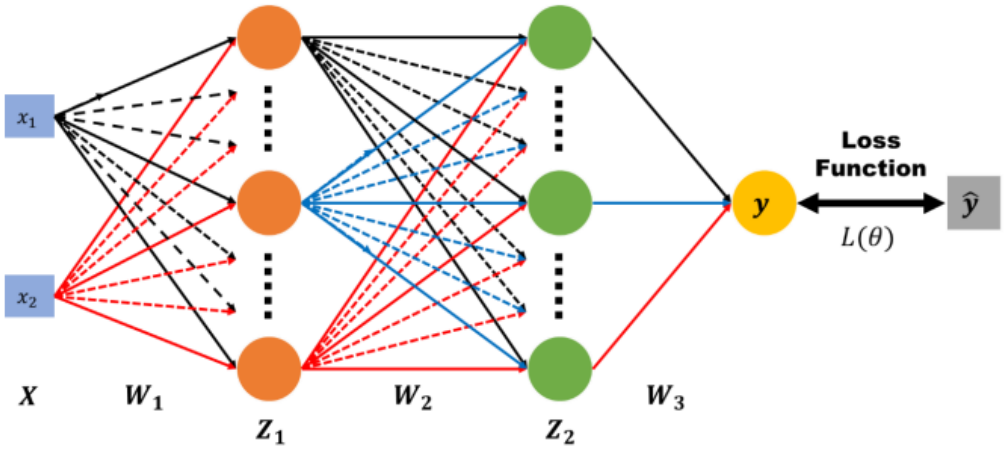
Lab1

中興　黃聖祐　L122502

March 20, 2024

1 Introduction

這是一個二分類問題，input 是二維的向量，透過兩層 hidden layers 的神經網路，得到一維的 output。我們所使用的 activation function 是 sigmoid function，loss function 用 binary cross-entropy loss，並用 Gradient Decent 作爲 optimizer。



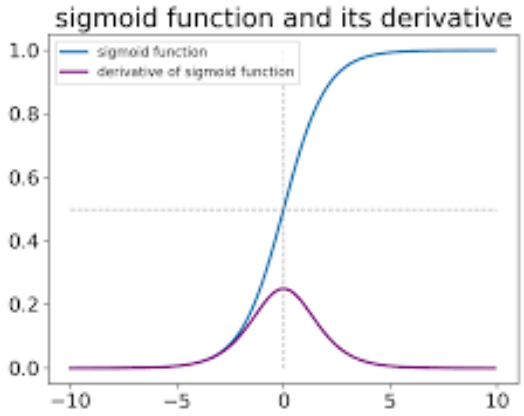
2 Experiment setups

A Sigmoid functions

$$sigmoid(x) = \frac{1}{1 + \exp -x}$$

sigmoid function 也稱爲 logistic function，因爲其連續、可微、平滑、單調，且介於 0 到 1 之間，所以適合作爲簡單模型的 activation function。下面是 sigmoid function 的微分:

$$sigmoid'(x) = sigmoid(x) \cdot (1 - sigmoid(x))$$

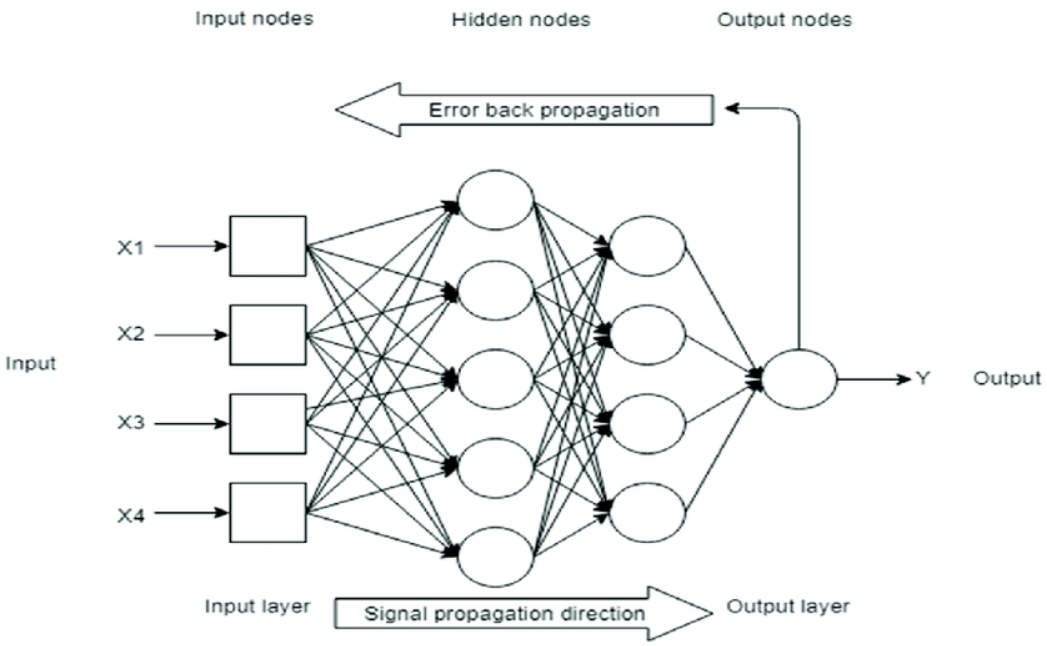


B Neural network

我設定兩層 hidden layers 皆爲 10 個 units，層層之間都會將前一層的 output 作爲下一層的 input，乘該層 weight matrix 並加上 bias，經過 activation function 後即爲 output。以下是變數的 size:

- Weight W = $[w_0, w_1, w_2]$: [10*1, 10*10, 1*10]
- Bias b = $[b_0, b_1, b_2]$: [10*1, 10*1, 1*1]
- z = $[z_0, z_1, z_2]$: [10*1, 10*1, 1*1] ，$z = x^T W + b$
- a = $[a_0, a_1, a_2]$: [10*1, 10*1, 1*1] ，$a = sigmoid(z)$

C Backpropagation

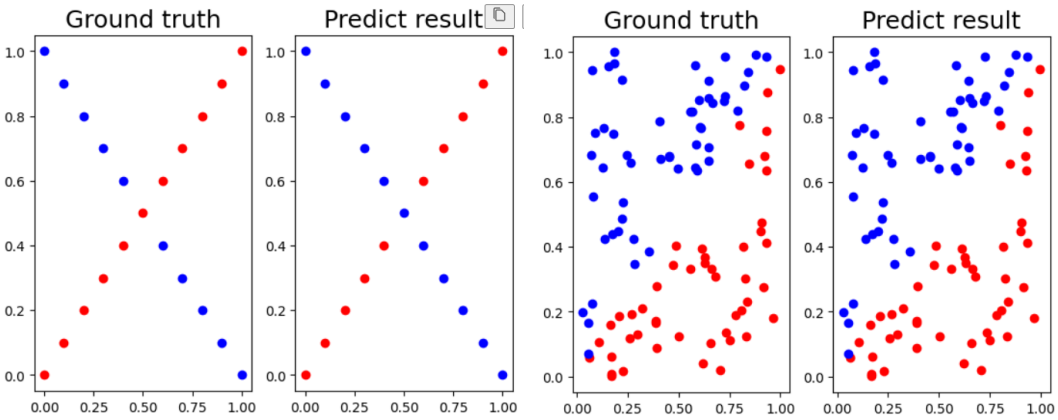


因爲是簡單的分類問題，所以用 Gradient Decent 就可以快速收斂。以下我將各個參數的導數列舉出來:

3 Results of my testing

A Screenshot and comparison figure

除了 xor 的正中央不一定每次都會分對，其餘都可以成功分類。



B The accuracy of my prediction

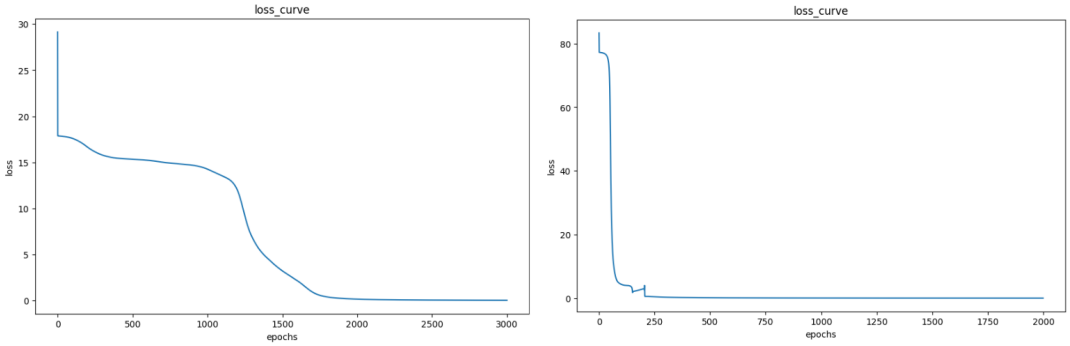
基本上都可以預測至 95% 以上，左圖 xor 準確率爲 95.2% ，右圖 linear 準確率爲 100%。稍後會說明不保證至 100% 的原因。

```
————————Terminated————————
[[0.00087576]
 [0.89171514]
 [0.00096954]
 [0.89115163]
 [0.00122056]
 [0.89067676]
 [0.00235178]
 [0.89061009]
 [0.03796381]
 [0.89120276]
 [0.89258374]
 [0.1192636 ]
 [0.89474614]
 [0.01871192]
 ...
 [0.9045103 ]
 [0.01425874]
 [0.90826042]]
accuracy: 95.23809523809523%
```

```
————————Terminated————————
[[0.99999139]
 [0.00000457]
 [0.00000247]
 [0.00003419]
 [0.99999907]
 [0.00002117]
 [0.99999961]
 [0.99999961]
 [0.00000277]
 [0.9999984 ]
 [0.9999995 ]
 [0.99981711]
 [0.96918642]
 [0.00000353]
 ...
 [0.00000494]
 [0.00009338]
 [0.00000616]]
accuracy: 100.0%
```

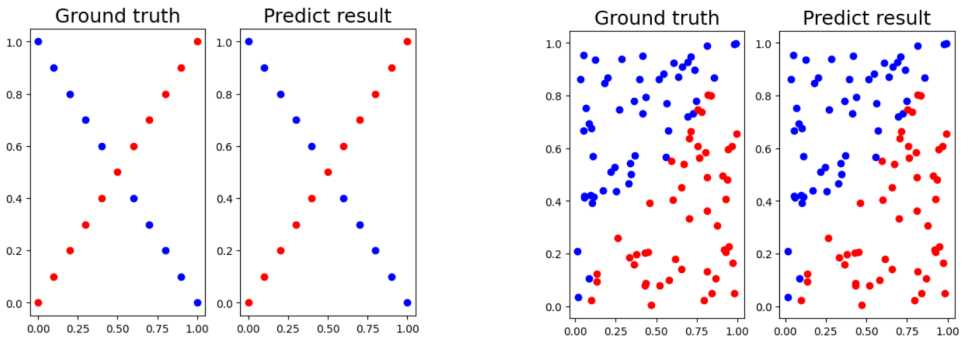
C Loss curve

在兩種資料集訓練中，約在 2000 epochs 時就可以很好的收斂。



D Anything you want to present

我設計 DNN 時有加入 bias ，但 xor 和 linear 這兩種數據明顯與原點有很大的線性關係，因此 bias 可能會導致不完全達到 100% 準確度。我有嘗試過不加 bias，十次的實驗皆爲 100% 的準確度。以下是不加 bias 時的實驗結果:



4 Discussion

A Try different learning rates

我的模型中，learning rate 皆設爲 0.1。以下我嘗試三組不同的 learning rate，分別爲 0.5、0.05、0.01，皆以十次實驗取平均。xor epochs 設爲 3000，linear epochs 設爲 2000。

lr	xor	linear
0.1	95.2	100
0.5	95.2	100
0.05	87.4	99.5
0.01	52.4	99.5

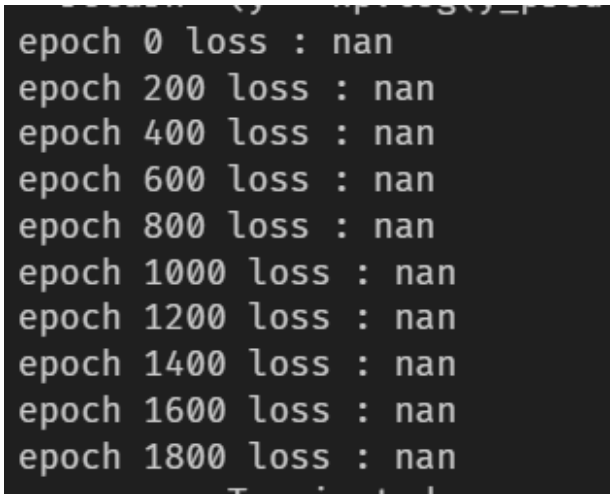
B Try different numbers of hidden units

我的模型中，兩層 hidden layer 皆爲 10 個 units。以下我嘗試四組不同的 hidden layer units 數組合，分別爲 [15, 10]、[20, 10]、[20, 20]，皆以十次實驗取平均。xor epochs 設爲 3000，linear epochs 設爲 2000。

hidden-pair	xor	linear
[10, 10]	95.2	100
[15, 10]	99.0	54.0
[20, 10]	99.5	52.3
[20, 20]	100	55.1

C Try without activation functions

因爲 loss function 爲 binary cross-entropy loss，其中包含 $\log(1 - y_{\text{pred}} + 1e^{-9})$，若沒有 activation function，y_{pred} 會大於 1，而 log 中不可有負值，因此 loss 爲 nan，因此準確度爲 0



D Something I want to share

對於原模型準確率不保證到 100%，我有三個看法。

1. 未設置種子碼，所以參數每次隨機亂數。因爲我的訓練 epochs 總數並沒有設很多，因此有時只能達到 95% 的精確度。若種子碼固定 36，則可以達 100% 準確率。

2. 在實驗過程中，也找到若是將 hidden layer size 設爲 [20, 20] 且 lr 爲 0.1 時，則 xor 的準確率將達到穩定的 100%，因此在 demo 時我將會以此最佳參數作演示。

3. 因爲這次 demo 時現場訓練，因此所設的訓練次數並無設太高，避免訓練時間過長。若設置 8000 epochs，用 [10, 10] 的 size pair，即可達到 100% 的準確度。

5 Extra

A Implement different optimizers

原先我使用的 optimizer 爲 Gradient Decent，在此我使用 Momentum:

$$\theta_{t+1} = \theta_t + v_t$$

其中，

$$v_t = \lambda \cdot v_{t-1} - \eta \cdot \nabla J(\theta_t)，v_0 = 0$$

B Implement different activation functions

原本在四層 (input layer, hidden layer *2, output layer) 轉換中都是以 sigmoid function 作爲 activation fucntion，在此我將 input layer 到第一層 hidden layer 中的 activation function 修改成 relu:

$$\text{ReLU}(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases}$$

其中它的導數爲

$$\frac{d}{dx}\text{ReLU}(x) = \begin{cases} 0, & \text{if } x < 0 \\ 1, & \text{if } x \geq 0 \end{cases}$$

會使用他是因爲接下來我將第一層改成 cnn。

C Implement convolutional layers

我將第一層改成用 3*1 的 kernel 去做卷積，並將第一層 hidden layer 改成 20 個 units，運用 zero-padding 以及 stride=1，我需要 10 個 kernel 才能讓 2*1 的 input 轉成 20 個 units。

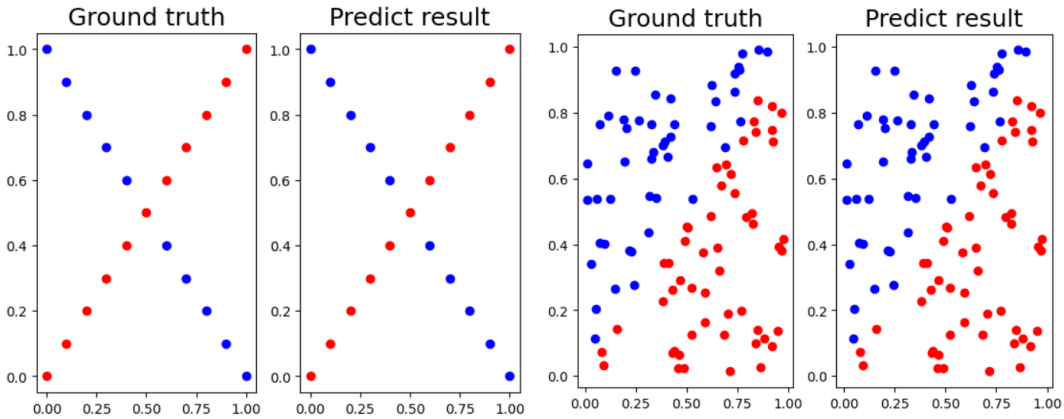


Figure 1: compare graph

Figure 2: accuracy

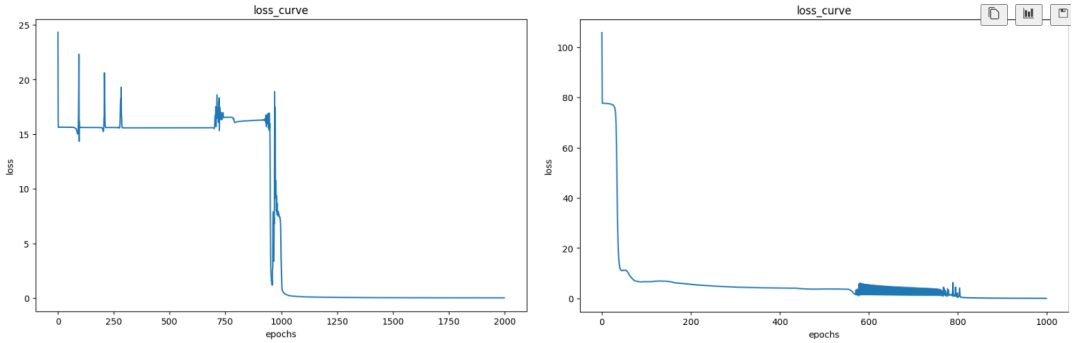


Figure 3: loss curve