
- anything you want to present

可以看到 EEGNet 的表現要比 DeepConvNet 的好很多，我認為可能的原因為 DeepConvNet 的層數過多，而導致 overfit，所以在 test 的結果上較不好。

B. Comparison figures
   - EEGNet



   - DeepConvNet



4. Discussion
   A. Anything you want to share

   1. 因為上學期有接觸過深度學習，對於建 model 和使用 Dataset 和 Dataloader 都有初步的理解，所以比起第一次大量數學的作業能更快的上手，所花的時間也較少。
   2. 這次的 model 相較於第四次作業的 resnet 是比較簡單一些，所以可以簡單的調整 learning rate 便可以達到要求的準確度，相信第四次作業需要更動較多的架構。