

DL Lab 1: flowers_102 CNN classifier

資料增強 (Data Augmentation)

- ◎ 計算 dataset 的 mean 和 std

將影像統一 resize 成 500*500 的大小，並計算其均值和標準差，以便後續進行 normalize。

```
import numpy as np

# 初始化累計值 (RGB三個通道)
mean_sum = np.zeros(3)
std_sum = np.zeros(3)
total_samples = 0

# 計算累計的均值和標準差
for images, _ in trainloader:
    batch_samples = images.size(0)
    images = images.view(batch_samples, images.size(1), -1)
    mean_sum += images.mean(2).sum(0).numpy()
    std_sum += images.std(2).sum(0).numpy()
    total_samples += batch_samples

# 計算最終的均值和標準差
dataset_mean = mean_sum / total_samples
dataset_std = std_sum / total_samples
print('Mean:', np.round(dataset_mean, 3))
print('Standard Deviation:', np.round(dataset_std, 3))
```

```
Mean: [0.436 0.376 0.285]
Standard Deviation: [0.266 0.212 0.218]
```

- ◎ 合併資料集 (Concatenate dataset)

將訓練集的影像做資料增強後，再與原始的訓練集影像合併，使資料量變兩倍。Batch size 設為 64。

```
augmentation_transform = transforms.Compose([
    transforms.RandomResizedCrop(500), # 隨機裁切為不同大小和寬高比，再縮放至500*500
    transforms.RandomHorizontalFlip(), # 隨機水平翻轉
    transforms.RandomRotation(degrees=30), # 隨機旋轉±30度
    transforms.ToTensor(),
    transforms.Normalize((0.436, 0.376, 0.285), (0.266, 0.212, 0.218))
])

original_transform = transforms.Compose([
    transforms.Resize(500),
    transforms.ToTensor(),
    transforms.Normalize((0.436, 0.376, 0.285), (0.266, 0.212, 0.218))
])
```

資料量

訓練集：12298 張，驗證集：1020 張，測試集：1020 張

模型架構

Residual Neural Network 18

⊙ Preprocessing layer

Layer	Channel	Kernel	Stride	Padding
Conv	64	7	2	3
BatchNorm	64			
ReLU / Leaky ReLU / PReLU				
MaxPool	64	3	2	1

⊙ BasicBlock：每個 layer 有 2 組，總共有 16 層卷積層。

Layer1	Channel	Kernel	Stride	Padding
Conv	64	3	1	1
BatchNorm	64			
ReLU / Leaky ReLU / PReLU				
Dropout				
Conv	64	3	1	1
BatchNorm	64			
Layer2	Channel	Kernel	Stride	Padding
Conv	128	3	2, 1	1
BatchNorm	128			
ReLU / Leaky ReLU / PReLU				
Dropout				
Conv	128	3	1	1
BatchNorm	128			
Layer3	Channel	Kernel	Stride	Padding
Conv	256	3	2, 1	1
BatchNorm	256			
ReLU / Leaky ReLU / PReLU				
Dropout				
Conv	256	3	1	1
BatchNorm	256			
Layer4	Channel	Kernel	Stride	Padding
Conv	512	3	2, 1	1
BatchNorm	512			
ReLU / Leaky ReLU / PReLU				
Dropout				
Conv	512	3	1	1
BatchNorm	512			

⊙ Average pooling、Fully connected

Convolutional Neural Network

為了進行比較，此 CNN model 使用上方 ResNet-18 拿掉 shortcut 後的架構，並使用 ReLU 作為激勵函數。

其他超參數設定

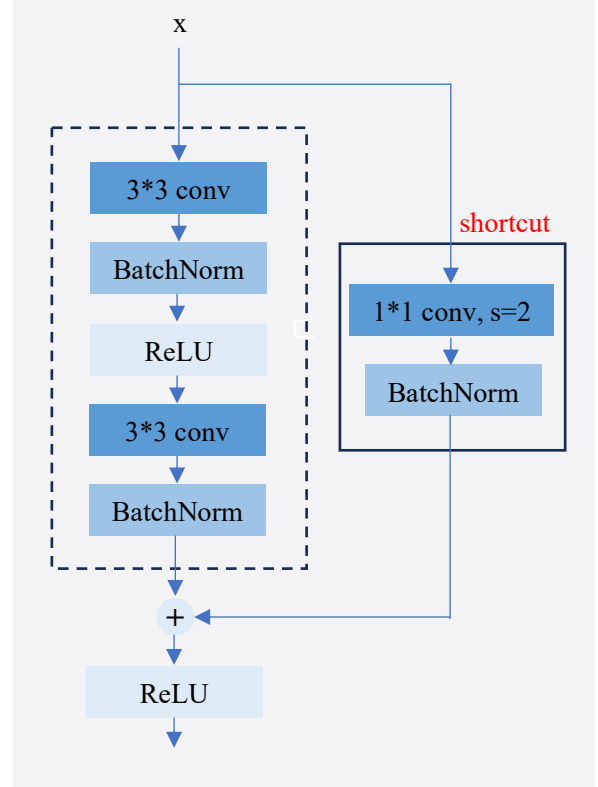
Loss function: CrossEntropyLoss

Optimizer: Adam

Learning rate: 0.001

Epoch: 140

當通道數變化時，如 64->128，即進行 shortcut，將輸入 x 升維並下採樣到 128。



模型比較

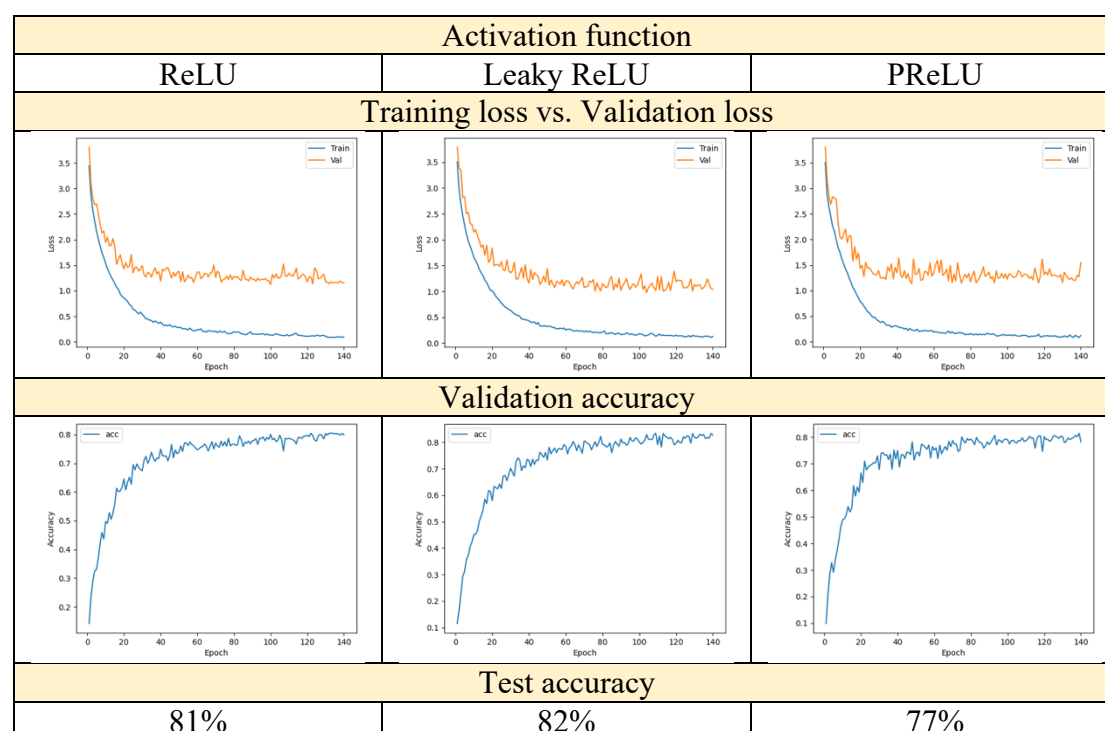
● 三種不同的激勵函數用在 ResNet-18 的比較

- 震盪幅度：PReLU>Leaky ReLU>ReLU
- 收斂速度：PReLU >ReLU>Leaky ReLU

說明：

在超參數相等的情況下，可以觀察到 PReLU 具有較快的學習速度，然而其震盪幅度卻較大。PReLU 是一種自適應斜率的激勵函數，能在負區域產生較小的梯度，這可能會減少負輸入對輸出的影響，進而降低輸出的震盪幅度。相比之下，Leaky ReLU 也能在負區域產生較小的斜率，但比 PReLU 的斜率小，因此震盪幅度會略大於 PReLU。而 ReLU 在負區域的梯度為零，可能導致輸出的震盪幅度較大，特別是在訓練初期。因此，理論上震盪幅度應該是 ReLU > Leaky ReLU > PReLU。

觀察以下圖表，Training loss 下降到 0.3 左右，而 Validation loss 則是停在 1 以上震盪，因此可能存在 Overfitting 的問題，我認為 0.001 的學習率可能過大。由於一個模型訓練了很多個 Epoch，又因資源有限無法進行太多嘗試，所以在未來訓練模型時應該要放慢學習速度，嘗試較小的學習率或在訓練過程中遞減學習率，以確保模型能夠更穩定的收斂。



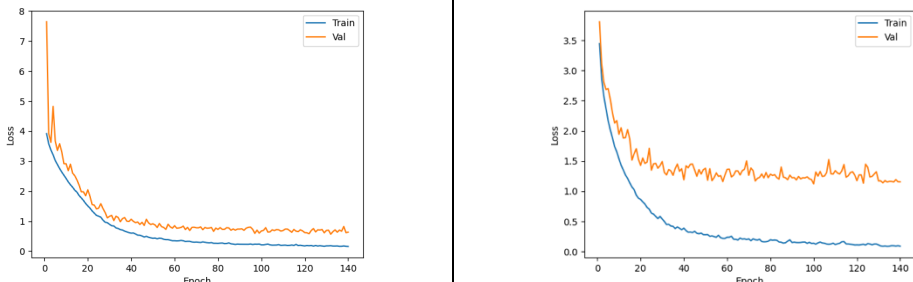
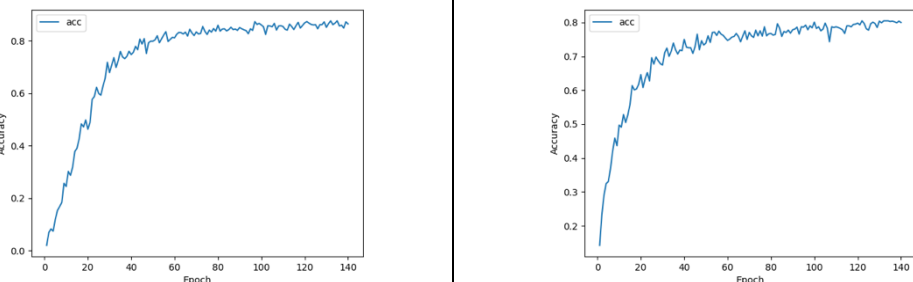
● 基於 ReLU 在 CNN 和 ResNet-18 的比較

- 震盪幅度：ResNet>CNN
- 收斂速度：ResNet>CNN

說明：

首先，解釋 ResNet 通常優於 CNN 的原因：ResNet 引入了殘差學習機制，即 shortcut 或跳躍連接，使模型能更順利地傳播梯度，提高收斂速度，並解決深度網絡訓練中的梯度消失或爆炸問題。同時，它也能輕鬆學習恒等映射，將輸入特徵直接添加到神經網路某些層的輸出中，使優化過程更容易。

然而，在我的模型訓練中，雖然 ResNet 的收斂速度較 CNN 快，但整體還是 CNN 的效果比 ResNet 更好。從以下圖表可以看到 Validation loss 的曲線趨近於 Training loss 的曲線，並且震盪幅度相對 ResNet 小很多。我認為有可能是我在這個資料集的超參數選擇比較適合 CNN，特別是在資料集較小的情況下，CNN 或許能更有效的避免 Overfitting。而 CNN 也擁有較小的模型規模，需要的計算資源較少，這使得在資源有限的情況下，它能更容易執行和部署。

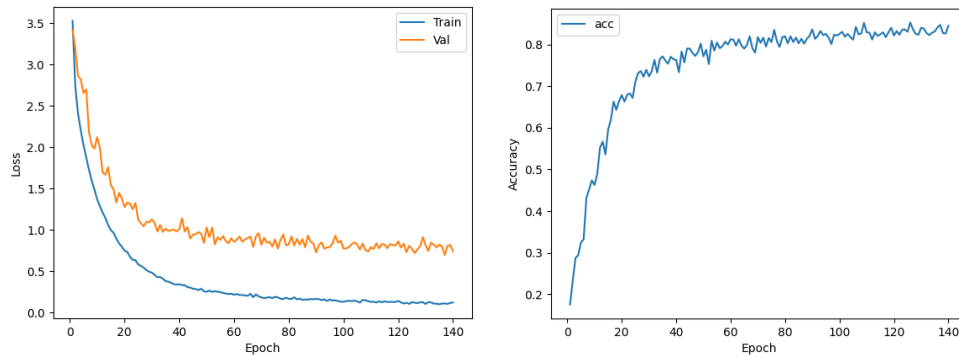
Model	
CNN	ResNet-18
	
Validation accuracy	
	
Test accuracy	
86%	81%

CNN's classification report

	precision	recall	f1-score	support
accuracy			0.86	1021
macro avg	0.88	0.86	0.86	1021
weight avg	0.88	0.86	0.86	1021

● 設定不同超參數的 ResNet-18 最佳結果

為了降低訓練時間，我將資料集 resize 成 320*320 (mean 不變，std 改為 0.264, 0.21, 0.216)，資料增強部分不使用 RandomRotation，並嘗試將 Learning rate 設為 0.0001，其餘超參數不變。使用 ResNet-18 且激勵函數為 ReLU 的準確率有到 84%。



ResNet's classification report

	precision	recall	f1-score	support
accuracy			0.84	1021
macro avg	0.86	0.84	0.84	1021
weight avg	0.86	0.84	0.84	1021

其他嘗試

1. 以準確率為 86% 的 CNN 和準確率為 84% 的 ResNet-18 做比較，分別 print 出兩者在測試集中準確率小於等於 50% 的類別，可以看到其中重疊的類別有 canterbury bells、siam tulip...。而這個 Flowers 資料集的 102 個類別的影像數量都不一樣，最少有 40 張，最多有 258 張。我認為有可能是這些類別的影像數量較少，因此未來可以嘗試對特定類別做資料增強，使得模型能有更充足的資料量進行學習及訓練。

2. 在模型的最後一層加上 Softmax 的效果並不好

有加 Softmax 的前三個訓練 Epoch：

```
Epoch: 1
  Batch: 193 of 193, Loss: 4.549, Validation Loss: 4.6071, Accuracy: 3.33%

Epoch: 2
  Batch: 193 of 193, Loss: 4.511, Validation Loss: 4.6034, Accuracy: 3.92%

Epoch: 3
  Batch: 193 of 193, Loss: 4.496, Validation Loss: 4.5919, Accuracy: 4.71%
```

沒加 Softmax 的前三個訓練 Epoch：

```
Epoch: 1
  Batch: 193 of 193, Loss: 3.585, Validation Loss: 3.9674, Accuracy: 10.98%

Epoch: 2
  Batch: 193 of 193, Loss: 2.939, Validation Loss: 3.2846, Accuracy: 20.88%

Epoch: 3
  Batch: 193 of 193, Loss: 2.600, Validation Loss: 2.7844, Accuracy: 29.80%
```