2023 第 2 學期 深度學習實驗 LAB01 Back-Propagation

學號: 411551022

姓名: 謝宇恆

1 Introduction

本實驗利用簡單的 linear neural network 去將兩群不同的點分群，並利用 back-propagation 調整每一個 neural unit 的權重。Figure 1 為兩個要解的問題分別是 Linear Problem 以及 XOR Problem。

Back-propagation 為調整 neural network 中每一個 neural unit function 的方法。Back-propagation 的演算法可分為兩部分: Propagation 與 Update weight。Propagation 可再分為三步: 正向傳播，Loss 計算以及反向傳播。正向傳播是將 input 輸入至網路中一層一層往下傳遞直到 output 層產生 Prediction 的資料。Loss 計算是將原始資料的 Label (Ground Truth)以及 Prediction 出來的資料進行運算，主要是為了觀察神經網路預測的資料與原始資料的差距。最後是反向傳播: 將 Loss 進行微分，從 output 層傳回去第一層的 hidden layer，計算出每一層 hidden layer 對於 Loss 的梯度，並依據該梯度調整該層的權重。

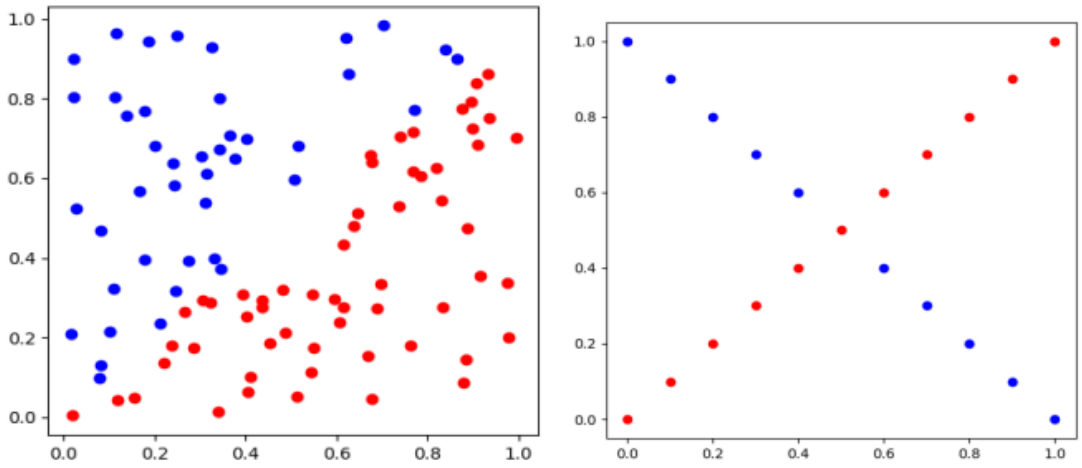

Figure 1. Linear Problem and XOR Problem

2 Experiments

2.1 Sigmoid function:

Sigmoid function 可以讓神經網路的輸出產生非線性的變化，讓神經網路可以應對更多不同種的情況。Sigmoid function 產生出來的結果如 Figure 2 所示。在每一個 neural unit 最後產生的結果加上一個 Sigmoid 的 activate function 就能將產生的結果限制在 0 與 1 之間。

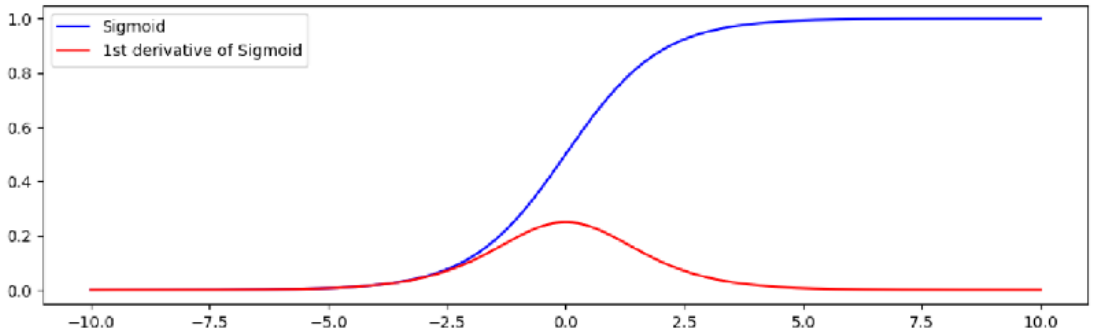

Figure 2. Sigmoid function

2.2 Neural Network

為了建構一個分類器，本實驗建立了一個 Linear Neural Network，整個實驗的網路架構圖如 Figure 3 所示。Input Layer 有兩個點，分別會對應到輸入資料 x 以及 y 座標。在 Hidden Layer 以及 Output layer 中的每一個 unit 都包含著兩個 unit，第一個是 neural unit 第二個是 activation function unit。實驗中的 Neural Unit 使用的是 unbiased function。Activation function 則是使用 sigmoid function。實驗中的 Learning rate 則是設定為 0.1。Loss function 我們使用的是 Root Mean Square Error (RMSE)。

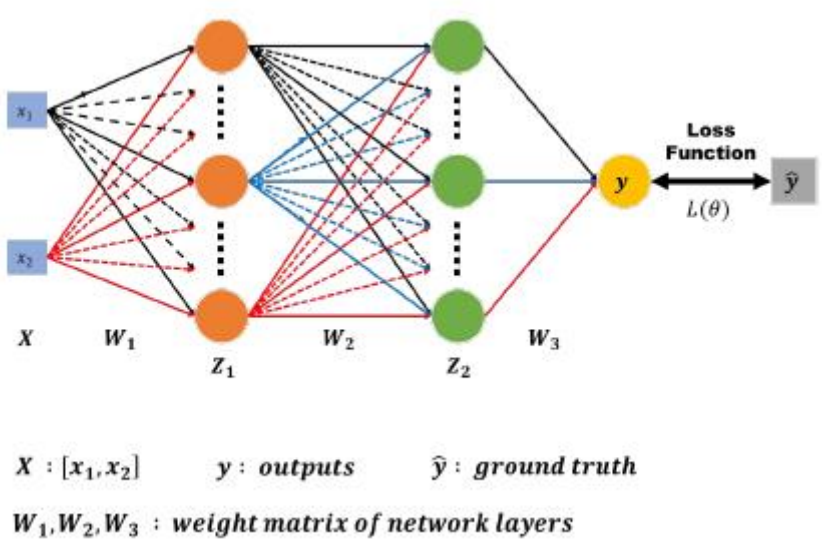

Figure 3. Neural Network Architecture

2.3 Back Propagation

本實驗的 Back Propagation 會將 Loss function 反向傳播至第一層的 hidden layer 以調整每一層的權重。我使用最簡單的方式將 Loss 對每一層的 weight matrix 進行偏微分並調整該層的 weight，方程式如下式。

$$(\hat{y} - y)^2 = \frac{\partial}{\partial w_n}(\hat{y} - y)^2, n \in (0,1,2) \quad (1)$$

3 Experiment Results

3.1 Screen shot and comparison results

Figure 4 和 Figure 5 為實驗的兩個 datasets 產生的圖以及 Predict 出來的 Results。

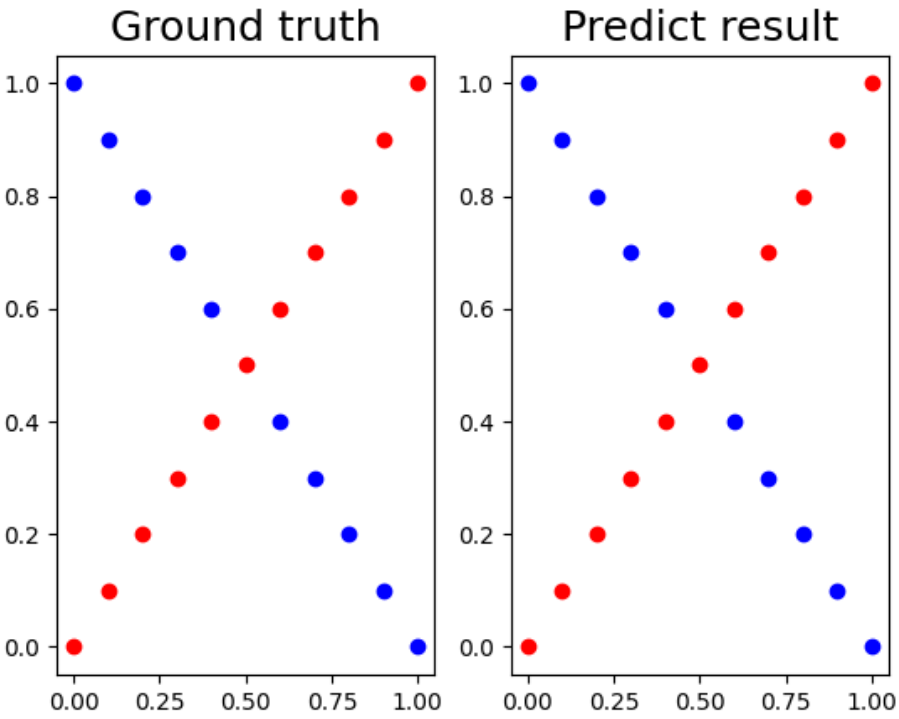

Figure 4. XOR Ground Truth and Predict Results

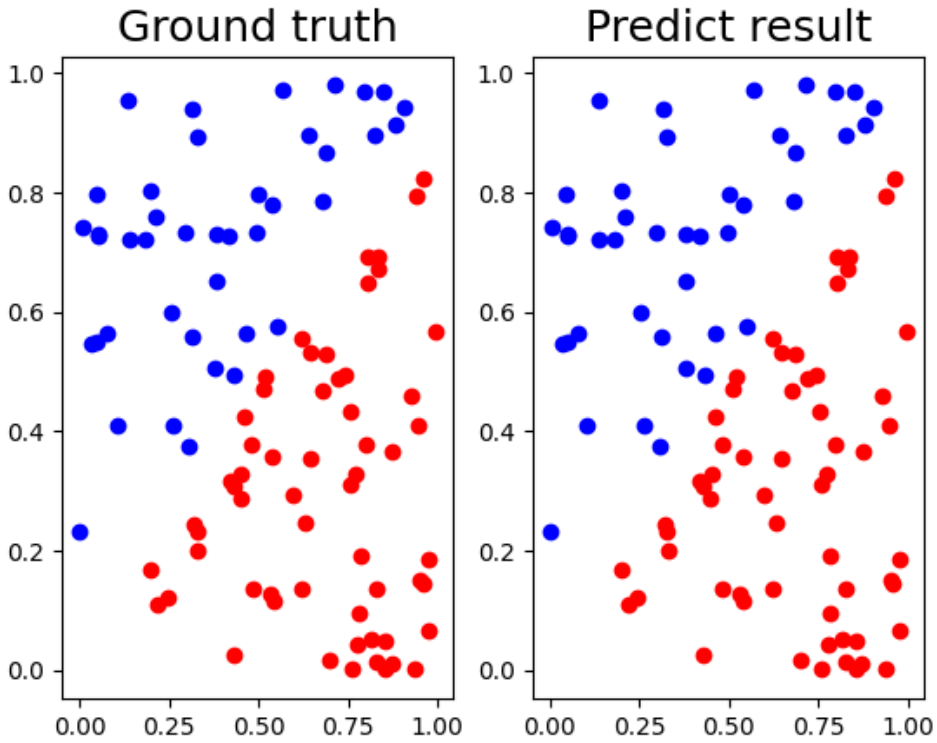

Figure 5 linear Ground Truth and Prediction Results

3.2 Accuracy of your Prediction:

本實驗進行了各 20 次，20 次的正確性平均在分別使用 XOR 以及

linear 的 data 時，都可以達到 99%。

3.3 Learning curve (loss, epoch curve)

兩種 data 的 loss 收斂情況分別顯示在 Figure 6 與 Figure 7。

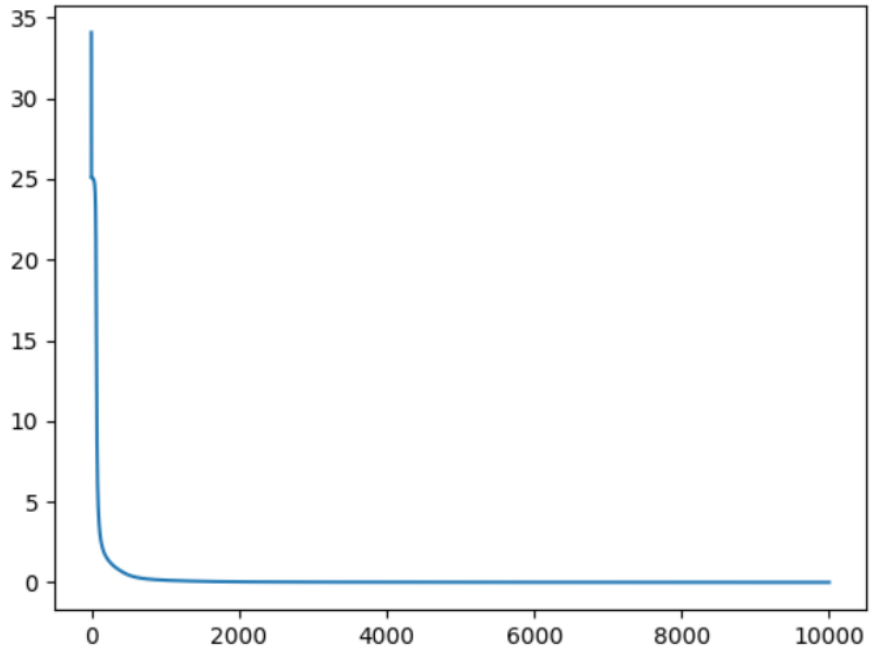

Figure 6. linear loss

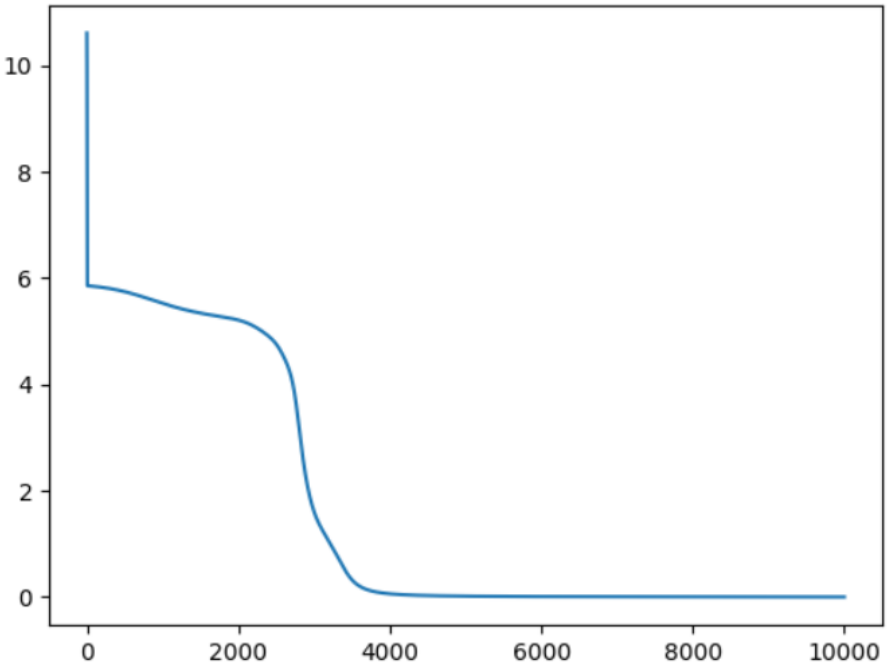

Figure 7. XOR loss

3.4 Anything you want to present

1. 不同的模型架構:

最一開始使用一層 hidden layer 的網路去測試本次實驗的資料，雖然 loss 收斂情形與兩層的 hidden layer 差不多，但是有時候

一層 Layer 的網路無法將資料有效的分群。

2. Input Data 以及 Weight Matrix 初始值的設定:

Input Data 以及 Weight Matrix 與 Loss 收斂的速度有相關，有時候一開始 loss 較低，反而無法有效的收斂，但是有時候一開始 loss 較高卻可以迅速的下降並收斂。

4 Discussion

4.1 Try Different learning rate

在這個實驗中，將會取三次的平均值作為報告值。

4.1.1 Learning Rate=1

Table 1. Learning Rate =1, the test accuracy and test loss.

	XOR	Linear
Test Accuracy	70.63%	99%
Test Loss	70.5%	0.1%

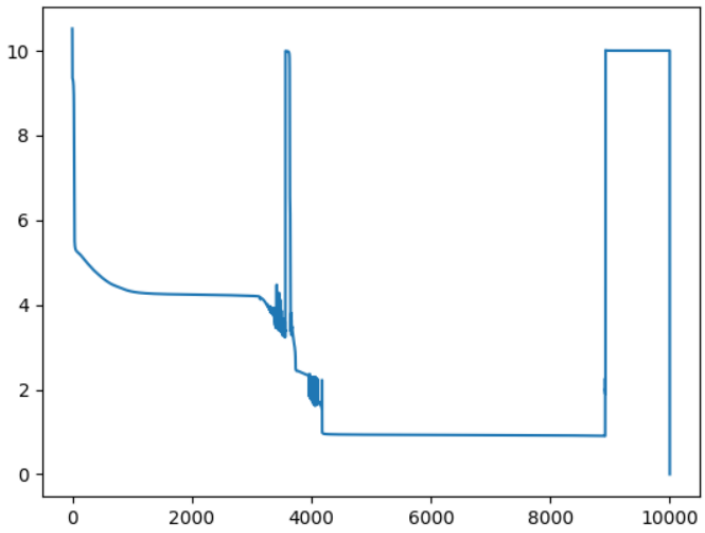

Figure 8. learning rate = 1, with XOR datasets

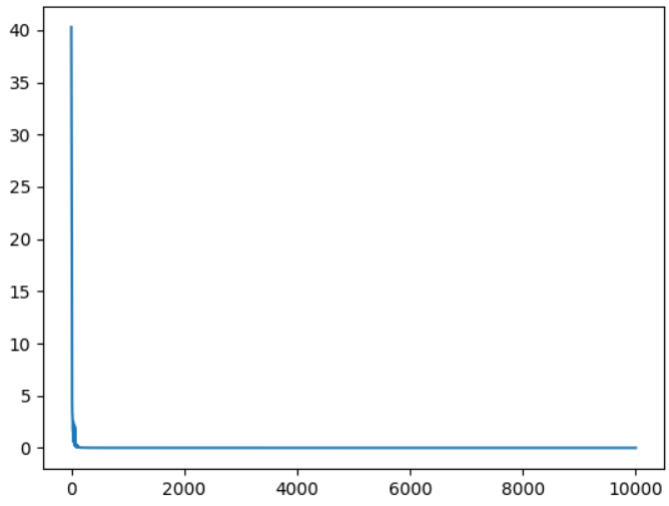

Figure 9. learning rate = 1, linear loss

4.1.2 Learning rate = 1e-2

Table 2 Learning Rate =1, the test accuracy and test loss.

	XOR	Linear
Test Accuracy	0	92
Test Loss	100	0.07

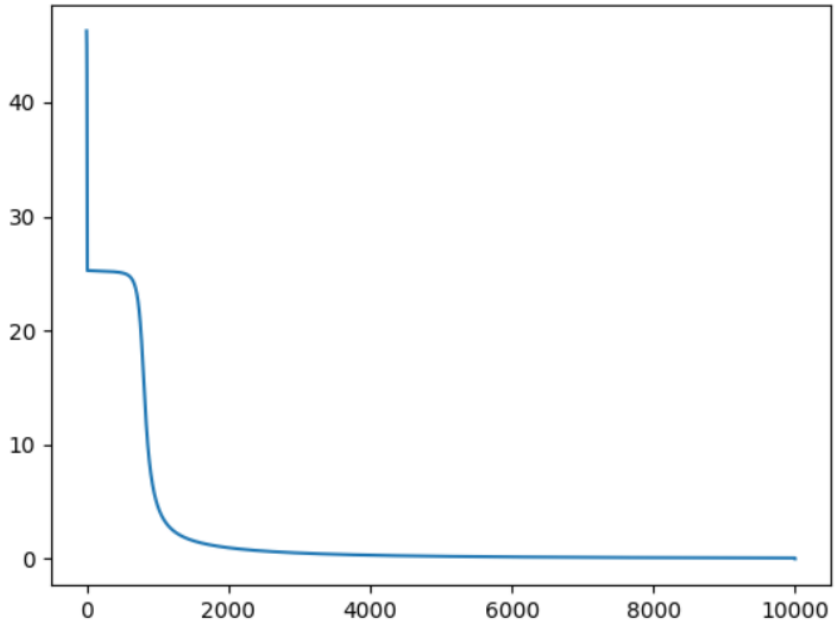

Figure 10. learning Rate=1e-2, linear loss

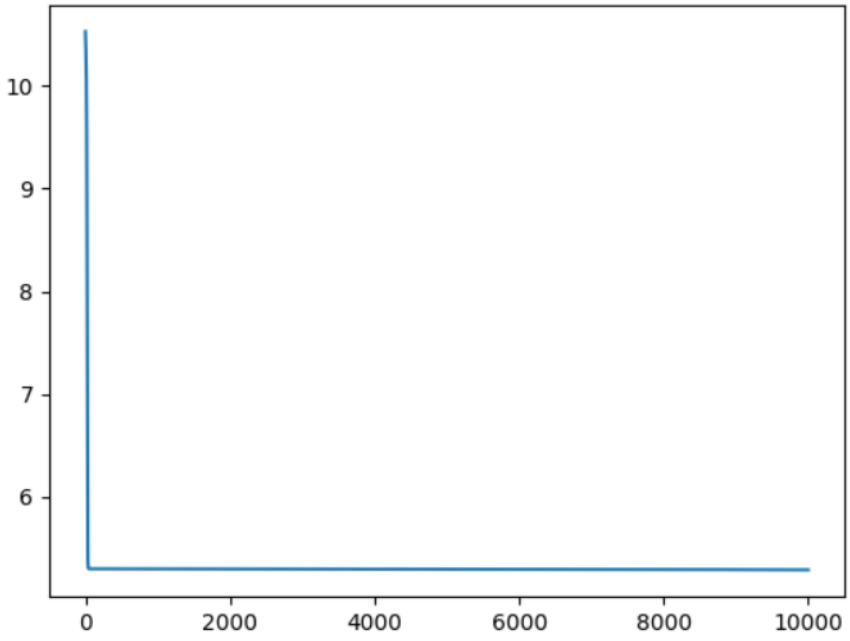

Figure 11. learning Rate=1e-2, XOR loss

在嘗試不同的 Learning Rate 後可以發現，當 Learning Rate 太大或太小都有

可能造成 loss 無法收斂，因此設定一個合適的 Learning Rate 是必須的。

4.2 Try Different number of units

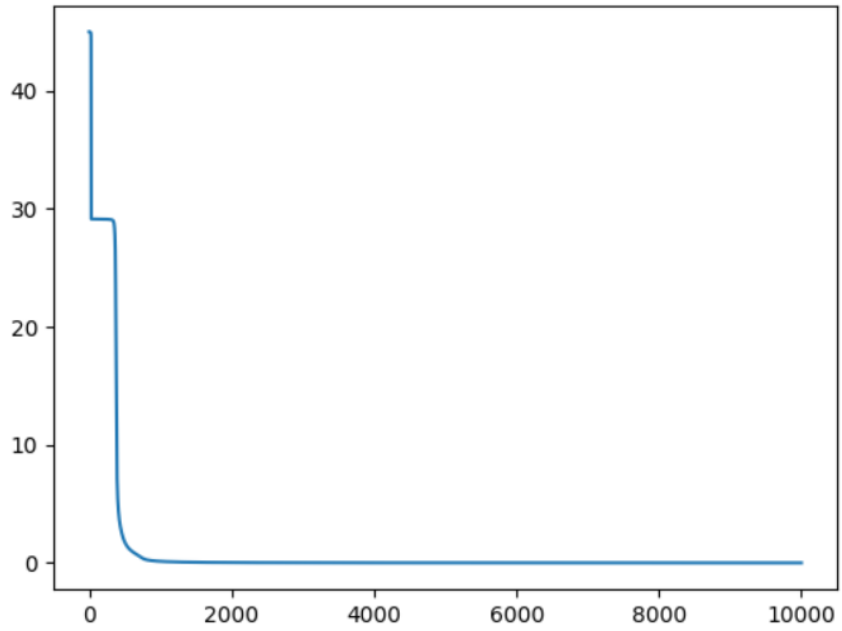

Figure 12. 20 units per 1 hidden layer, loss, Linear datasets

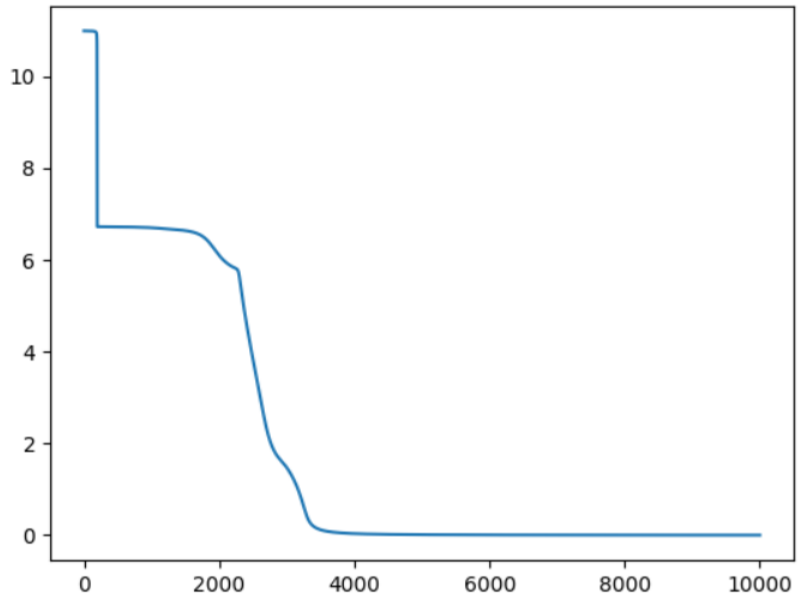

Figure 13. 20 units per 1 hidden layer, loss, XOR datasets

本實驗嘗試不同數量的 units 去觀察 loss 以及 Accuracy，由於網路的目的較為簡單所以多數量的 Neural Units 跟少數量的 Neural Units 的結果不會差到太多，只是更多數量的 Neural Units 會增加計算時間。

4.3 Try without activation functions

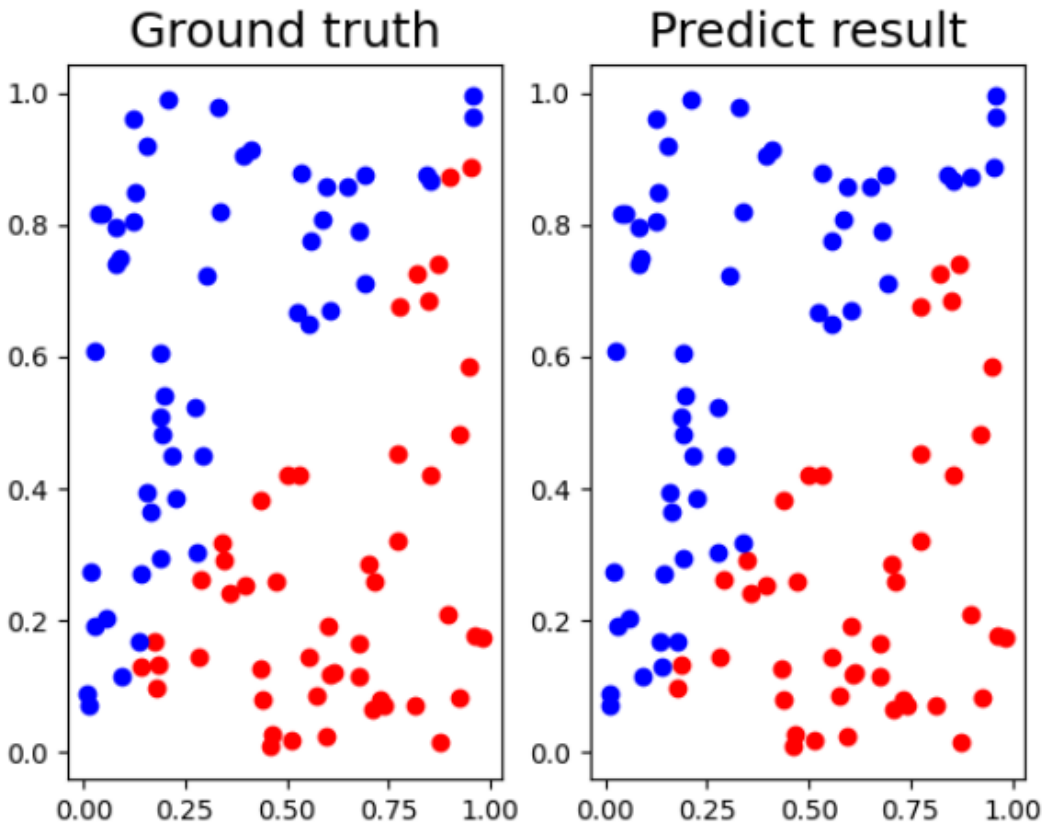

Figure 14. w/o activation function in linear problem

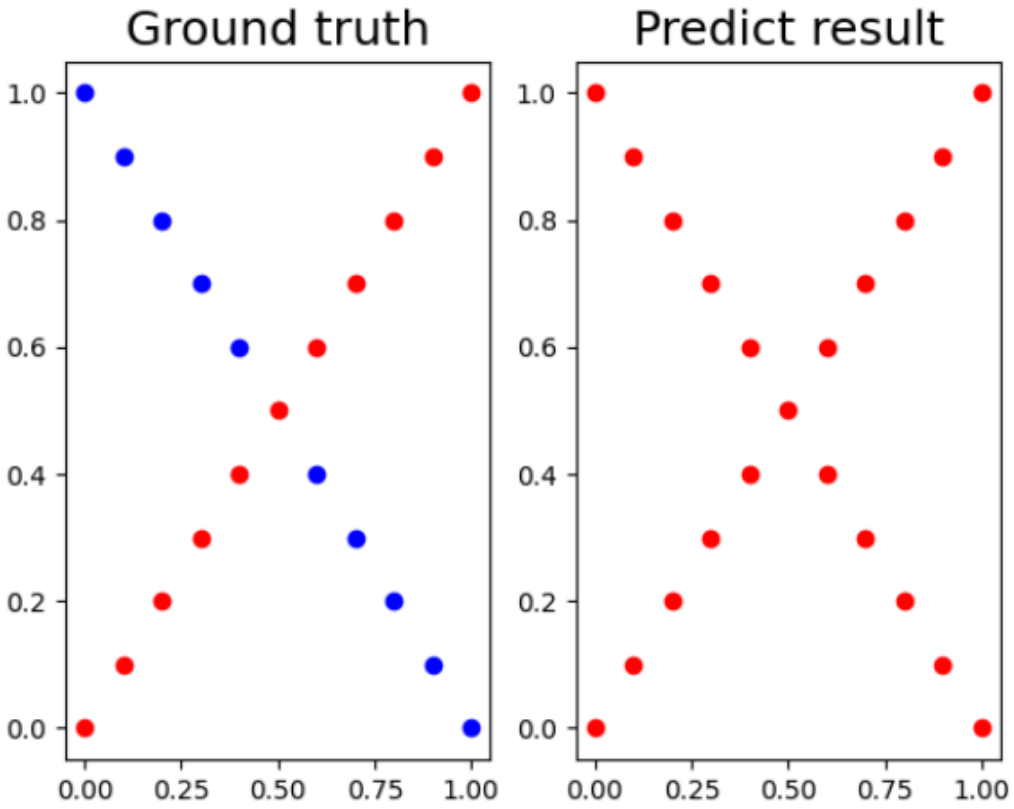

Figure 15. w/o activation in XOR Problem

Activation function 主要是為了讓 output 產生非線性的變化，雖然在 Figure 14 中可以觀察到沒有 activation function 依然可以分群，但是分群的準確性較低，Figure 15 則無法分群，因此 activation function 依據要解的問題可選擇要不要加入到 Neural network 中。