HW3

109062580 李佳陽

Date:2020/12/27

1. Model architecture 如下圖:

```
Input shape: [8, 3, 32, 32]---------->[batch size, channels, height, width]
Model Architecture:
----------------------------------------------------------------
       Layer (type)               Output Shape         Param #
================================================================
           Conv2d-1            [8, 8, 32, 32]             224
             ReLU-2            [8, 8, 32, 32]               0
        MaxPool2d-3            [8, 8, 16, 16]               0
           Conv2d-4           [8, 16, 16, 16]           1,168
             ReLU-5           [8, 16, 16, 16]               0
        MaxPool2d-6             [8, 16, 8, 8]               0
           Linear-7                   [8, 100]         102,500
             ReLU-8                   [8, 100]               0
           Linear-9                     [8, 3]             303
================================================================
Total params: 104,195
Trainable params: 104,195
Non-trainable params: 0
```

說明:

(conv1): Conv2d(input channel=3, output channel=8, kernel size=3, stride=1, padding(1, 1), pad=1, bias=True)

(relu11): ReLU()

(max_pool1): MaxPooling2d(kernel size=2, stride=2)

(conv2): Conv2d(input channel=8, output channel=16, kernel size=3, stride=1, padding(1, 1), pad=1, bias=True)

(relu12): ReLU()

(max_pool2): MaxPooling2d(kernel size=2, stride=2)

(flatten): Flatten()

(fc1): Dense(input dims=16*8*8, output dims=100)

(relu13): ReLU()

(fc2): Dense(input dims=100, output dims=3)

做後一層 fc2 輸出會經過 softmax，輸出每個 class 的機率。

Loss function:

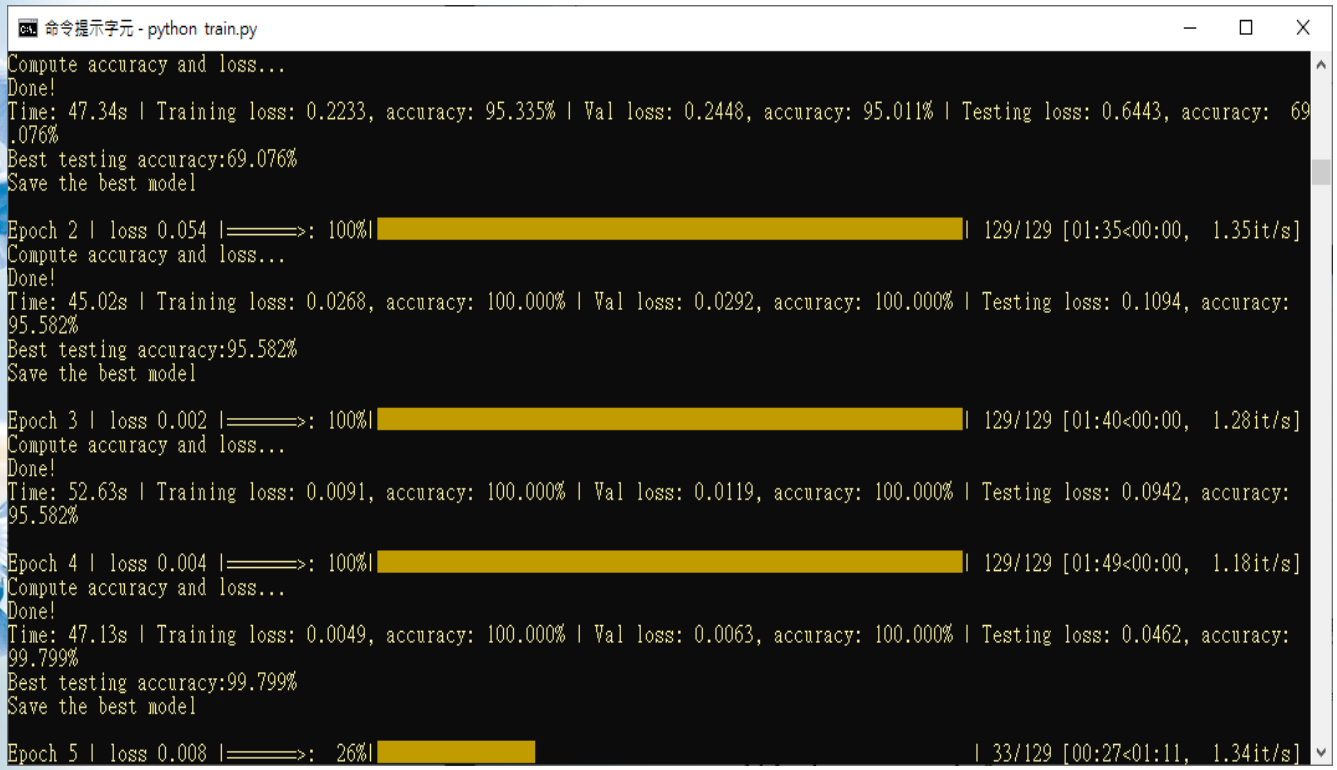
使用 cross entropy loss function.

Testing result:

```
C:\Users\zxcz1\Downloads\DL\HW\HW03>python test.py
Data preprocessing...
Done!(time:1.05s)
Start testing!
Testing finished(time:49.19)
Training loss: 0.0013, accuracy: 100.000% | Validation loss: 0.0011, accuracy: 100.000% | Testing loss: 0.0527 , accuracy: 99.598%
```

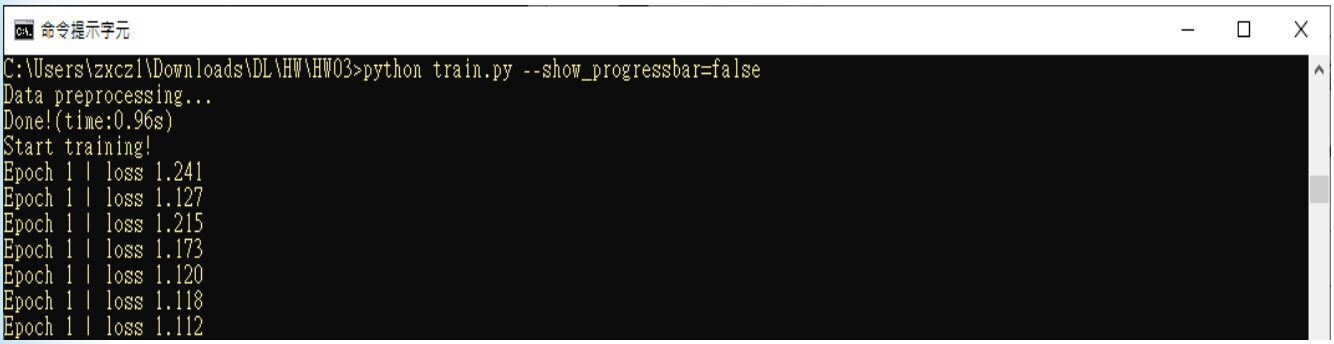
2. 訓練過程

每個 epoch 訓練完會對 training data, validation data 和 testing data 進行 accuracy 和 loss 的計算。

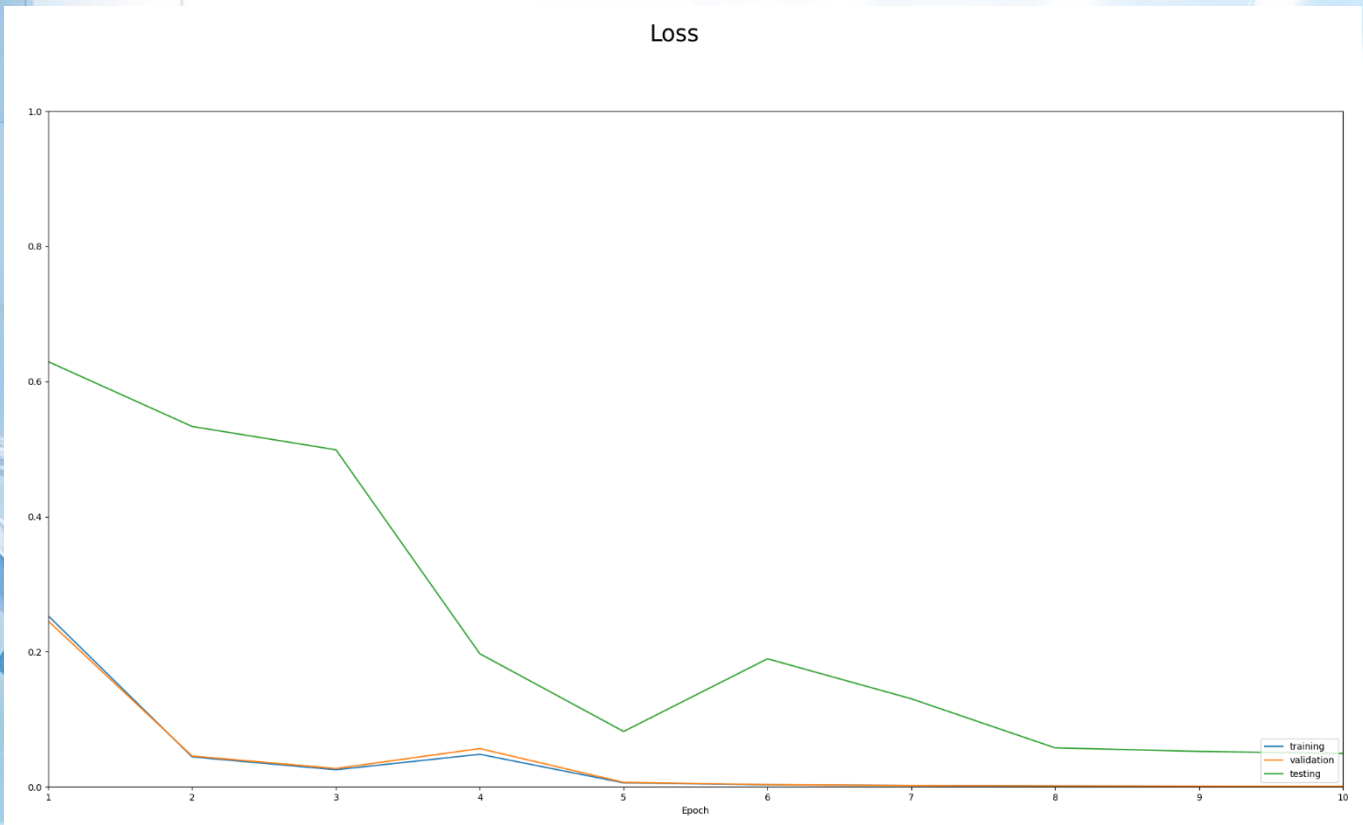


說明:使用 progress bar 顯示目前 epoch 的訓練進度、目前 epoch 已執行時間和剩餘時間和每秒多少個 iteration。

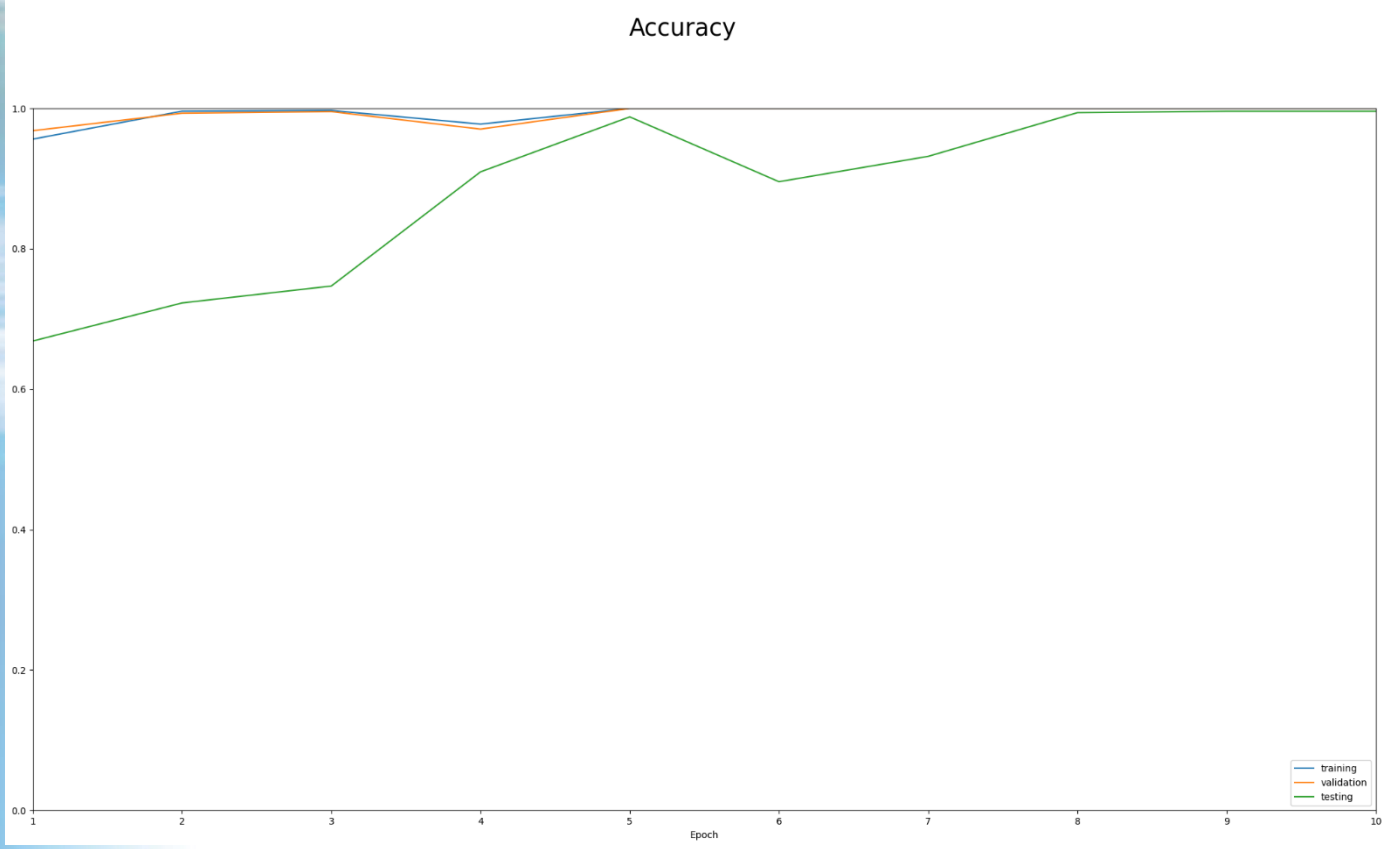
另外也可輸入指令: python train.py --show_progressbar=false 不使用 progressbar 訓練。更多 argparse 指令可參考 readme.txt 或輸入指令: argparse python train.py -help。



3. Training loss, validation loss and testing loss，其中，每個 epoch 的 loss 都是對 training data 全部迭代優化後所計算出。



Training accuracy, validation accuracy and testing accuracy，其中，每個 epoch 的 accuracy 都是對 training data 全部迭代優化後所計算出。



4. Problem encountered

Problem 1:

loss 沒有變化。

Solution:

存取每個 layer 的 parameters 和 gradients，並使用 pyplot show 出變化曲線，觀察哪個 layer 出了問題。

Problem 2:

資料分割不均勻，導致每次 testing 的結果差異大。

Solution:

因每個 class 目錄下的圖片具有固定順序排列(如下面 2 張圖)，因此讀取完後須 Shuffle。此外，在訓練時，每個 epoch 也會對 training data Shuffle。

前部分圖片

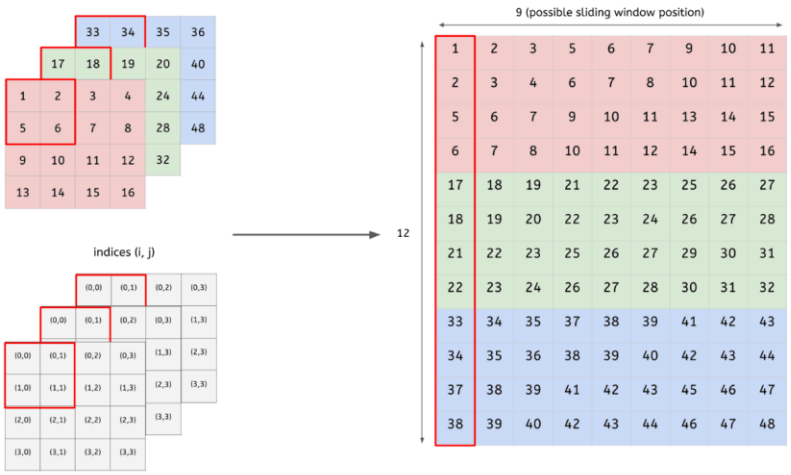
後部分圖片

5. Other Improvement

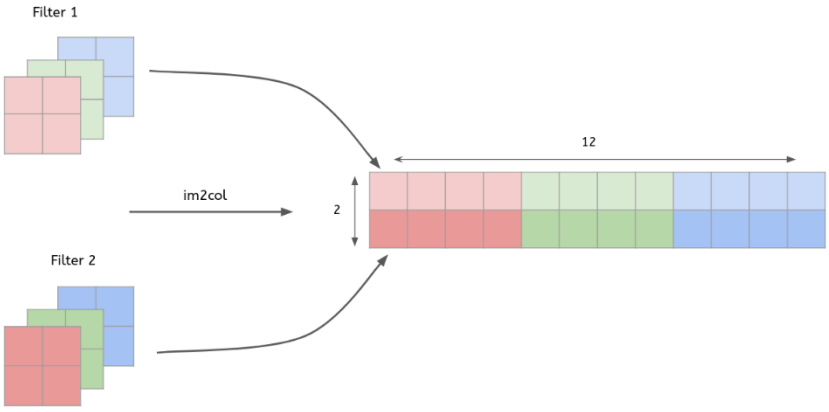
Convolutional layer:

在本作業的 Conv 運算並非使用一般方法，而是使用快速的矩陣相乘法。

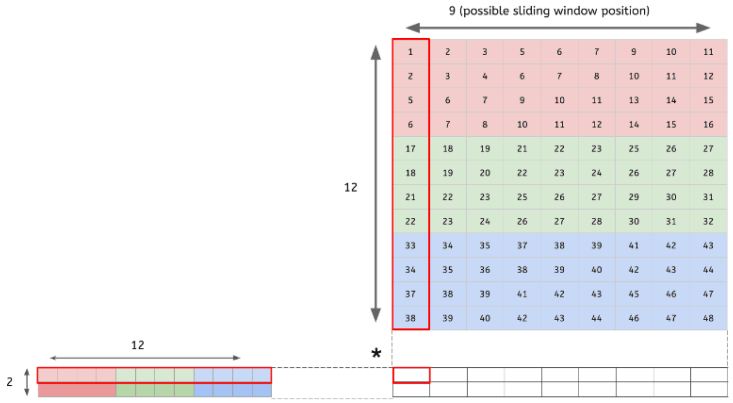
A. 將輸入圖片轉換成矩陣



B. 將 filter 轉換成 1 維



C. 將轉換完後的圖片和 filter 進行矩陣相乘運算



6. 結論

本次作業雖然在 testing data 上可到達 99%的準確度，因 random shuffle data 的關係，最後的 testing 結果可能會不如預期，雖然上述 4.提到的方法可降低 testing 結果的不穩定性，不過解決此問題的方法還是需透過適當的 data augmentation，讓 testing 結果更穩定。