

## **State all the hyperparameters you need for training and how you tune them**

- learning\_rate = 0.001
- epochs = 40
- eval\_batch\_size = 32,
- lr = 0.001,
- lr\_decay\_step = 30,
- lr\_gamma = 0.1,
- momentum = 0.9,
- train\_batch\_size = 64,
- weight\_decay = 0.001

因為使用原本 sample code 的 weight\_decay = 0.0005, 根據 learning curve 的結果顯示, 會發生 overfitting 的現象, 所以後來我把 weight\_decay = 0.001 調高兩倍, 來減緩 overfitting 的發生。

optimizer 我使用 Adam, 而不是原本 sample code 的 SGD。

所有的 conv layer 的 kernel size 都設為 3。

## **Show the structure of your best model.**

```
model:  
CNN(  
    (cnn1): Conv2d(3, 32, kernel_size=(3, 3), stride=(1, 1))  
    (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)  
    (relu1): ReLU()  
    (cnn2): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1))  
    (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)  
    (relu2): ReLU()  
    (maxpool1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)  
    (cnn3): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1))  
    (bn3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)  
    (relu3): ReLU()  
    (cnn4): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1))  
    (bn4): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)  
    (relu4): ReLU()  
    (maxpool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)  
    (fc1): Linear(in_features=4096, out_features=10, bias=True)  
)
```

## **Explain the design of your model and what you've observed.**

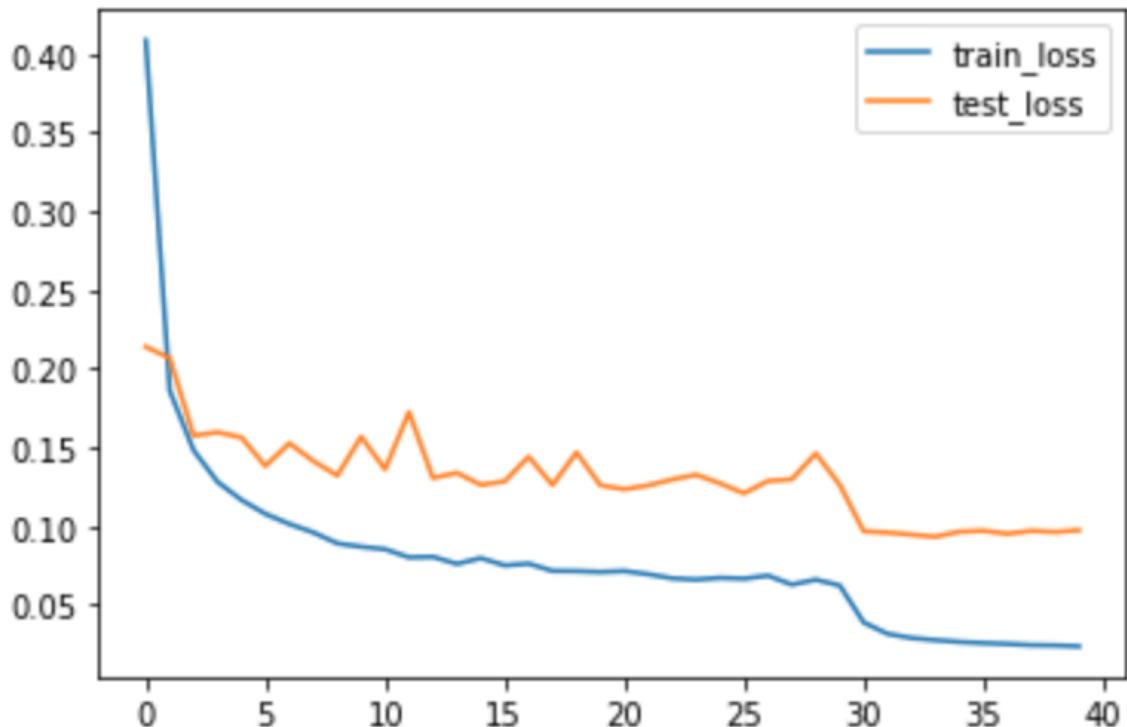
原本的 sample code 裡的 kernel 第一層是用 size = 7, 但我之後改成 size = 5 了。

原本的 sample code 是使用 stride = 2, 我則是讓 stride = 1。

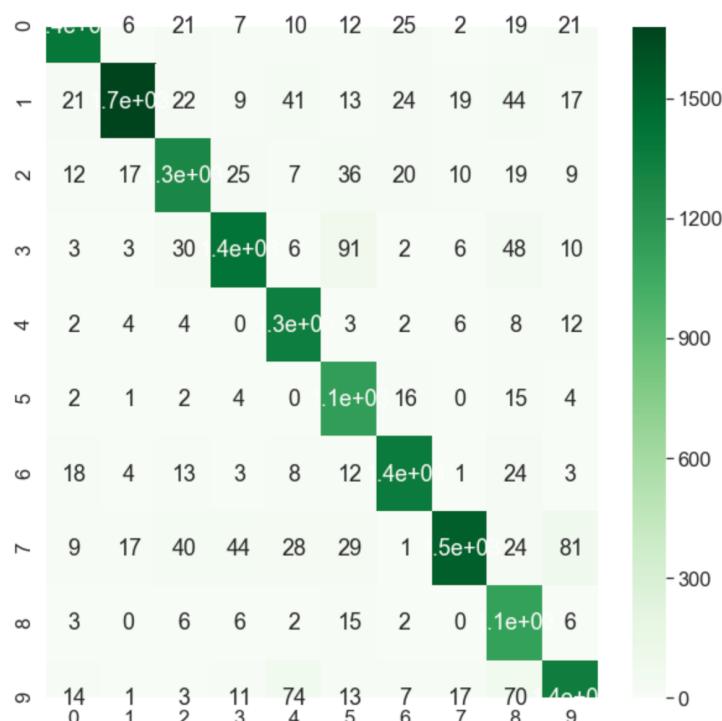
原本的 sample code 是使用 average pooling, 我則是讓 max pooling。

在實驗過程中, 回想起以前曾聽說 network 越 deep, 通常效果會越好, 實驗結果也發現確實如此。所以我最後使用的 model 的設計理念是讓 network 越 deep 越好。並且讓 kernel size 畫可能的小。所以我的 kernel size 使用 3。並且用 4 層的 convolution layer 搭配 2 層的 max pooling來進行訓練。

**Plot the learning curve during training (CrossEntropy Loss).**



**Plot the confusion matrix for validation set, and briefly explain what you've observed.**



特別容易判斷錯誤的：

- 真正的 label 是 3, 但 predict 出來是 5
- 真正的 label 是 3, 但 predict 出來是 8
- 真正的 label 是 7, 但 predict 出來是 9

- 真正的 label 是 9, 但 predict 出來是 4
- 真正的 label 是 9, 但 predict 出來是 8

完全沒預測錯誤的：

- 真正的 label 是 4, 但 predict 出來是 3
- 真正的 label 是 5, 但 predict 出來是 4
- 真正的 label 是 5, 但 predict 出來是 7
- 真正的 label 是 8, 但 predict 出來是 1
- 真正的 label 是 8, 但 predict 出來是 7

用 model 預測出來的結果，和人類用肉眼看會發生錯誤的可能情況類似。

例如錯誤發生機率最高的 3 和 5，因為數字下方都有弧線，所以可以了解為什麼 model 會判斷錯誤。