HW6 Two Layer Neural Network 優化

M113040105 劉東霖

壹.用到的 function:

一.nn_get_search_params:

如下圖所示，這是我調了很多次參數和觀察了很多準確率之後，決定使用這一組參數觀察。

```
learning_rates = [1.0]
hidden_sizes = [4096]
regularization_strengths = [0,1e-3,]
learning_rate_decays = [1.0,0.95,0.9]
```

二.find_best_net:

(1) 如下圖所示，先把 train 和 validation 的資料從字典裡面取出，並把 learning_rates 和 hidden_size 和正則化強度和 learning_rate_decays 從 nn_get_search_params 取出。

```
x_train,y_train,x_val,y_val=data_dict['X_train'],data_dict['y_train'],data_dict['X_val'],data_dict['y_val']
learning_rates, hidden_sizes, regularization_strengths, learning_rate_decays = nn_get_search_params()
```

for 4 個迴圈準備跑前面的超參數。

```
for lr in learning_rates:
    for hidden_size in hidden_sizes:
        for reg in regularization_strengths:
            for lr_decay in learning_rate_decays:
```

(2) 如下圖所示，利用前面超參數裡面的 hidden_size 來建立 model。

3*32*32 為輸入圖片的大小，10 為輸出的類別個數。

```
net = TwoLayerNet(3 * 32 * 32, hidden_size, 10, device=x_train.device, dtype=x_train.dtype)
```

(3) 如下圖所示，把 train 和 validation 代入 model 裡面去訓練，並把目前迴圈執行到的 learning_rates 和正則化強度和 learning_rate_decays 帶入 model 裡測試，batch_size 的部分設為 1000。疊代次數設為 3000，因為我發現疊代越多次準確率越高，但相對跑一次訓練需要花很久的時間。

```
stats  =  net.train(x_train,y_train,x_val,y_val,
    num_iters=3000,  batch_size=1000,
    learning_rate=lr,  learning_rate_decay=lr_decay,
    reg=reg,  verbose=False)
```

(4) 如下圖所示，把 validation 的資料代入 model 去預測，並計算準確率。最後再把目前執行到的超參數和準確率印出來。

```
y_pred=net.predict(x_val)
acc=torch.sum(y_pred==y_val)/y_val.shape[0]
print('{lr:',lr,',hidden_size:',hidden_size,',reg:',reg,',lr_decay:',lr_decay,'}:acc',acc.item())
```

(5) 如果目前的準確率大於最佳的準確率，就把目前的準確率和 model 和 loss 和一些東西記錄下來。

```
if(acc>best_val_acc):
    best_val_acc=acc
    best_net=net
    best_stat=stats
```

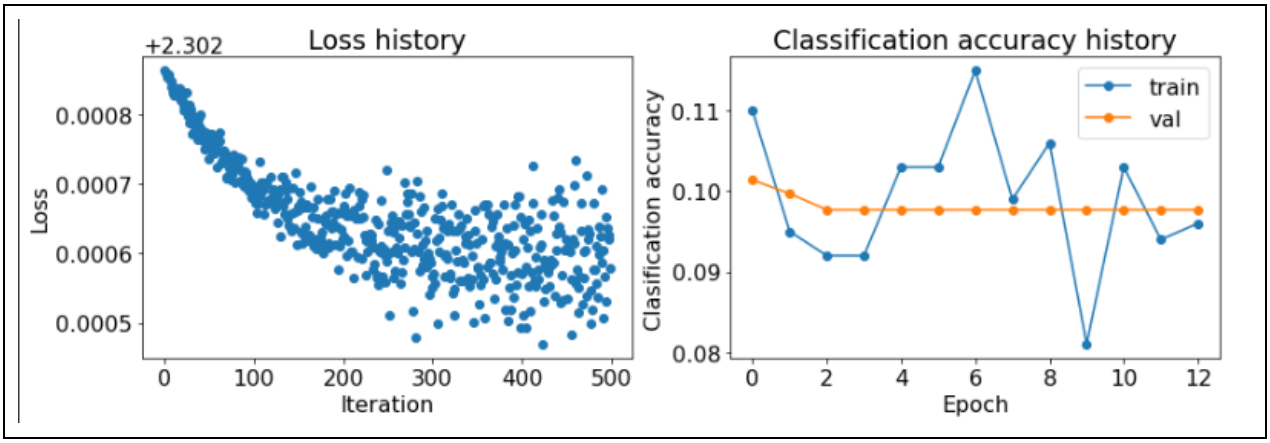
貳.執行結果:

(1) 如下圖所示，當疊代次數 500 次和 batch_size=1000 和 hidden_size=36 和 learning_rate=0.01 和 learning_rate_decay=0.95 和正則化強度=0.25 時，validation 準確率為 9.77%，loss 為 2.302571，可

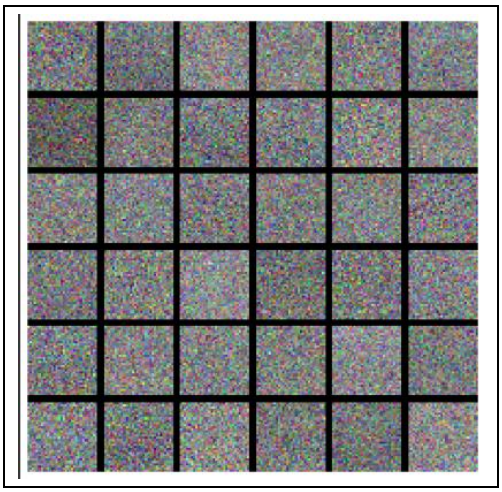
以看到 loss 雖然在下降，但是 train 的準確率搖擺不定，validation 的準確率一直停在某個值。

```
net = TwoLayerNet(input_size, hidden_size, num_classes, dtype=data_dict['X_train'].dtype, device=data_dict['X_train'].device)

# Train the network
stats = net.train(data_dict['X_train'], data_dict['y_train'],
                  data_dict['X_val'], data_dict['y_val'],
                  num_iters=500, batch_size=1000,
                  learning_rate=1e-2, learning_rate_decay=0.95,
                  reg=0.25, verbose=True)

# Predict on the validation set
y_val_pred = net.predict(data_dict['X_val'])
val_acc = 100.0 * (y_val_pred == data_dict['y_val']).double().mean().item()
print('Validation accuracy: %.2f%%' % val_acc)
```

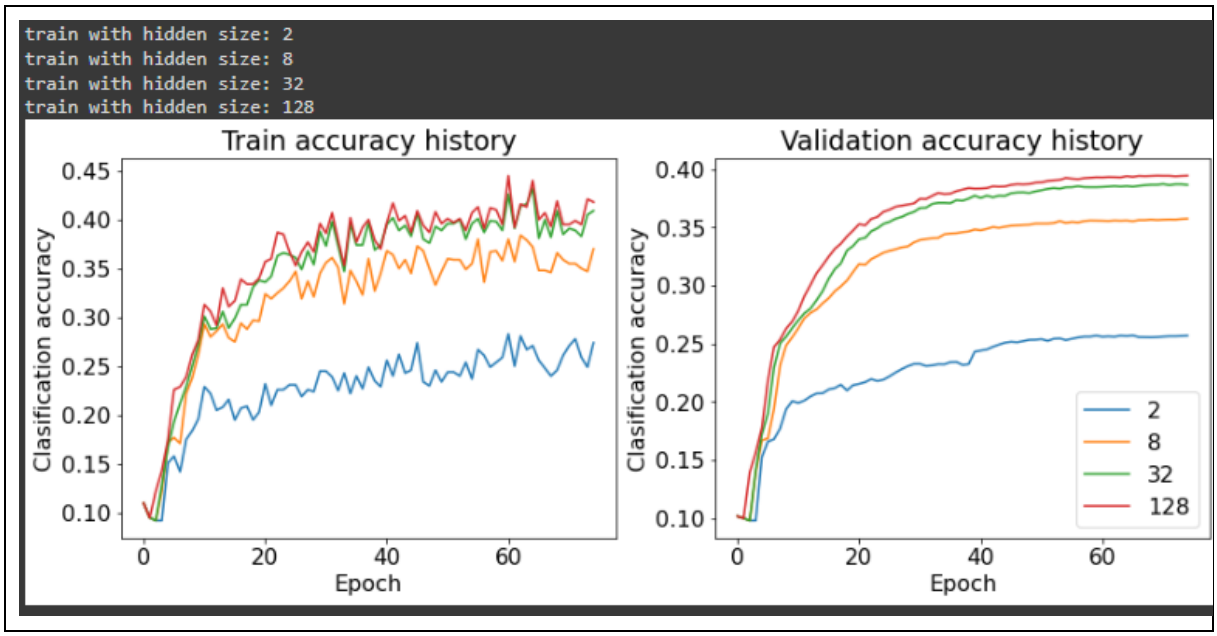
```
iteration 0 / 500: loss 2.302864
iteration 100 / 500: loss 2.302695
iteration 200 / 500: loss 2.302669
iteration 300 / 500: loss 2.302552
iteration 400 / 500: loss 2.302571
Validation accuracy: 9.77%
```



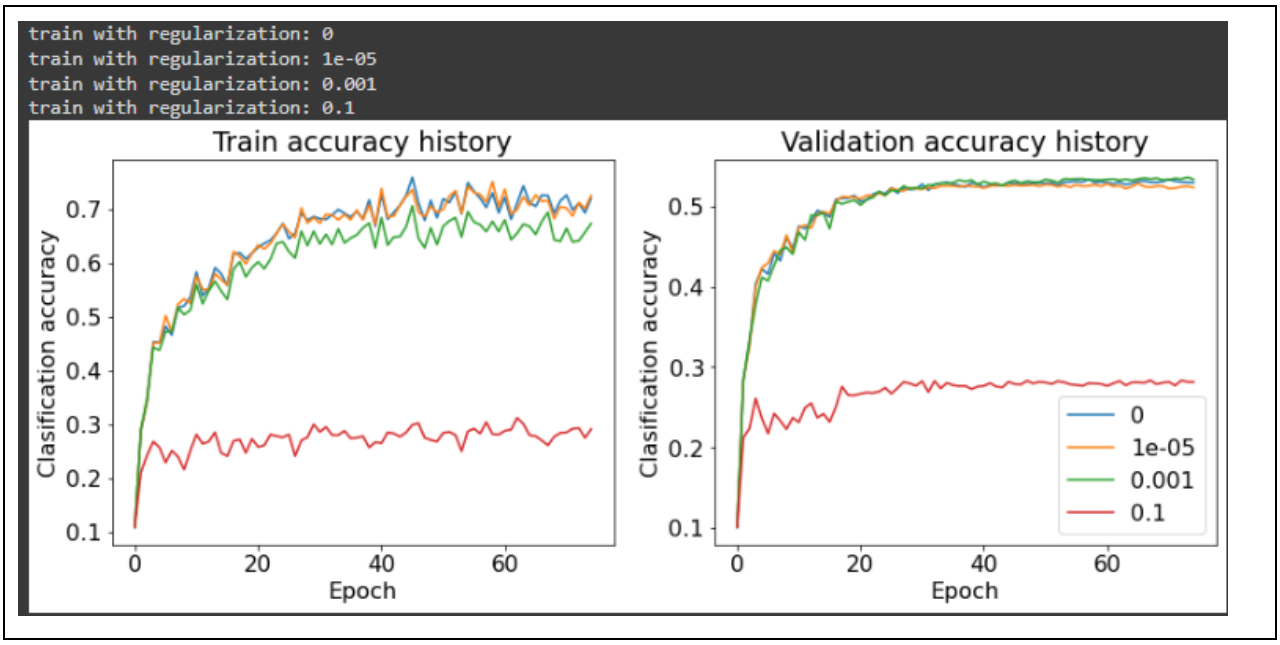
(2) 如下圖所示，把 weight 視覺化發現一堆雜訊。



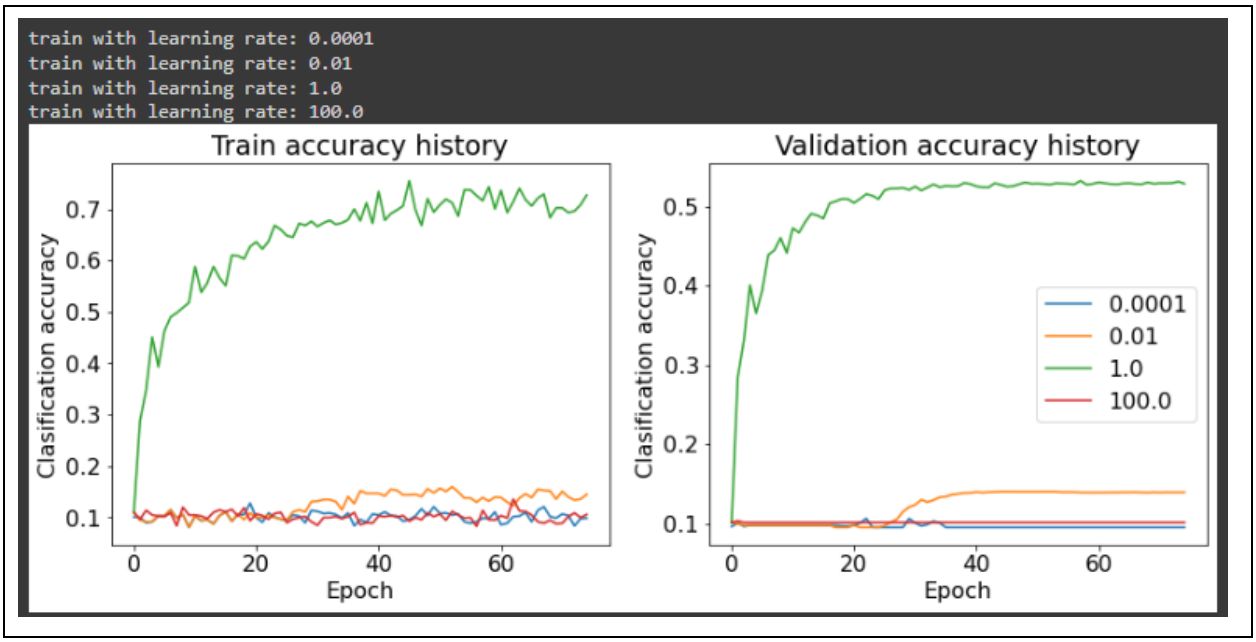
(3) 如下圖所示，當疊代次數 3000 次和 batch_size=1000 和 learning_rate=0.1 和 learning_rate_decay=0.95 和正則化強度=0.001 時，發現 hidden_size 越大準確率越高，但 train 的準確率有點搖擺不定。



(4) 如下圖所示，當疊代次數 3000 次和 batch_size=1000 和 hidden_size=128 和 learning_rate=1.0 和 learning_rate_decay=0.95 時，發現正則化強度在 0.1 時準確率很低，其它都很接近。



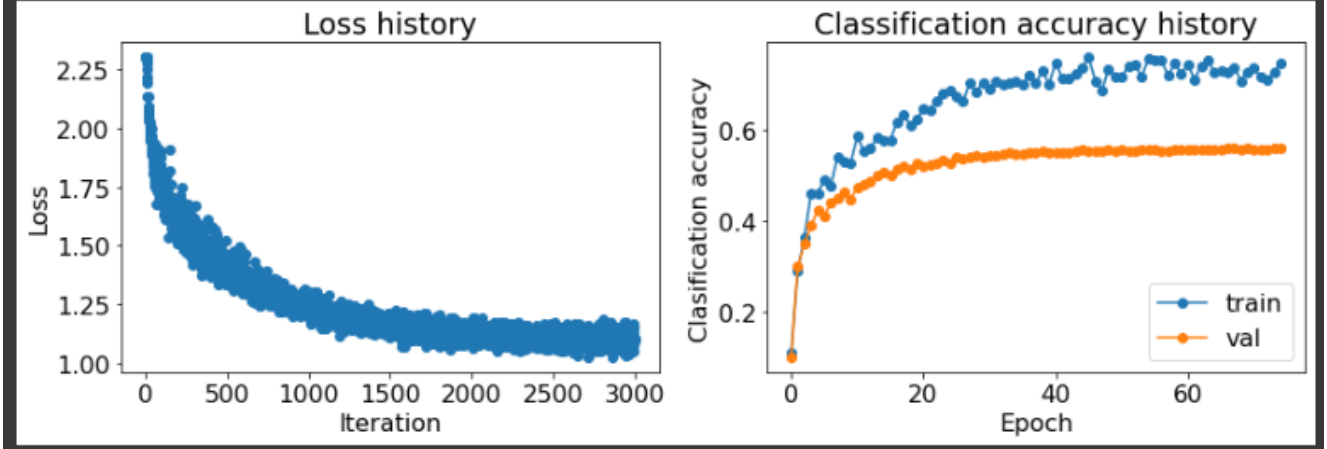
(5) 如下圖所示，當疊代次數 3000 次和 batch_size=1000 和 hidden_size=128 和 learning_rate_decay=0.95 和正則化強度=1e-4 時，發現當 learning_rate=1.0 時準確率特別高。



(6) 用不同的 learning rate 和 hidden_Size 和正則化強度和 learning_rate_decay 來找出最佳的 validation 準確率，發現當 learning rate=1.0 和 hidden_size=4096 和正則化強度=0.001 和

learning_rate_decay=0.95 時，validation 準確率為 55.68%，test 的準確率為 55.25%。

```
{lr: 1.0 ,hidden_size: 4096 ,reg: 0 ,lr_decay: 1.0 }:acc 0.1013999953866005
{lr: 1.0 ,hidden_size: 4096 ,reg: 0 ,lr_decay: 0.95 }:acc 0.5562999844551086
{lr: 1.0 ,hidden_size: 4096 ,reg: 0 ,lr_decay: 0.9 }:acc 0.5397999882698059
{lr: 1.0 ,hidden_size: 4096 ,reg: 0.001 ,lr_decay: 1.0 }:acc 0.1013999953866005
{lr: 1.0 ,hidden_size: 4096 ,reg: 0.001 ,lr_decay: 0.95 }:acc 0.5568000078201294
{lr: 1.0 ,hidden_size: 4096 ,reg: 0.001 ,lr_decay: 0.9 }:acc 0.5307999849319458
tensor(0.5568, device='cuda:0') <two_layer_net.TwoLayerNet object at 0x7f1815ac1f70> {'loss_history': [2.30271
```




```
Saved in drive/My Drive/Colab Notebooks/DL2023/HW6/HW6/nn_best_model.pt
```

```
Best val-set accuracy: 55.68%
```

```
Test accuracy: 55.25%
```

(7) 如下圖所示，把 weight 視覺化