

隊長：R06922002 姚嘉昇 組員：R06942054 潘仁傑、R06522620 王仁蔚

分工表

	1-1 Part1	1-1 Part2	1-2 Part1	1-2 Part2	1-2 Part3	1-2 Bonus	1-3 Part1	1-3 Part2	1-3 Part3	1-3 Bonus	report
姚嘉昇 R06922002											
王仁蔚 R06522620											
潘仁傑 R06942054											

HW1-1 Part1 Simulate a Function

- Describe the models you use, including the number of parameters (at least two models) and the function you use. (0.5%)

以下 model 之數字是每一層 Dense 的數量

Model0 : 1→5→10→10→10→10→10→5→1 Total parameters: 571

Model1 : 1→10→18→15→4→1 Total parameters: 572

Model2 : 1→190→1 Total parameters: 571

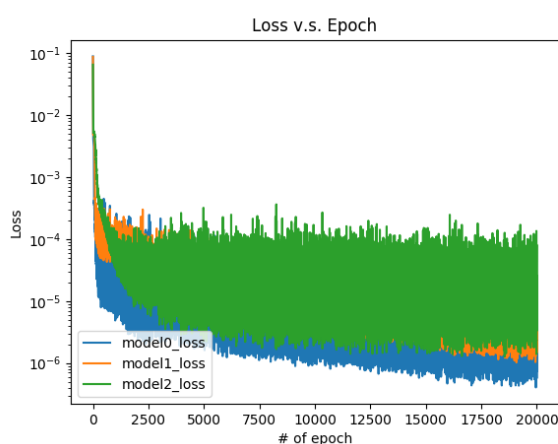
Function1 : $\sin(5 * \pi * x) / (5 * \pi * x)$

Function2 : $\text{sgn}(\sin(5 * \pi * x))$

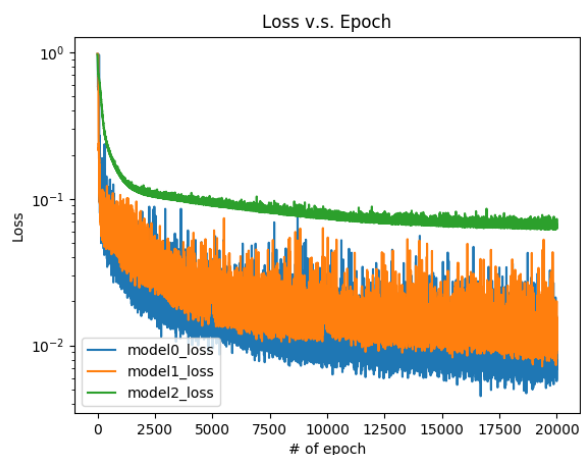
Domain : `np.linspace(0.0001, 1.0, num=10000)`

`loss='mse', optimizer='adam', batch_size = 128, epochs = 20000`

- In one chart, plot the training loss of all models. (0.5%)

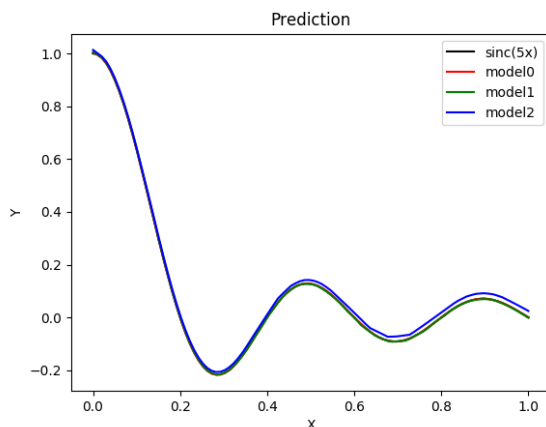


Function1

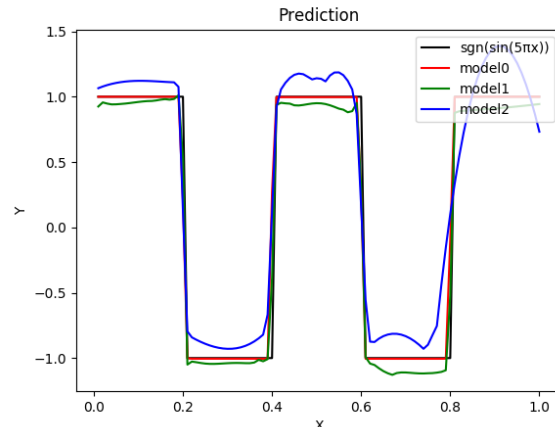


Function2

- In one graph, plot the predicted function curve of all models and the ground-truth function curve. (0.5%)



Function1



Function2

- Comment on your results. (1%)

在第 2 點的兩張圖中，我們可以發現，雖然在訓練的過程中三個 model 的 loss 曲線會震盪，但 model0 的震盪的 loss 最低值都比起其他兩者來的低，model2 震盪的 loss 最高值在 2500 個 epoch 後都會比其他兩者來的高。

而在第 3 點左邊的圖中，三個 model 都非常接近 $\text{sinc}(5x)$ 這個函數，且在第 2 點左邊的圖中，三者的 loss 都在 10 的 $-4 \sim -5$ 次方中，因此我們認為這三種 model 都有足夠的能力學習這個函數。

第 3 點右邊的圖中，最深的 model0 完全與目標函數重疊，loss 最低值在 10 的 -2 次方左右；最淺的 model2 與目標函數看起來相差甚遠，且其 loss 比 model0 高了十倍，因此在這個函數的學習上，我們觀察到同參數但較深的 model 有更好的表現。

- Use more than two models in all previous questions. (bonus 0.25%)

如第 1 點，我們使用了三個 model

- Use more than one function. (bonus 0.25%)

如第 1 點，我們使用了兩個 function

HW1-1 Part2 Train on Actual Tasks

- Describe the models you use and the task you chose. (0.5%)

CIFAR10

Model0: CNN(49, (5, 5)) + Dense(512) + Dense(10) paramters:1,238,678

Model1: CNN(49, (3, 3)) + CNN(49, (3, 3)) + Dense(512) + Dense(10) paramters:1,257,984

Model2: CNN(32, (3, 3)) + CNN(32, (3, 3)) + CNN(64, (3, 3)) + CNN(64, (3, 3)) + Dense(512) + Dense(10) paramters:1,250,858

batch_size = 32, epochs = 100, optimizer = rmsprop(lr=0.0001, decay=1e-6)

loss = 'categorical_crossentropy'

MNIST

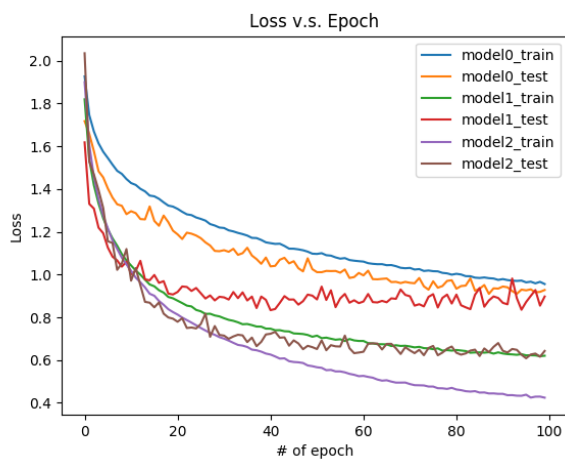
Model0: CNN(17, (3, 3)) + desne(10) parameters: 133,460

Model1: CNN(12, (5, 5)) + CNN(16, (5, 5)) + desne(10) parameters: 130,578

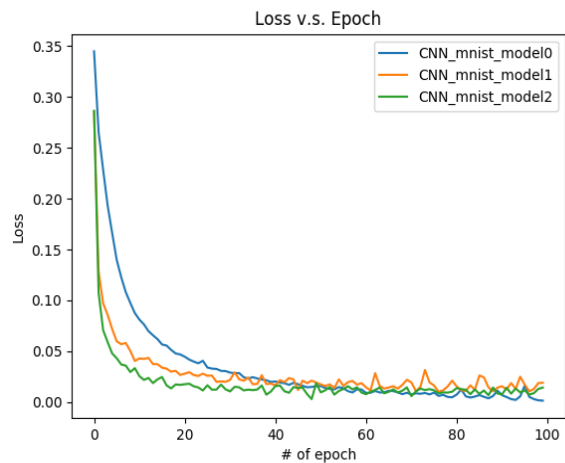
Model2 CNN(8, (3, 3)) + CNN(12, (3, 3)) + CNN(16, (3, 3)) + desne(10) parameters: 128,150

batch_size = 128, epochs = 100, activation='selu', padding='SAME'
loss = 'categorical_crossentropy', optimizer = 'adam'

- In one chart, plot the training loss of all models. (0.5%)

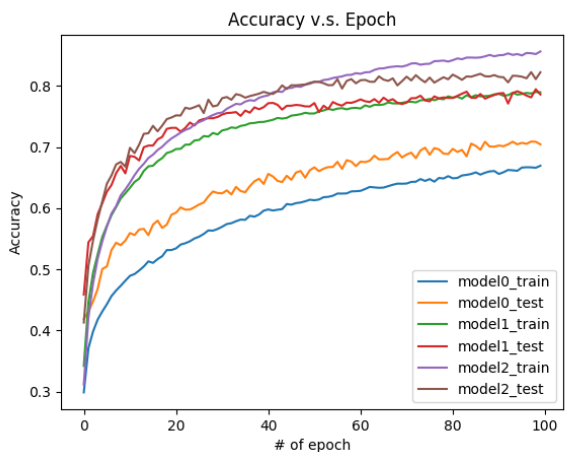


CIFAR10

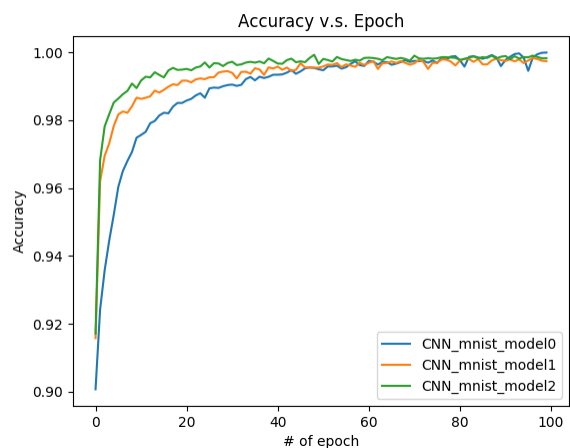


MNIST

- In one chart, plot the training accuracy. (0.5%)



CIFAR10



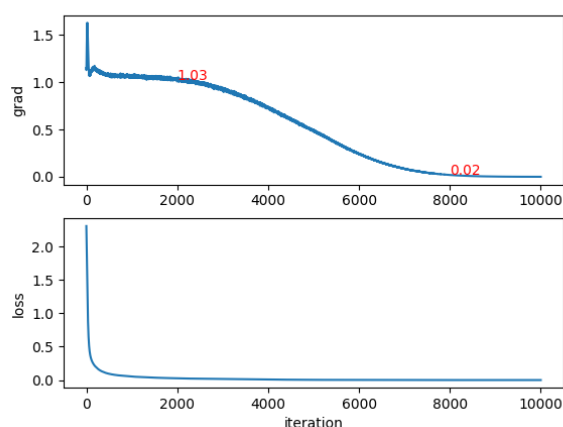
MNIST

- Comment on your results. (1%)

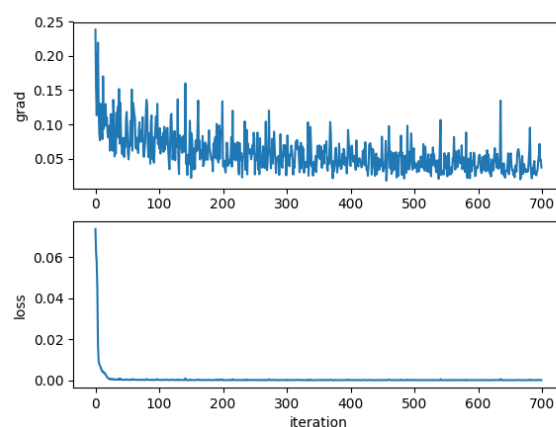
在第2、3點的左圖。我們可以觀察到 loss 排序不論是 training 或 testing 都是最深的 model2 最，最寬胖的 model0 最高；acc 也是最深的表現最好，最胖的表現最差，因此在 CIFAR10 上，比較深的 model 表現較寬的 model 好。

HW1-2 Part 2 Observe gradient norm during training.

1. Plot one figure which contain gradient norm to iterations and the loss to iterations. (1%)



MNIST



SINC

2. Comment your result. (1%)

根據 gradient norm to iterations and the loss to iterations 兩張圖，我們可以觀察 Loss 及 iteration 的圖，訓練後期的 loss 都已經收斂至 0，按照 gradient decent 的概念，直觀會認為 gradient 應該已經收斂到 0，也就是 local minimum，但我們拉出 grad 並作圖的話，我們發現其實 gradient 並不會等於 0，左圖是下降，直到 8000 個 epoch 後才趨近 0，而右圖則是持續地在 0.05~0.15 震盪。

HW1-2 Part 3 What happens when gradient is almost zero?

1. State how you get the weight which gradient norm is zero and how you define the minimal ratio. (2%)

設定：

Task : $Y = \text{sinc}(5 \cdot X)$

Domain : $X = \text{np.linspace}(0.0001, 1.0, \text{num}=5000)$

Model : Dense(1) + Dense(5) + Dense(10) + Dense(10) + Dense(5) + Dense(1)

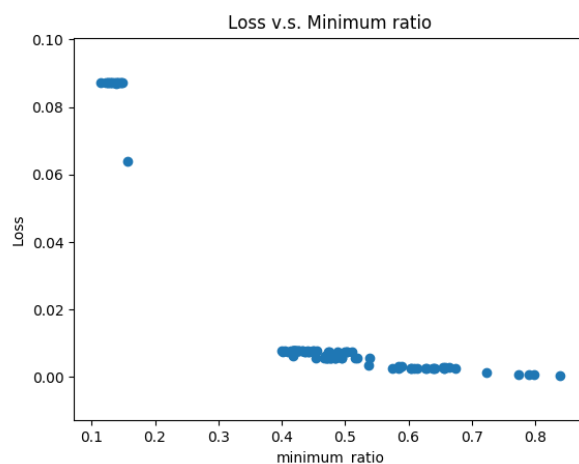
訓練方式：loss_func 用 MSE 訓練 64 個 epoch 後，將 loss_func 改成

epoch_grad_g 繼續訓練 1024 個 epoch，若 epoch_grad_g < 5e-3，則 early stop

Weight 偏移方式：每個 weight 隨機加上一個 -1~+1 的 random 值

Minimal ratio 定義：偏移後的 loss 比原本高的個數除上總個數

2. Train the model for 100 times. Plot the figure of minimal ratio to the loss. (2%)



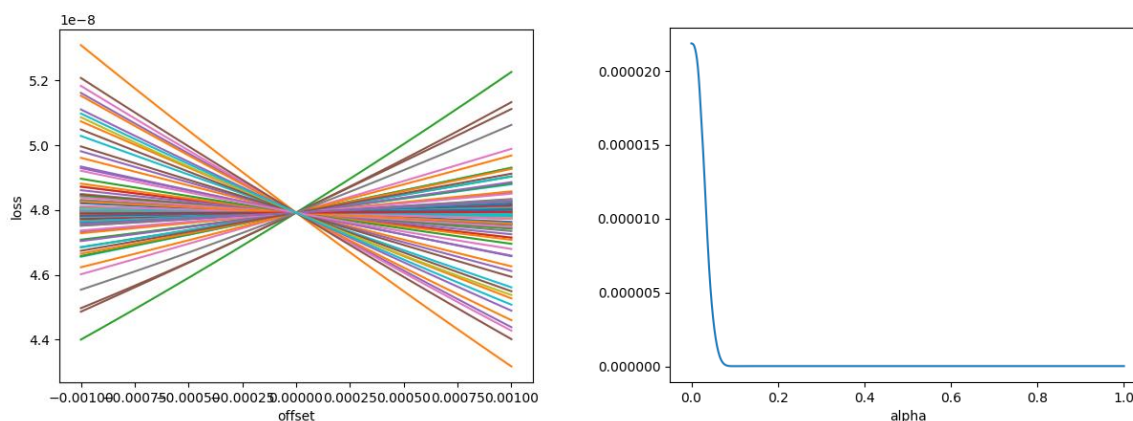
3. Comment your result. (1%)

根據上圖，loss 在 0.09 附近時(圖左上方)，minimal_ratio 約莫在 0.1 左右，也就是說，附近的點的 loss 比 0.09 大的比例只有一成；同理 loss 在 0 附近時(圖右下方)，minimal_ratio 約 0.7 到 0.85 之間，比例約八成。

我們的實驗結果顯示當 loss 越低時，minimum_ratio 越高，也就是 ratio 越高越靠近 local minimal。minimal ratio 越低則代表這個點有可能是 saddle point，所以 loss 才會偏高。

HW1-2 Bonus (1%)

1. Use any method to visualize the error surface.



2. Concretely describe your method and comment your result.

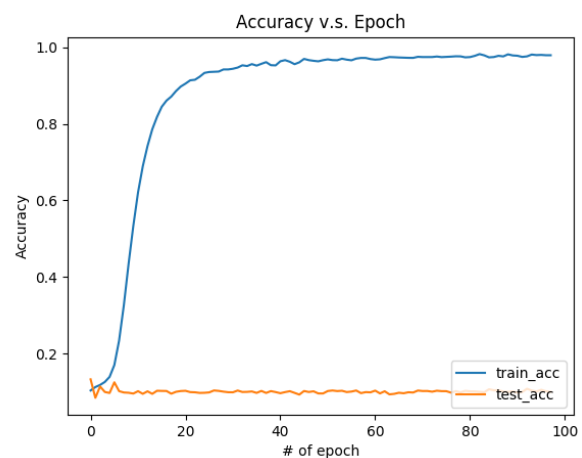
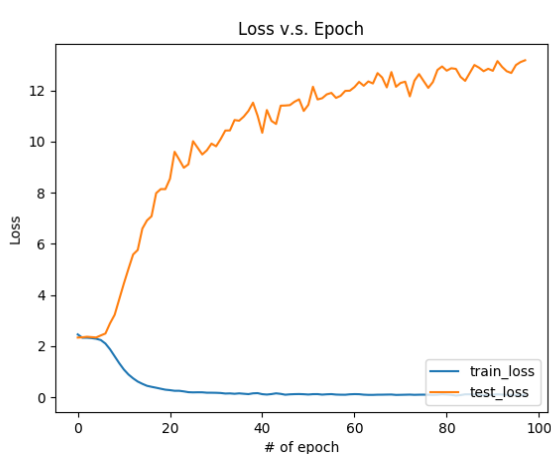
我們使用 HW1-2 Part 3 的 model 架構，並嘗試了兩種 visualize the error surface 的方法：

- 第一種(上左圖)：抽取訓練過的 model weights，依次對每一個參數加上 offset $-0.001 \sim 0.001$ ，每一個參數得到一條 loss 對其參數做 offset 的直線。根據此圖，我們可以發現各個參數加上正 offset 後，loss 有的上升，有的卻下降，可以推判 offset=0 的 loss 可能是落在 saddle point，若是落在 local minimum，不管在任何參數上做微量的 offset，loss 應當都會上升。

- 第二種(上右圖)：將最後的 model weights 跟初始的 model weights 做插值， $\alpha=0$ 時是初始的 model weights， $\alpha=1$ 時是 train 完的 model weights。也就是說，在高維空間中，初始的 weights 會筆直地往 final 的 weights 前進，我們將圖 zoom in 後，一直都是遞減而沒有像助教投影片中會震盪，也許是我們初始的 weight 剛好落在一個平滑的曲面，其 gradient 恰好朝向 final weights 的方向，也許是我們的 model 太過簡易，loss 曲線不夠複雜。

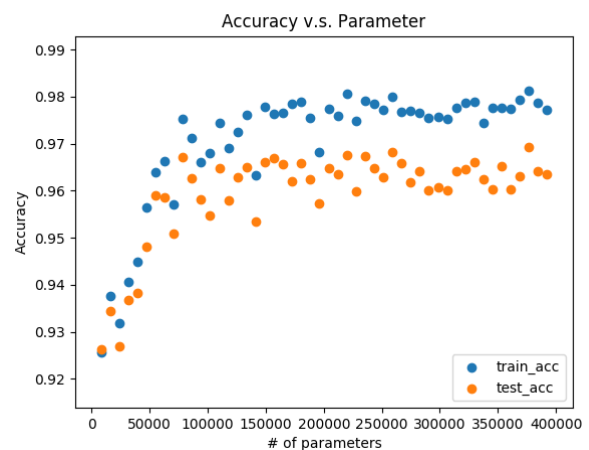
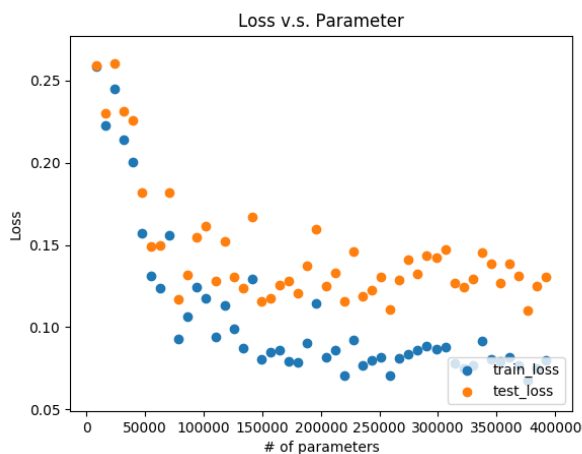
HW1-3 Part 1 Can network fit random variables?

1. Describe your settings of the experiments. (e.g. which task, learning rate, optimizer) (1%)
MNIST
batch_size = 128
optimizer=Adam(lr=0.001)
Activation('selu')
Model: CNN(16, (3, 3))+ CNN(16, (3, 3)) + CNN(32, (3, 3)) + Dense(512) + Dense(10)
parameters:1,238,678
2. Plot the figure of the relationship between training and testing, loss and epochs. (1%)



HW1-3 Part 2 Number of parameters V.S. Generalization

1. Describe your settings of the experiments. (e.g. which task, the 10 or more structures you choose) (1%)
MNIST、batch_size = 1024、epochs = 20
50 個 model：每個 model 都有一層 CNN，且 kernel_size=(3, 3)，但第 i 個 model 的 CNN 使用 i + 1 個 filter，每個 model 最後接 Dense(10)
loss='categorical_crossentropy', optimizer='adam'
2. Plot the figures of both training and testing, loss and accuracy to the number of parameters. (1%)



3. Comment your result. (1%)

根據上圖，當 parameters 越多時，不論 train or test 的 loss 就逐漸下降，超過 50000 之後，就沒有下降趨勢。當 parameter 數超過五萬，可以判定 model 的學習能力已經能夠處理 MNIST 這個 dataset 的問題，也沒有因為 parameter 太多而 Overfitting。

HW1-3 Part 3 Flatness V.S. Generalization

1. Part 1

1.1 Describe the settings of the experiments (e.g. which task, what training approaches) (0.5%)

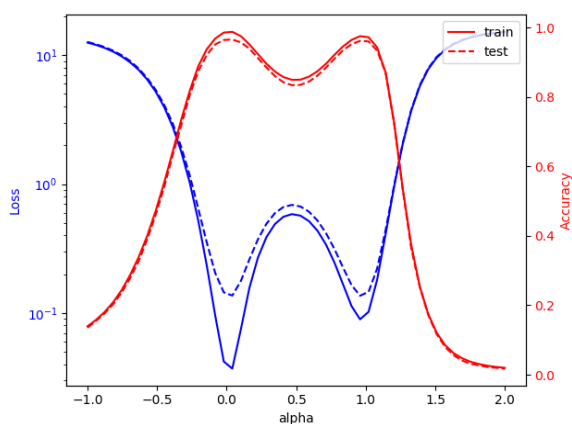
使用 MNIST，架構為 CNN(16, (3,3), selu) + Dense(10)

所有 model 都使用 `loss='categorical_crossentropy'`，`optimizer='adam'`

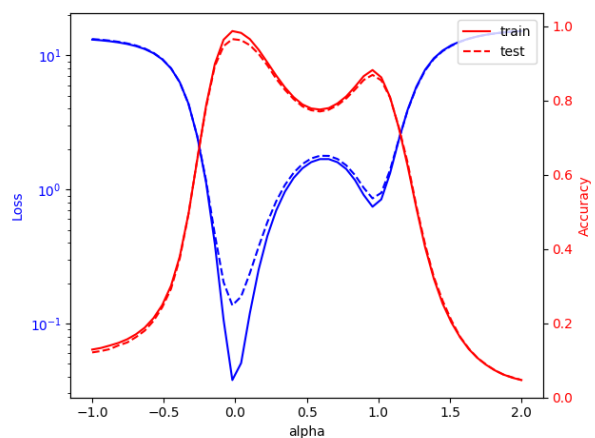
Batch Size = 64 v.s. 1024 各一個 model，並使用插指法繪圖

Learning Rate = $1e-3$ vs. $1e-2$ 各一個 model，並使用插指法繪圖

1.2 Plot the figures of both training and testing, loss and accuracy to the number of interpolation ratio. (1%)



Batch Size = 64 v.s. 1024



Learning Rate = $1e-3$ vs. $1e-2$

1.3 Comment your result. (1%)

上左圖在 $\alpha=0$ 時是 Small Batch Size, $\alpha=1$ 是 Large Batch Size, 上右圖在 $\alpha=0$ 時是 Small Learning Rate, $\alpha=1$ 是 Large Learning Rate。

以左圖為例, 在 training 時小的 Batch Size 的 loss 比起大的 Batch Size 來的低, 但並沒有向老師上課所說小的 Batch Size 會比較寬, 大的 Batch Size 會比較尖, 但其 testing 的 loss, 比起 training 的 loss 都來的高, 如同老師上課所說, 可能是 testing 的 loss 曲線會跟 loss 相像但會在某個維度有些微偏移。

2. Part 2:

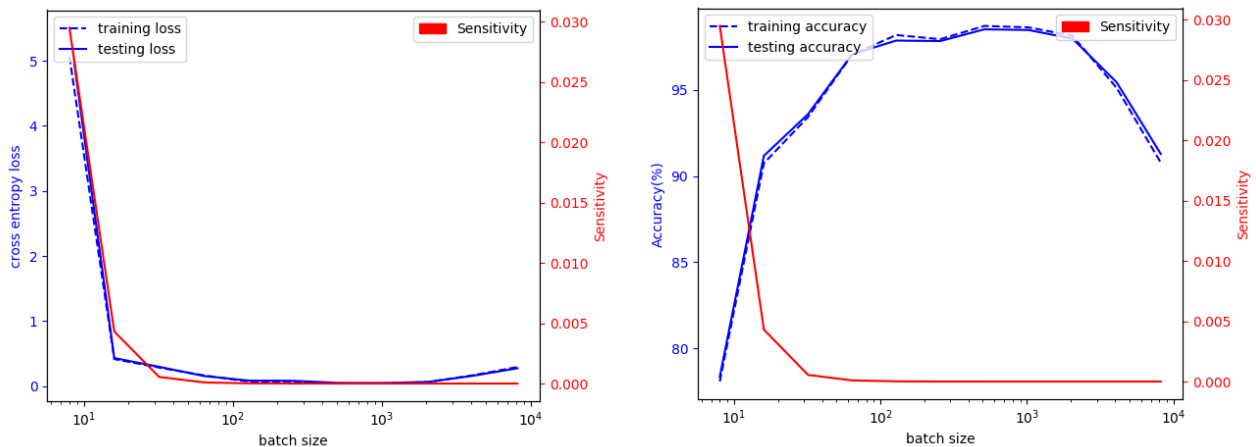
1.1 Describe the settings of the experiments (e.g. which task, what training approaches) (0.5%)

TASK : MNIST and CIFAR10

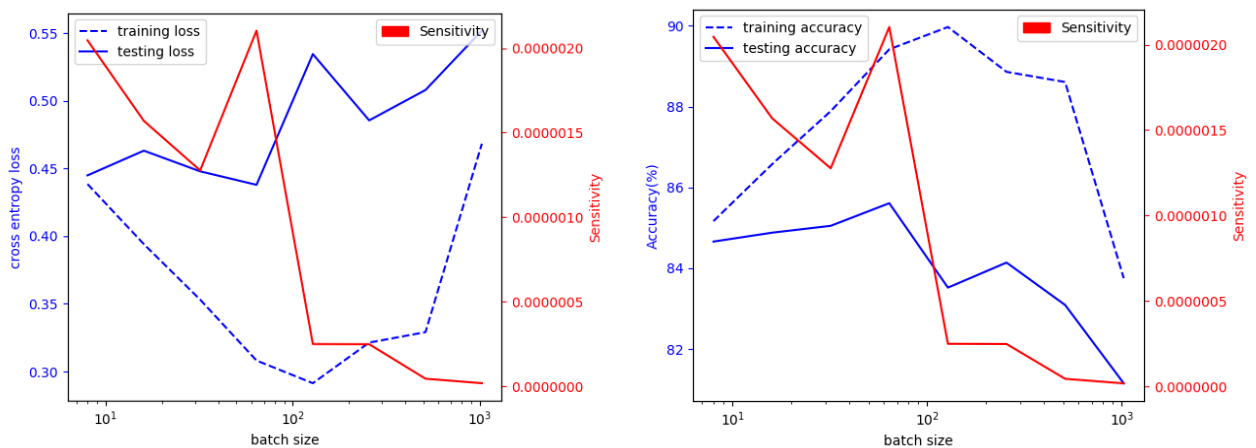
sensitivity 定義 : Frobenius norm of gradients of loss to input

1.2 Plot the figures of both training and testing, loss and accuracy, sensitivity to your chosen variable. (1%)

MNIST



CIFAR-10



1.3 Comment your result. (1%)

以上四張圖可以看出，當 Batch Size 越來越大時，sensitivity 會趨近 0，也就是說這個 network 對 data 的敏感度越來越低，也可以觀察在 MNIST 及 CIFAR10 上當 batch Size 大到一定程度時 Acc 會開始降低。敏感度低意味著 model 較易受資料偏移影響，因此其 generalization 能力較強，理當在 training acc 和 testing acc 間的差距小，不過在 cifar-10 的圖上並沒有這樣表現，推測可能我們計算的是 loss 對 input 而非 output 對 input 所產生的敏感度。

3. Bonus : Use other metrics or methods to evaluate a model's ability to generalize and concretely describe it and comment your results.

我們計算 Sharpness 的方法是同老師投影片的 Definition2，在 train 完的 weights 附近加上 random offset，這些點最高的 Loss 與原本 Loss 相減即是我們的 sharpness，若 sharpness 越大則該 model 之 weight 落在較尖陡的 local minimum。

每個 model epoch 都是 20，model 架構為一層 CNN+Dense。在每個 model 的 weight 附近採樣 1000 個樣本點，範圍是 $\epsilon = 1e-4$ 。下圖為 MNIST 上計算 loss&sharpness v.s. batch_size 以及 accuracy and sharpness v.s. batch_size，從圖可明顯發現，當 batch size 越大時，sharpness 值就越大，代表收斂到非常陡峭的 local minimum，但 batch size 在 10~1000 的範圍內 sharpness 明顯震盪，推測可能是因為偏移量太大或是，weight 的每個維度都有 random offset，造成 random offset 過的 weight 跳出 local minimum 所在之位置。

