Lab4-1 - EEG classification Report

0816124 Hao-Chen Lin

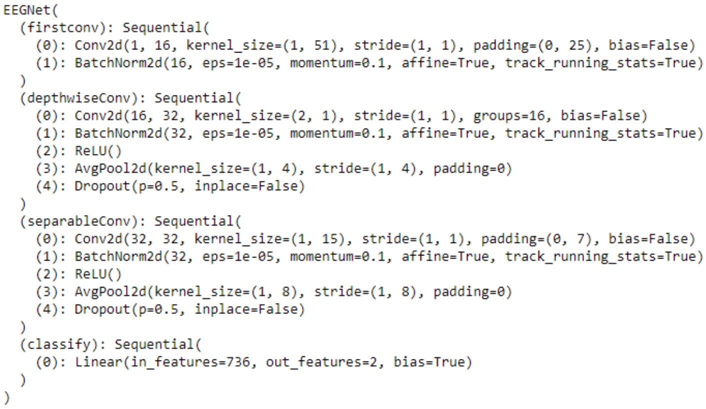
April 8, 2022

1 Introduction

In this lab, we need to implement simple EEG classification models which are EEGNet, DeepConvNet with BCI competition dataset. Additionally, we need to try different kinds of activation function including ReLU, Leaky ReLU, ELU.

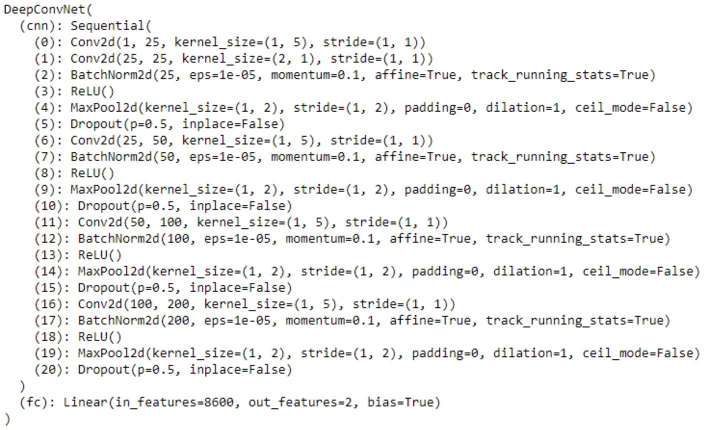
2 Experiment set up

2.1 The detail of your model



```
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): ReLU()
    (3): AvgPool2d(kernel_size=(1, 4), stride=(1, 4), padding=0)
    (4): Dropout(p=0.5, 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): ReLU()
    (3): AvgPool2d(kernel_size=(1, 8), stride=(1, 8), padding=0)
    (4): Dropout(p=0.5, inplace=False)
  )
  (classify): Sequential(
    (0): Linear(in_features=736, out_features=2, bias=True)
  )
)
```

Figure 1: The detail of EEGNet.



```
DeepConvNet(
  (cnn): Sequential(
    (0): Conv2d(1, 25, kernel_size=(1, 5), stride=(1, 1))
    (1): Conv2d(25, 25, kernel_size=(2, 1), stride=(1, 1))
    (2): BatchNorm2d(25, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (3): ReLU()
    (4): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (5): Dropout(p=0.5, inplace=False)
    (6): Conv2d(25, 50, kernel_size=(1, 5), stride=(1, 1))
    (7): BatchNorm2d(50, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (8): ReLU()
    (9): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (10): Dropout(p=0.5, inplace=False)
    (11): Conv2d(50, 100, kernel_size=(1, 5), stride=(1, 1))
    (12): BatchNorm2d(100, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (13): ReLU()
    (14): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (15): Dropout(p=0.5, inplace=False)
    (16): Conv2d(100, 200, kernel_size=(1, 5), stride=(1, 1))
    (17): BatchNorm2d(200, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (18): ReLU()
    (19): MaxPool2d(kernel_size=(1, 2), stride=(1, 2), padding=0, dilation=1, ceil_mode=False)
    (20): Dropout(p=0.5, inplace=False)
  )
  (fc): Linear(in_features=8600, out_features=2, bias=True)
)
```

Figure 2: The detail of DeepConvNet.

2.2 Explain the activation function

2.2.1 ReLU

$$ReLU(x) = max(0, x)$$

2.2.2 Leaky ReLU

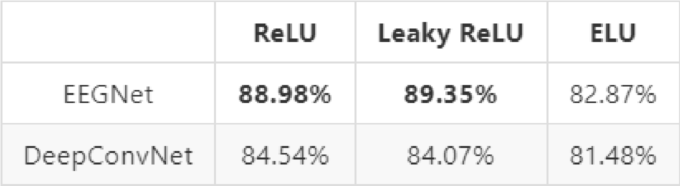
$$LeakyRELU(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \text{negative slope} \times x, & \text{otherwise} \end{cases}$$

2.2.3 ELU

$$ELU(x) = max(0, x) + min(0, \alpha * (exp(x) - 1))$$

2.3 Experimental results

2.3.1 The highest testing accuracy.



	ReLU	Leaky ReLU	ELU
EEGNet	**88.98%**	**89.35%**	82.87%
DeepConvNet	84.54%	84.07%	81.48%

Figure 3: The highest testing accuracy of all combinations.

2.3.2 Comparison figures

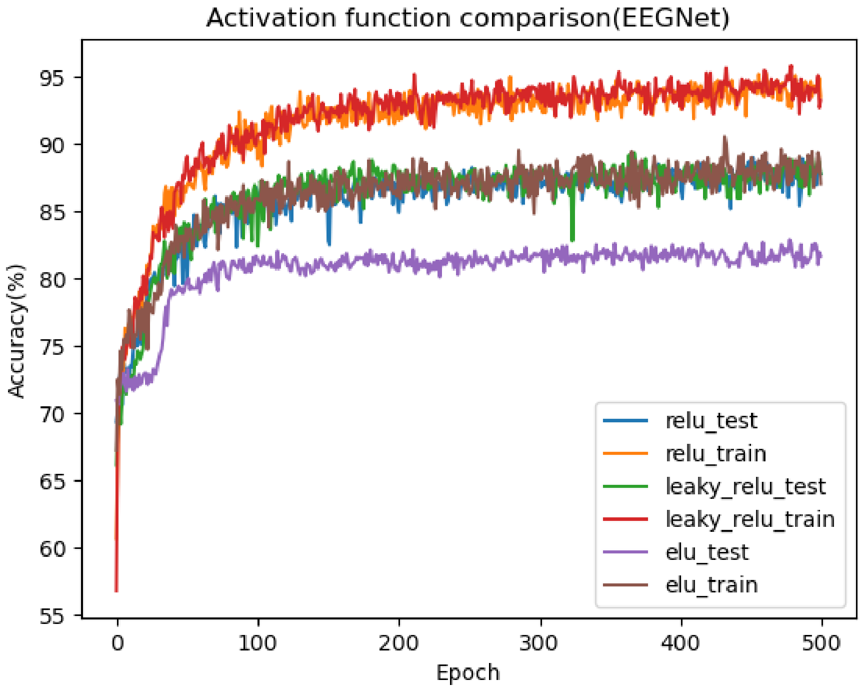


Figure 4: Comparison figure for EEGNet.

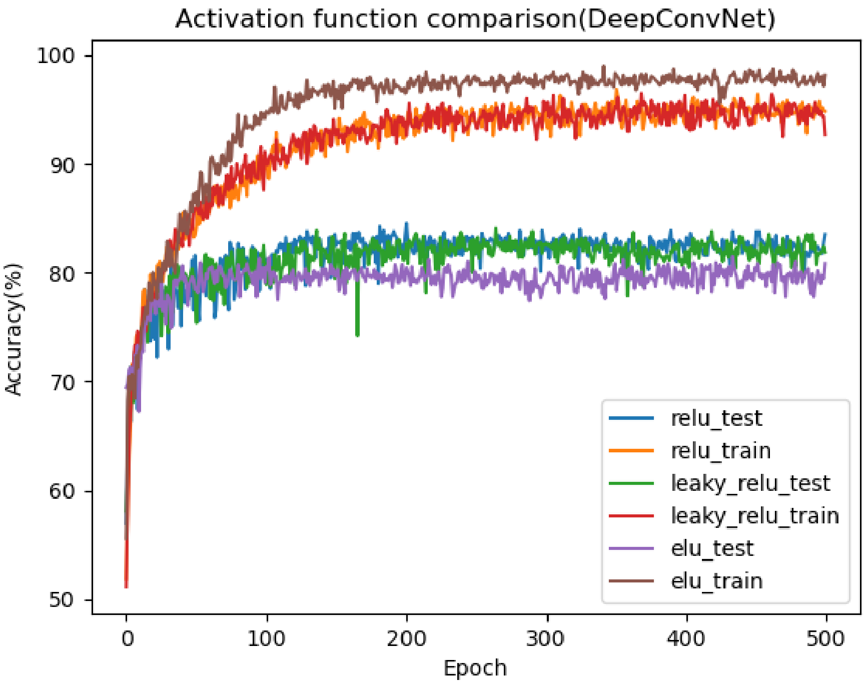


Figure 5: Comparison figure for DeepConvNet.

2.4 Discussion

I use grid search to find the best hyperparameters to this task, and i choose Adam as optimizer with lr=1e-3, weight decay=5e-3, and EEGNet with Leaky ReLU as its activation function has the highest testing accuracy in the end.