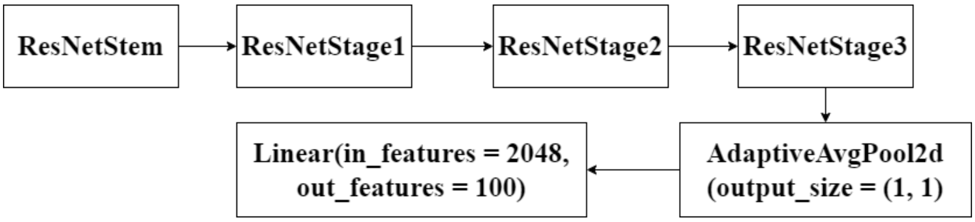
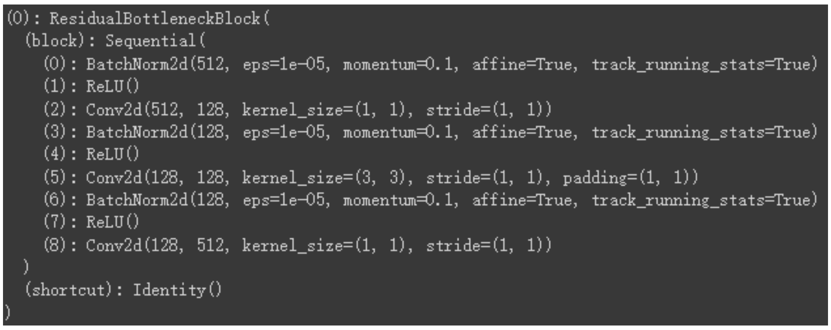
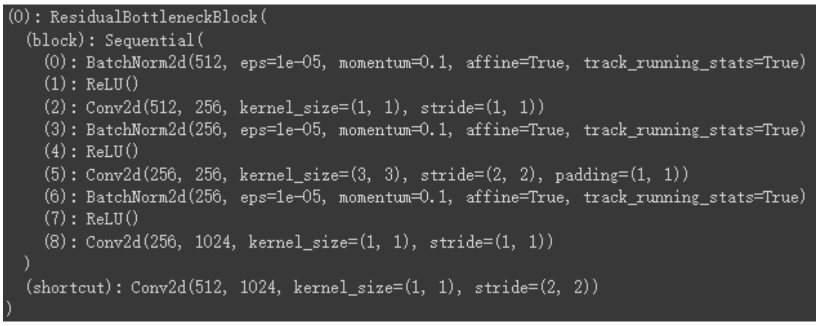
Deep Learning 2024 Midterm exam_B103012002_林凡皓

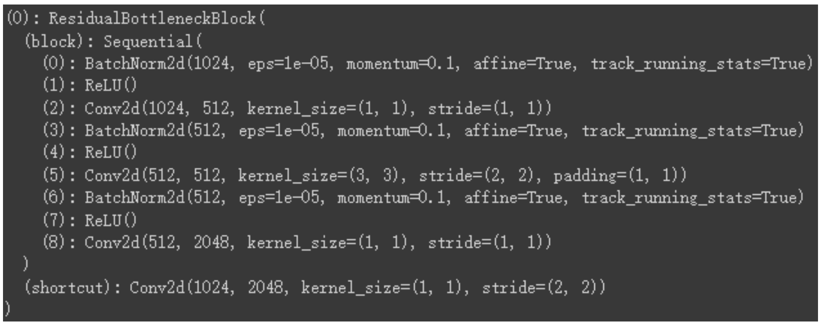
1. 訓練架構或使用的技巧：


ResNetStem
ResNetStage1
ResNetStage2
ResNetStage3
AdaptiveAvgPool2d (output_size = (1, 1)
Linear(in_features = 2048, out_features = 100)

ResNetStage 採用五個 ResidualBottleneckBlock 串接，ResidualBottleneckBlock 架構如下

<table>
<tr><td>ResNetStage1</td><td>

```
(0): ResidualBottleneckBlock(
  (block): Sequential(
    (0): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (1): ReLU()
    (2): Conv2d(512, 128, kernel_size=(1, 1), stride=(1, 1))
    (3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): ReLU()
    (5): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (6): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): ReLU()
    (8): Conv2d(128, 512, kernel_size=(1, 1), stride=(1, 1))
  )
  (shortcut): Identity()
)
```
</td></tr>
<tr><td>ResNetStage2</td><td>

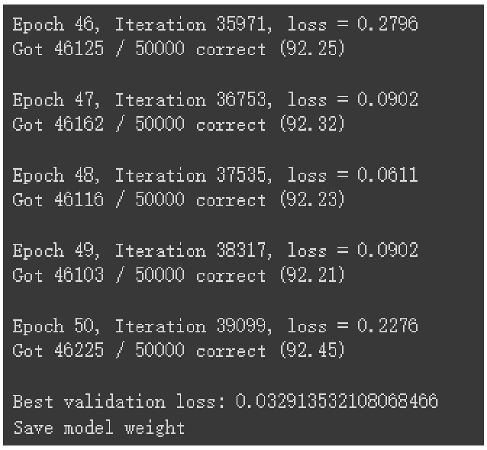
```
(0): ResidualBottleneckBlock(
  (block): Sequential(
    (0): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (1): ReLU()
    (2): Conv2d(512, 256, kernel_size=(1, 1), stride=(1, 1))
    (3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): ReLU()
    (5): Conv2d(256, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (6): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): ReLU()
    (8): Conv2d(256, 1024, kernel_size=(1, 1), stride=(1, 1))
  )
  (shortcut): Conv2d(512, 1024, kernel_size=(1, 1), stride=(2, 2))
)
```
</td></tr>
<tr><td>ResNetStage3</td><td>

```
(0): ResidualBottleneckBlock(
  (block): Sequential(
    (0): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (1): ReLU()
    (2): Conv2d(1024, 512, kernel_size=(1, 1), stride=(1, 1))
    (3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): ReLU()
    (5): Conv2d(512, 512, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (6): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (7): ReLU()
    (8): Conv2d(512, 2048, kernel_size=(1, 1), stride=(1, 1))
  )
  (shortcut): Conv2d(1024, 2048, kernel_size=(1, 1), stride=(2, 2))
)
```
</td></tr>
</table>

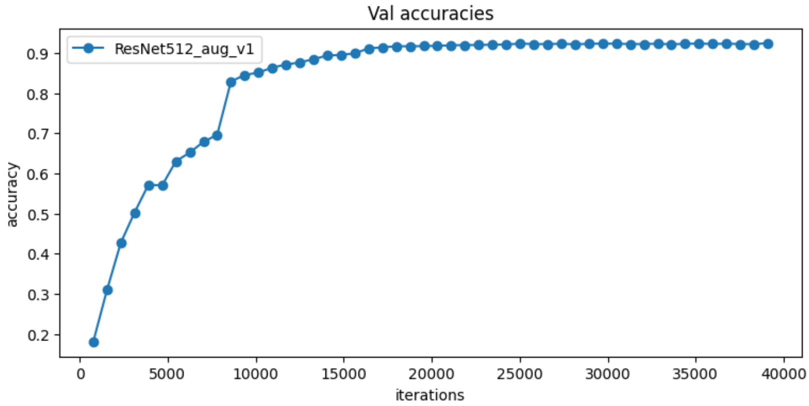
訓練技巧部分，我透過 ColorJitter、RandomVerticalFilp、RandomHorizontalFlip 來做 data augmentation，最終訓練資料數量為 250000 張(200000 張為 training data，50000 張為 validation data)。

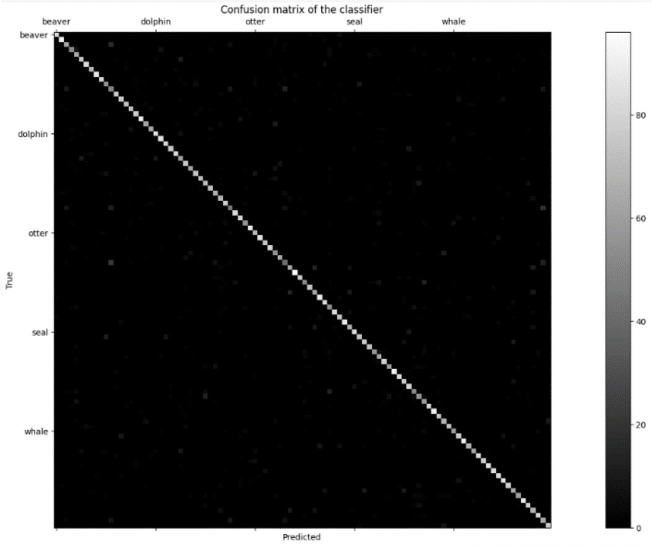
為了避免訓練時錯過最佳模型，我在每個 epochs 結束時，比較當前 epoch 的 loss 與先前最佳 loss，將比較好的模型儲存。

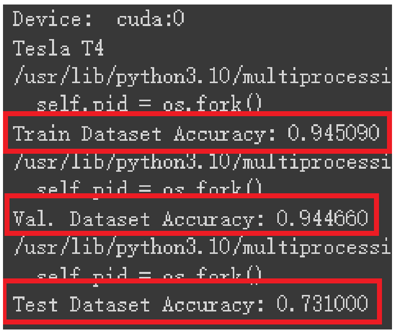
2. 訓練正確率：



```
Epoch 46, Iteration 35971, loss = 0.2796
Got 46125 / 50000 correct (92.25)

Epoch 47, Iteration 36753, loss = 0.0902
Got 46162 / 50000 correct (92.32)

Epoch 48, Iteration 37535, loss = 0.0611
Got 46116 / 50000 correct (92.23)

Epoch 49, Iteration 38317, loss = 0.0902
Got 46103 / 50000 correct (92.21)

Epoch 50, Iteration 39099, loss = 0.2276
Got 46225 / 50000 correct (92.45)

Best validation loss: 0.032913532108068466
Save model weight
```







```
Device:  cuda:0
Tesla T4
/usr/lib/python3.10/multiprocessi
  self.pid = os.fork()
Train Dataset Accuracy: 0.945090
/usr/lib/python3.10/multiprocessi
  self.pid = os.fork()
Val. Dataset Accuracy: 0.944660
/usr/lib/python3.10/multiprocessi
  self.pid = os.fork()
Test Dataset Accuracy: 0.731000
```

3. 測試正確率：

(單位：%)

data1	data2	data3	data4	data5
20.10	73.80	43.80	96.50	44.20