關於 ResNet:

網路越深不一定能增加模型效果。而 ResNet 透過 Residual block 和 Shortcut connection 來克服深度帶來的 Gradient vanishing 和收斂的問題。

本次實驗使用 ResNet34 在三種不同 activation function(Sigmoid, LeakyReLU, ReLU)下訓練 flowers102，並且用 ReLU 訓練 200 epochs 來觀察結果。

- Training strategy and additional function:

✧ Data augmentation and concatenate

```
transform1 = transforms.Compose(
    [transforms.ToTensor(),
     transforms.Resize([224,224]),
     transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
     ])
transform2 = transforms.Compose(
    [transforms.ToTensor(),
     transforms.RandomRotation(degrees=45),
     transforms.Resize([224,224]),
     transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
     ])
```

```
trainset = ConcatDataset([trainset1, trainset2])
```

準確率在訓練初期使用此策略有提升 10%以上所以保留

✧ 增加 Fully connected layer 強化特徵學習來提高 accuracy

```
        self.fc = nn.Linear(512 * block.expansion, num_classes)
#         self.fc2 = nn.Linear(250, num_classes)
```

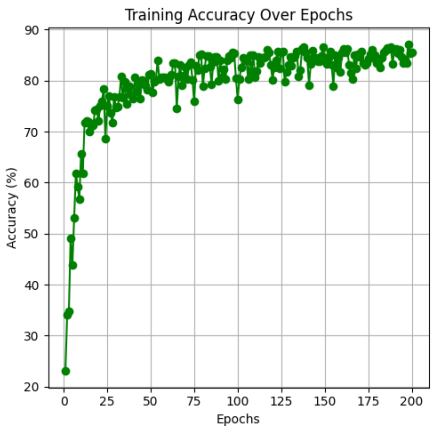
增加 fully connected layer 到最終分類層數的策略在準確率沒有顯著提升，並且增加 GPU 負擔所以在此試驗中不採用

✧ 用 learning rate 隨 epoch 遞減的 strategy

```
optimizer = torch.optim.Adam(model.parameters(), lr=0.0001)
scheduler = StepLR(optimizer, step_size=20, gamma=0.5)
```

在準確度上沒有明顯幫助，但是在協助收斂穩定度上有明顯幫助

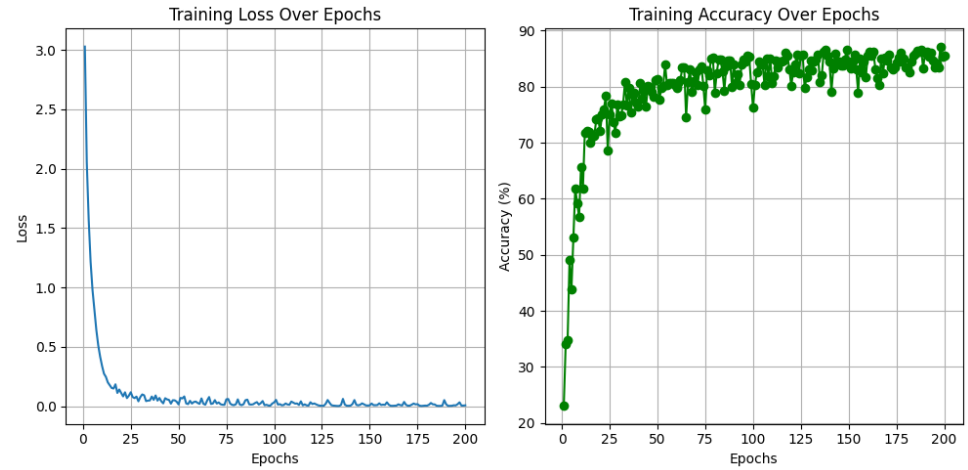
Before



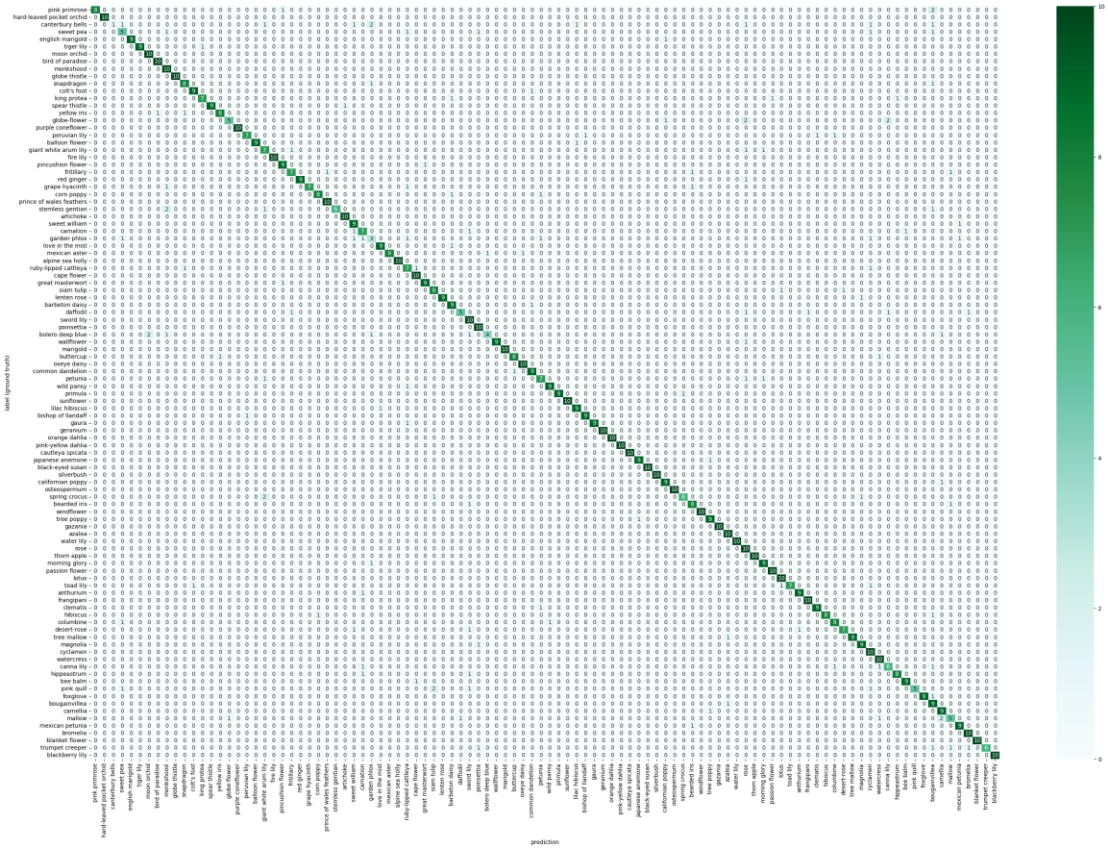
After

- Performance comparation

✧ 200 epochs (ReLU)

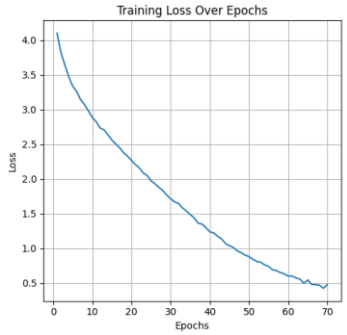


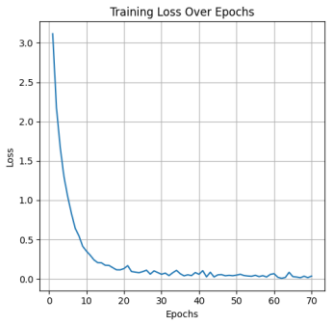
Best accuracy of the network on the test images: 87.16 %

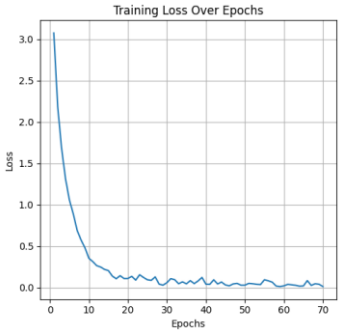


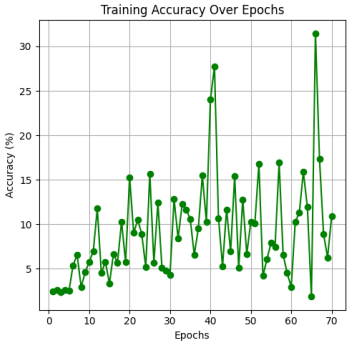
✧ ResNet34 with different activation function

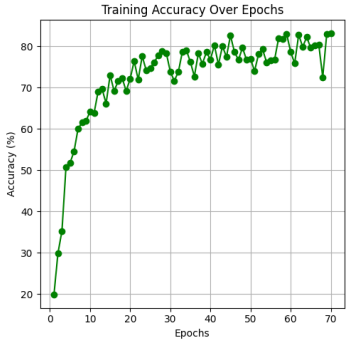
Sigmoid LeakyReLU ReLU



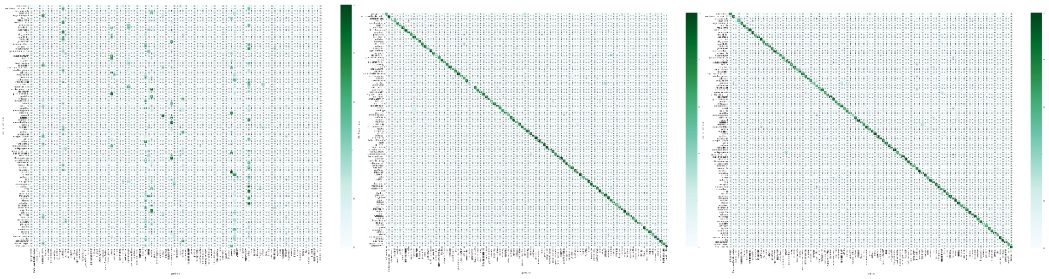









```
Best accuracy of the network on the test images:
31.47 %                85.10 %                83.24 %
```

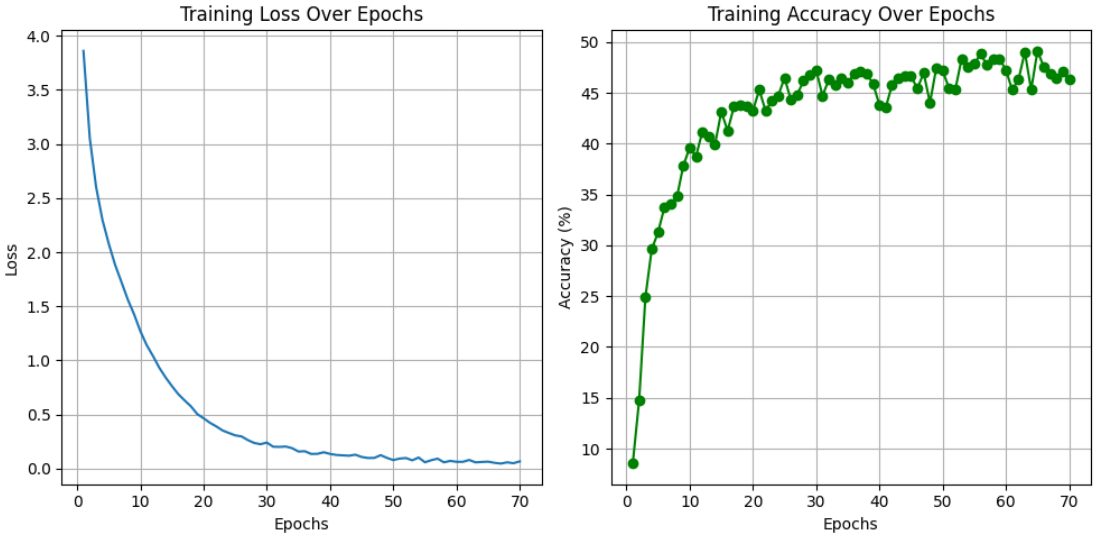


Performance of 3 activation function(70 epochs):

LeakyReLU >= ReLU > Sigmoid

觀察下來三種 activation function 在 flowers102 上面 LeakyReLU 更快到達最後的準確度附近，並且準確度波動幅度比 ReLU 還小，而因為 Sigmoid 通常在二元分類中使用，所以我們可以從圖表中看到他在訓練成果中不盡理想。

- Different of ResNet and Lab0, and the reason of better performance




```
Best accuracy of the network on the test images: 49.12 %
```

在 epoch 為 70 下，ResNet34 ReLU 為 83.24% Lab0 為 49.12%，可以明顯觀察出 Resnet34 有更好的學習結果。Resnet 是在架構中加入 Residual block 其中包括 Shortcut connection 來避免在使用深層訓練的時候所造成的 Gradient vanishing 問題。而關於計算的部分是計算 residual 而不是整個函式所以整個過程更加容易而且也凸顯了誤差之間的變化程度。也因此可以比 CNN 在用更深層的網路架構上，有更好的準確度的效果。

如果經過模型第一層出來的H(x)為3.0，則F(x)=3.0–2.9=0.1
經過模型第二層出來的H(x)為3.1，則F(x)=3.1–2.9=0.2

一般網路層的誤差變化率會是(3.1–3.0)/3.0=**3.33%**
殘差網路層的誤差變化率會是(0.2–0.1)/0.1=**100%**

H(x)為要學習的函數，透過 F(x)=H(x)-x 知曉計算 residual 更加有效，最後透過移項得到->H(x)=F(x)+x，得到所要學習的函數。而 Shortcut connection 即可產出 F(x)+x，並且不會增加額外運算量或負擔。