**Lab1: back-propagation**

Student id / name: A113599 / 楊淨富

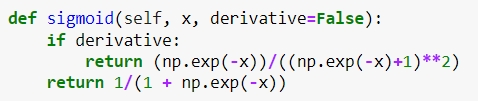
**I. Introduction:**

利用兩層hidden layer的Neural network來訓練二維的資料(range from 0.0 to 1.0) 即其所對應到的一維ground truth (0 or 1)。因為資料集較小，故training set同時也是testing set。

**II. Experiment setups**

1. **Activation functions**

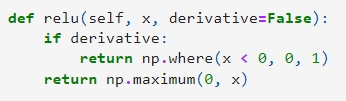
* Sigmoid function:



Sigmoid function是一種activation function，其輸出為0至1。

derivative若設為True則是sigmoid微分後的函數。反之則為sigmoid function。

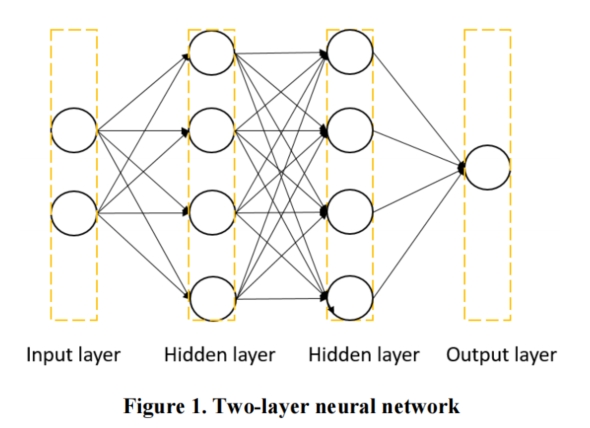
* Relu funtion:



Relu也是一種activation function，假定輸入為x，則輸出為max(0, x)。

比較麻煩的是其在0這個點無法進行微分(因為左右兩點的斜率不同)。

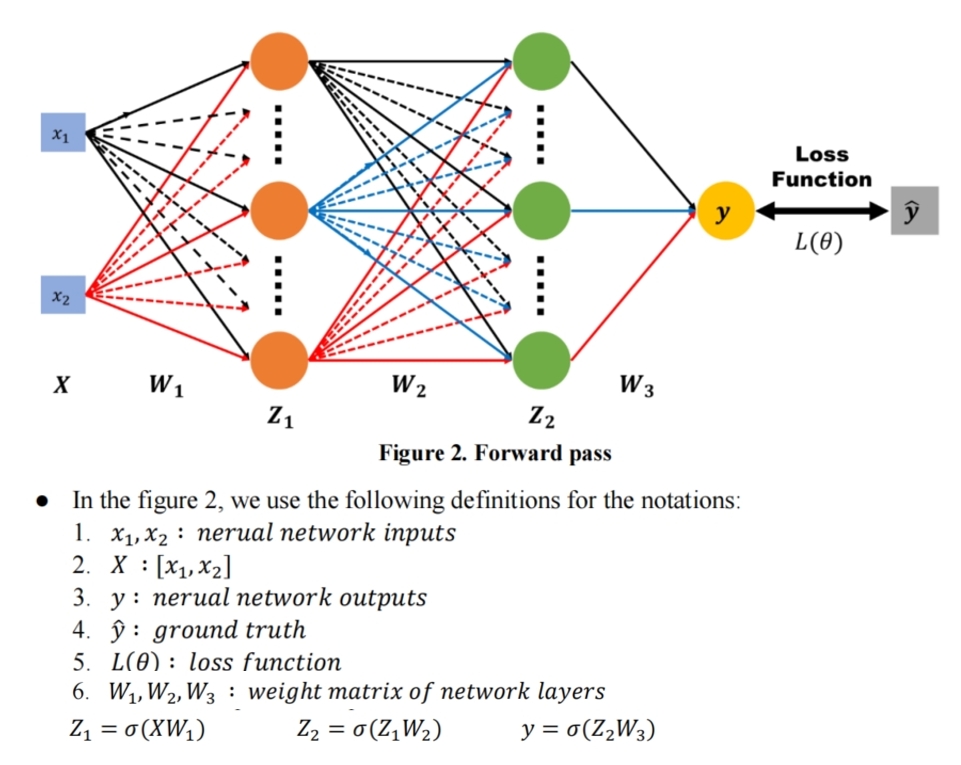
1. **Neural network**



* Layers:
* Input layer: 由於input是二維，所以兩個neuron。
* Hidden layer: 要求至少兩層hidden layers。我預設兩層(h1, h2)各4個neuron。
* Output layer: 由於label是0或1，所以一個neuron。
* Training process:

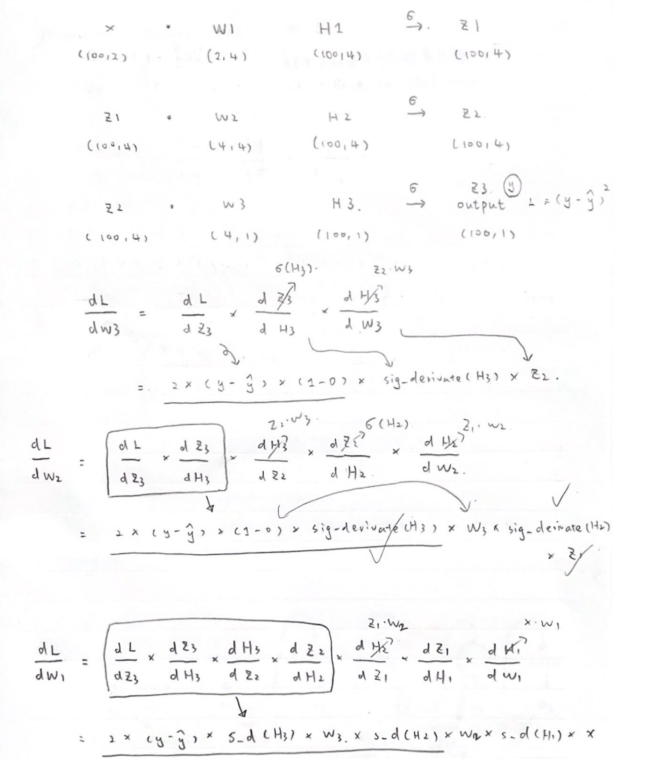
1. 先隨機初始化parameters (此lab即weights)
2. 透過feed\_forward得出output
3. 進行back\_propagation(將Loss funtion對各weights做微分，算出各weights之gradient)
4. 透過optimize更新各weights
5. 重複進行Step 2 - 4

**C. Back propagation:**



* 由output依序向前更新各weights
* 更新方法為:

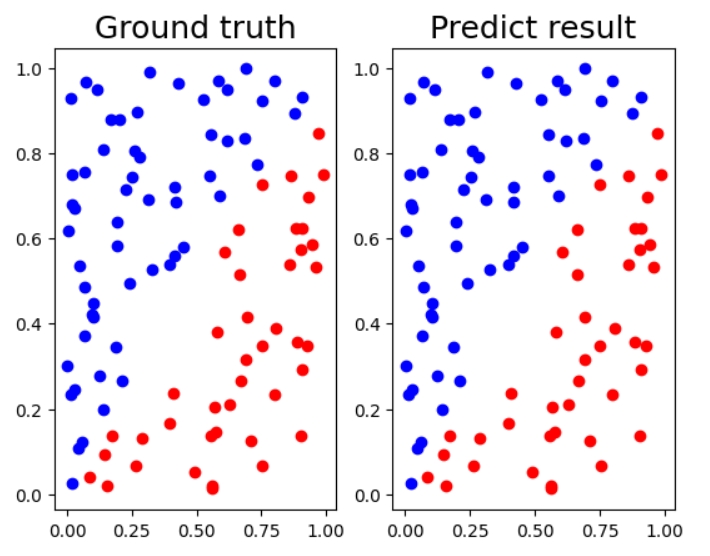
將對Loss function對各weight做偏微分，計算出各weight之gradients後，算出new\_weight = old\_weight - learning\_rate \* weight\_gradient。如下圖所示(依序更新W3,W2,W1)



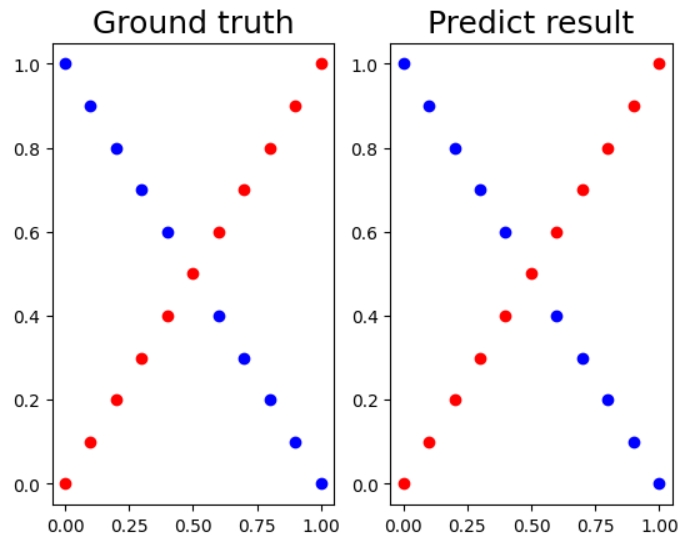
**III. Results of your testing**

**A. Screenshot and comparison figure**

* Case 1: Linear

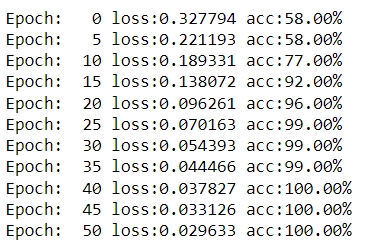


* Case 2: XOR

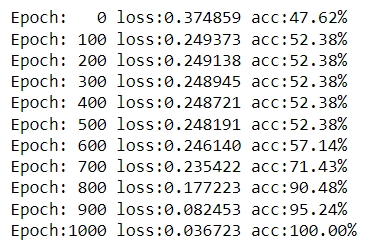


**B. Show the accuracy of your prediction**

* Case 1: Linear

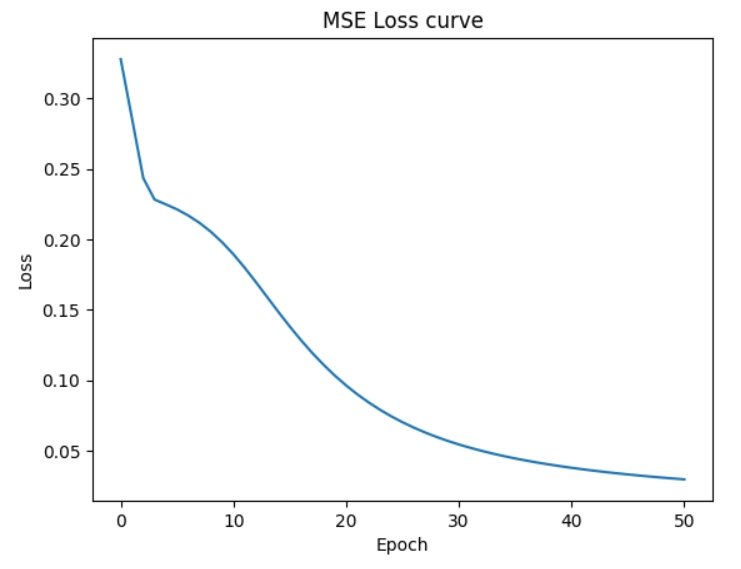


* Case 2: XOR

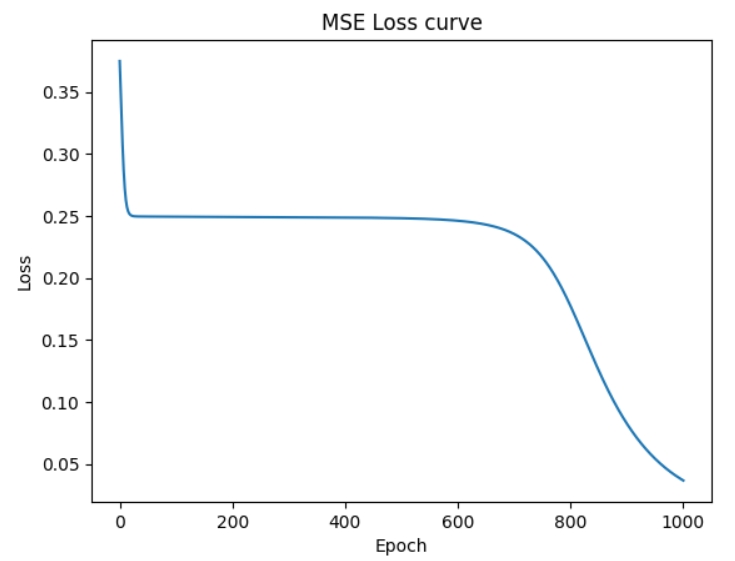


**C. Learning curve(loss, epoch curve)**

* Case 1: Linear



* Case 2: XOR



**4. Discussion**

**A. Try different learning rates**

* **Case 1: Linear**

|  |  |  |
| --- | --- | --- |
| learing\_rate | Epochs to acc=100% | Loss curve |
| 0.1 | 38 |  |
| 0.01 | 366 |  |
| 0.5 | 560 |  |

* **Case 2: XOR**

|  |  |  |
| --- | --- | --- |
| learing\_rate | Epochs to acc=100% | Loss curve |
| 0.1 | 916 |  |
| 0.01 | 9100 |  |
| 0.5 | 188 |  |

**B. Try different numbers of hidden units**

* **Case 1: Linear**

|  |  |  |
| --- | --- | --- |
| Hidden layers | Epochs to acc=100% | Loss curve |
| [2, 4, 4, 1] | 38 |  |
| [2, 6, 6, 1] | 33 |  |
| [2, 10, 10, 1] | 36 |  |

* **Case 2: XOR**

|  |  |  |
| --- | --- | --- |
| Hidden layers | Epochs to acc=100% | Loss curve |
| [2, 4, 4, 1] | 916 |  |
| [2, 6, 6, 1] | 661 |  |
| [2, 10, 10, 1] | 587 |  |

1. **Try without activation functions**

* **Case 1: Linear**

|  |  |
| --- | --- |
| With sigmoid | Without sigmoid |
| Learning\_rate = 0.1 | Learning\_rate = 0.0001  (if use 0.1, cannot converge) |
|  |  |
|  |  |

* **Case 2: XOR**

|  |  |
| --- | --- |
| With sigmoid | Without sigmoid |
| Learning\_rate = 0.1 | Learning\_rate = 0.01  (if use 0.1, cannot converge) |
|  |  |
|  |  |

**5. Extra**

1. **Implement different optimizers**

* **Case 1: Linear**

|  |  |
| --- | --- |
| **Sigmoid + SGD** | **Sigmoid + momentum** |
|  |  |
|  |  |
|  |  |

* **Case 2: XOR**

|  |  |
| --- | --- |
| **Sigmoid + SGD** | **Sigmoid + momentum** |
|  |  |
|  |  |
|  |  |

1. **Implement different activation functions**

* **Case 1: Linear**

|  |  |
| --- | --- |
| **Relu + SGD** | **Relu + momentum** |
|  |  |
|  |  |
|  |  |

* **Case 2: XOR**

|  |  |
| --- | --- |
| **Relu + SGD** | **Relu + momentum** |
|  |  |
|  |  |
|  |  |