EAI

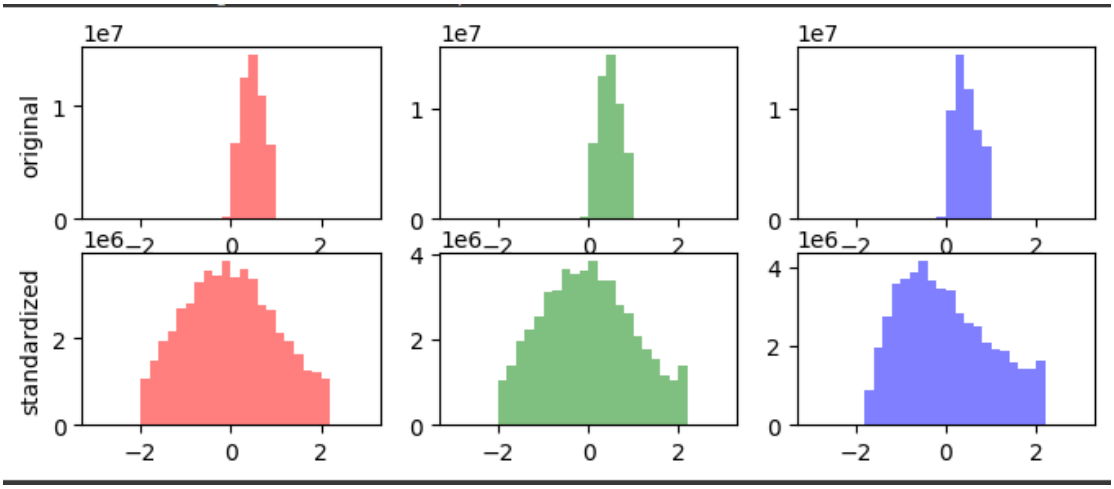
Lab 3
Report

系級	113 電機乙
學號	F64096114
姓名	郭家佑

- 說明不同 tuning 方式的原理及如何實作

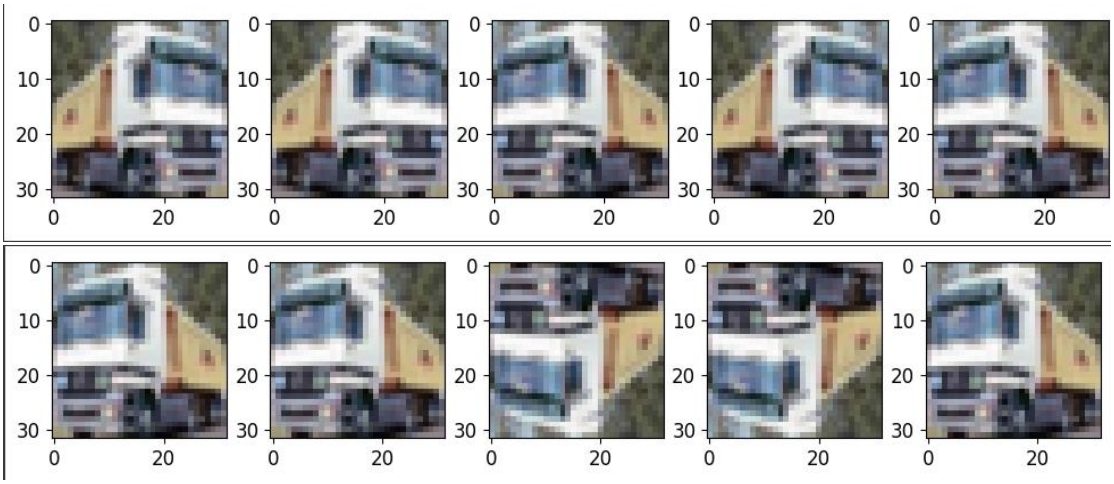
  1.standardization

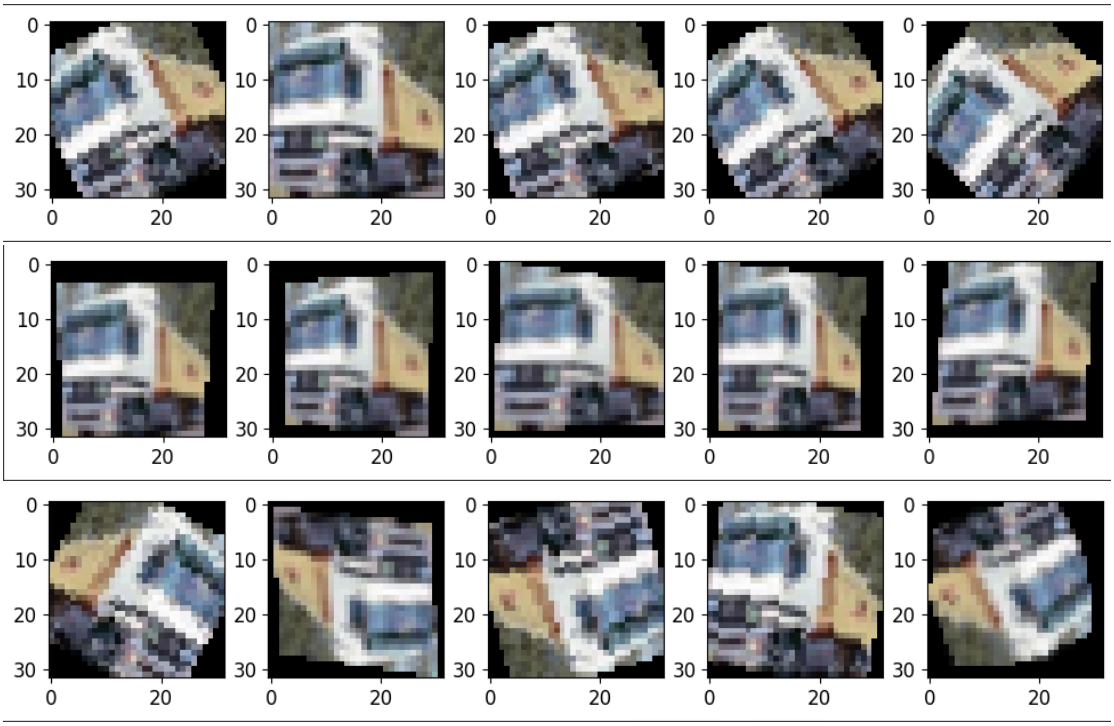
  先算出平均與標準差，在利用 transform 套件裡的 Normalize 來標準化。



  2.Data augmentation

  使用水平、垂直、旋轉、翻轉來增加訓練資料量。

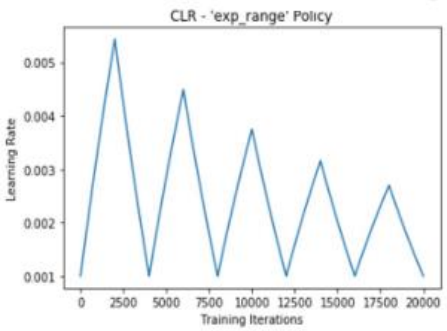




3. Dynamic LR rate

利用類似 exp_range Cyclical Learning Rates 來動態調整學習率。

```
[11] #  learning  rate  shedule
     def  adjust_learning_rate(optim,  epoch):
            #  define  your  lr  scheduler
            if  epoch  %  2  ==  0:
                optimizer.param_groups[0]['lr']  +=  optimizer.param_groups[0]['lr']*0.2
            else:
                optimizer.param_groups[0]['lr']  -=  optimizer.param_groups[0]['lr']*0.2
            #  for  param_group  in  optim.param_groups:
            #          param_group['lr']  =  lr
```



4. Different batch size

將原先 128 的 batch_size 變大，訓練時間減少許多。

- 說明並比較不同 tuning 方式如何造成影響

1. Standardization

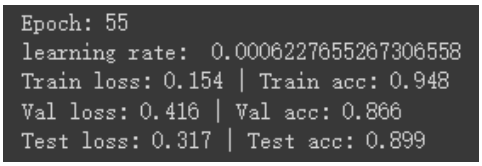
   使資料靠平均集中，可使 back propagation 時不會受到 gradient vanishing 影響。

2. Data augmentation

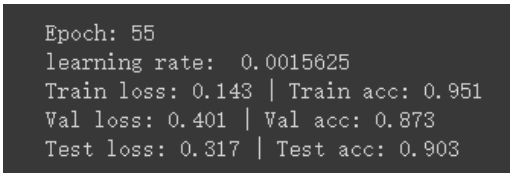
   增加訓練集資料可使在少的資料集中也可以有好的訓練結果

3. Dynamic LR rate

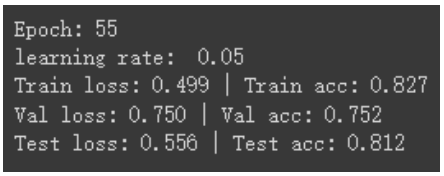
   有動態調整 LR 皆會比 static LR 好很多。



自製 dynamic LR



StepLR



Static LR

4. Different batch size

較大的 batch size 雖然訓練時間快很多，但是準確率沒辦法跟小的 batch_size 同水平。

```
Epoch: 55
learning rate:  0.0015625
Train loss: 0.254 | Train acc: 0.911
Val loss: 0.496 | Val acc: 0.833
Test loss: 0.420 | Test acc: 0.864
```

大 batch_size

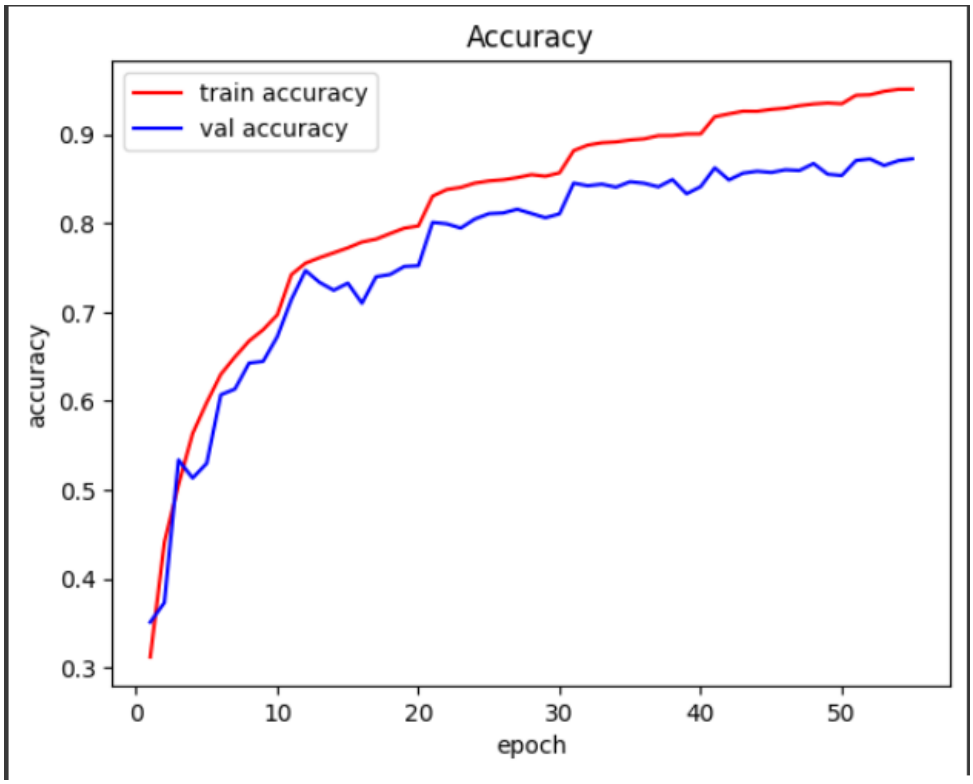
```
Epoch: 55
learning rate:  0.0015625
Train loss: 0.143 | Train acc: 0.951
Val loss: 0.401 | Val acc: 0.873
Test loss: 0.317 | Test acc: 0.903
```

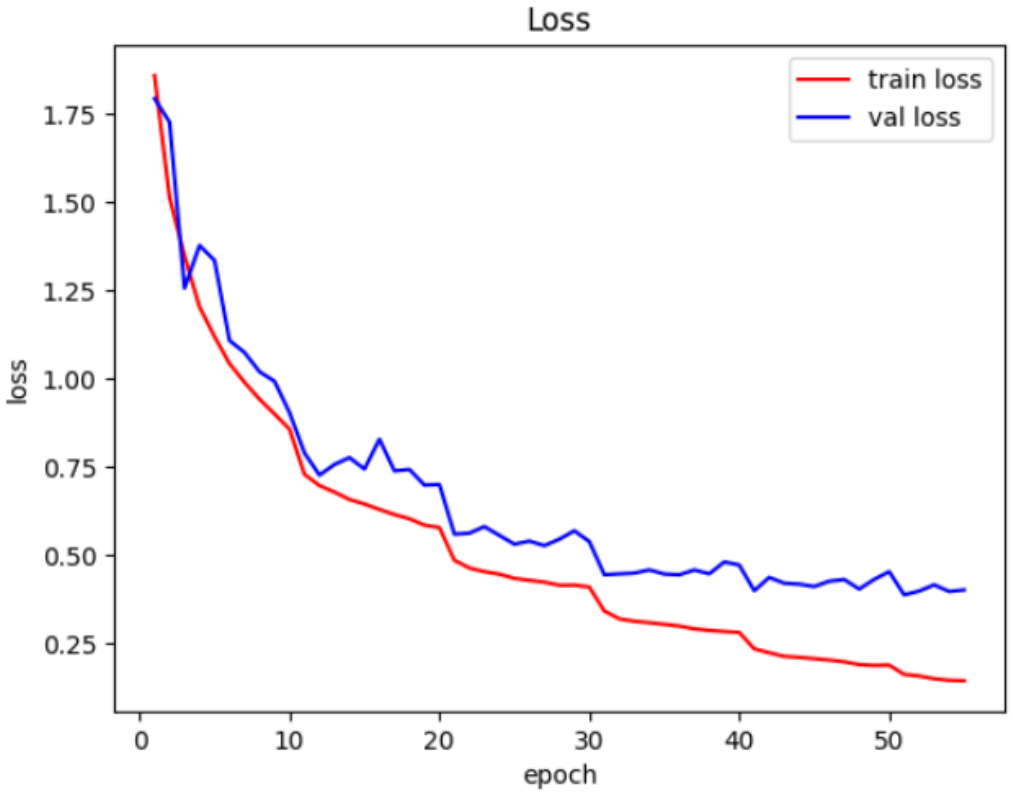
小 batch_size

- 截圖並說明各項結果

由 Accuracy 的圖形結果可看出有了 Dynamic learning rate 的幫助下能帶來顯著的改善，尤其是再改動後的那次，都能使原本已經快要飽和的準確率又跳高了一截，幫助整體預測的準度，最終測試達 90%。

```
Epoch: 55
learning rate:  0.0015625
Train loss: 0.143 | Train acc: 0.951
Val loss: 0.401 | Val acc: 0.873
```

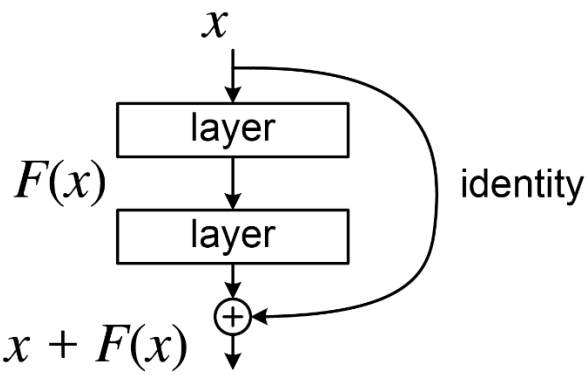





```
Test loss: 0.317 | Test acc: 0.903
```

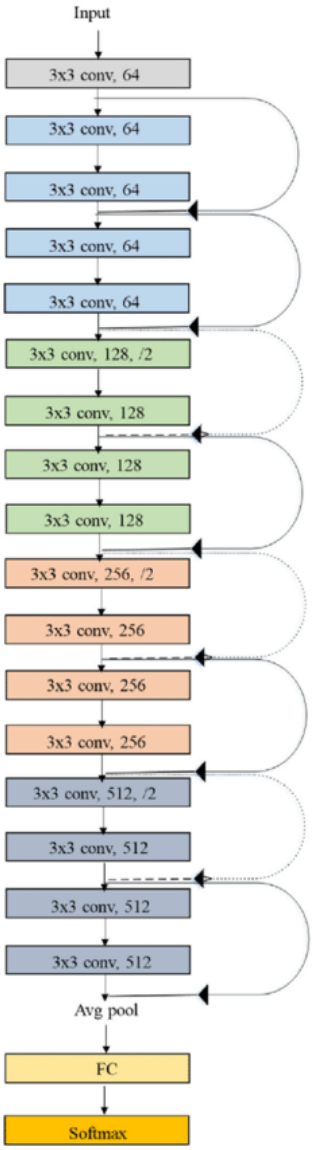
- 如何搭建 Resnet18

  首先是搭建 block 網路，其內部由兩個 Convolution 和兩個 batch normalization 和一個 ReLU 層所組成，最後一層要與 input 做 shortcuut 的動作再 ReLU 輸出 。



接著利用先前搭建的 block 網路來建 ResNet18，透過

make_layer 來串接所建立的子 layer，由於每大層都要有兩個 block，因此 num_blocks=2，而剩下的參數就對照 ResNet18 各自填入對應的值，最後再利用這五大層加上頭尾的 convolution 和 fully-connected 就建立完成。



- 實作過程中遇到的困難及你後來是如何解決的

  一開始其實不太懂 LR 要如何動態調整，經過搜尋後大約有了想法，又發現助教有在 code 裡面幫我們加上一般的遞減 LR 了，因此就模仿他寫出來。此外，原先對於 ResNet 的 short cut 路徑為何就能讓網路一直加深而能繼續收斂有點好奇，後來經過論文公式的推導與作者本人在 YT 的解說後就清楚了。