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。

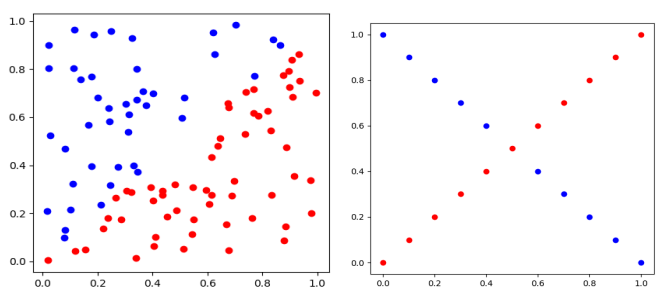


Figure 1. Linear Problem and XOR Problem

1. Experiments
   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。

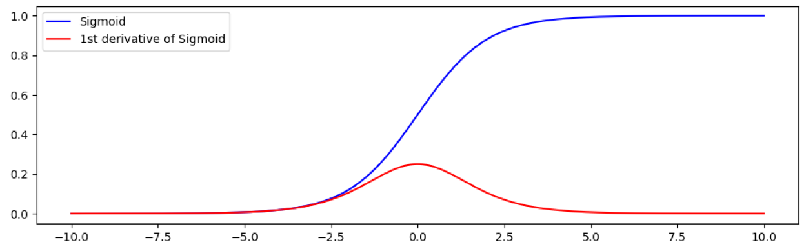


Figure 2. Sigmoid function

* 1. 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)。

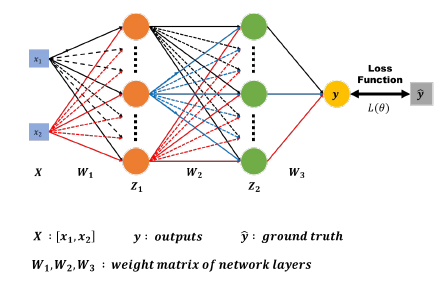


Figure 3. Neural Network Architecture

* 1. 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。

(1)

1. Experiment Results
   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。

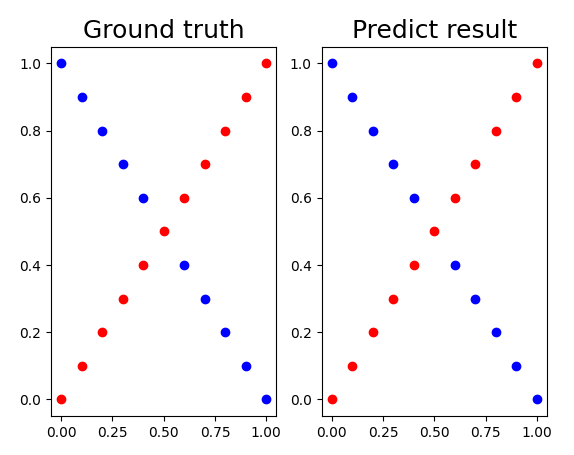


Figure 4. XOR Ground Truth and Predict Results

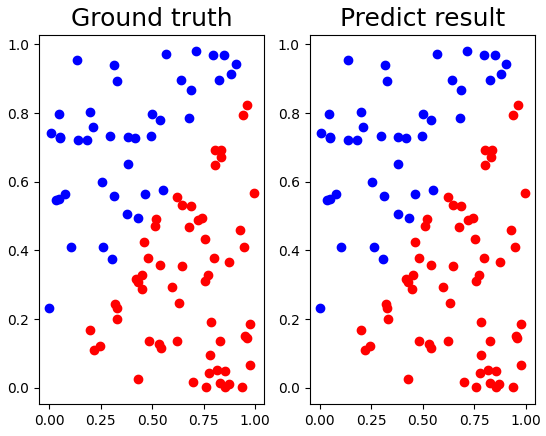


Figure 5 linear Ground Truth and Prediction Results

* 1. 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%。

* 1. Learning curve (loss, epoch curve)

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

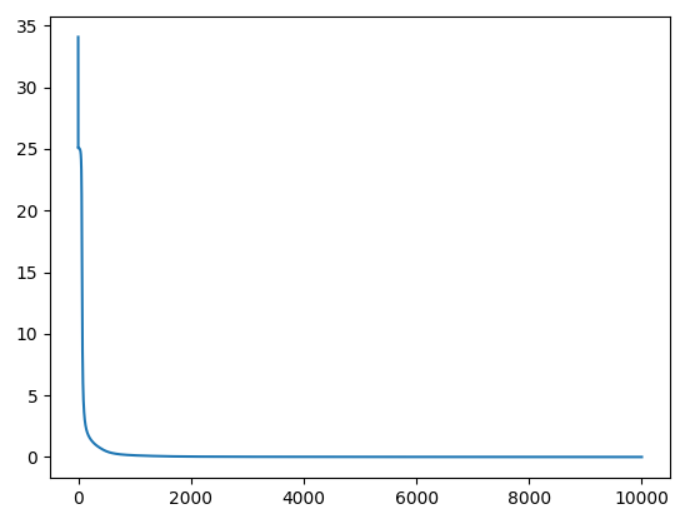


Figure 6. linear loss

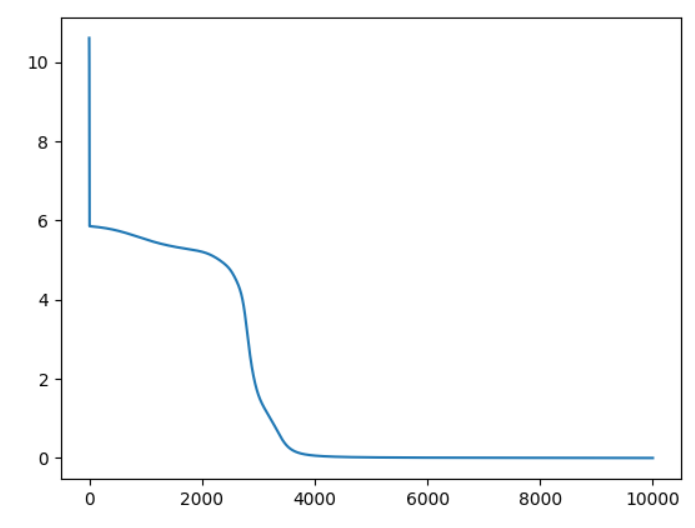


Figure 7. XOR loss

* 1. 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。

1. 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。

1. Discussion
   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。

* + 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% |

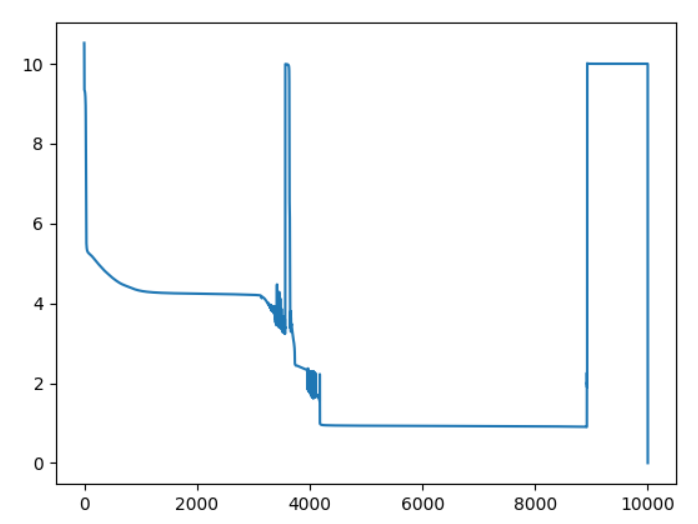


Figure 8. learning rate = 1, with XOR datasets

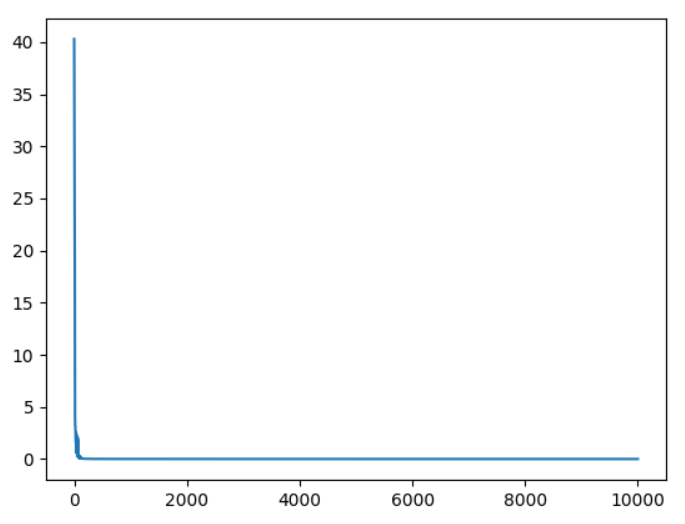


Figure 9. learning rate = 1, linear loss

* + 1. 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 |

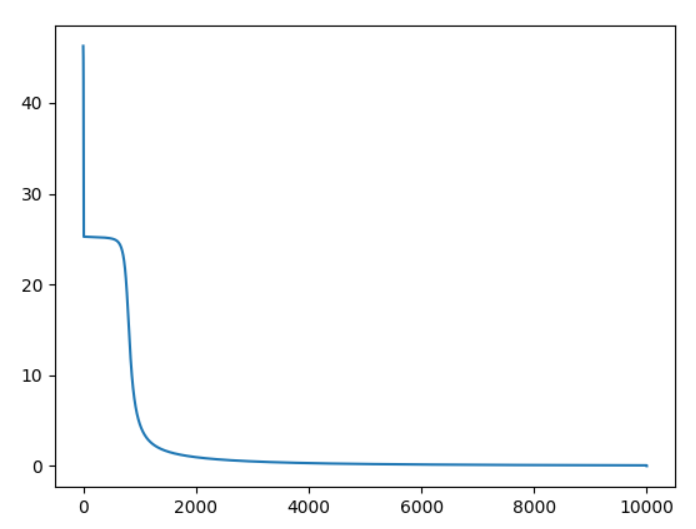


Figure 10. learning Rate=1e-2, linear loss

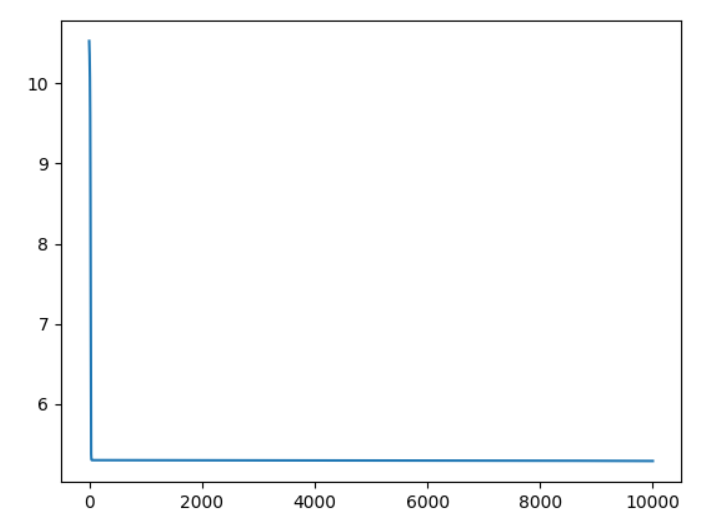


Figure 11. learning Rate=1e-2, XOR loss

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。

* 1. Try Different number of units

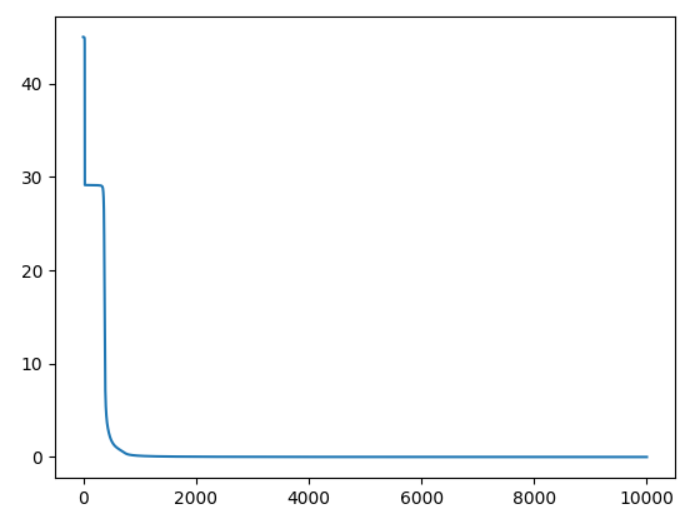


Figure 12. 20 units per 1 hidden layer, loss, Linear datasets

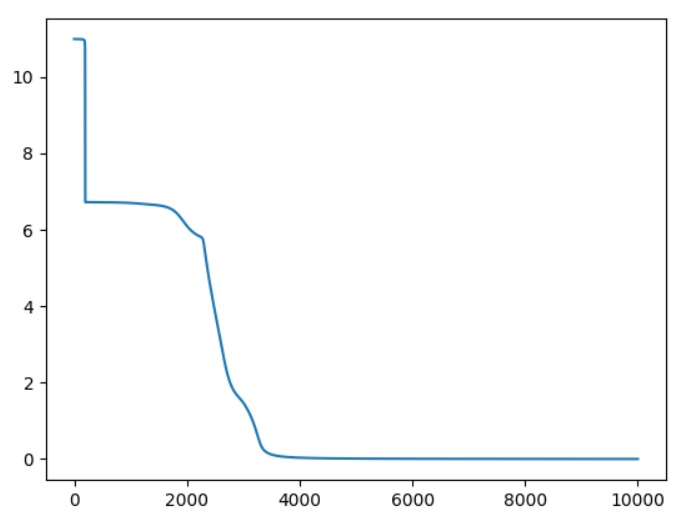


Figure 13. 20 units per 1 hidden layer, loss, XOR datasets

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。

* 1. Try without activation functions

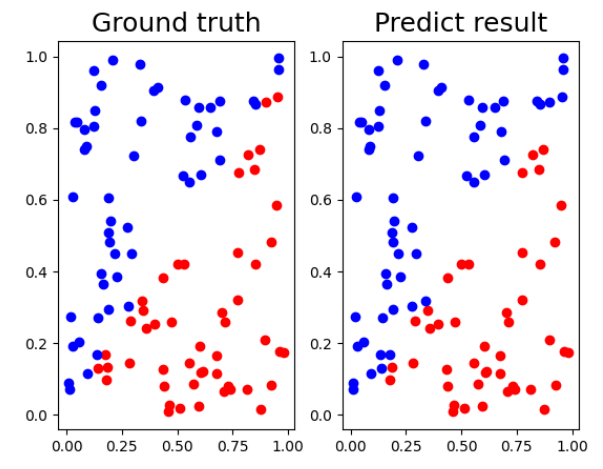


Figure 14. w/o activation function in linear problem

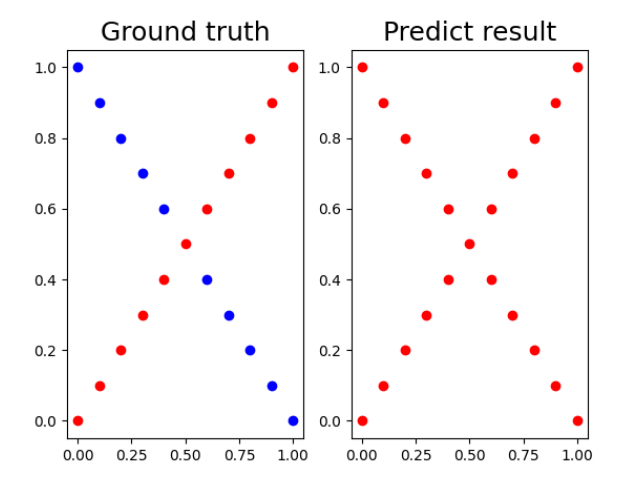


Figure 15. w/o activation in XOR Problem

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。