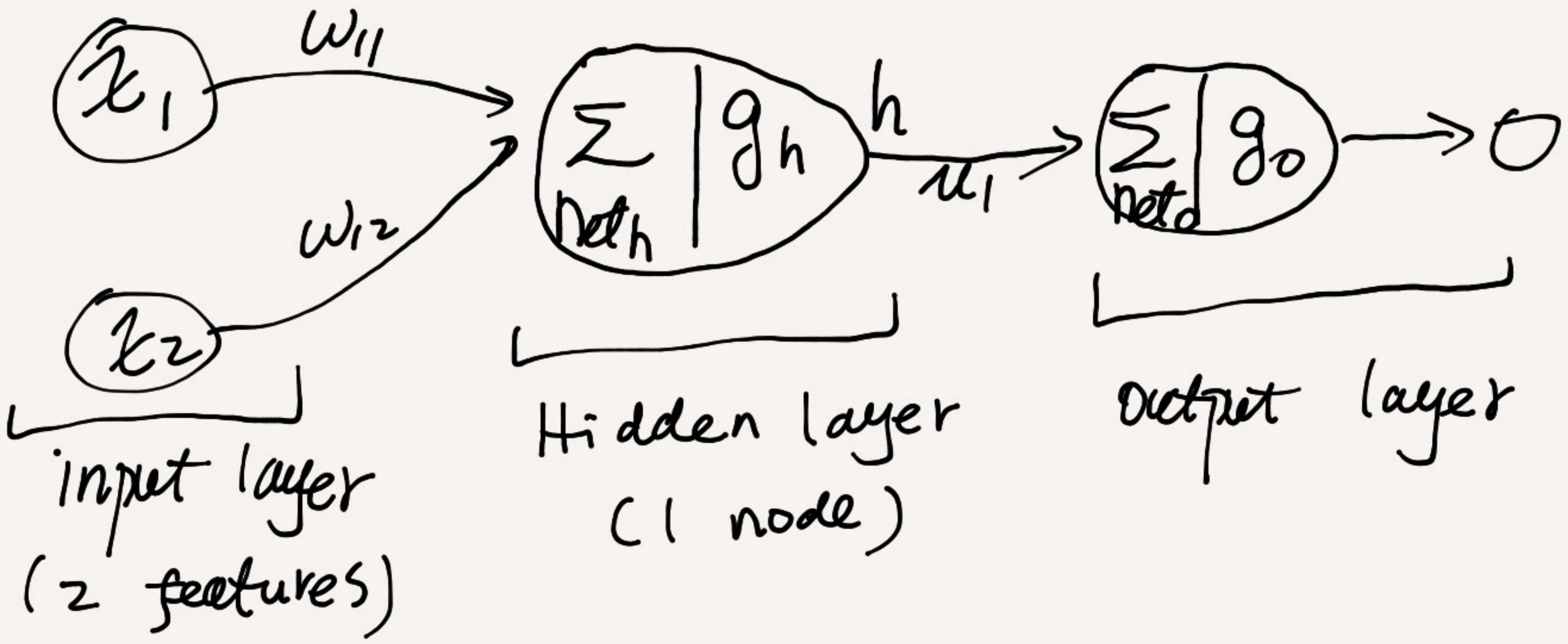
Lecture 9. ANNs (Chapter 2. The science of DL)

3-layer NN:



4) Final model function

$$F(x) = g_o(u_1 \cdot h) = g_o(u_1 \cdot g_h(net_h))$$

$$= g_o(u_1 \cdot g_h(w_{11} \cdot x_1 + w_{12} \cdot x_2))$$

1. NN model

1) Input layer. no calculations.

\# input nodes = # features.

2) Hidden layer(s)

net_h (pre-activation / net input): $net_h = w_{11} \cdot x_1 + w_{12} \cdot x_2$

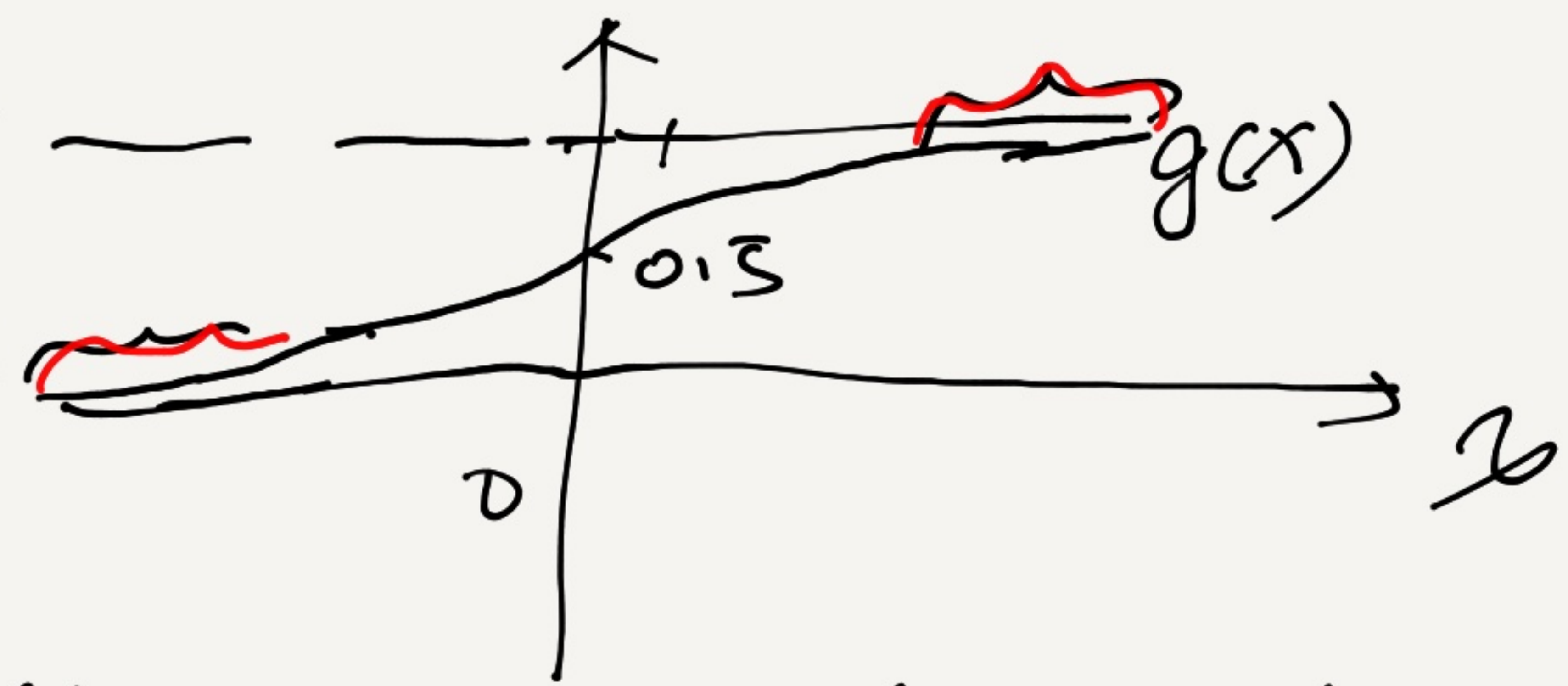
g_h (activation function): $h = g_h(net_h) = g_h(w_{11} x_1 + w_{12} \cdot x_2)$

3) Output layer. $net_o = u_1 \cdot h$, $o = g_o(net_o) = g_o(u_1 h)$

2. Activation functions.

1) Sigmoid (logistic) function.

$$g(x) = \frac{1}{1+e^{-x}} \in (0,1)$$



prior to 2011, it was the most popular activation function

It can be used for both the $\underbrace{\text{hidden layer(s)}}_{g_h}$ and $\underbrace{\text{output layer}}_{g_o}$

If $x \to +\infty$, or $x \to -\infty$, $g(x)$ becomes very flat.

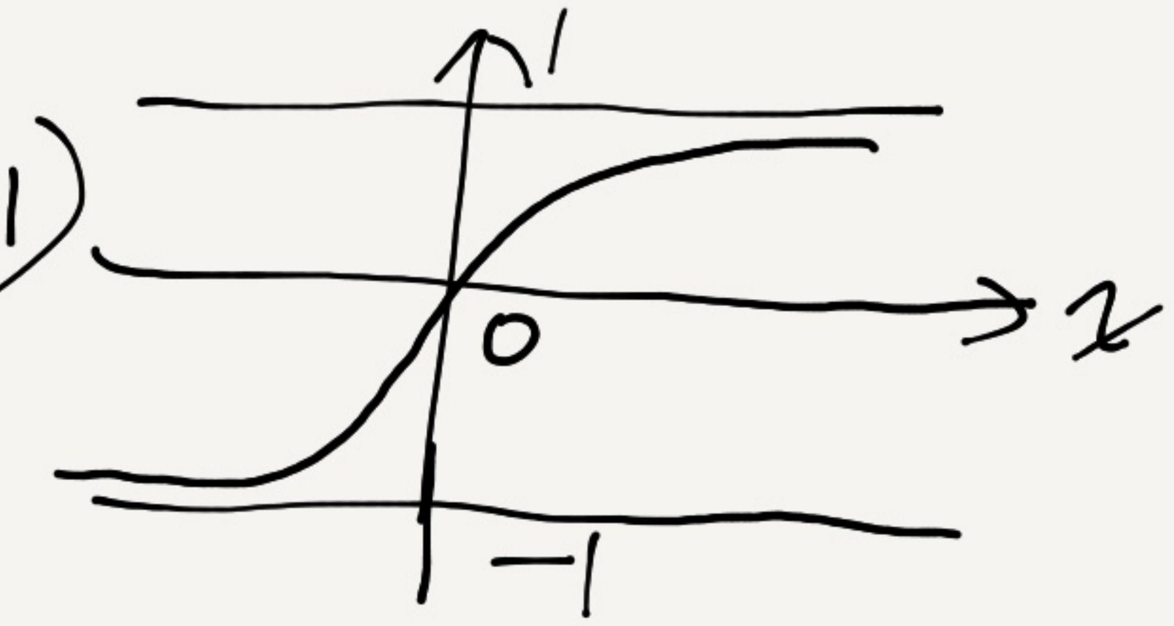
$$\to \nabla g(x) \to 0$$

$$w^{i+1} = w^{i} - \varepsilon \cdot \overbrace{\nabla g(x)}^{\to 0} \to w^{i+1} = w^{i}$$

<u>gradient vanishing</u>

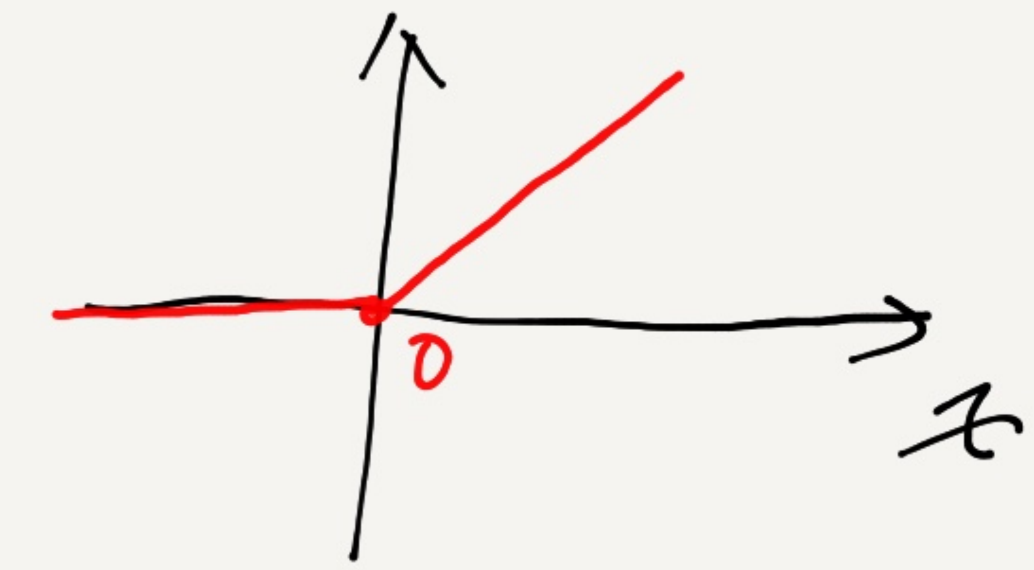
2) hyporbolic tangent.

$$g(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \in (-1, 1)$$



3) Rectified Linear Units (ReLu)

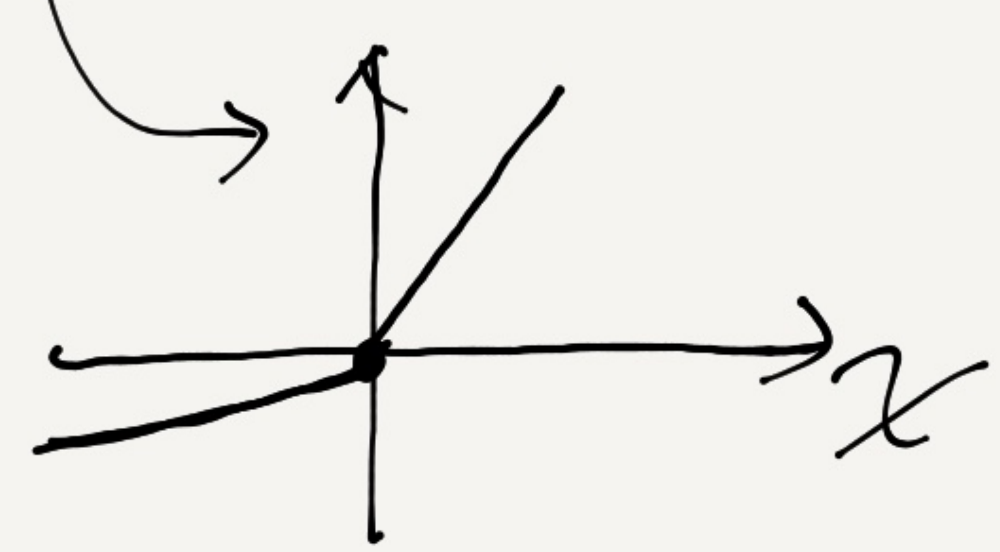
$$g(x) = \max\{0, x\} = \begin{cases} x, & \text{if } x > 0 \\ 0 & \text{otherwise.} \end{cases}$$



if $x > 0$, $\nabla g(x) = 1$

ReLu becomes the default activation function for hidden layers

leaky ReLu: $g(x) = \underbrace{\max\{0, x\}}_{x>0} - \underbrace{\alpha \cdot \max\{0, -x\}}_{x<0} = \begin{cases} x, & x > 0 \\ -\alpha x & x \le 0 \end{cases}$

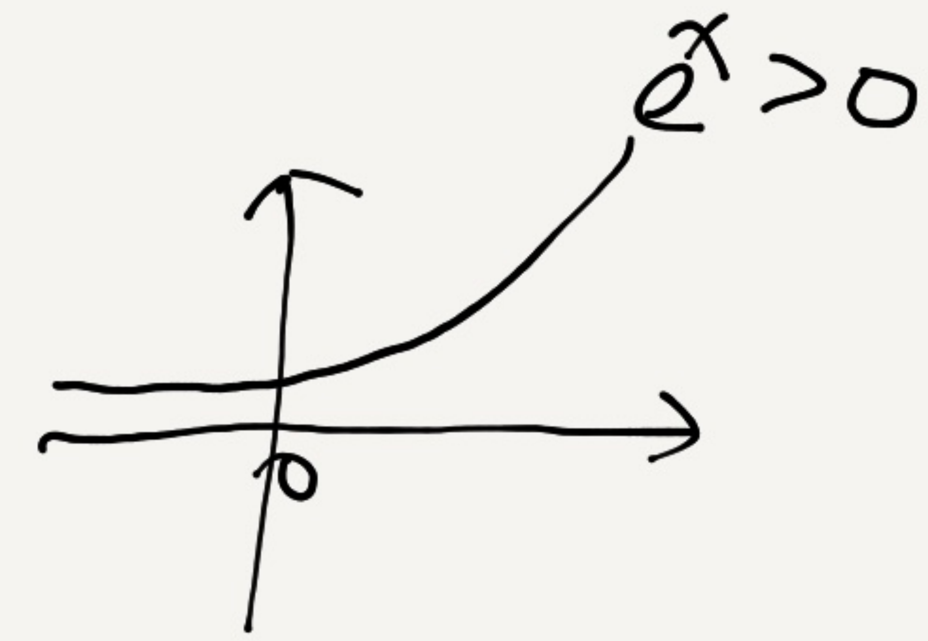


Swiss (2017): $f(x) = x \cdot \text{sigmoid}(\beta x)$

Smooth transition at point 0.

4) softmax function. (output layer)

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} \qquad \mathrm{softmax}(x) = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_m \end{pmatrix} \qquad S_i = \frac{e^{x_i}}{\sum_{i=1}^{m} e^{x_i}}$$



$$s_i > 0, \quad \sum_{i=1}^{m} S_i = 1$$

Softmax is good for multi-class classification problem.

$x \rightarrow$ model $\rightarrow$

class 1 $\rightarrow s_1 \in (0,1)$

class 2 $\rightarrow s_2 \in (0,1)$

$\vdots$

class m $\rightarrow S_m \in (0,1)$

3, Loss function

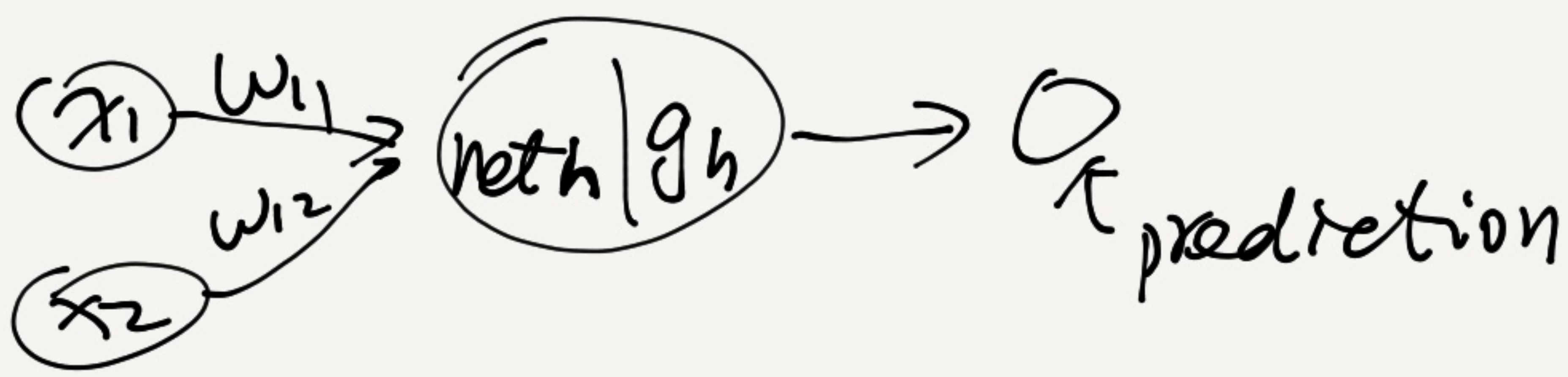
$$L_0 = \frac{1}{2}(y - o)^2 \quad \text{(single data sample)}$$

prediction

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - o_i)^2 \quad \text{(multiple samples)}$$

4. Optimization.

1) 2-layer NN



model : $O = g_h(w_{11} \cdot x_1 + w_{12} \cdot x_2)$

Loss : $L(w_{11}, w_{12}) = \frac{1}{2}(y - O)^2$

$$= \frac{1}{2}\Big(y - \overbrace{g_h(w_{11}x_1 + w_{12}x_2)}^{O}\Big)^2$$

Optimization using GD :

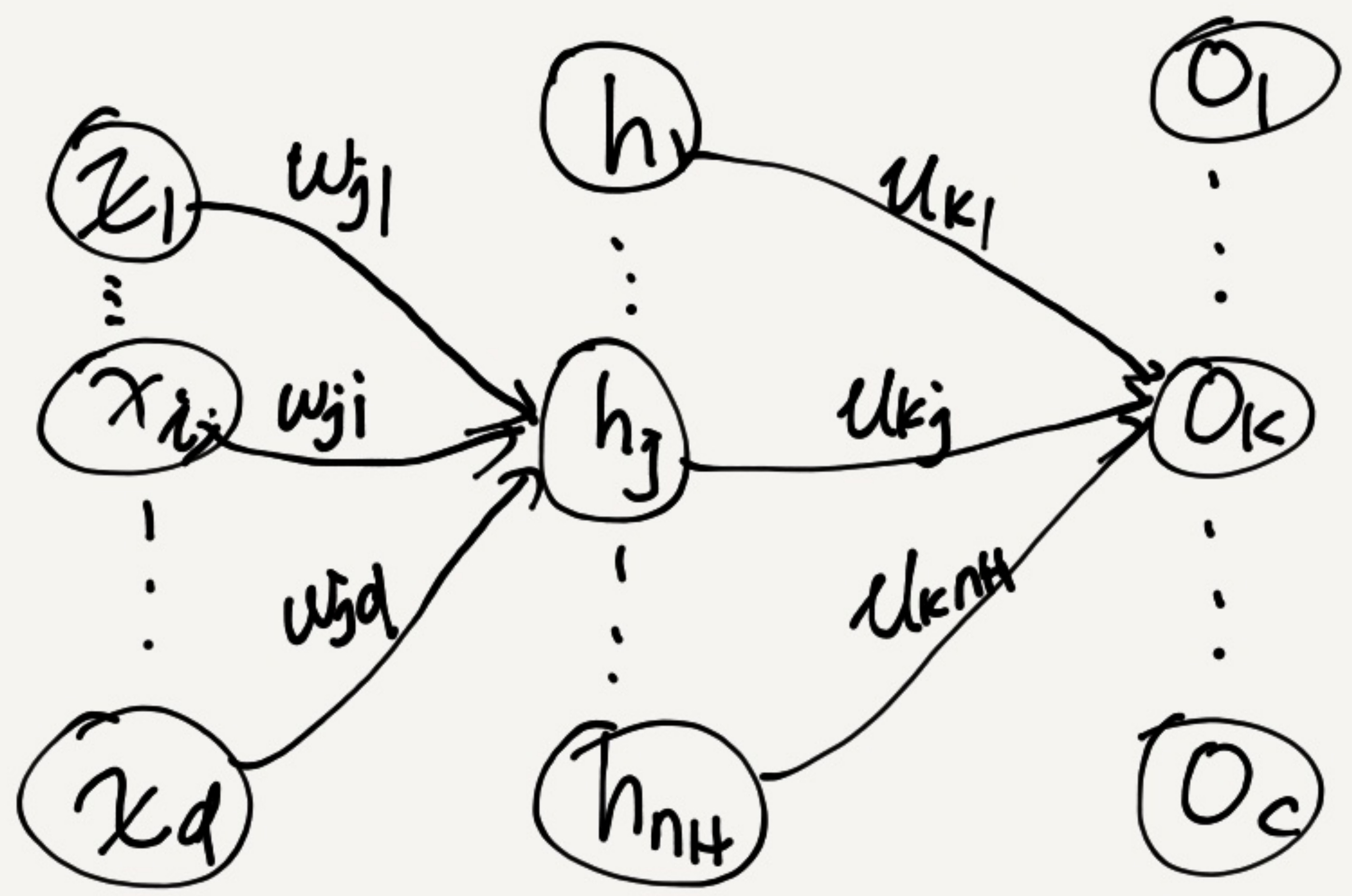
Weight update
$$\begin{cases} w_{11}^{i+1} = w_{11}^{i} - \varepsilon \cdot \nabla_{w_{11}} L \\ w_{12}^{i+1} = w_{12}^{i} - \varepsilon \cdot \nabla_{w_{12}} L \end{cases}$$

$$\nabla_{w_{11}} L = \frac{\partial L}{\partial O} \cdot \frac{\partial O}{\partial w_{11}} = \overbrace{2 \cdot \frac{1}{2}(y - O) \cdot -1}^{\frac{\partial L}{\partial O}} \cdot g_h' \cdot x_1$$

$$= \underbrace{(O - y) \cdot g_h'}_{} \cdot x_1$$

$$\nabla_{w_{12}} L = \underbrace{(O - y) \cdot g_h'}_{} \cdot x_2$$

2) 3-layer NN



d: # of features

n_H: # of Hidden nodes.

C: # of output nodes.

Model:

$$h_j = g_h(net_j) \quad net_j = \sum_{i=1}^{d} w_{ji} \cdot x_i \,. \quad j=1,\cdots,n_H$$

$$O_k = g_o(net_k) \quad net_k = \sum_{j=1}^{n_H} u_{kj} \cdot h_j \,. \quad k=1,\cdots,c$$

Loss: $\mathcal{L}(W,U) = \frac{1}{2} \sum_{i=1}^{c} (y_k - \underbrace{O_k}_{\text{prediction}})^2$

model parameters.

$$n_H \times d + n_H \times C$$
$$= n_H \times (d+c)$$

GD / SGD: $w_{ji} = w_{ji} - \varepsilon \cdot \nabla_{w_{ji}} \mathcal{L}$?

$$u_{kj} = u_{kj} - \varepsilon \cdot \nabla_{u_{kj}} \mathcal{L} \quad ?$$