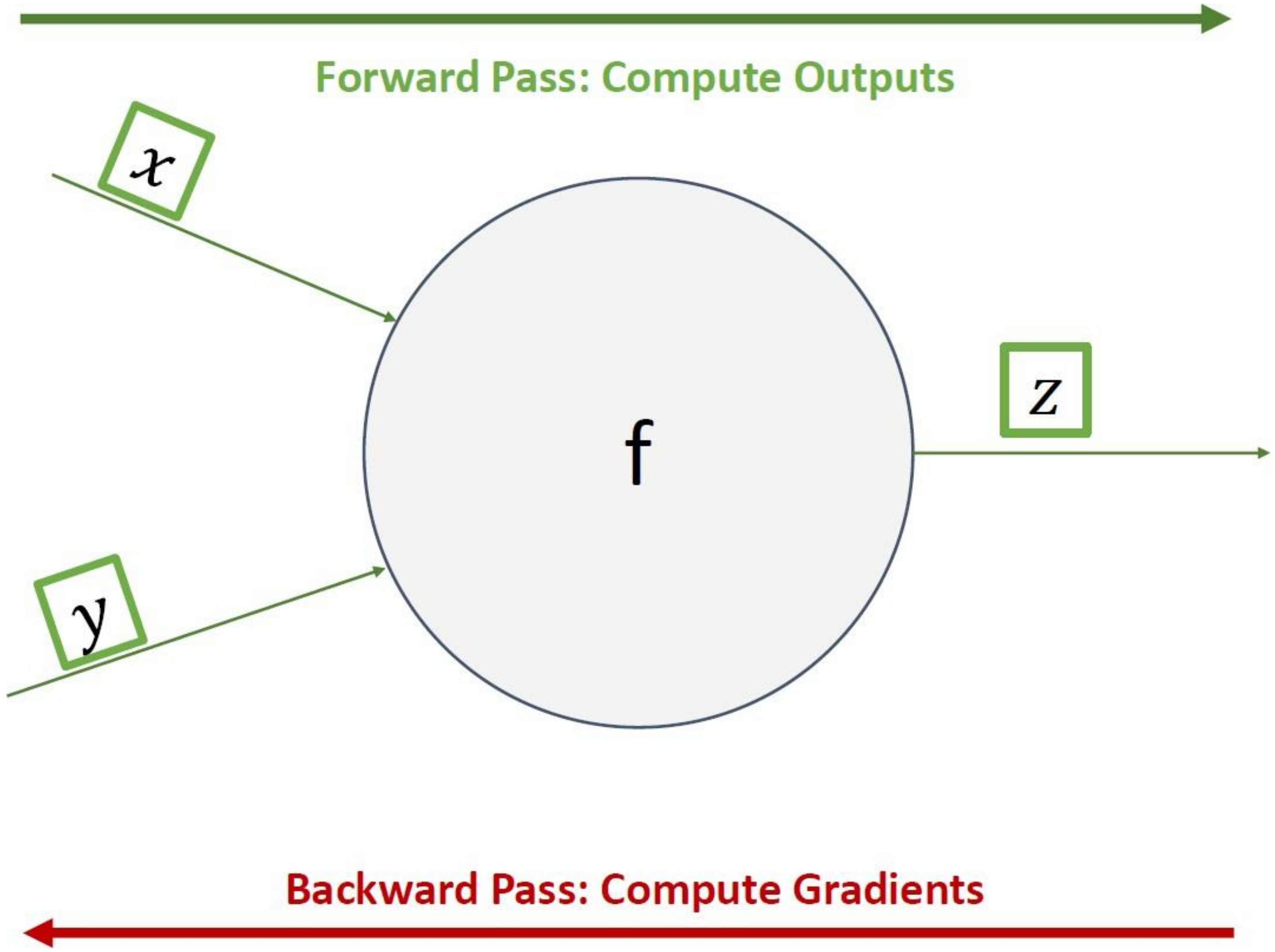
Introduction to Backpropagation

Sejong RCV – 임근택

Forward Pass / Backward Pass



Loss function

A **loss function** tells how good our current classifier is

Low loss = good classifier
High loss = bad classifier

(Also called: **objective function**; **cost function**)

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^{N}$$

Where x_i is an vector (image) and y_i is an integer-valued label

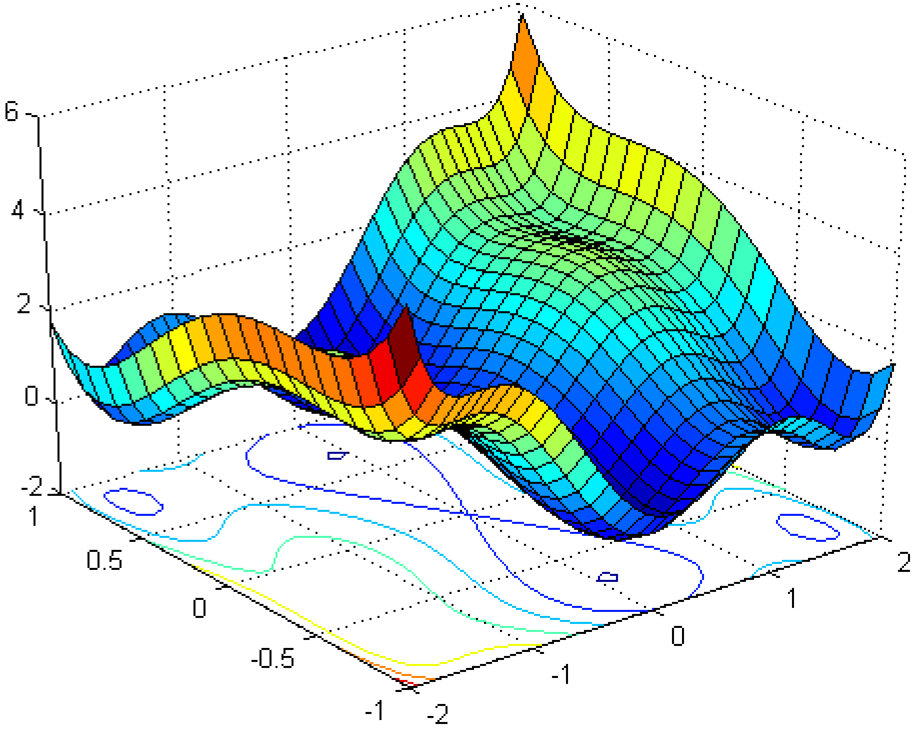
Loss for a single example is

$$L_i = L(f(x_i, W), y_i)$$

Loss for the dataset is average of per-example losses:

분류기의 Loss ➔

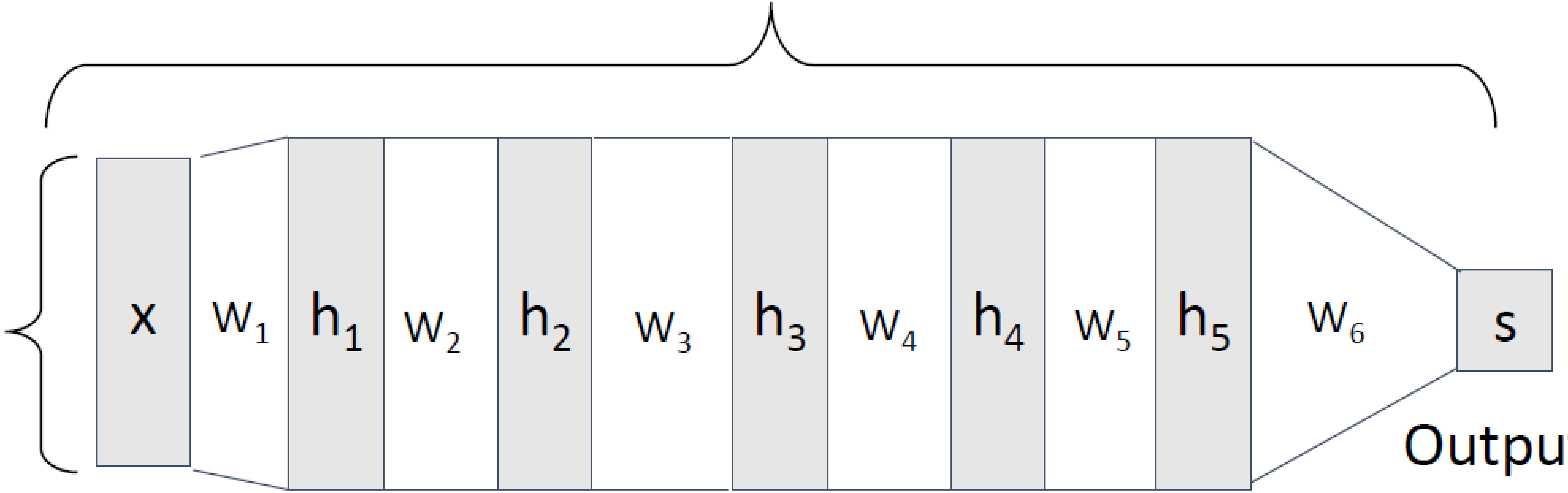
$$\phi(W) = \frac{1}{N}\sum_{i=1}^{N} L(f(x_i, W), y_i)$$



Deep Neural Network

Depth = number of layers

Width:
Size of
each
layer



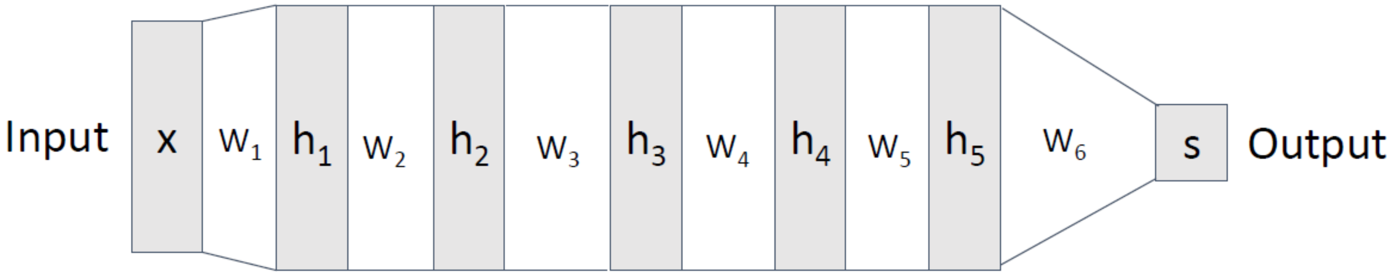
Output: 10

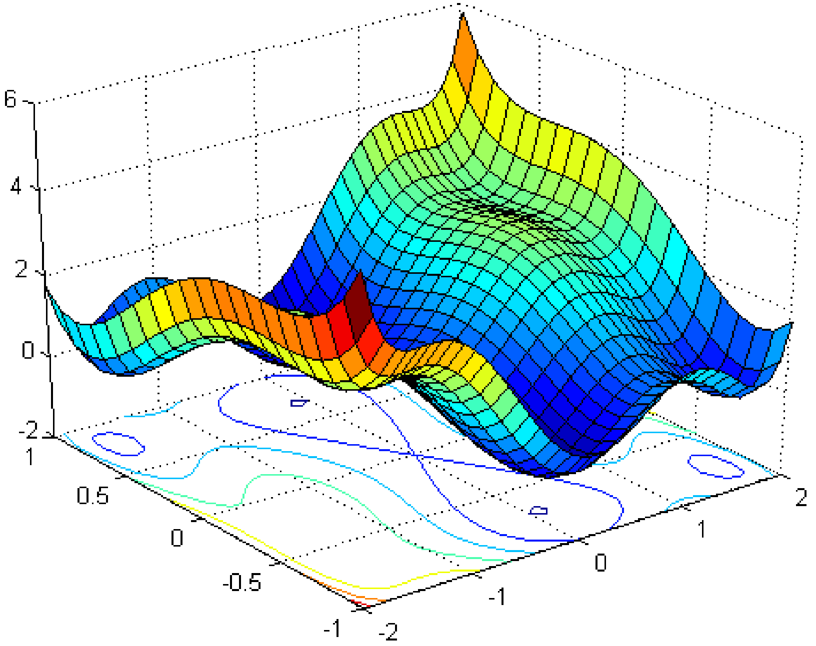
Input:
3072

Gradient Descent

Goal:
find the **w** minimizing some loss function L.

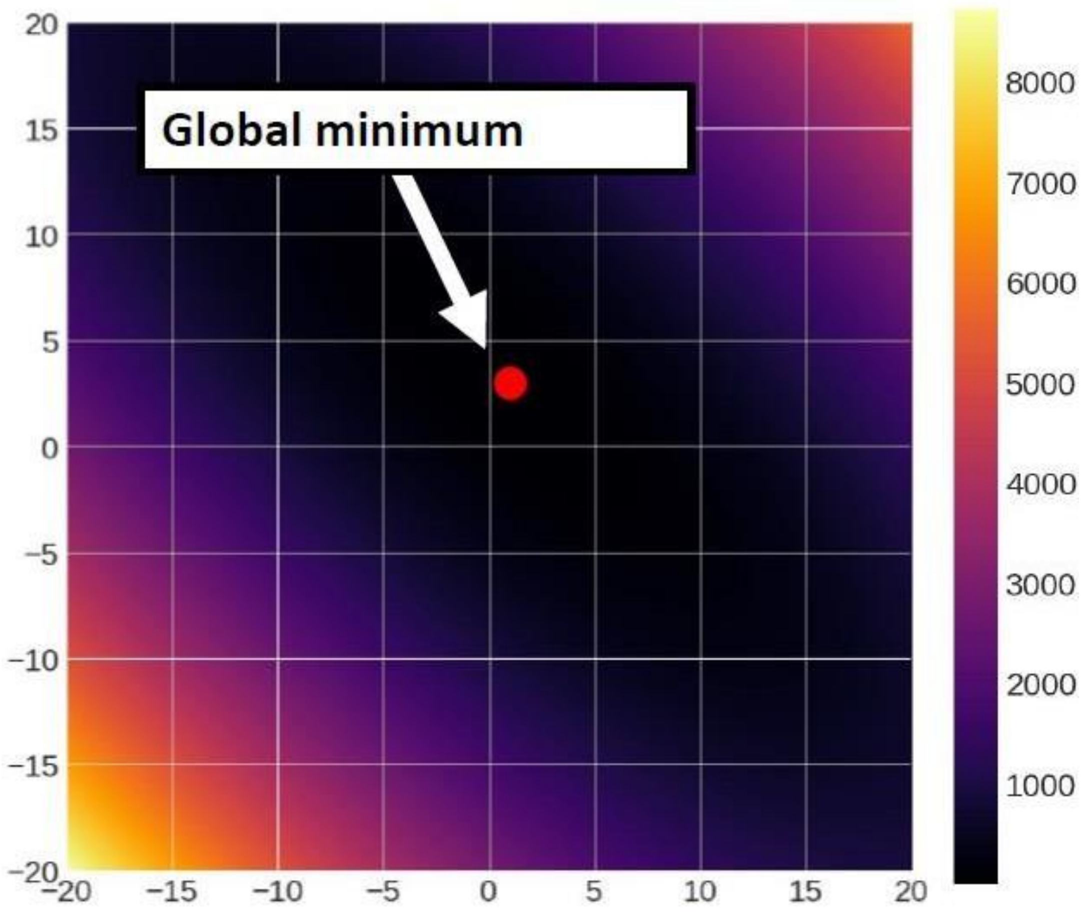
$$\arg\min_{\boldsymbol{w} \in R^N} L(\boldsymbol{w})$$





Gradient Descent

$$f(x,y) = (x+2y-7)^2 + (2x+y-5)^2$$



Gradient Descent

Method: at each step, move in direction of negative gradient

w0 = *initialize*() #initialize

for iter in range(numIters):

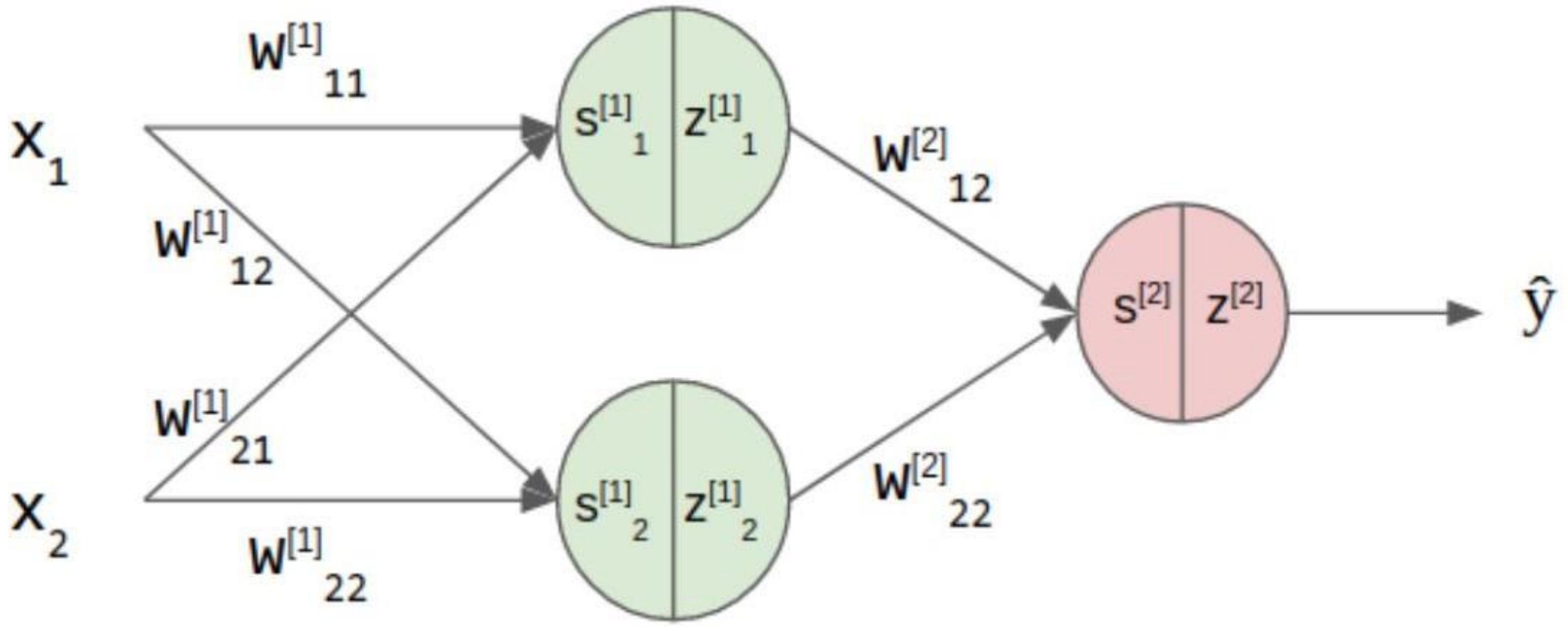
    g = $\nabla_{w} L(\boldsymbol{w})$ # eval gradient

    w = **w** + -*stepsize****g** # update w

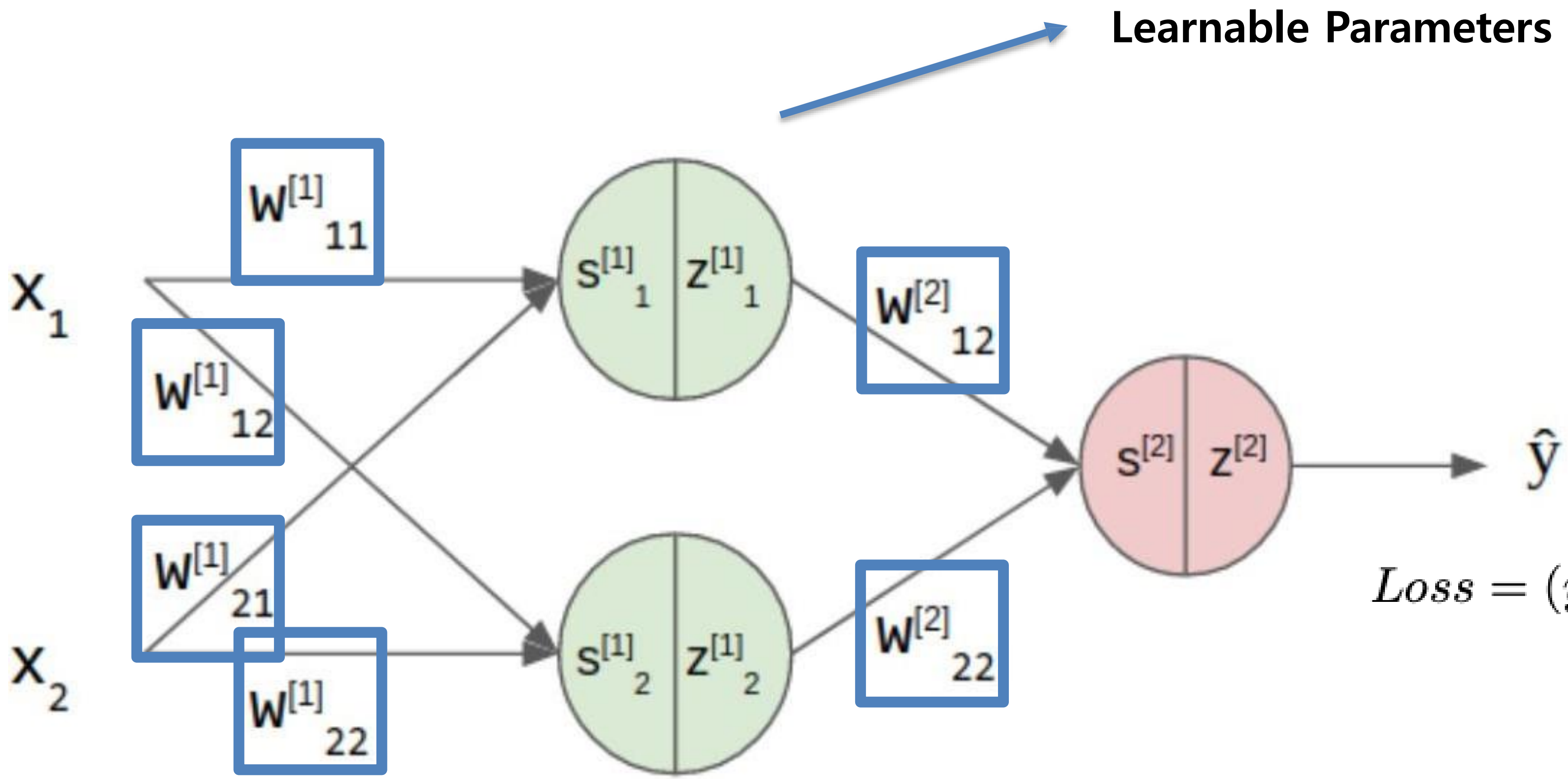
return **w**

Learning rate

Backward Pass (Backpropagation)



Backward Pass (Backpropagation)



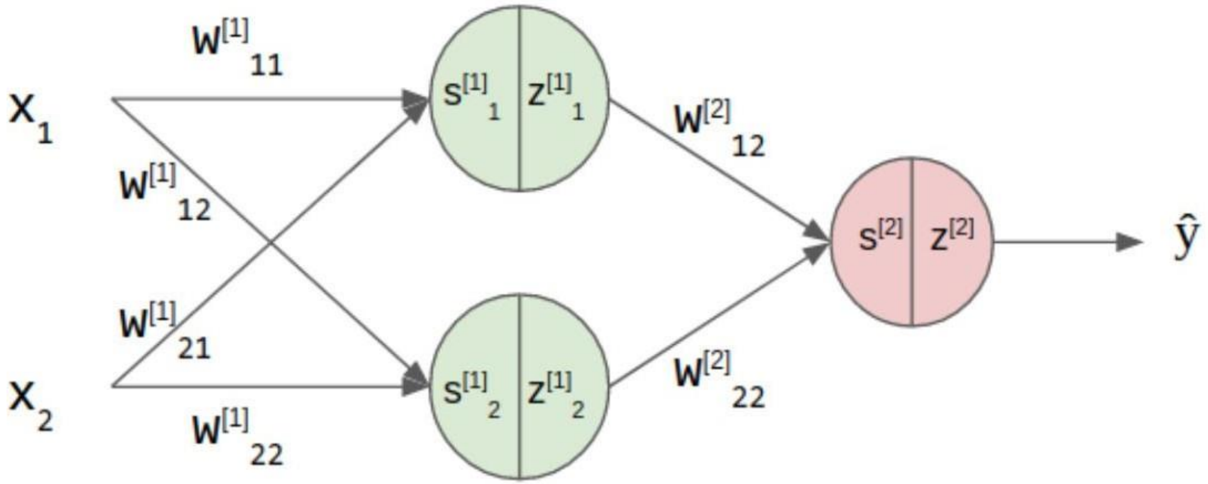
$$Loss = (y - \hat{y})^2$$

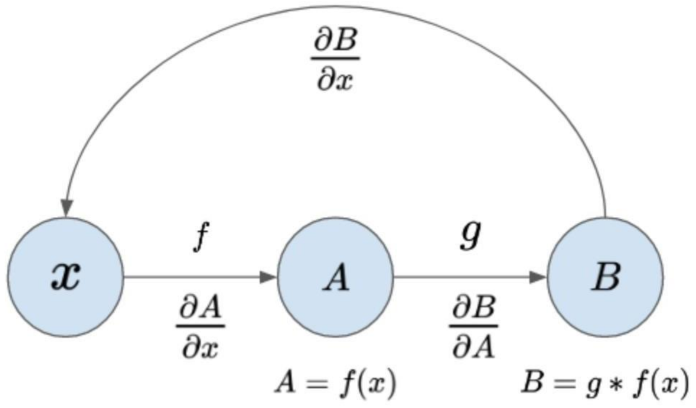
Then, How to calculate gradient on each layer ?

$$\frac{\partial Loss}{\partial w_{11}^{(1)}}, \frac{\partial Loss}{\partial w_{12}^{(1)}}, \dots, \frac{\partial Loss}{\partial w_{22}^{(2)}},$$

Backward Pass (Backpropagation)

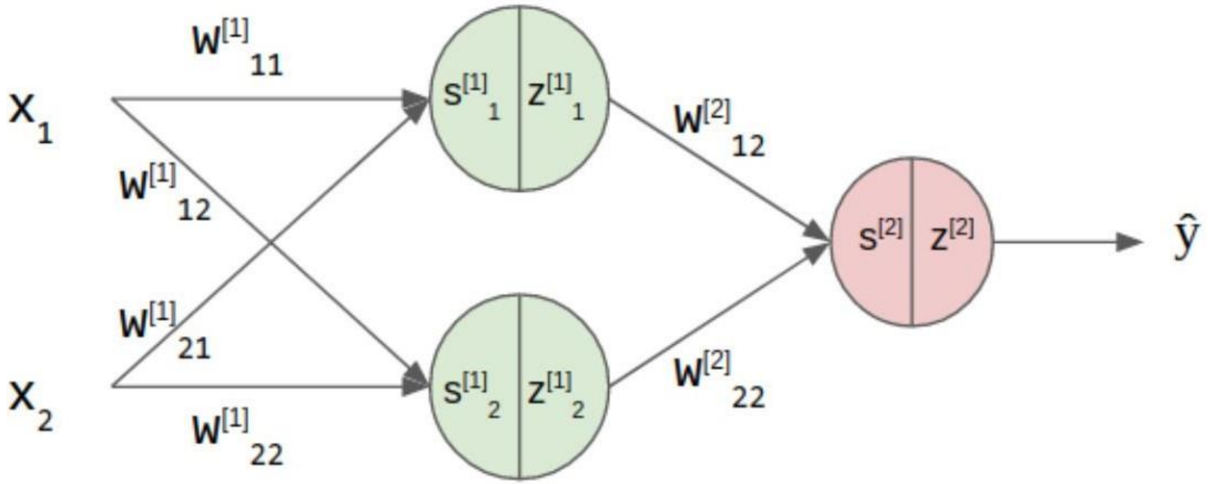
Let's calculate gradient using chain rule !





Backward Pass (Backpropagation)

Let's calculate gradient using chain rule !



$$s_1^{[1]} = w_{11}^{[1]} x_1 + w_{21}^{[1]} x_2$$

$$z_1^{[1]} = tanh(s_1^{[1]})$$

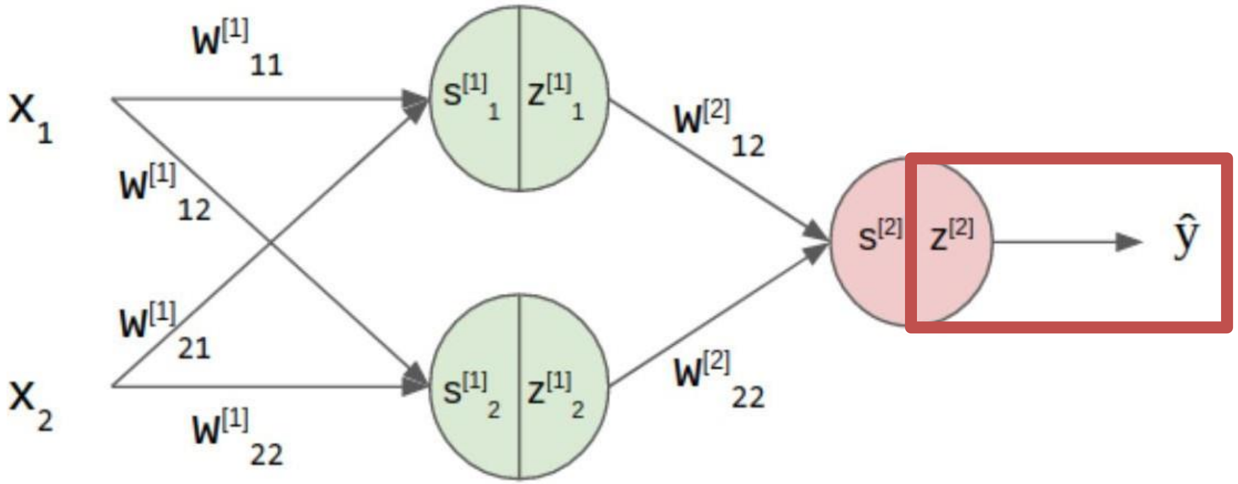
$$s^{[2]} = w_{12}^{[2]} z_1^{[1]} + w_{22}^{[2]} z_2^{[1]}$$

$$z^{[2]} = tanh(s^{[2]})$$

$$L = (y - z^{[2]})^2$$

Backward Pass (Backpropagation)

Let's calculate gradient using chain rule !



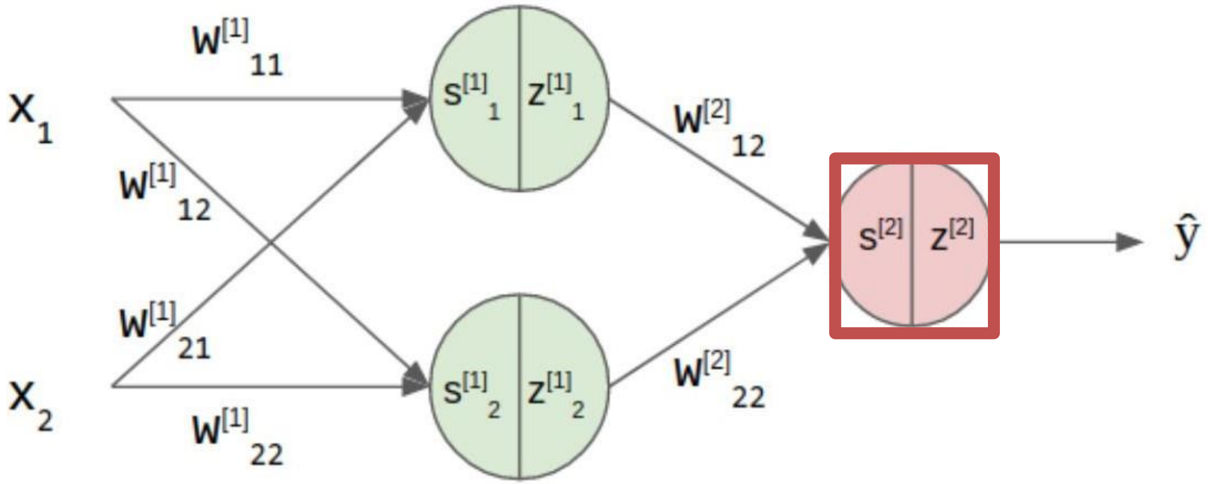
$$L = (y - z^{[2]})^2$$

$$\frac{\partial L}{\partial z^{[2]}} = 2(y - z^{[2]})$$

$$\frac{\partial L}{\partial w_{11}^{[1]}} = \frac{\partial L}{\partial z^{[2]}} \times \dots$$

Backward Pass (Backpropagation)

Let's calculate gradient using chain rule !



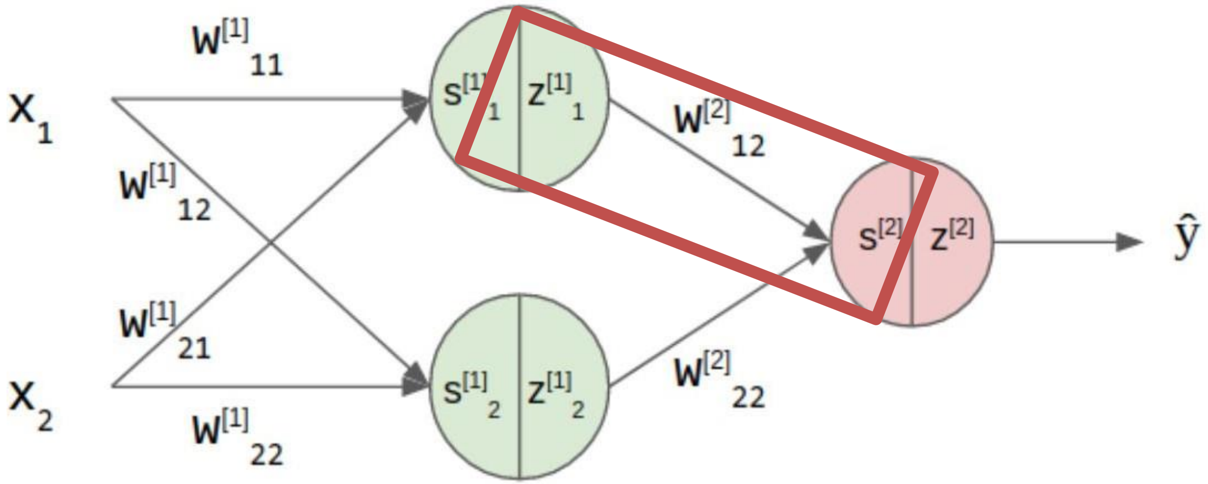
$$z^{[2]} = tanh(s^{[2]})$$

$$\frac{\partial z^{[2]}}{\partial s^{[2]}} = (1 - (z^{[2]})^2)$$

$$\frac{\partial L}{\partial w_{11}^{[1]}} = \frac{\partial L}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial s^{[2]}} \times \dots$$

Backward Pass (Backpropagation)

Let's calculate gradient using chain rule !



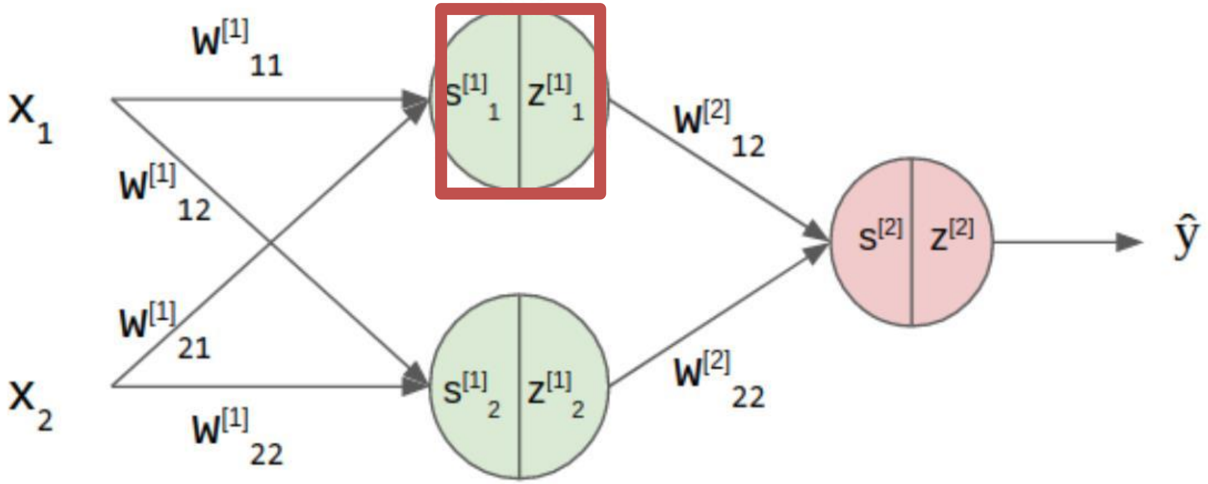
$$s^{[2]} = w_{12}^{[2]} z_1^{[1]} + w_{22}^{[2]} z_2^{[1]}$$

$$\frac{\partial s^{[2]}}{\partial z_1^{[1]}} = w_{12}^{[2]}$$

$$\frac{\partial L}{\partial w_{11}^{[1]}} = \frac{\partial L}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial s^{[2]}} \times \frac{\partial s^{[2]}}{z_1^{[1]}} \times \dots$$

Backward Pass (Backpropagation)

Let's calculate gradient using chain rule !



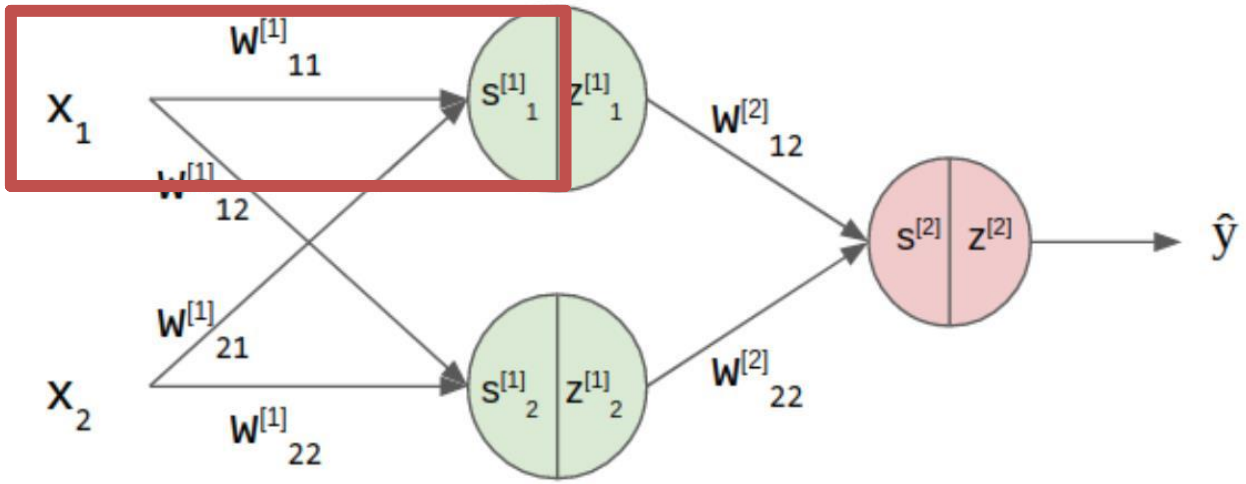
$$z_1^{[1]} = tanh(s_1^{[1]})$$

$$\frac{\partial z_1^{[1]}}{\partial s_1^{[1]}} = (1 - (s_1^{[1]})^2)$$

$$\frac{\partial L}{\partial w_{11}^{[1]}} = \frac{\partial L}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial s^{[2]}} \times \frac{\partial s^{[2]}}{z_1^{[1]}} \times \frac{\partial z_1^{[1]}}{\partial s_1^{[1]}} \times \dots$$

Backward Pass (Backpropagation)

Let's calculate gradient using chain rule !



$$s_1^{[1]} = w_{11}^{[1]} x_1 + w_{21}^{[1]} x_2$$

$$\frac{\partial s_1^{[1]}}{\partial w_{11}^{[1]}} = x_1$$

$$\frac{\partial L}{\partial w_{11}^{[1]}} = \frac{\partial L}{\partial z^{[2]}} \times \frac{\partial z^{[2]}}{\partial s^{[2]}} \times \frac{\partial s^{[2]}}{z_1^{[1]}} \times \frac{\partial z_1^{[1]}}{\partial s_1^{[1]}} \times \frac{\partial s_1^{[1]}}{\partial w_{11}^{[1]}}$$