HW3 Report EEG classification

tags: `DL and Practice`

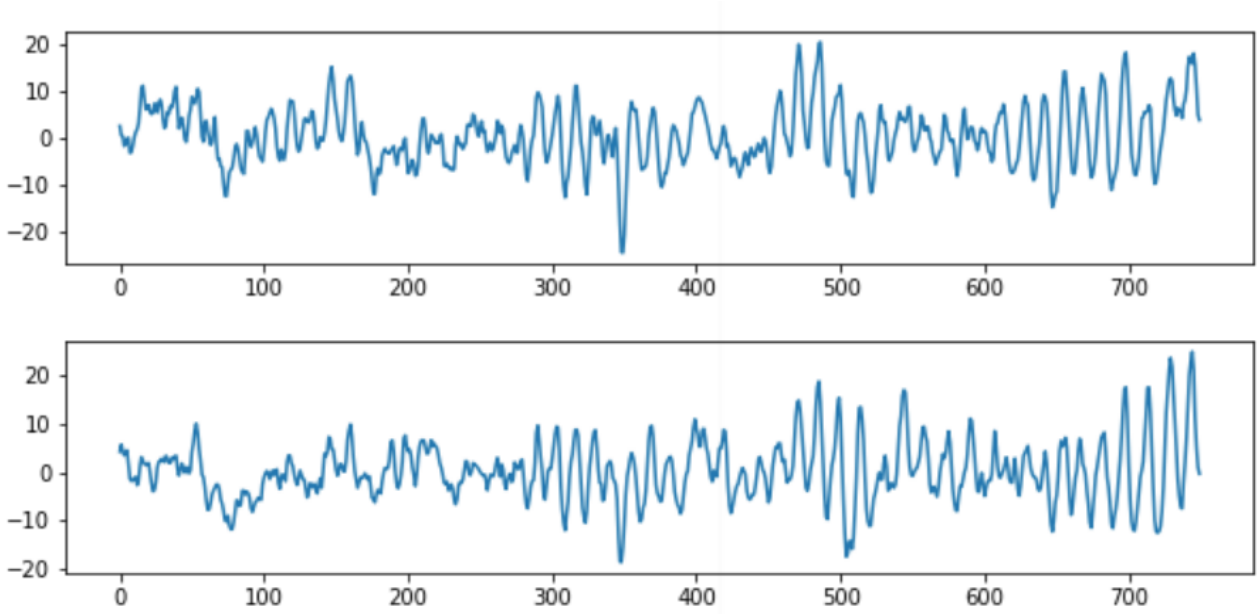
電控碩一 黃柏叡 309512074

Report

Introduction

使用EEGNet與DeepConvNet處理分類問題，訓練資料的shape為(C=1, H=2, W=750)，training資料及testing資料各有1080筆 這次作業最主要的目標:

- 利用三種activation functions找出兩個model中最高的accuracy
- 將accuracy的趨向圖像化



Experimental Setup

A. 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)
  )
  (depthwiseConv): 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): ELU(alpha=1.0)
    (3): AvgPool2d(kernel_size=(1, 4), stride=(1, 4), padding=0)
    (4): Dropout(p=0.25)
  )
  (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): ELU(alpha=1.0)
    (3): AvgPool2d(kernel_size=(1, 8), stride=(1, 8), padding=0)
    (4): Dropout(p=0.25)
  )
  (classify): Sequential(
    (0): Linear(in_features=736, out_features=2, bias=True)
  )
)
```

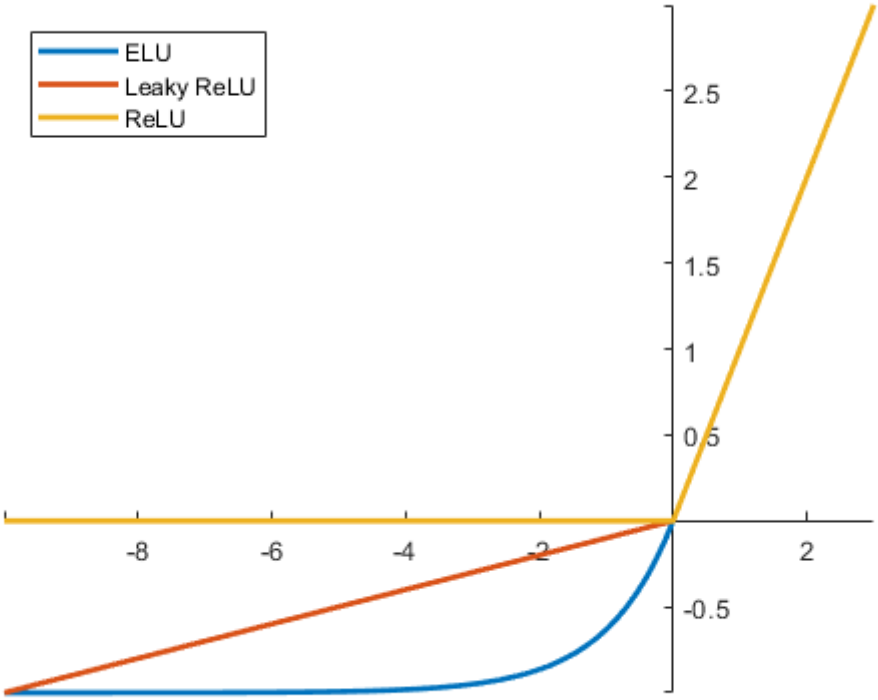
使用depthwise-separable convolution, 是基於傳統convolution的輕量化版本，可以降低參數數量，提升訓練與evaluate的速度，並且不會影響accuracy的精準度

- DeepConvNet

Layer	# filters	size	# params	Activation	Options
Input		(C, T)			
Reshape		(1, C, T)			
Conv2D	25	(1, 5)	150	Linear	mode = valid, max norm = 2
Conv2D	25	(C, 1)	25 * 25 * C + 25	Linear	mode = valid, max norm = 2
BatchNorm			2 * 25		epsilon = 1e-05, momentum = 0.1
Activation				ELU	
MaxPool2D		(1, 2)			
Dropout					p = 0.5
Conv2D	50	(1, 5)	25 * 50 * C + 50	Linear	mode = valid, max norm = 2
BatchNorm			2 * 50		epsilon = 1e-05, momentum = 0.1
Activation				ELU	
MaxPool2D		(1, 2)			
Dropout					p = 0.5
Conv2D	100	(1, 5)	50 * 100 * C + 100	Linear	mode = valid, max norm = 2
BatchNorm			2 * 100		epsilon = 1e-05, momentum = 0.1
Activation				ELU	
MaxPool2D		(1, 2)			
Dropout					p = 0.5
Conv2D	200	(1, 5)	100 * 200 * C + 200	Linear	mode = valid, max norm = 2
BatchNorm			2 * 200		epsilon = 1e-05, momentum = 0.1
Activation				ELU	
MaxPool2D		(1, 2)			
Dropout					p = 0.5
Flatten					
Dense	N			softmax	max norm = 0.5

傳統的CNN架構 C->(CBAMD)->(CBAMD)->(CBAMD)->(CBAMD)->fully connected C:Conv2D B:BatchNorm A:Activation function P:MaxPooling D:Dropout

B. Explain the activation function



- ReLU

$$\text{ReLU} = \max(0, x)$$

- Leaky ReLU

$$f(x) = \max(0.01x, x)$$

- ELU

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha(e^x - 1), & \text{otherwise} \end{cases}$$

這三者的差異出現在input小於0時 在進行backpropagation時，LeakyReLU跟ELU就算input值小於0，仍然會有gradient output，然而此時ReLU梯度即為0。LeakyReLU跟ELU的公式設計就是為了解決當input小於0時，ReLU會發生的梯度消失問題

Experimental results

A. The highest testing accuracy

- Screenshot with two models
  - EEGNet(lr=0.0005)

```
epoch290  acc_test:82.87%
epoch300  loss:0.0037  acc:90.556%
epoch300  acc_test:84.17%
epoch310  loss:0.0036  acc:89.444%
epoch310  acc_test:83.98%
epoch320  loss:0.0039  acc:90.741%
epoch320  acc_test:83.89%
epoch330  loss:0.0038  acc:90.093%
epoch330  acc_test:84.91%
epoch340  loss:0.0039  acc:89.630%
epoch340  acc_test:83.61%
epoch350  loss:0.0036  acc:91.574%
epoch350  acc_test:83.70%
ReLU  best accuracy EEG:  86.66666666666667 %
LeakyReLU  best accuracy EEG:  87.5 %
ELU  best accuracy EEG:  84.9074074074074 %
```

  - DeepConvNet(lr=0.001)

```
epoch290  acc_test:79.07%
epoch300  loss:0.0017  acc:96.852%
epoch300  acc_test:79.63%
epoch310  loss:0.0017  acc:96.481%
epoch310  acc_test:80.46%
epoch320  loss:0.0021  acc:95.000%
epoch320  acc_test:78.33%
epoch330  loss:0.0018  acc:96.667%
epoch330  acc_test:77.22%
epoch340  loss:0.0017  acc:96.574%
epoch340  acc_test:80.37%
epoch350  loss:0.0018  acc:96.296%
epoch350  acc_test:78.24%
ReLU  best accuracy Deep:  85.37037037037037 %
LeakyReLU  best accuracy Deep:  84.16666666666667 %
ELU  best accuracy Deep:  81.57407407407408 %
```

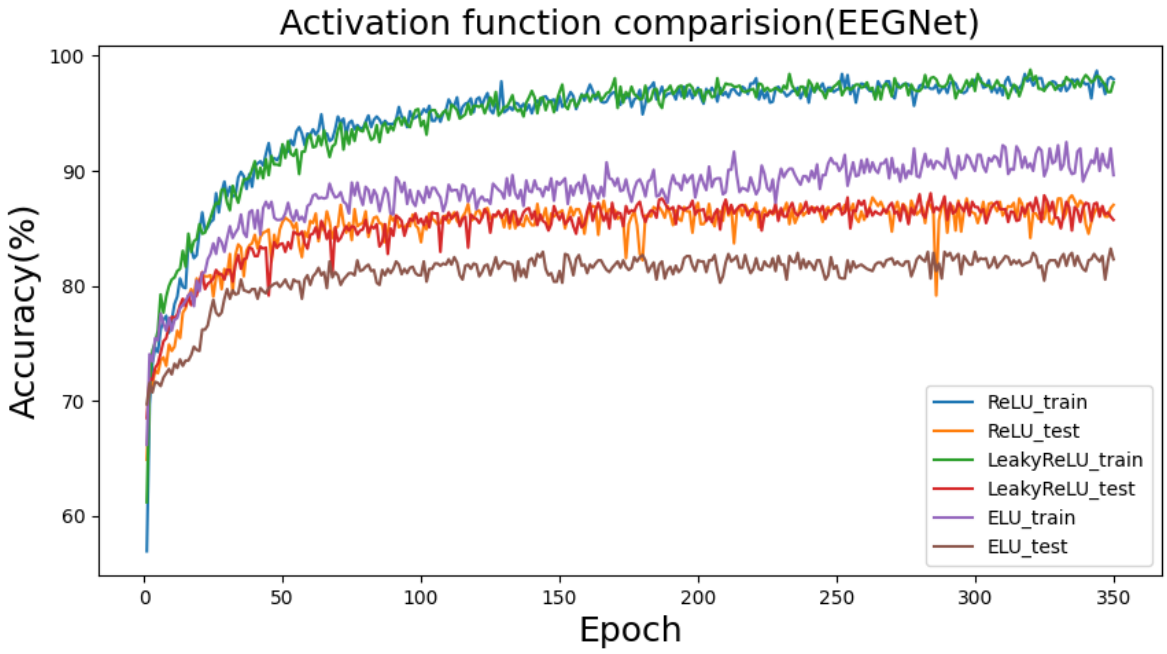
	ReLU	**Leaky ReLU**	**ELU**
EEGNet	86.7%	87.5%	84.9%
DeepConvNet	85.37%	84.17%	81.57%

- anything you want to present
  - epochs: 350
  - optimizer: Adam
  - criterion: CrossEntropy
  - batch size: 64
  - learning rate for EEGNet: 0.0005

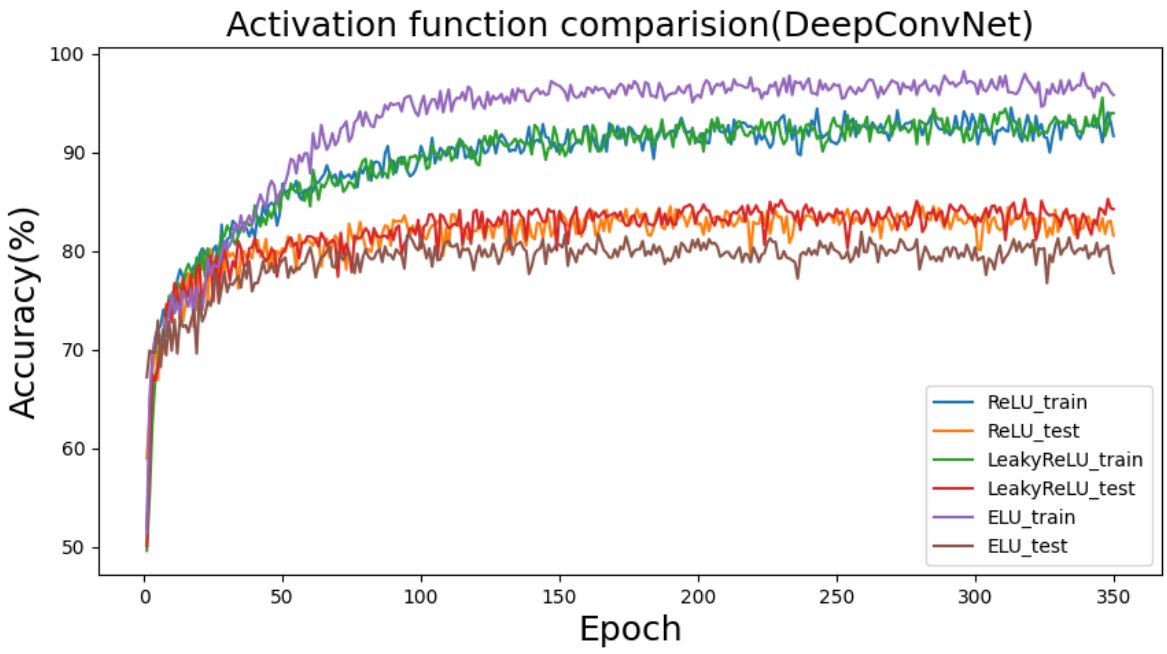
- learning rate for DeepConvNet: 0.001

B. Comparison figures

- EEGNet(lr=0.0005)



- DeepConvNet(lr=0.001)



Discussion

Anything you want to share

- Dropout 可以解決overfitting problem。此外，model不能學習到training data的所有特徵(因為dropout)，model學習的是資料的pattern

- 因為一開始讀近來的data是numpy格式，所以需要先用Tensordataset()轉成tensor格式在放入DataLoader
- data以及model都需要加上 .to(device)才能放入gpu加速運算，tensor可以在gpu跟cpu上運算，而numpy只能在cpu上運算

```
device=torch.device('cuda' if torch.cuda.is_available() else 'cpu')
```