多工所碩一 楊奕儒
ID：312553038

DLP HW2

(1) Introduction

In this lab, I implemented EEG classification models which are EEGNet, DeepConvNet with BCI competition dataset. Additionally, I tried different kinds of activation function including ReLU, Leaky_ReLU and ELU.

(2) Experiment set up

(A)The detail of your model

EEGNet：

Base on the slide, I constructed the Neural network with self.firstconv, self.depthwiseConv and self.seperableConv. After the convolutional layers, the output is flattened and fed into a fully connected layer with 736 input features and 2 output features. The final output of the model is the result of this fully connected layer. Additionally, the code includes a train_and_test function, which is used to train and test the EEGNet model on provided data loaders.

DeepConvNet：

The model consists of multiple convolutional blocks, each containing a convolutional layer, batch normalization, an activation function. The forward function sequentially passes the input through each of the convolutional blocks, flattens the output, and then feeds it to the fully connected layer to obtain the final output of the model. With out.view(out.shape[0],-1) in forward function, we can define flatten the data.

(B) Explain the activation function

(a) Relu

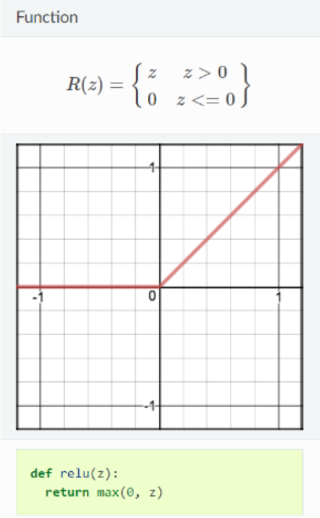
Fast, but when input data < 0, gradient will set to 0, may cause “dying ReLu problem” .

(b) LeakyReLU

It possesses the characteristics of ReLU but with a non-zero gradient when the input x is less than 0, which can address the "dying ReLU" problem.

(c)Elu

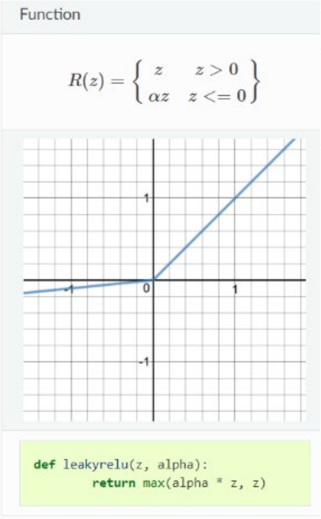
ELU is a function that tends to converge values to zero faster and produce more accurate results. But when input data > 0 , it may blow up.



Function

$$R(z) = \begin{Bmatrix} z & z > 0 \\ 0 & z <= 0 \end{Bmatrix}$$

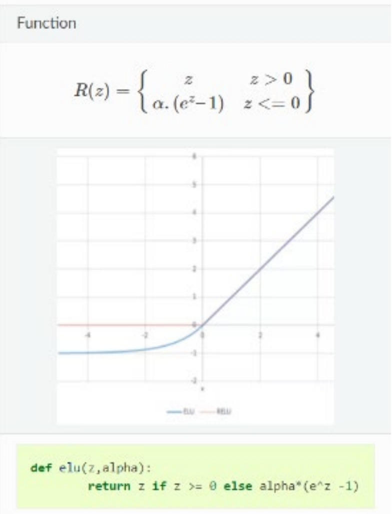
```
def relu(z):
  return max(0, z)
```



Function

$$R(z) = \begin{Bmatrix} z & z > 0 \\ \alpha z & z <= 0 \end{Bmatrix}$$

```
def leakyrelu(z, alpha):
        return max(alpha * z, z)
```



Function

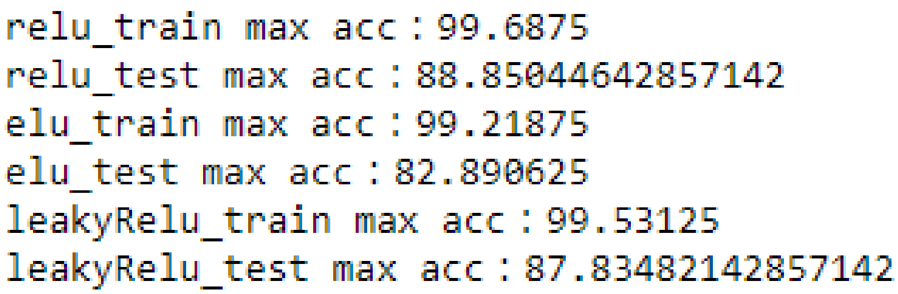
$$R(z) = \begin{Bmatrix} z & z > 0 \\ \alpha.(e^z-1) & z <= 0 \end{Bmatrix}$$

```
def elu(z,alpha):
        return z if z >= 0 else alpha*(e^z -1)
```

(3) Experimental results

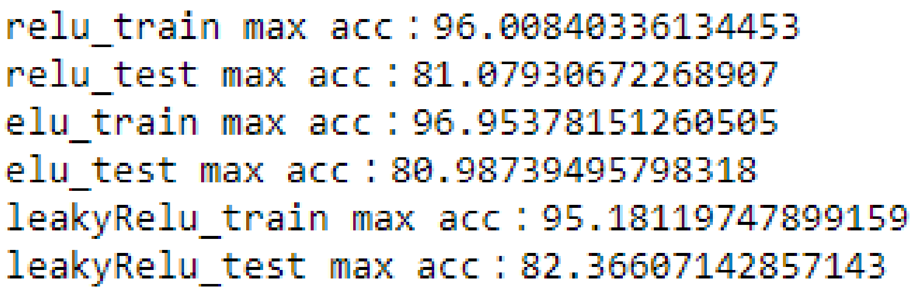
(a) The highest testing accuracy

EEGNet：



```
relu_train max acc：99.6875
relu_test max acc：88.85044642857142
elu_train max acc：99.21875
elu_test max acc：82.890625
leakyRelu_train max acc：99.53125
leakyRelu_test max acc：87.83482142857142
```

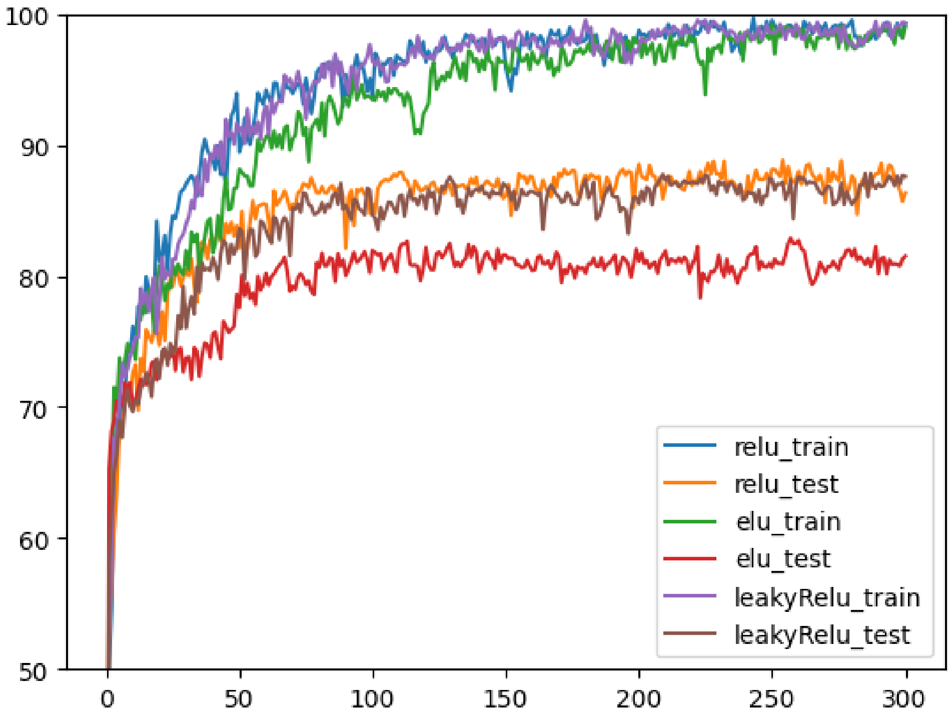
DeepConvNet：



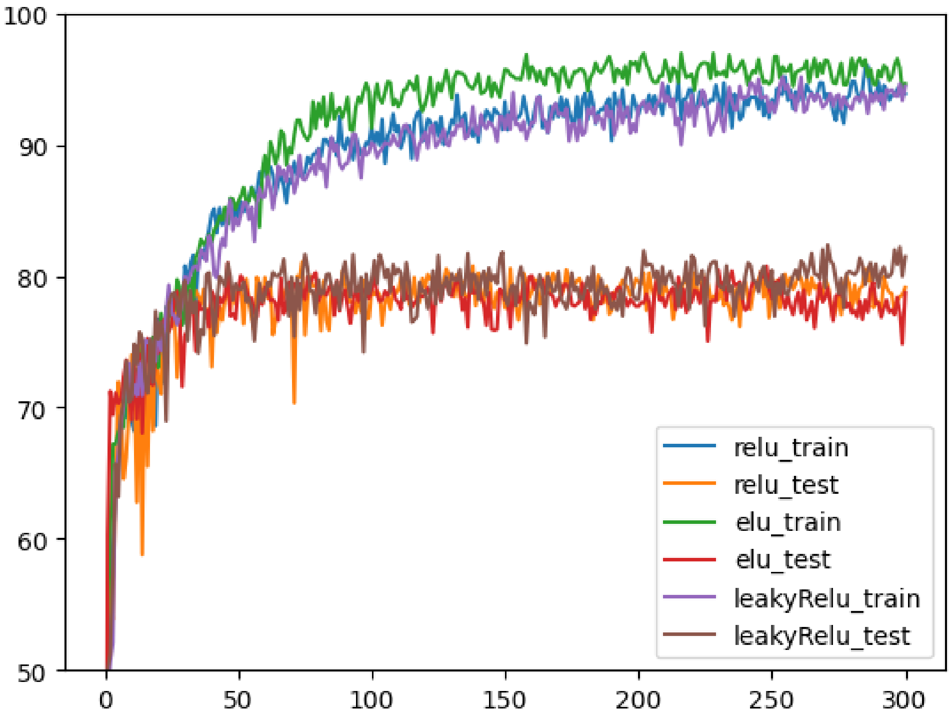
```
relu_train max acc：96.00840336134453
relu_test max acc：81.07930672268907
elu_train max acc：96.95378151260505
elu_test max acc：80.98739495798318
leakyRelu_train max acc：95.18119747899159
leakyRelu_test max acc：82.36607142857143
```

(b) Comparison figures：

EEGNet：



DeepConvNet：



(4) Discussion

(a)

Before this lab, I had not used data_loader and data_set. To understand their purposes and the meaning of their parameters, I spent some time familiarizing myself with them.

(b)

During the execution of this lab, I noticed that my GPU utilization was consistently low. Even after adjusting the batch size and the num_workers, there was no significant improvement. This is an aspect that I will need to consider and find a solution for in the future.