Testing our NN on a real dataset: CIFAR-10

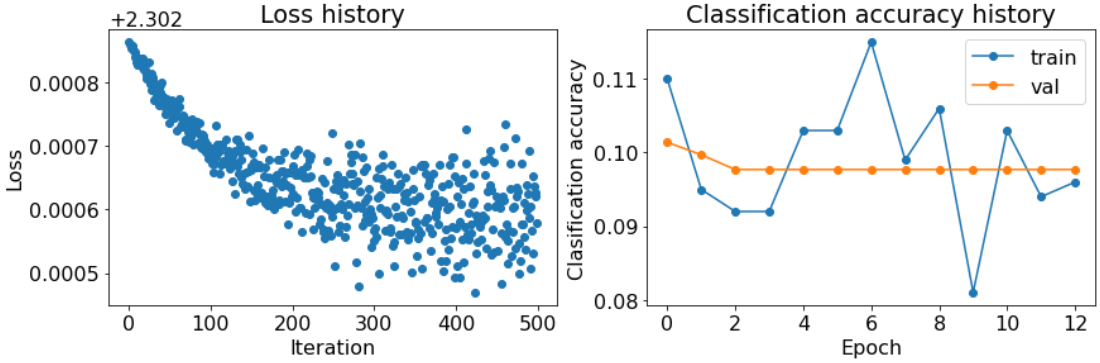
1. 使用 CIFAR10 準備 Training Dataset 和 Testing Dataset，shape 如下

```
Train data shape:  torch.Size([40000, 3072])
Train labels shape:  torch.Size([40000])
Validation data shape:  torch.Size([10000, 3072])
Validation labels shape:  torch.Size([10000])
Test data shape:  torch.Size([10000, 3072])
Test labels shape:  torch.Size([10000])
```

2. 利用 TwoLayerNet 代表我們的神經網路
3. 使用 Stochastic gradient descent 來訓練神經網路模型，一次跑一個 mini-batch 樣本，算出一次梯度或是 mini-batch 梯度的平均後就更新一次。在這我們使用預測參數

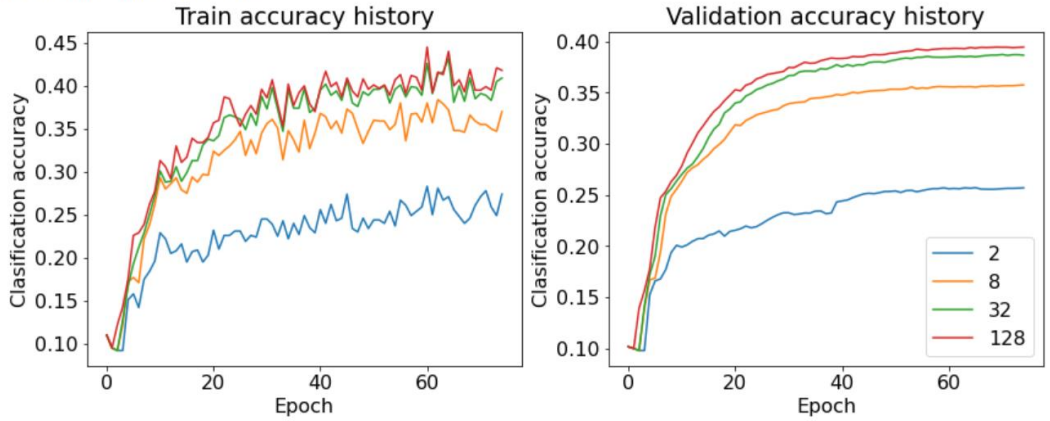
```
iteration 0 / 500: loss 2.302864
iteration 100 / 500: loss 2.302695
iteration 200 / 500: loss 2.302669
iteration 300 / 500: loss 2.302552
iteration 400 / 500: loss 2.302571
Validation accuracy: 9.77%
```

4. 由於驗證集準確率只有 9.77%，我們可以發現模型表現是非常差的。透過繪製 loss function 圖形查看錯誤可能為何，透過下圖，我們可以得知在 train 到 200 多次時，模型變得越來越不趨於線性，因此我們推斷 learning rate 可能太低了，同時 size 使用太少，應增加模型訓練樣本數。



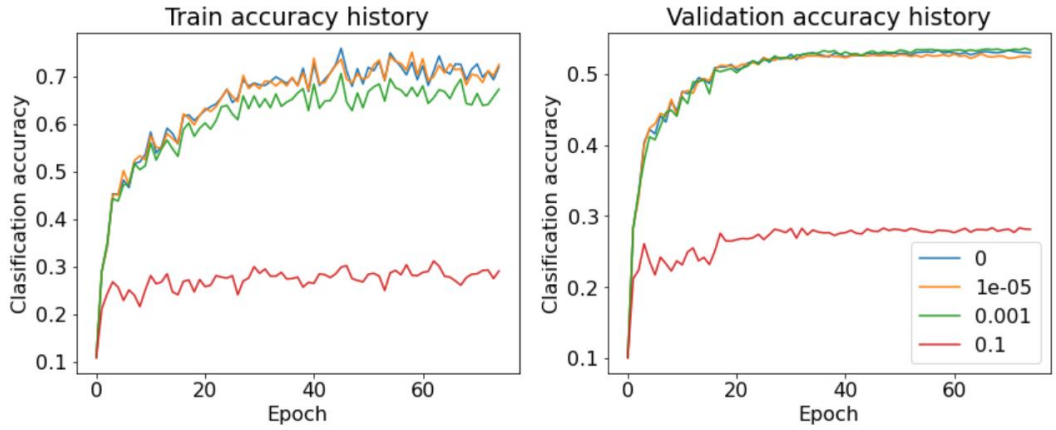
5. 透過增加 hidden layer size 來增加 capacity

```
train with hidden size: 2
train with hidden size: 8
train with hidden size: 32
train with hidden size: 128
```



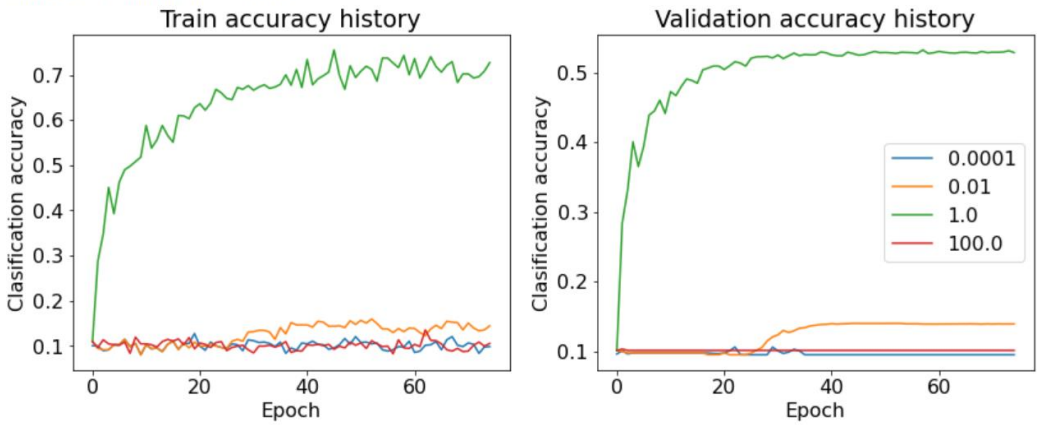
6. 另外一個避免 underfitting 的方法則是透過調整正規化的參數，由結果可知，若是將正規化參數調整過高，則也會傷害到驗證集模型的表現。

```
train with regularization: 0
train with regularization: 1e-05
train with regularization: 0.001
train with regularization: 0.1
```



7. 調整 learning rate 的大小來看相對應的模型表現變化，可了解到當 learning rate 過小時，往解跳的速率太小；太大則容易震盪太大，跳不進最佳解內。

```
train with learning rate: 0.0001
train with learning rate: 0.01
train with learning rate: 1.0
train with learning rate: 100.0
```

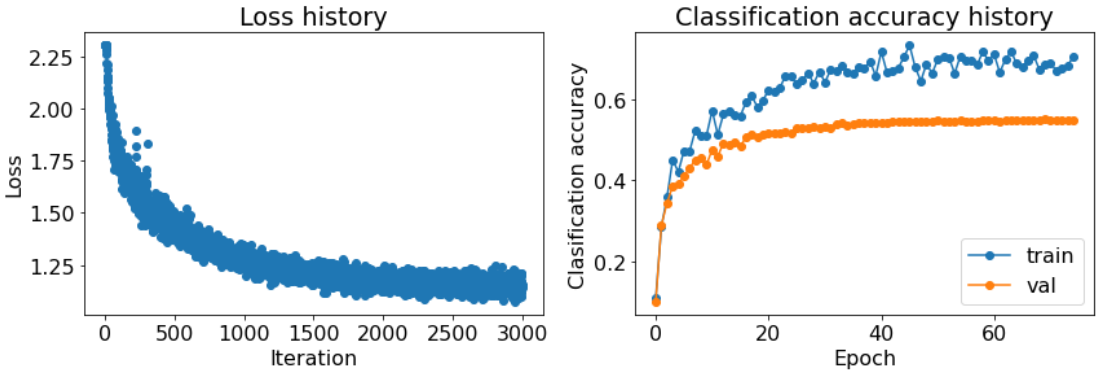


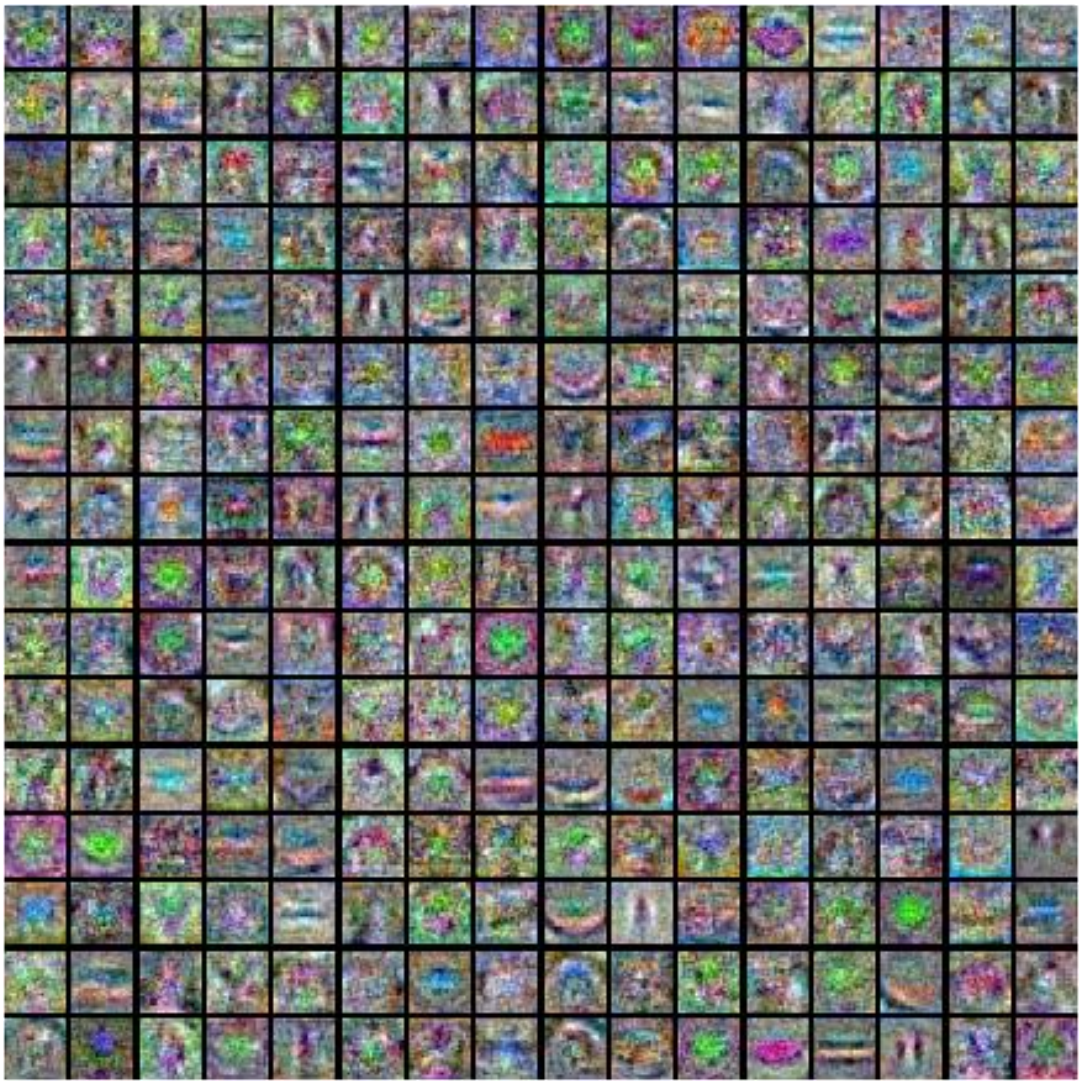
8. 將 learning rate 代入(0.1, 1.0, 2.0)
   hidden size 代入(64, 128, 256)
   regularization strength 代入(0.001, 0.0001)
   learning rate decay 代入(0.95)
   從中找出最佳神經網路參數，並且最佳驗證集表現為 55.07%

```
train with hidden_size: 64
train with learning_rate: 0.1
train with regularization: 0.001
train with hidden_size: 128
train with learning_rate: 0.1
train with regularization: 0.001
train with hidden_size: 256
train with learning_rate: 0.1
train with regularization: 0.001
train with hidden_size: 64
train with learning_rate: 1.0
train with regularization: 0.001
train with hidden_size: 128
train with learning_rate: 1.0
train with regularization: 0.001
train with hidden_size: 256
train with learning_rate: 1.0
train with regularization: 0.001
train with hidden_size: 64
train with learning_rate: 2.0
train with regularization: 0.001
train with hidden_size: 128
train with learning_rate: 2.0
train with regularization: 0.001
```

```
train with hidden_size: 256
train with learning_rate: 2.0
train with regularization: 0.001
train with hidden_size: 64
train with learning_rate: 0.1
train with regularization: 0.0001
train with hidden_size: 128
train with learning_rate: 0.1
train with regularization: 0.0001
train with hidden_size: 256
train with learning_rate: 0.1
train with regularization: 0.0001
train with hidden_size: 64
train with learning_rate: 1.0
train with regularization: 0.0001
train with hidden_size: 128
train with learning_rate: 1.0
train with regularization: 0.0001
train with hidden_size: 256
train with learning_rate: 1.0
train with regularization: 0.0001
train with hidden_size: 64
train with learning_rate: 2.0
train with regularization: 0.0001

train with hidden_size: 128
train with learning_rate: 2.0
train with regularization: 0.0001
train with hidden_size: 256
train with learning_rate: 2.0
train with regularization: 0.0001
0.5507000088691711
```





9. 將測試集丟入測試，準確率如下

```
Test accuracy: 54.50%
```