NYCU DL
Lab1 - Backpropagation

TA 曾昱仁 Yu-Jen Tseng

July 9, 2024

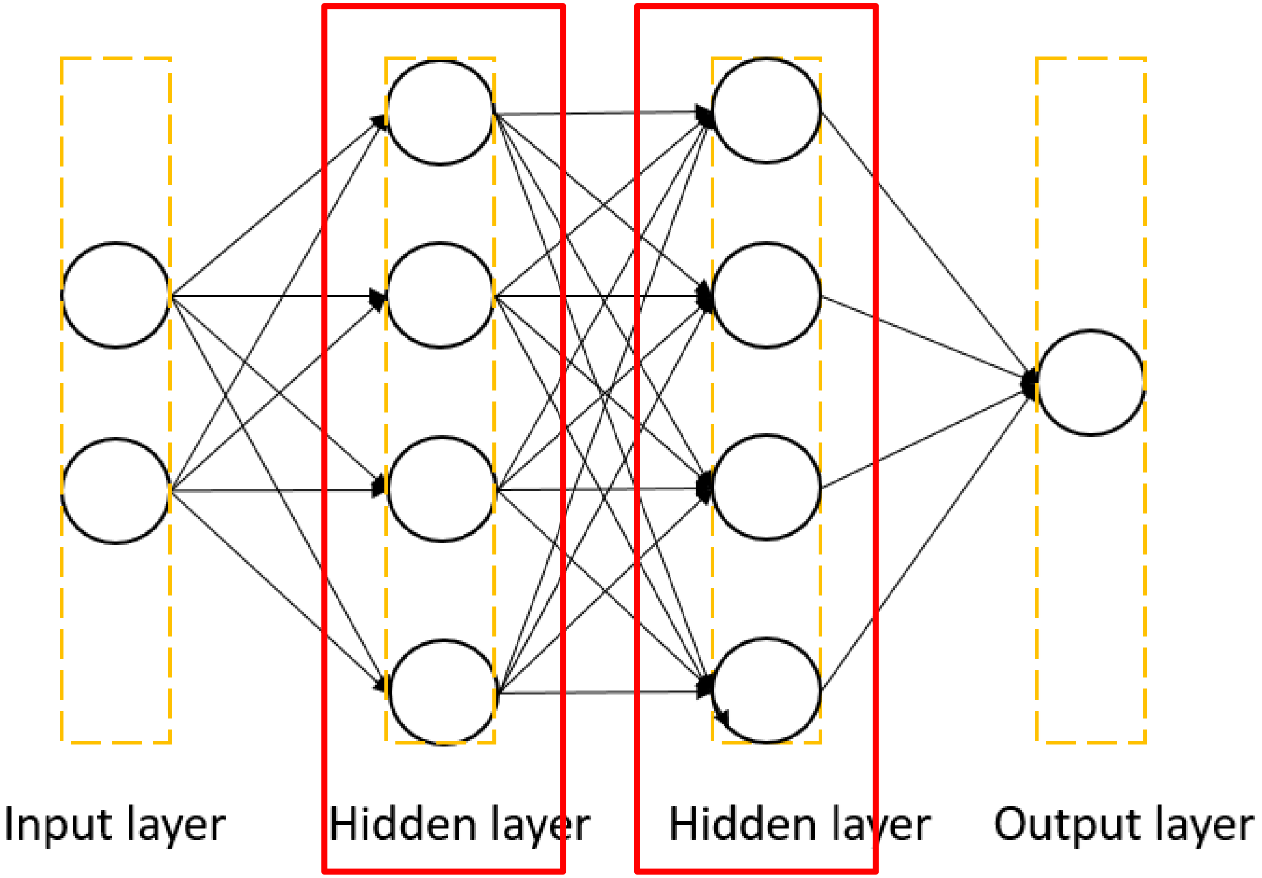
Outline

- Lab Objective
- Lab Description
- Scoring Criteria
- Time Schedule

Lab Objective

Lab Objective

- In this lab, you will need to understand and implement a simple neural network with forward and backward pass using two hidden layers



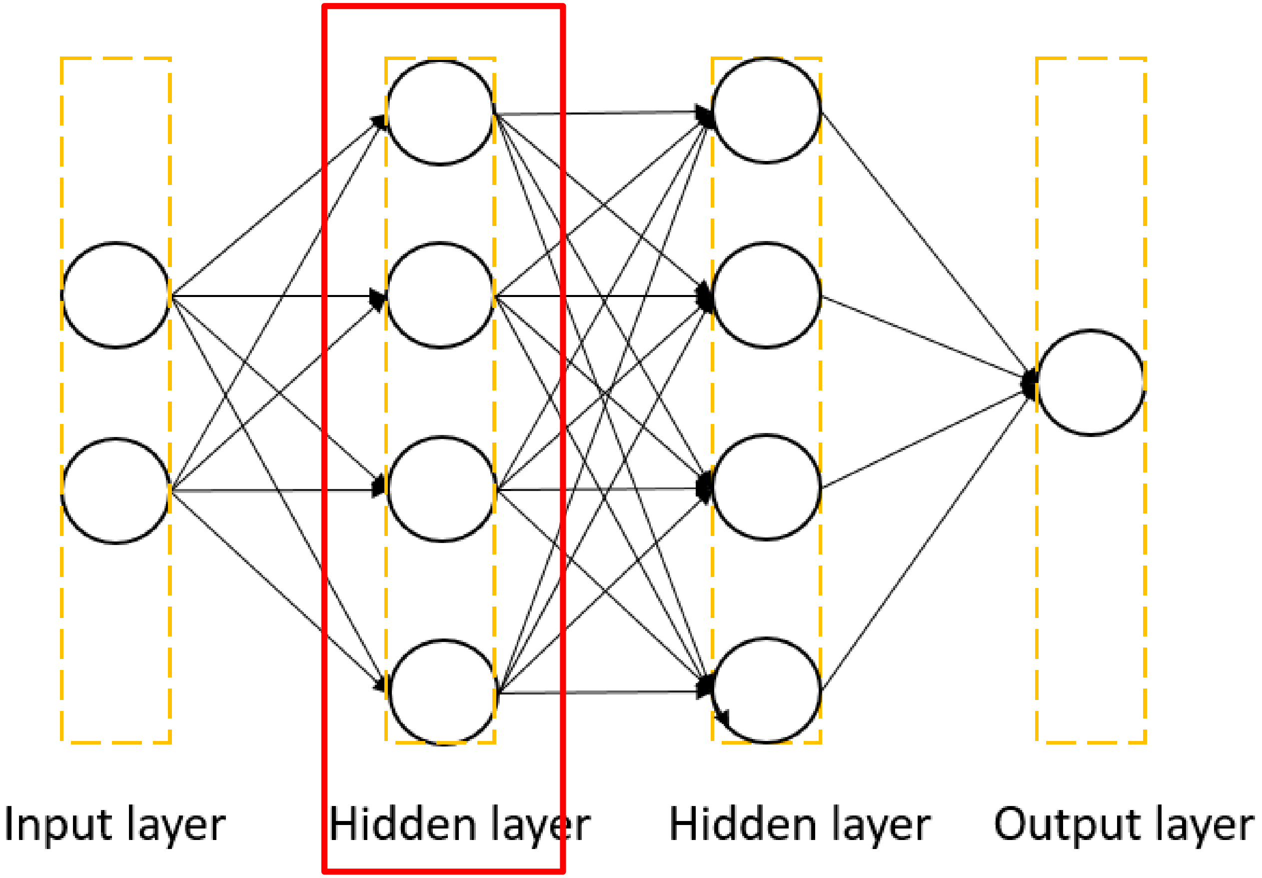
Lab Description

Lab Description

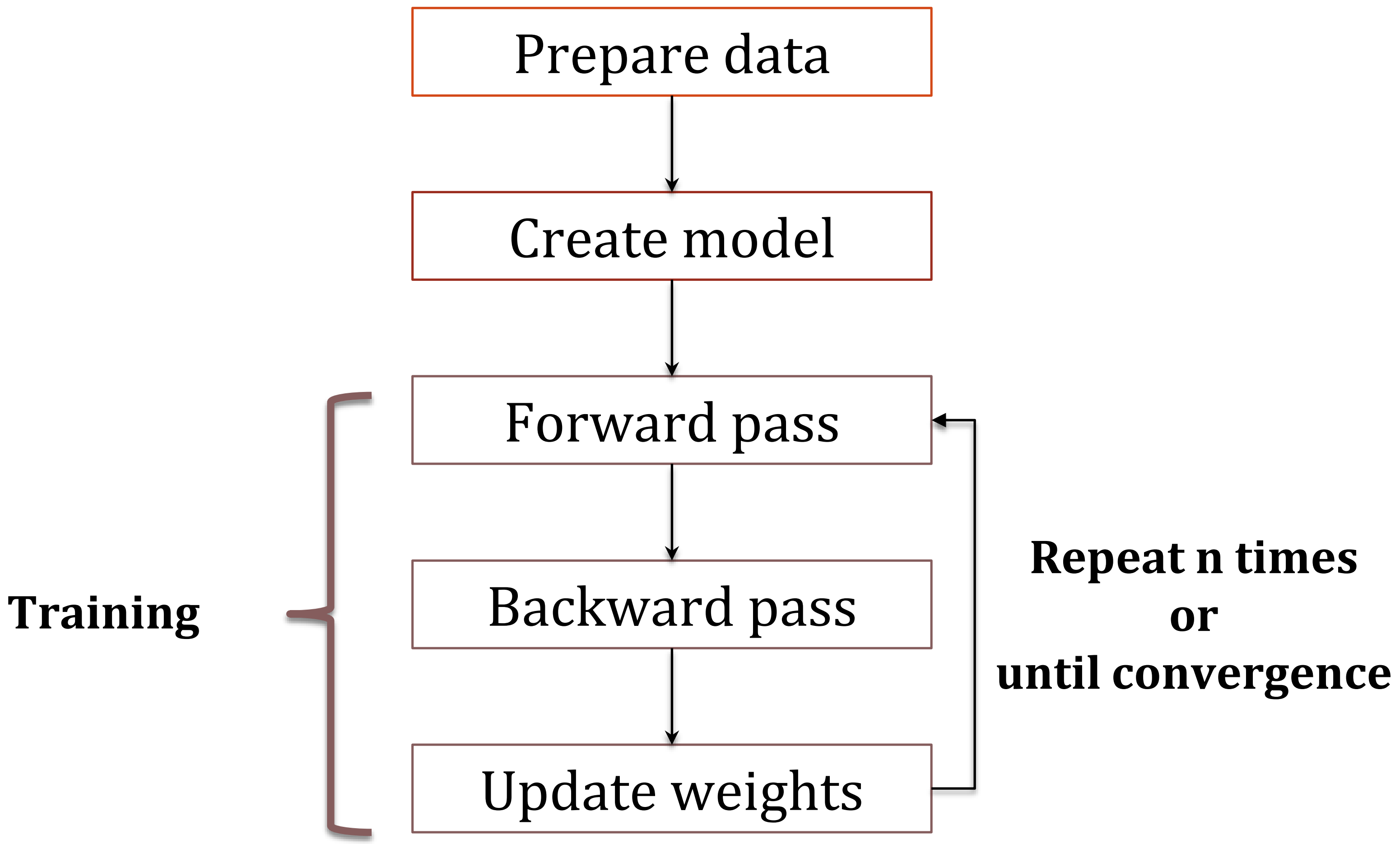
- Implement a simple neural network with two hidden layers.
- You can only use **Numpy** and other **python standard libraries**.
  - **PyTorch, TensorFlow,….. are not allowed.**
- Visualize the results in report.
  - Plot your comparison figure showing the predictions and ground truth.
  - Plot your learning curve (loss, epoch).
  - Print the accuracy of your prediction.
  - matplotlib… is allowed in this part.

Lab Description

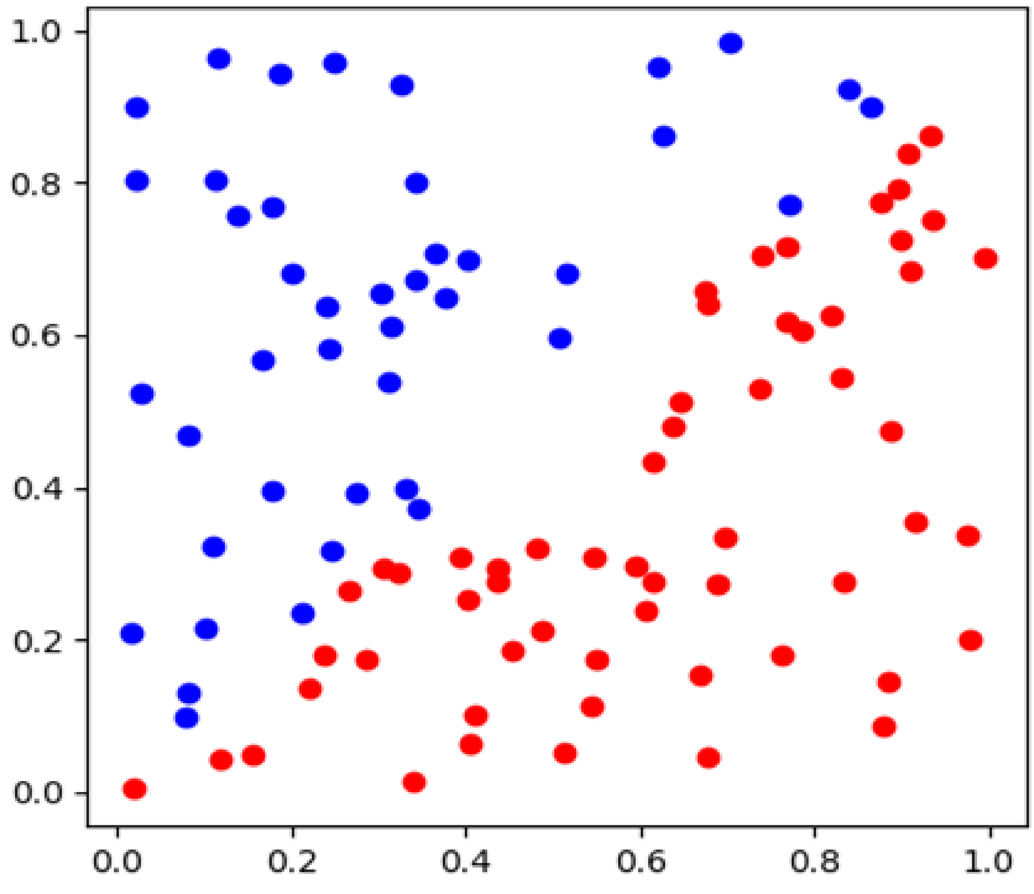
- Each Layer should contain at least **one transformation** (e.g. Linear, CNN,...) and **one activation function** (e.g. sigmoid, tanh....)



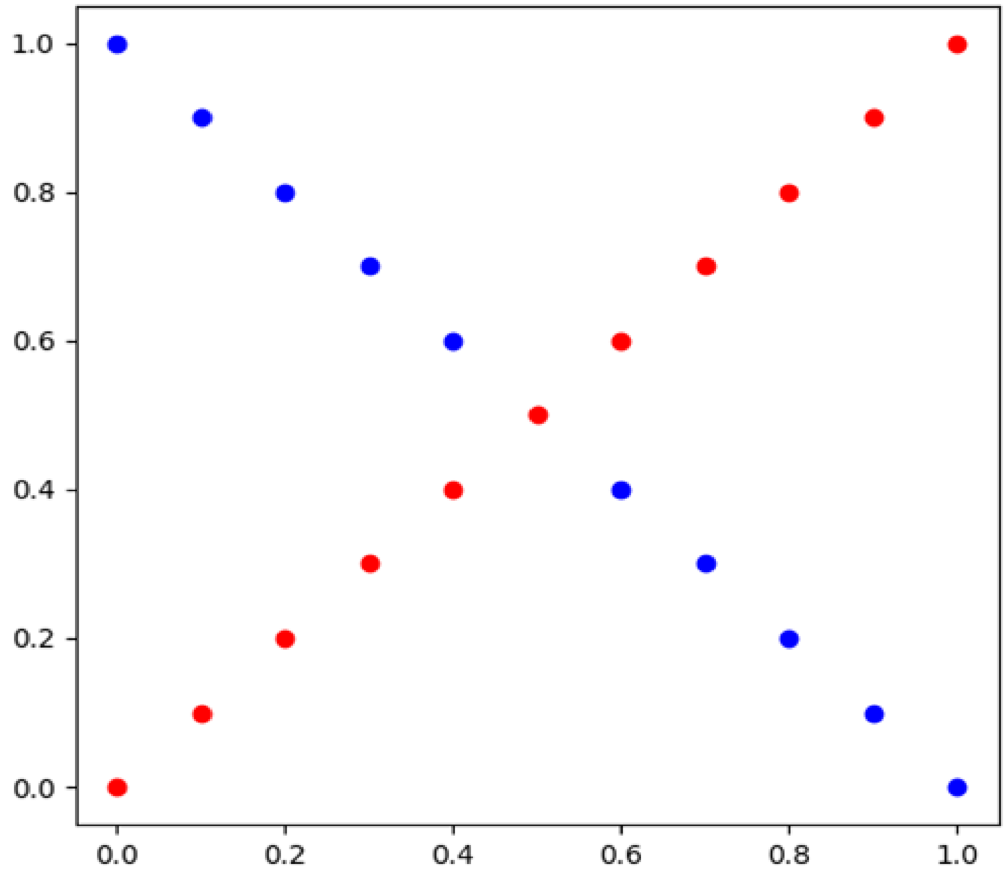
Lab Description – Flowchart



Lab Description - Data



generate_linear()



generate_XOR_easy()

Input data generate

Training & Testing use same data

```
def generate_linear(n=100):
    import numpy as np
    pts = np.random.uniform(0, 1, (n, 2))
    inputs = []
    labels = []
    for pt in pts:
        inputs.append([pt[0], pt[1]])
        distance = (pt[0]-pt[1])/1.414
        if pt[0] > pt[1]:
            labels.append(0)
        else:
            labels.append(1)
    return np.array(inputs), np.array(labels).reshape(n, 1)
```

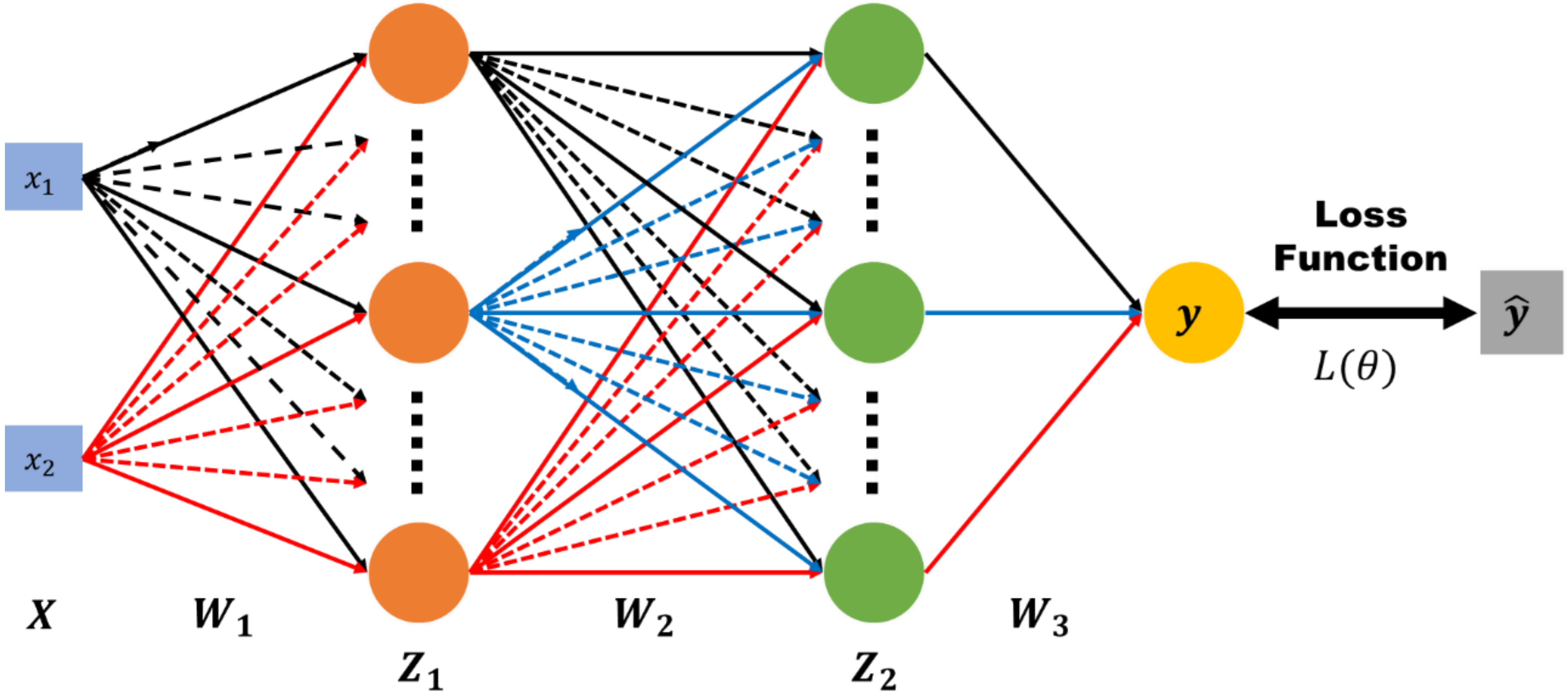
```
def generate_XOR_easy():
    import numpy as np
    inputs = []
    labels = []

    for i in range(11):
        inputs.append([0.1*i, 0.1*i])
        labels.append(0)

        if 0.1*i == 0.5:
            continue

        inputs.append([0.1*i, 1-0.1*i])
        labels.append(1)

    return np.array(inputs), np.array(labels).reshape(21, 1)
```

Don't overwrite these functions!!!

Function usage

```
x, y = generate_linear(n=100)
x, y = generate_XOR_easy()
```

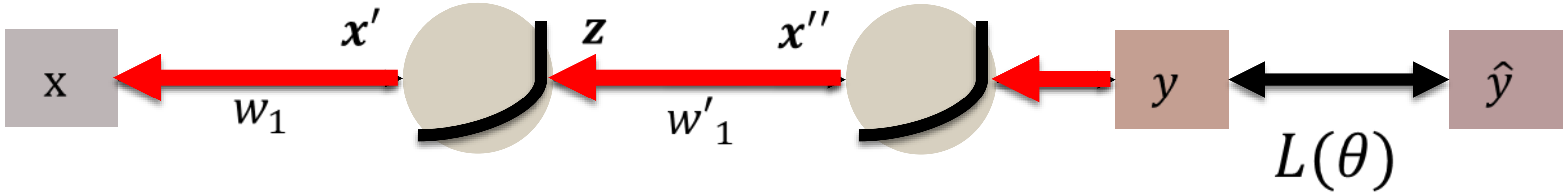
Lab Description – Architecture



X : $[x_1, x_2]$ $\quad$ y : *outputs* $\quad$ $\hat{y}$: *ground truth*

W_1, W_2, W_3 : *weight matrix of network layers*

Lab Description – Backward



$\boldsymbol{x' = xw_1}$ $\boldsymbol{z = \sigma(x')}$ $\boldsymbol{x'' = zw'_1}$ $\boldsymbol{y = \sigma(x'')}$

Chain rule

$$y = g(x) \qquad z = h(y)$$

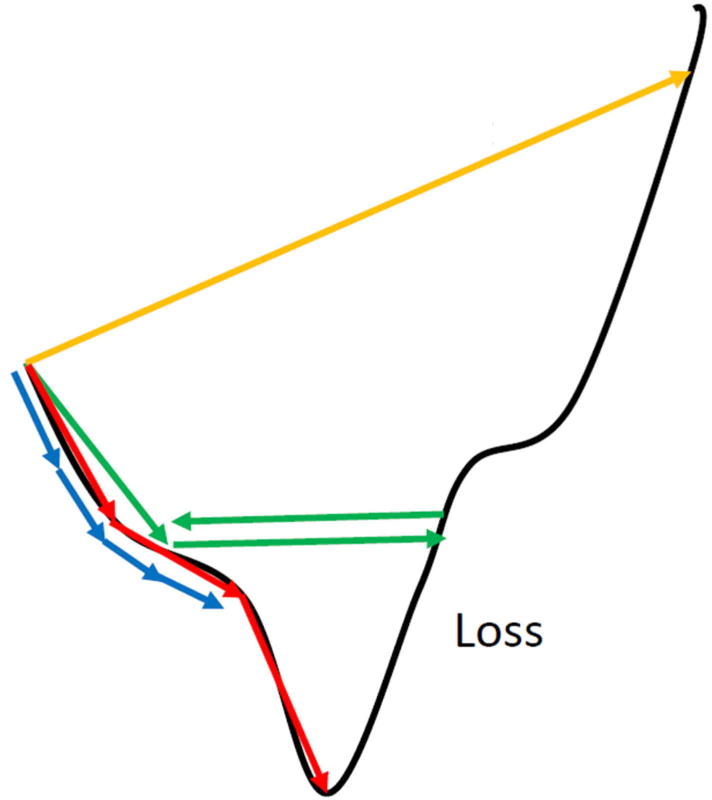
$$\mathbf{x} \overset{\mathbf{g}()}{\longrightarrow} \boldsymbol{y} \overset{\boldsymbol{h}()}{\longrightarrow} \boldsymbol{z}$$

$$\frac{\boldsymbol{dz}}{\boldsymbol{dx}} = \frac{\boldsymbol{dz}}{\cancel{\boldsymbol{dy}}} \frac{\cancel{\boldsymbol{dy}}}{\boldsymbol{dx}}$$

$$\begin{aligned}
\frac{\partial L(\theta)}{\partial w_1} &= \frac{\partial y}{\partial w_1} \frac{\partial L(\theta)}{\partial y} \\
&= \frac{\partial x''}{\partial w_1} \frac{\partial y}{\partial x''} \frac{\partial L(\theta)}{\partial y} \\
&= \frac{\partial z}{\partial w_1} \frac{\partial x''}{\partial z} \frac{\partial y}{\partial x''} \frac{\partial L(\theta)}{\partial y} \\
&= \frac{\partial x'}{\partial w_1} \frac{\partial z}{\partial x'} \frac{\partial x''}{\partial z} \frac{\partial y}{\partial x''} \frac{\partial L(\theta)}{\partial y}
\end{aligned}$$

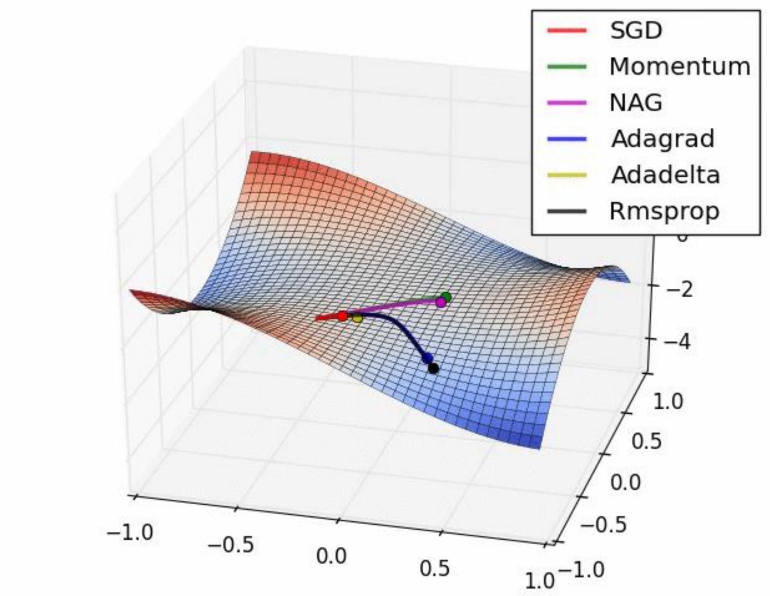
Lab Description – Gradient descent

Network Parameters $\theta = \{w_1, w_2, w_3, w_4, \cdots\}$



$$\theta^1 = \theta^0 - \rho \nabla L(\theta^0)$$
$$\theta^2 = \theta^1 - \rho \nabla L(\theta^1)$$
$$\theta^3 = \theta^2 - \rho \nabla L(\theta^2)$$

ρ : Learning rate



http://www.denizyuret.com/2015/03/alec-radfords-animations-for.html

Lab Description - Prediction

- In the training, you need to print loss

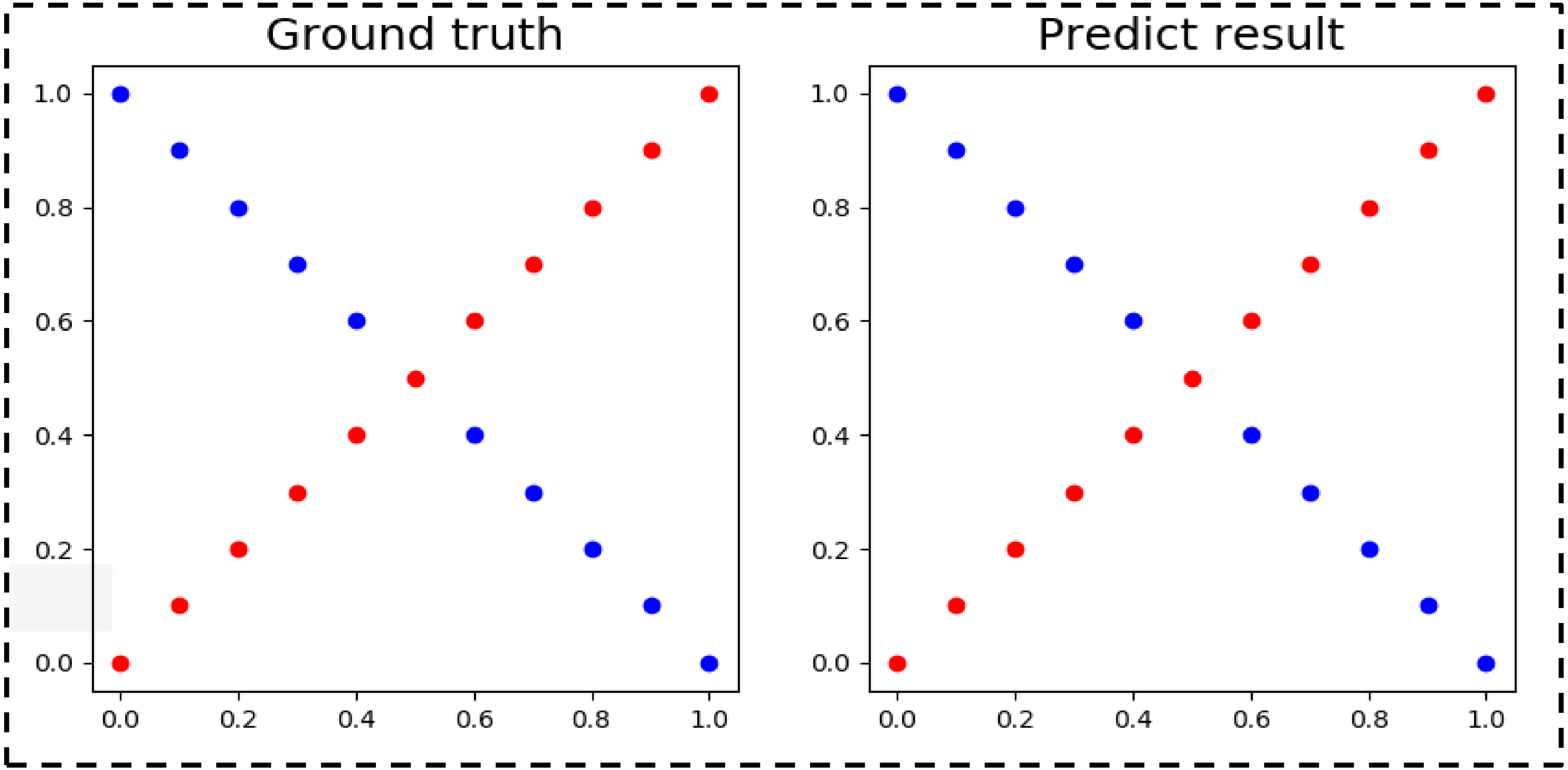
```
epoch 10000 loss : 0.16234523253277644
epoch 15000 loss : 0.2524336634177614
epoch 20000 loss : 0.1590783047540092
epoch 25000 loss : 0.22099447030234853
epoch 30000 loss : 0.3292173477217561
epoch 35000 loss : 0.40406233282426085
epoch 40000 loss : 0.43052897480298924
epoch 45000 loss : 0.4207525735586605
epoch 50000 loss : 0.3934759509342479
epoch 55000 loss : 0.3615008372106921
epoch 60000 loss : 0.33077879872648525
epoch 65000 loss : 0.30333537090819584
epoch 70000 loss : 0.2794858089741792
epoch 75000 loss : 0.25892812312991587
epoch 80000 loss : 0.24119780823897027
epoch 85000 loss : 0.22583656353511342
epoch 90000 loss : 0.21244497028971704
epoch 95000 loss : 0.2006912468389013
```

- In the testing, you need to show your predictions, also the accuracy

```
Iter91 |     Ground truth: 1.0 |     prediction: 0.99943 |
Iter92 |     Ground truth: 1.0 |     prediction: 0.99987 |
Iter93 |     Ground truth: 1.0 |     prediction: 0.99719 |
Iter94 |     Ground truth: 1.0 |     prediction: 0.99991 |
Iter95 |     Ground truth: 0.0 |     prediction: 0.00013 |
Iter96 |     Ground truth: 1.0 |     prediction: 0.77035 |
Iter97 |     Ground truth: 1.0 |     prediction: 0.98981 |
Iter98 |     Ground truth: 1.0 |     prediction: 0.99337 |
Iter99 |     Ground truth: 0.0 |     prediction: 0.20275 |
loss=0.03844 accuracy=100.00%
```

Lab Description - Prediction

- Visualize the predictions and ground truth at the end of the training process



Scoring Criteria

Scoring Criteria

- Report (40%)
- Demo(60%)
  - Experimental results (40%)
    - **You have to achieve at least 90% of accuracy to get the demo score.**
  - Questions (20%)
- Extra (10%)
  - Implement different optimizers. (2%)
  - Implement different activation functions. (3%)
  - Implement convolutional layers. (5%)

Report format

1. Introduction (20%)
2. Experiment setups (30%):
   A. Sigmoid functions
   B. Neural network
   C. Backpropagation
3. Results of your testing (20%)
   A. Screenshot and comparison figure
   B. Show the accuracy of your prediction
   C. Learning curve (loss, epoch curve)
   D. Anything you want to present
4. Discussion (30%)
   A. Try different learning rates
   B. Try different numbers of hidden units
   C. Try without activation functions
   D. Anything you want to share
5. Extra (10%)
   A. Implement different optimizers. (2%)
   B. Implement different activation functions. (3%)
   C. Implement convolutional layers. (5%)

Time Schedule

Important Date

- Assignment Deadline: 7/16 (Tue.) 18:30
- Demo date: 7/16 (Tue.)
- Zip all files in one file
  - Report (report.pdf)
  - Source code
- Name it like「DL_LAB1_yourstudentID_name.zip」
  - ex:「DL_LAB1_312554018_曾昱仁.zip」
  - **If the zip file name or the report spec have format error, you will get a penalty of 5 points.**

Time schedule

	LAB1 Back-Propagation	LAB2 CNN	LAB3 CNN	LAB4 VAE	LAB5 MaskGIT	LAB6 Generative Models
Announce	7/9 (Tabc)	7/16 (Tabc)	7/23 (Tabc)	7/30 (Tabc)	8/6 (Tabc)	8/13 (Tabc)
DEMO	7/16 (Tabc)	7/23 (Tabc)	7/30 (Tabc)	TBD	TBD	No demo

Demo schedule

- Demo date: 7/16 (Tue.)
  - The Lab 2 announcement is scheduled for 7/16 at the beginning of the class, and the demo will start at around 7 pm.
  - Each person will be allocated approximately 5 minutes.
  - If the scheduled time is inconvenient, please contact the respective TA to arrange an alternative demo session.

link

	Google meet link		
	TA	曾昱仁	OK/Please come in
14:00	student 1		
14:05	student 2		
14:10	student 3		
14:15			
14:20			
14:25			
14:30			

Reference

1. *http://www.denizyuret.com/2015/03/alec-radfords-animations-for.html*
2. *http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML17_2.html*