📖

DL Class Lab1

Lab1 : Backpropagation

▼ 1. Introduction

✏️ Lab1 是僅以 numpy library 去實現 **Backpropagation** 的算法，並建立 regression model 訓練 **linear** 與 **XOR** 分佈資料。透過這次業加強了對於 gradient-based training 的 neuron network 之實作與訓練技能。藉由這次機會練習建立 python 之模組化的物件，增加整體程式的有效擴充性，進而達到易維護與易測試的目的，下方為個物件的介：

▼ A. Activation:

- Attributes:
  - forward_func → dict, for calling the forward pass easier.
  - backward_func → dict, for calling the backward pass easier.
  - func_name → str, to determine which activation method should be called.
  - pre_input → numpy array, to record the previous input.
  - pre_output → numpy array, to recrod the previous forward output.
- Methods:
  - For forward pass:
    - forward
    - sigmoid
    - relu
    - softplus
  - For backward pass:
    - backward
    - derivative_sigmid
    - derivative_relu
    - derivative_softplus
  - For update some trainable activation funcitons:
    - update

▼ B. Linear_Laryer:

- Attributes:
  - input_dim → int, the dimention of the input.

- output_dim → int, the dimention of the output.
- w → numpy array, the weights of linear layer with the shape of (output_dim, input_dim)
- delta_w → numpy array, to record the gradient of the weights.
- momentum_w → numpy array, to record the momentum of the weights.
- beta → float, hyper parameter for updating the momentum_w.
- pre_input → numpy array, to record the previous input of forward pass.

- Methods:
  - forward
  - backward
  - update
  - L2_Norm → for normalize the weights with the method of l2 norm.

▼ C. Model:

- Attributes:
  - history → dict, to record the information of traning and testing.
  - layers → list, the list contains the whole layers of model
- Methods
  - Build
  - fit → for training the model
  - forward
  - backward
  - update
  - Record_History
  - Show_Training_Stage_Info
  - predict_2_binary
  - Compute_Accuracy
  - evaluate

▼ 2. Experiment Setups

- Device & Software:
  - Device: Azure Server
  - OS: ubuntu 18.04
  - python version: 3.8
  - numpy version: 1.22.4

- Standar Model Setups:
  - Architecture:

- 2-3-Sigmoid-3-Sigmoid-1, which means the code below:

```
model.Build([
    Linear_Layer(2, 3),
    Activation('sigmoid'),
    Linear_Layer(3, 3),
    Activation('sigmoid'),
    Linear_Layer(3, 1),
])
```

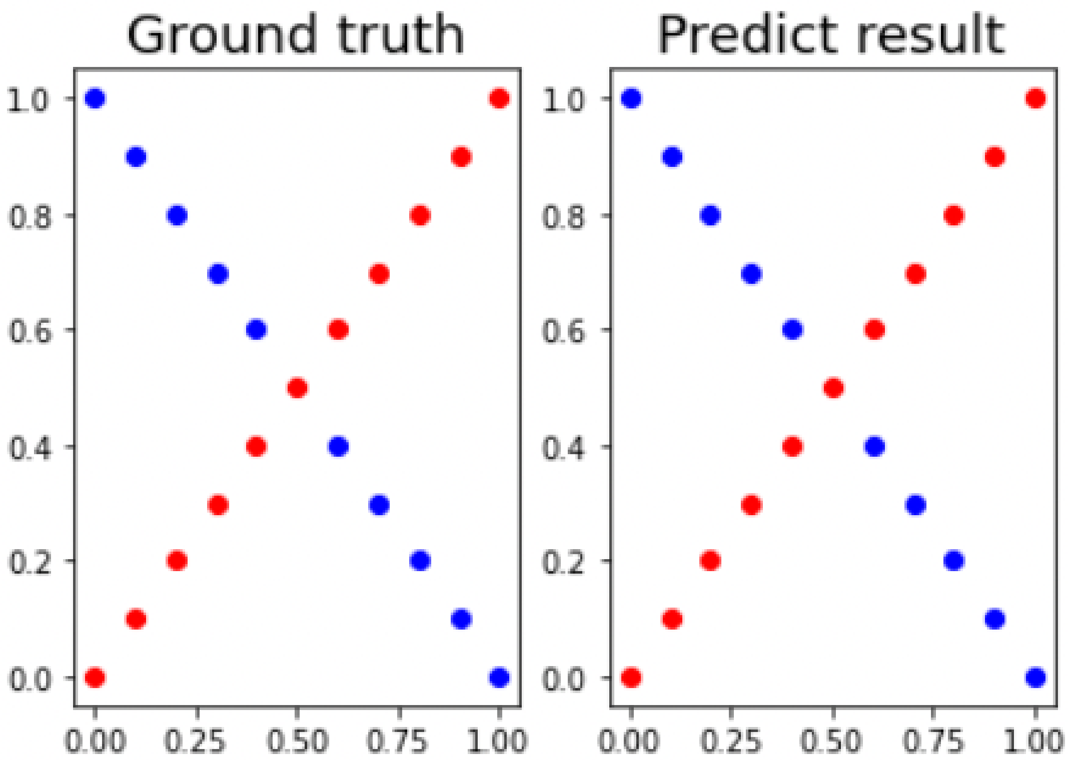
- Epochs: 30000
- Learning Rate: 0.1
- Optimizer: sgdm
- Loss Function: Mean Square Error

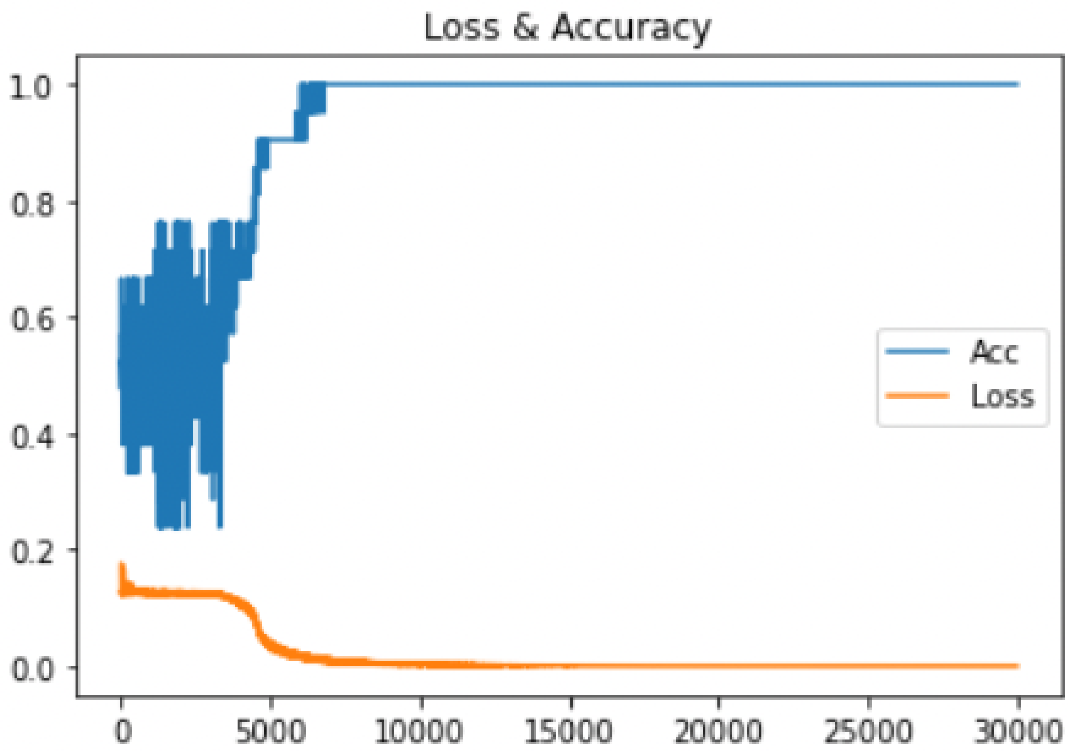
3. Results of Your Testing

A. XOR Dataset

Standar Model Result-XOR

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
XOR	0.1	sgdm	2-3-Sigmoid-3-Sigmoid-1	30000	100%

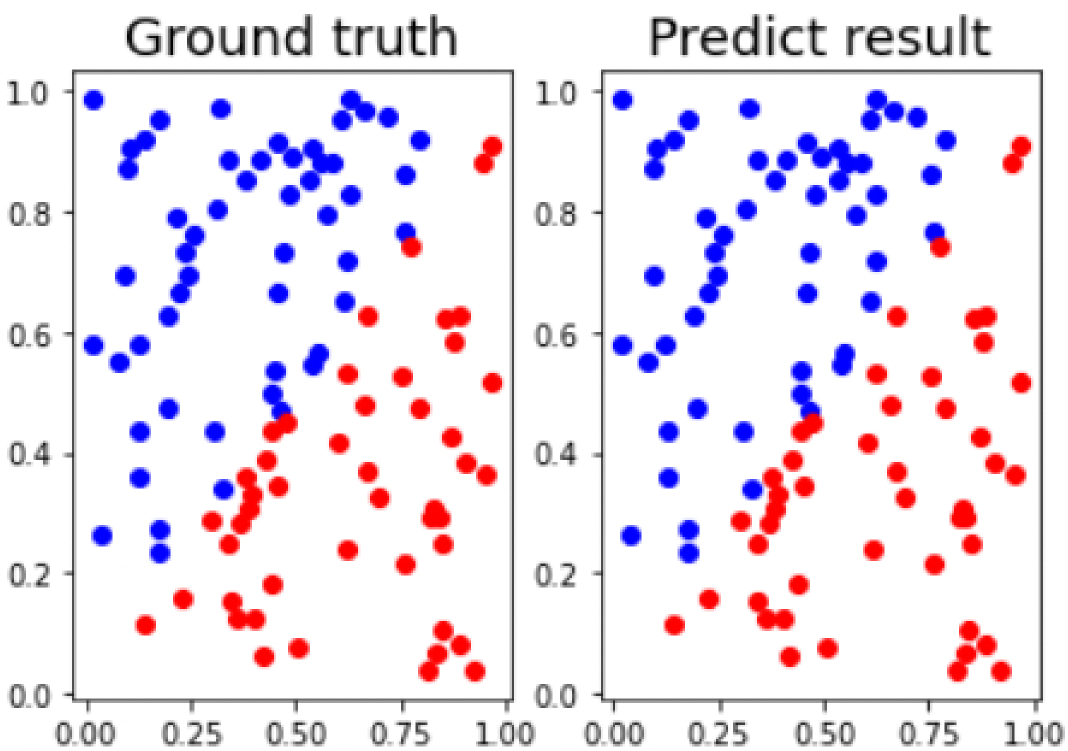


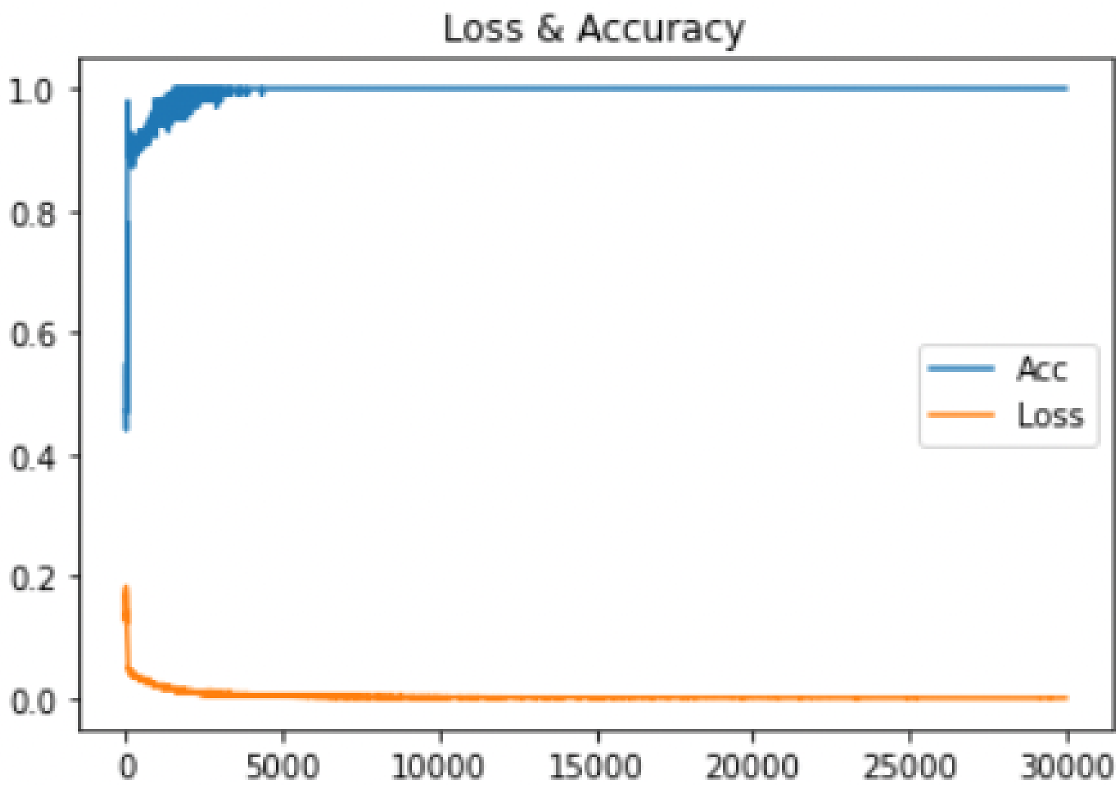


▼ **B. Linear Dataset**

Standar Model Result-Linear

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
Linear	0.1	sgdm	2-3-Sigmoid-3-Sigmoid-1	30000	100%





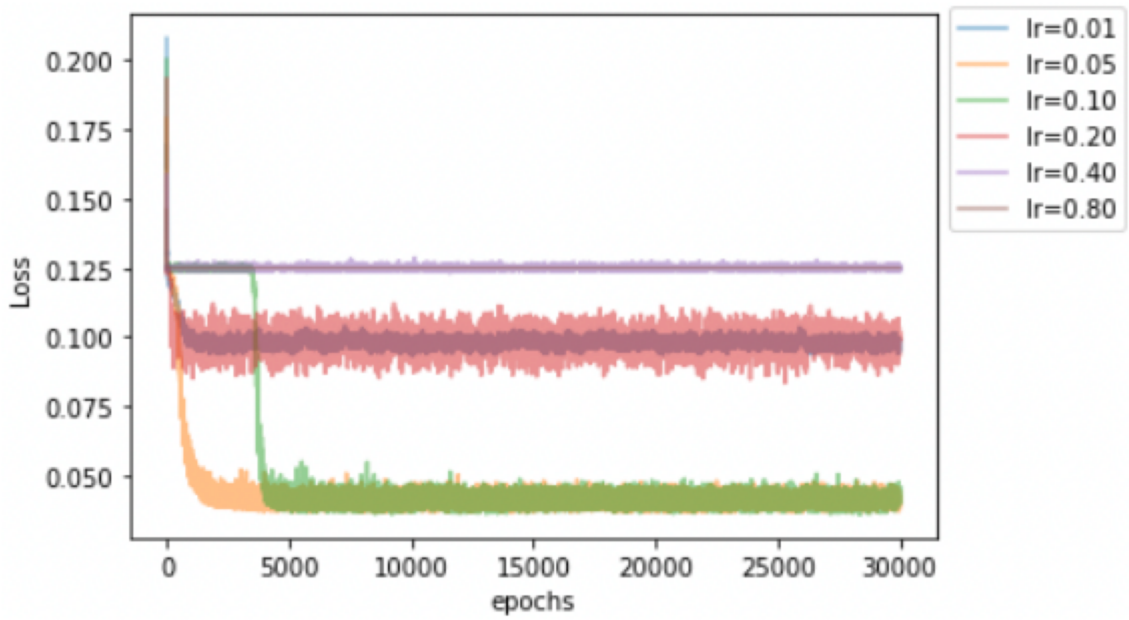
4. Discussion

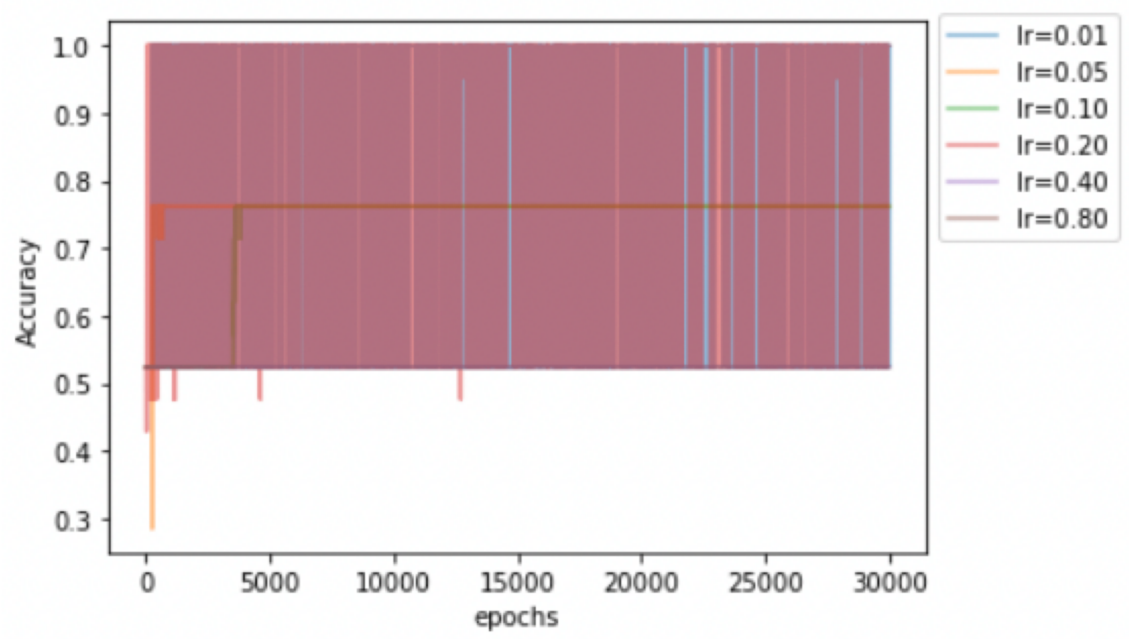
A. Try different learning rates

✏️ 以下方的表可以顯示出 Learning Rate 對訓練模型影響是非常顯著的，所以如何調整 Learning Rate 並更新參數這也是個很大的坑。

Differenet Learning Rate

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
XOR	0.01	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	30000	52%
XOR	0.05	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	30000	76%
XOR	0.1	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	30000	100%
XOR	0.2	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	30000	52%
XOR	0.4	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	30000	52%
XOR	0.8	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	30000	52%



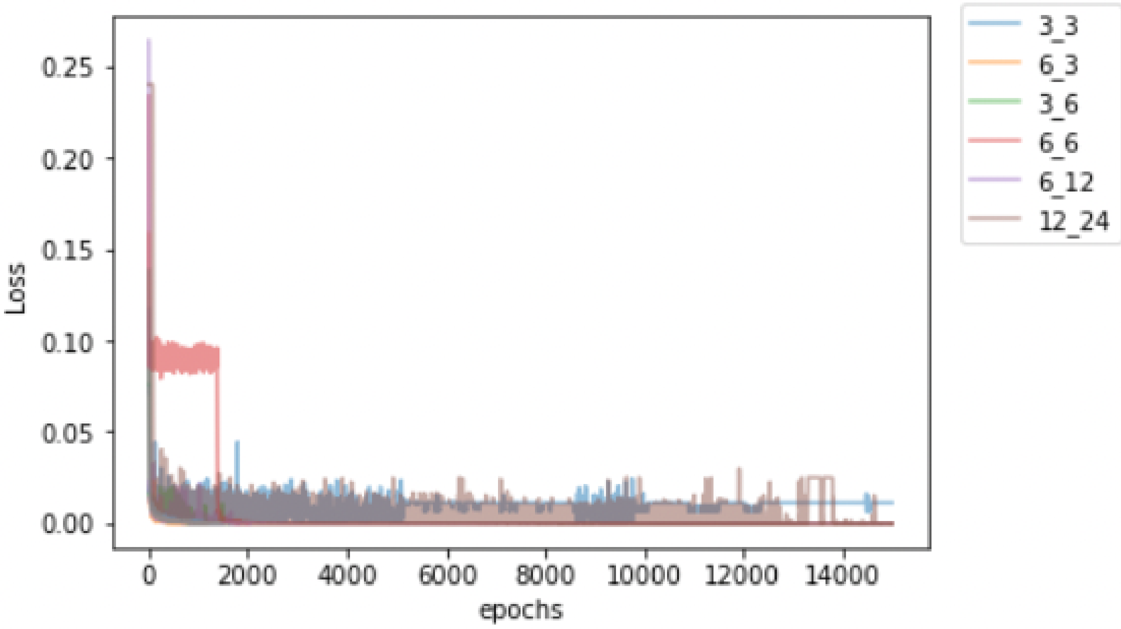


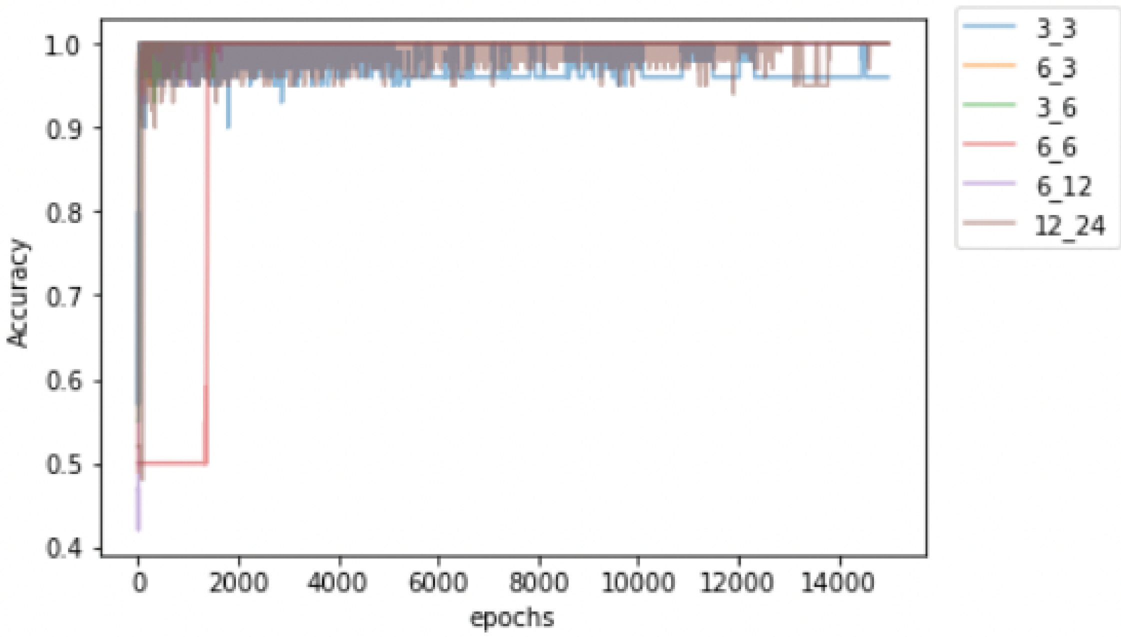
▼ B. Try different numbers of hidden units

✏️ 以下方的圖表得知模型的參數量足夠多的情況下，就能達到不錯的效果。

Differenet Number of Hidden Units

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
Linear	0.1	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	15000	96%
Linear	0.1	sgdm	2-6-ReLU-3-ReLU-1-Sigmoid	15000	100%
Linear	0.1	sgdm	2-3-ReLU-6-ReLU-1-Sigmoid	15000	100%
Linear	0.1	sgdm	2-6-ReLU-6-ReLU-1-Sigmoid	15000	100%
Linear	0.1	sgdm	2-6-ReLU-12-ReLU-1-Sigmoid	15000	100%
Linear	0.1	sgdm	2-12-ReLU-24-ReLU-1-Sigmoid	15000	100%





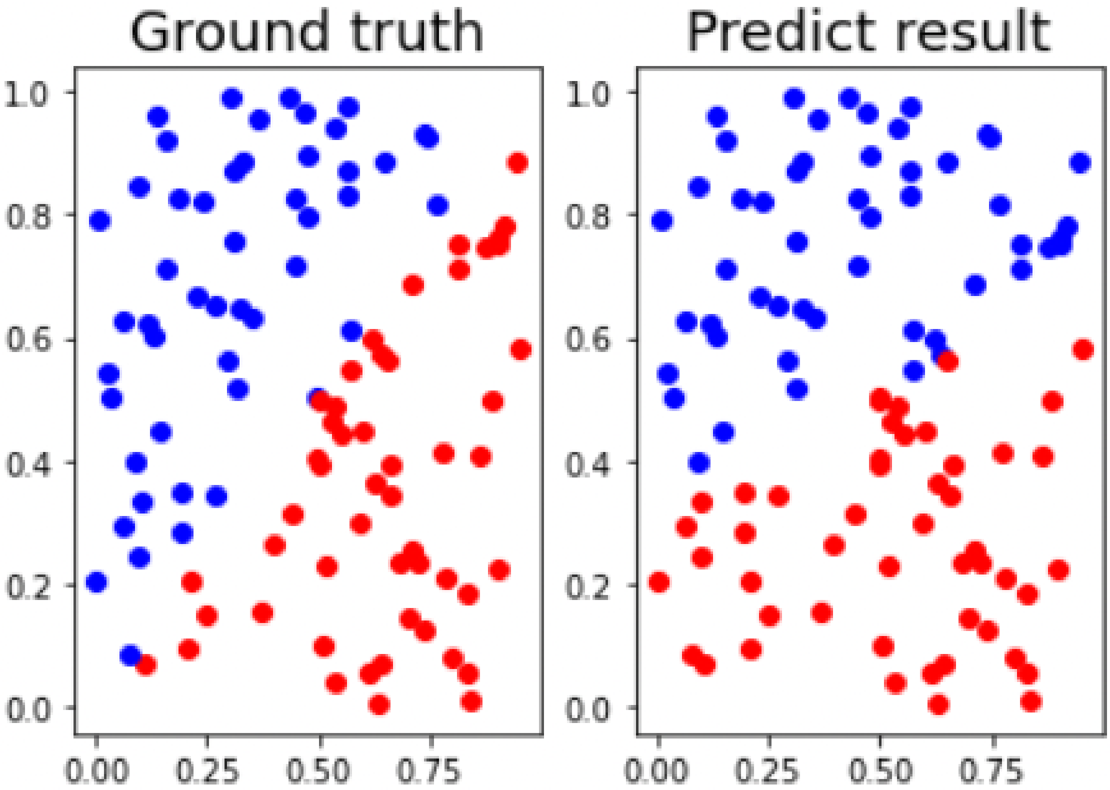
▼ C. Try without activation funcitons

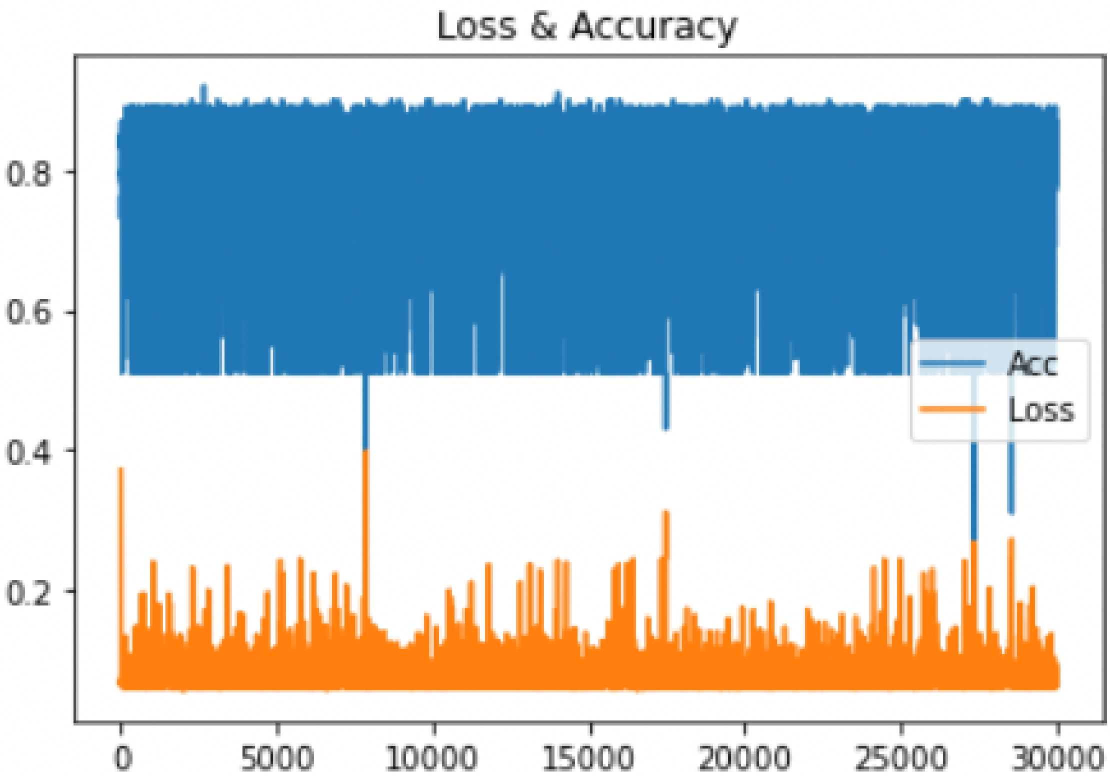
✏️ 下面圖表表示出缺少 activation functions 會導致訓練困難以及準確度下降。

Linear Data:

Without Activation Functions Comparision

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
Linear	0.1	sgdm	2-3-3-1	30000	80%

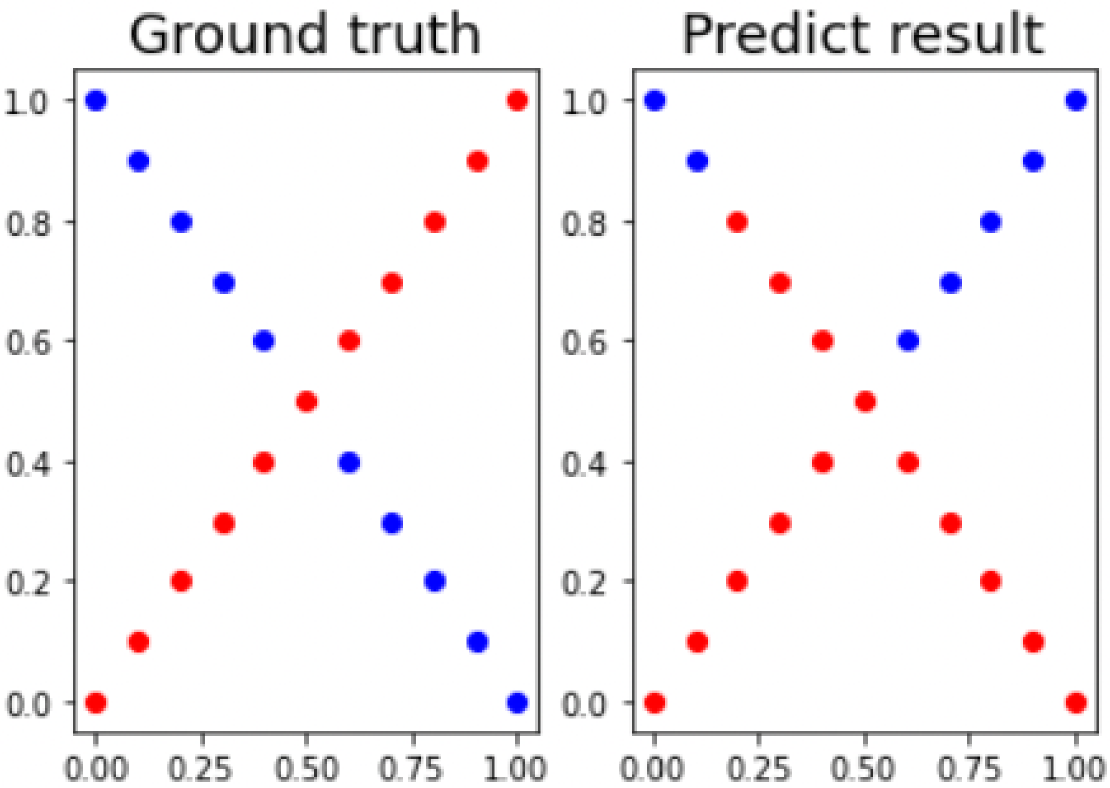


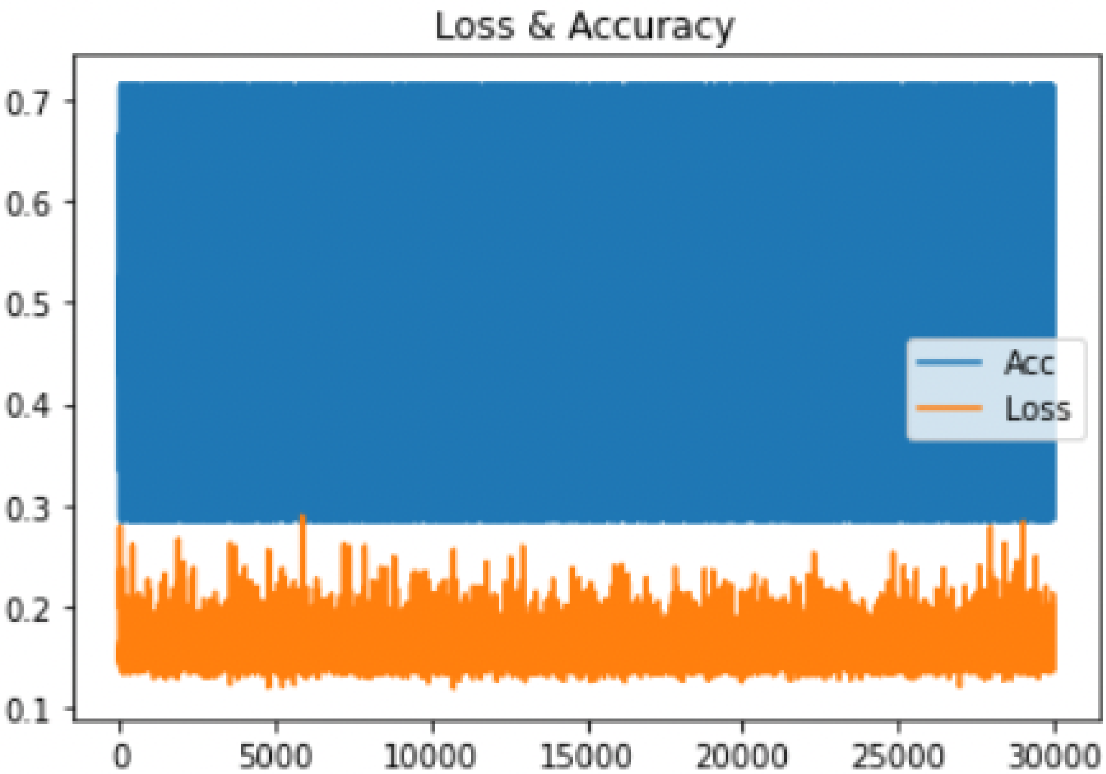


XOR Data:

Without Activation Functions Comparision

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
XOR	0.1	sgdm	2-3-3-1	30000	38%



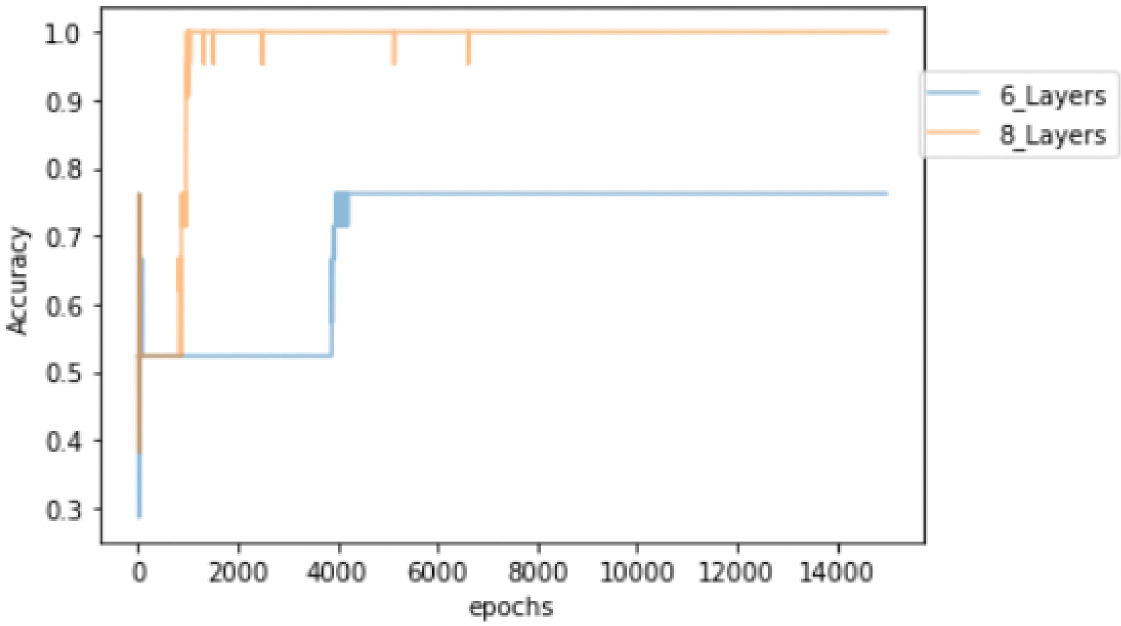


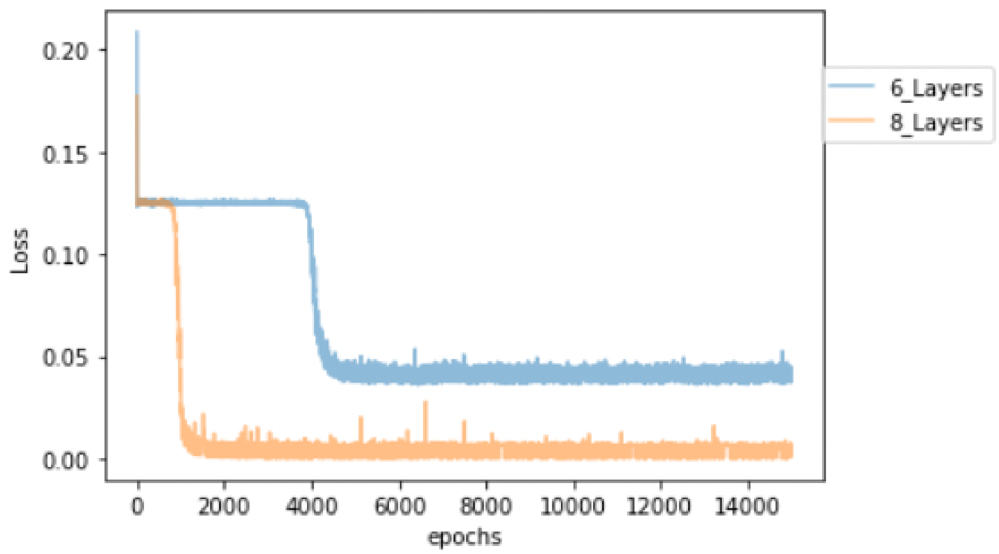
▼ D. Try different number of layers

> ✏️ 下方圖表表示，層數較多的網路可以有較佳的效果。

Different Number of Layers

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
XOR	0.1	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	15000	76%
XOR	0.1	sgdm	2-3-ReLU-3-ReLU-3-ReLU-1-Sigmoid	15000	100%



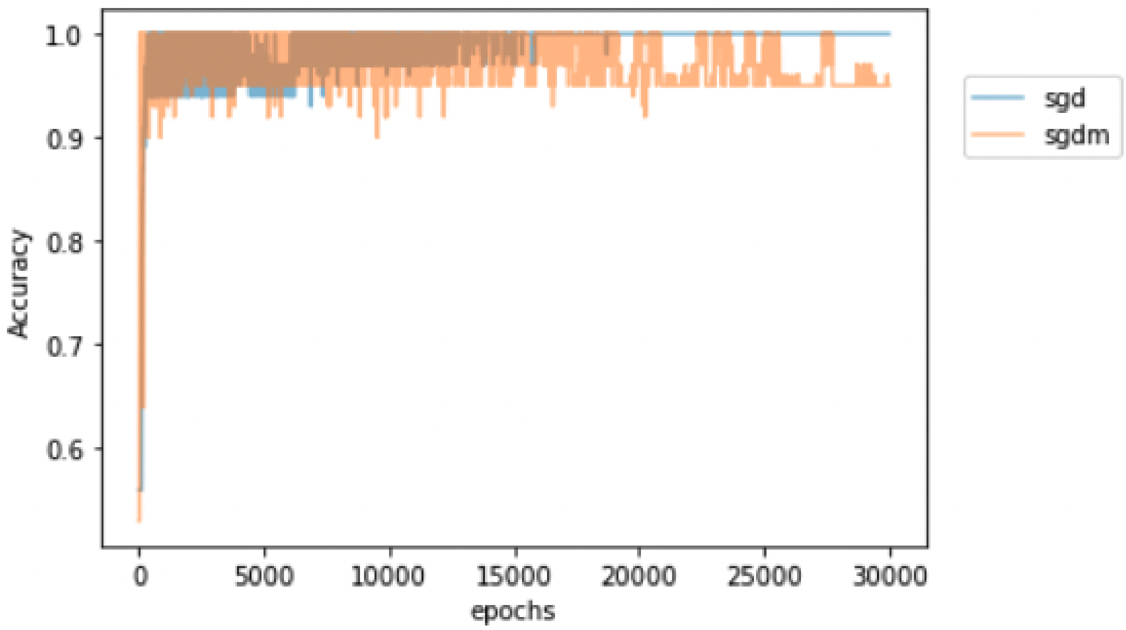


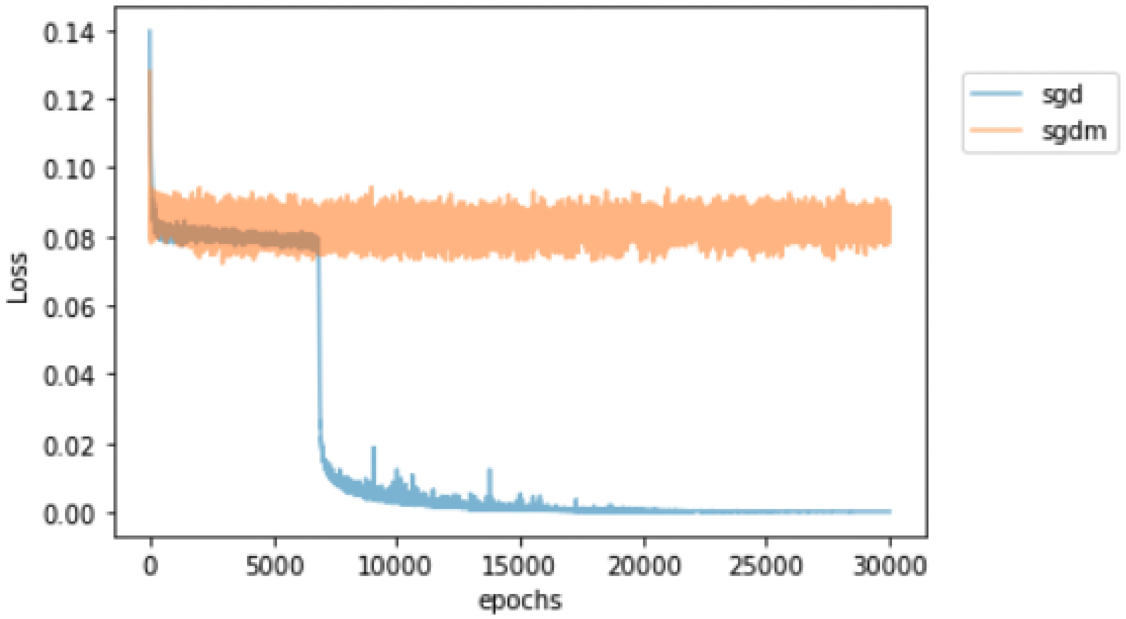
▼ E. Implement different optimizers

✏️ 圖顯示 sgdm 的震盪幅度比 sgd 高。

Different Optimizers

Datasets	Learning Rate	Optimizer	Architectures	Epochs	Accuracy
Linear	0.1	sgd	2-3-ReLU-3-ReLU-1-Sigmoid	30000	100%
Linear	0.1	sgdm	2-3-ReLU-3-ReLU-1-Sigmoid	30000	95%





▼ F. Manuscript

✏️ 在實作 backpropagation 時，手算證明的過程：

$$z^1_{3\times1} = W^1_{3\times2} \times X_{2\times1} = \begin{bmatrix} W^1_{11} & W^1_{21} \\ W^1_{12} & W^1_{22} \\ W^1_{13} & W^1_{23} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} z^1_1 \\ z^1_2 \\ z^1_3 \end{bmatrix}$$

$$\delta X = W^{1T} \times \delta z^1 = \begin{bmatrix} W^1_{11} & W^1_{12} & W^1_{13} \\ W^1_{21} & W^1_{22} & W^1_{23} \end{bmatrix} \begin{bmatrix} \delta z^1_1 \\ \delta z^1_2 \\ \delta z^1_3 \end{bmatrix}$$

$$\delta X = W^{1T} \times \delta z^1 =$$

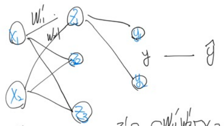
$$\delta z^1 = W^{2T} \times \delta z^2 = \begin{bmatrix} W^2_{11} & W^2_{12} & W^2_{13} & W^2_{14} \\ W^2_{21} & W^2_{22} & W^2_{23} & W^2_{24} \\ W^2_{31} & W^2_{32} & W^2_{33} & W^2_{34} \end{bmatrix}$$

$$z^2_{4\times1} = W^2_{4\times3} \times z^1_{3\times1} = \begin{bmatrix} W^2_{11} & W^2_{21} & W^2_{31} \\ W^2_{12} & W^2_{22} & W^2_{32} \\ W^2_{13} & W^2_{23} & W^2_{33} \\ W^2_{14} & W^2_{24} & W^2_{34} \end{bmatrix} \begin{bmatrix} z^1_1 \\ z^1_2 \\ z^1_3 \end{bmatrix}_{3\times1} = \begin{bmatrix} z^2_1 \\ z^2_2 \\ z^2_3 \\ z^2_4 \end{bmatrix}$$

$$\delta z^2_{4\times1} \times z^{1T}_{1\times3} = \begin{bmatrix} \delta z^2_1 \\ \delta z^2_2 \\ \delta z^2_3 \\ \delta z^2_4 \end{bmatrix} \begin{bmatrix} z^1_1 & z^1_2 & z^1_3 \end{bmatrix}$$

$$= \begin{bmatrix} W^2_{11} & W^2_{21} & W^2_{31} \\ W^2_{12} & W^2_{22} & W^2_{32} \end{bmatrix}$$

$$\begin{cases} \delta \underset{X}{z} = W^T \times \delta \\ \delta W = \delta \times z^T \\ W^t = W^{t-1} + \eta \times \delta W^{t-1} \end{cases}$$



$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad z^1 = \begin{bmatrix} z^1_1 \\ z^1_2 \\ z^1_3 \end{bmatrix} = \begin{bmatrix} W^1_{11} & W^1_{21} \\ W^1_{12} & W^1_{22} \\ W^1_{13} & W^1_{23} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} W^2_{11} & W^2_{21} & W^2_{31} \\ W^2_{12} & W^2_{22} & W^2_{32} \end{bmatrix} \begin{bmatrix} z^1_1 \\ z^1_2 \\ z^1_3 \end{bmatrix}$$

$u = (y - \hat{y})$ — ①

$L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}u^2$ — ②

$\frac{\partial L}{\partial u} = u = (y - \hat{y})$ — ③

$\frac{\partial L}{\partial y} = \frac{\partial u}{\partial y}\frac{\partial L}{\partial u} = (y - \hat{y})$ — ④ $\quad W^2_{2\times3} \; (y-\hat{y})_{2\times1}$

$\frac{\partial L}{\partial z^1} = \frac{\partial y}{\partial z^1}\frac{\partial L}{\partial y} = \frac{\partial (W^2 z^1)}{\partial z^1}(y - \hat{y}) = W^{2T}(y - \hat{y})$ ⑤

check: $(W^2)^T (y-\hat{y}) = \begin{bmatrix} W^2_{11} & W^2_{12} \\ W^2_{21} & W^2_{22} \\ W^2_{31} & W^2_{32} \end{bmatrix} \begin{bmatrix} \delta y_1 \\ \delta y_2 \end{bmatrix}$

$$\delta z^1 = \begin{bmatrix} W^2_{11}\delta y_1 + W^2_{12}\delta y_2 \\ W^2_{21}\delta y_1 + W^2_{22}\delta y_2 \\ W^2_{31}\delta y_1 + W^2_{32}\delta y_2 \end{bmatrix} \text{ v.s } \begin{bmatrix} z^1_1 \\ z^1_2 \\ z^1_3 \end{bmatrix}$$

$$\delta W^2 = \frac{\partial L}{\partial W^2} = \frac{\partial y}{\partial W^2}\frac{\partial L}{\partial y} = \frac{\partial (W^2 z^1)}{\partial W^2}(y - \hat{y})$$

$$\frac{\partial}{\partial W^2}\begin{bmatrix} W^2_{11}z^1_1 + W^2_{21}z^1_2 + W^2_{31}z^1_3 \\ W^2_{12}z^1_1 + W^2_{22}z^1_2 + W^2_{32}z^1_3 \end{bmatrix}_{2\times1} \begin{bmatrix} \delta y_1 \\ \delta y_2 \end{bmatrix}_{2\times1}$$

$$\Downarrow$$

$$\delta W^2_{2\times3} = \begin{bmatrix} \delta W^2_{11} & \delta W^2_{21} & \delta W^2_{31} \\ \delta W^2_{12} & \delta W^2_{22} & \delta W^2_{32} \end{bmatrix}$$

$$= \begin{bmatrix} z_1\delta y_1 & z_2\delta y_1 & z_3\delta y_1 \\ z_1\delta y_2 & z_2\delta y_2 & z_3\delta y_2 \end{bmatrix}_{2\times3}$$

v.s. $\Rightarrow \begin{bmatrix} \delta y_1 \\ \delta y_2 \end{bmatrix}_{2\times1} \underbrace{\begin{bmatrix} z_1 & z_2 & z_3 \end{bmatrix}_{1\times3}}_{z^T} = \delta y \times z^T$

$$= \begin{bmatrix} z_1\delta y_1 & z_2\delta y_1 & z_3\delta y_1 \\ z_1\delta y_2 & z_2\delta y_2 & z_3\delta y_2 \end{bmatrix}$$

$\delta z = W^T \times \delta\theta$

$\delta W = \delta\theta \times z^T$ $\qquad a(x) = \frac{1}{1+e^{-x}}$

$y = \Rightarrow$ sigmoid $\frac{1}{1+e^{-x}}$ $\qquad \frac{\partial y}{\partial u} = a$

$\frac{\partial y}{\partial x} = \frac{\partial a}{\partial x}\frac{\partial y}{\partial a} = \frac{\partial a}{\partial x}$