Lecture 10. Backpropagation Algorithm. (BP)

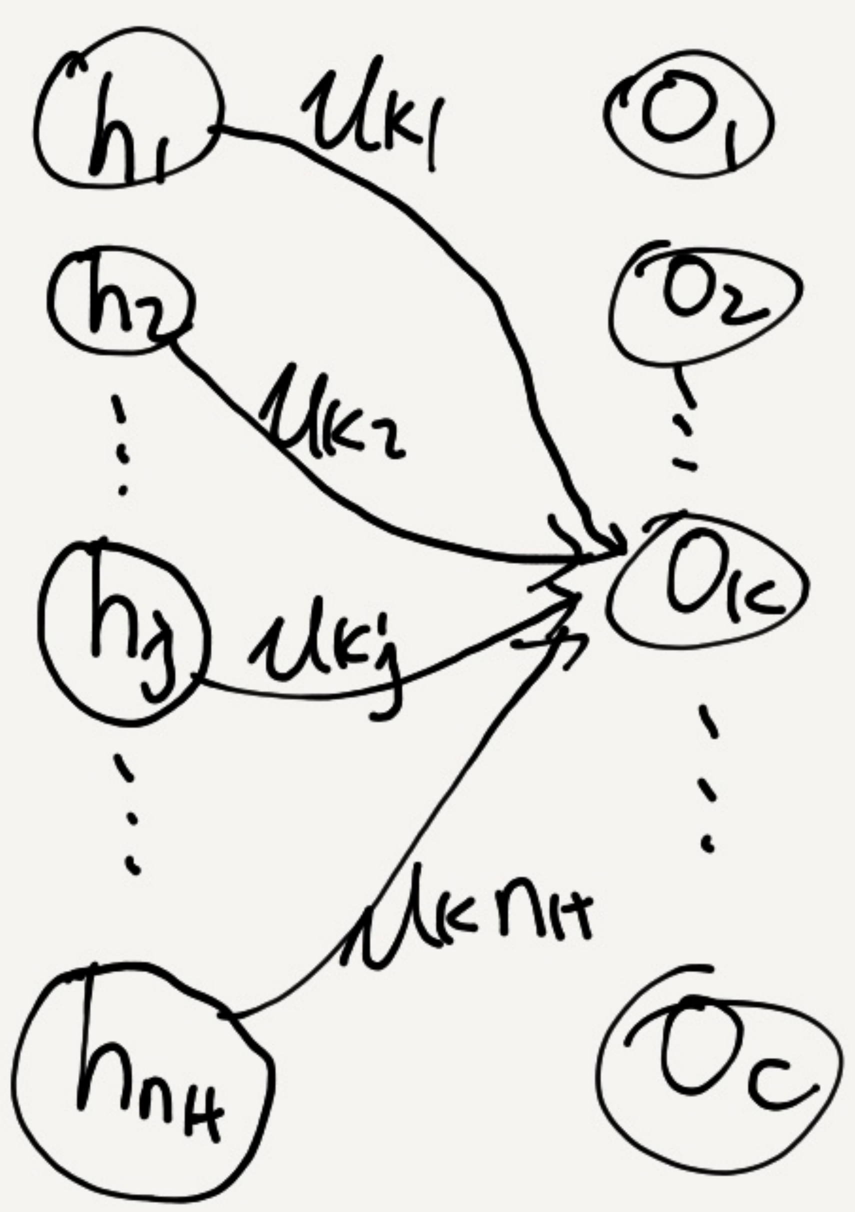
1. BP is an approach to calculate the gradients for NNs.

   BP is not an optimization algorithm.

   In our 3-layer NN, BP will

   (1) Calculate gradients for each weight/parameter between the output layer and the hidden layer.

$$
\mathcal{U} = \begin{array}{l} \left[\begin{array}{llll} u_{11}, & u_{12}, & \cdots, & u_{1n_H} \\ \vdots & & & \vdots \\ u_{k1}, & u_{k2}, & \cdots, & u_{kn_H} \\ \vdots & & & \\ u_{c1}, & u_{c2}, & \cdots, & u_{cn_H} \end{array}\right]_{c \times n_H} \end{array}
\begin{array}{l} \rightarrow O_1 \\ \\ \rightarrow \text{all weights of } O_k \\ \\ \rightarrow O_c \end{array}
$$

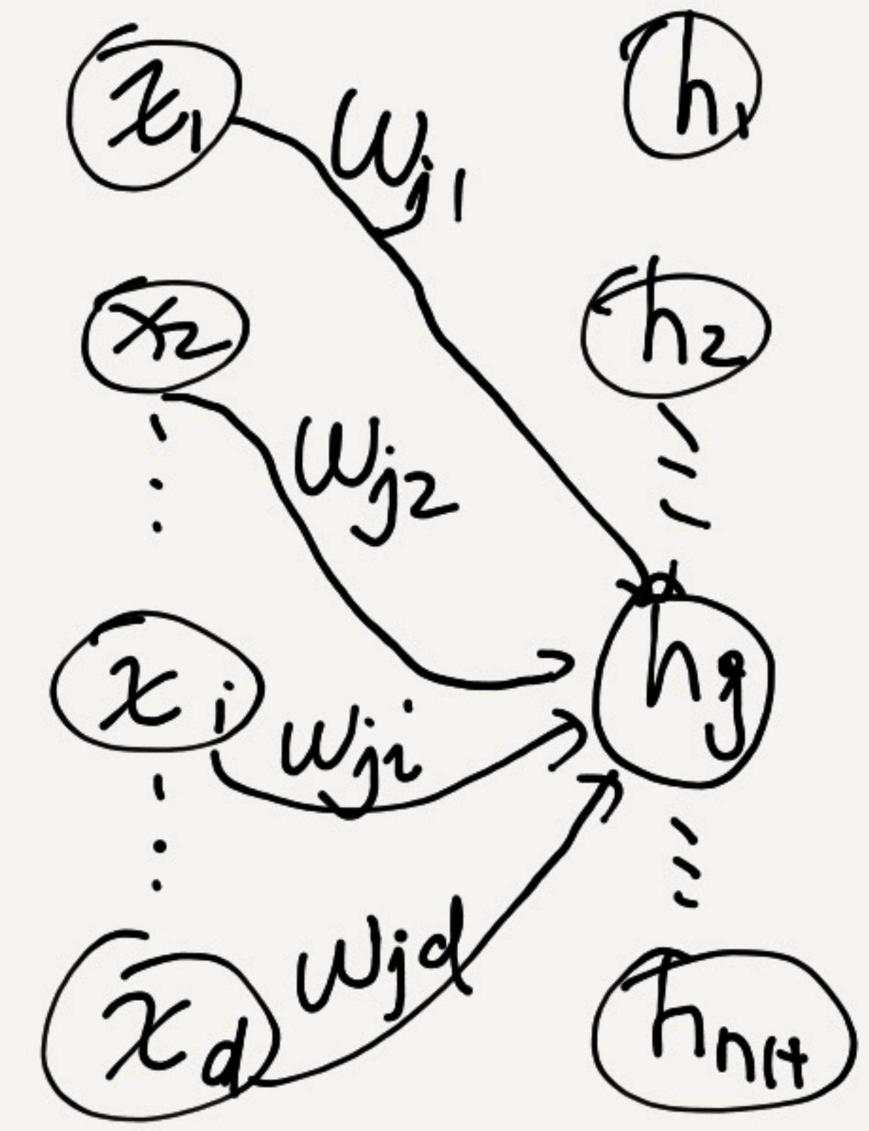


For $k = 1, 2, \cdots, C$ (# of output node)

$$\nabla_{u_{k1}} L, \ \nabla_{u_{k2}} L, \ \nabla_{u_{kj}} L, \ \cdots, \ \nabla_{u_{kn_H}} L$$

(2) calculate gradient for each weight between hidden and input layer.

$$W = \begin{array}{c} \begin{bmatrix} w_{11}, & w_{12}, & \cdots, & w_{1d} \\ \vdots & \vdots & & \vdots \\ w_{j1}, & w_{j2}, & \cdots, & w_{jd} \\ \vdots & & & \\ w_{n_H1}, & w_{n_H2}, & \cdots, & w_{n_Hd} \end{bmatrix} \end{array} \begin{array}{l} \to h_1 \\ \\ \to h_j \\ \\ \to h_{n_H} \end{array}$$

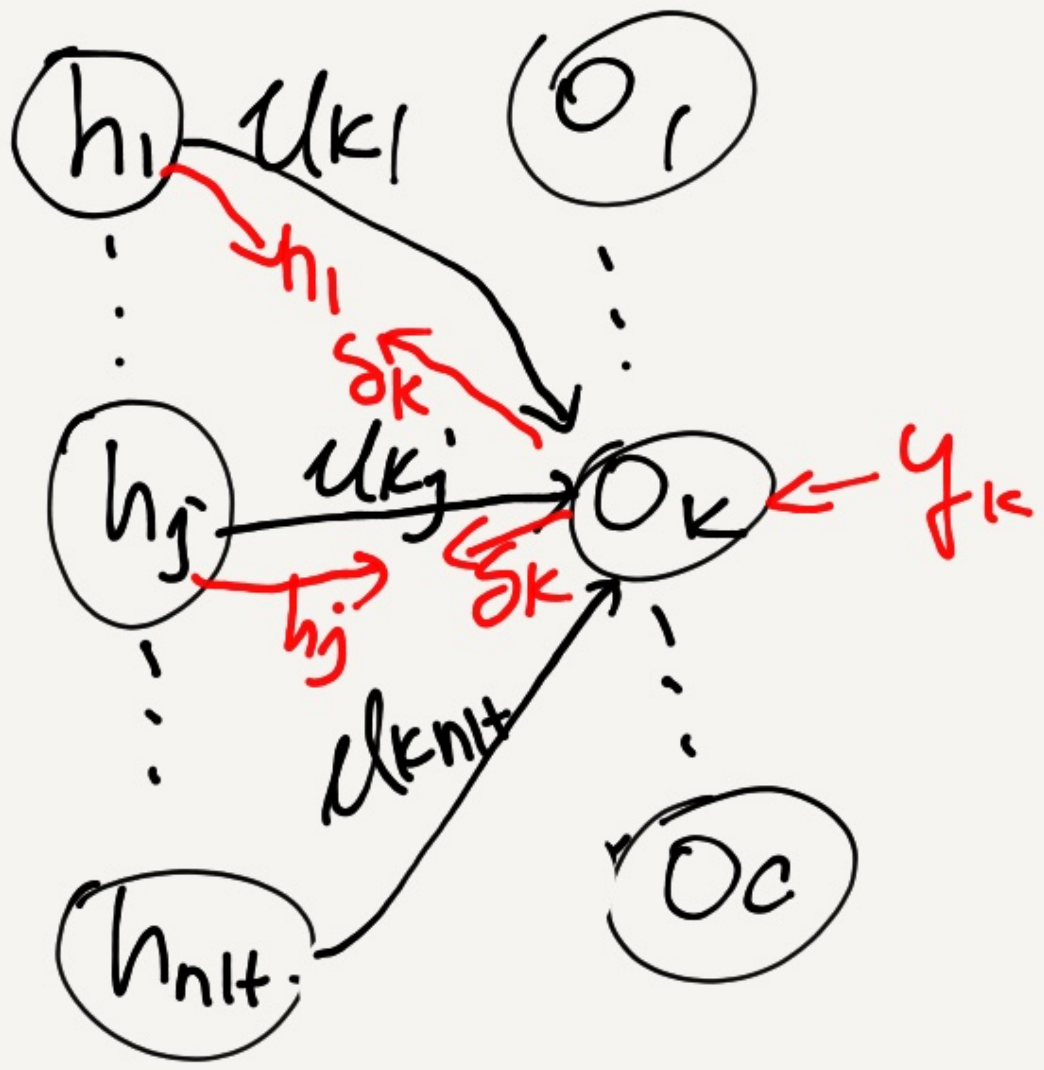


For $j = 1, 2, \cdots, n_H$

$$\nabla_{w_{j1}} L, \; \nabla_{w_{j2}} L, \; \cdots, \; \nabla_{w_{jd}} L.$$

2. $\nabla_{u_{kj}} \mathcal{L}$, $\quad k = 1, 2, \cdots, c.$

$j = 1, 2, \cdots, n_H$



prediction

Loss: $\mathcal{L}(w, u) = \frac{1}{2} \sum_{k=1}^{c} (y_k - O_k)^2$

model $\quad O_k = g_o(net_k) \qquad net_k = \sum_{j=1}^{n_H} u_{kj} \cdot h_j$

chain

$\boxed{\mathcal{L} \longrightarrow g_o \longrightarrow net_k}$

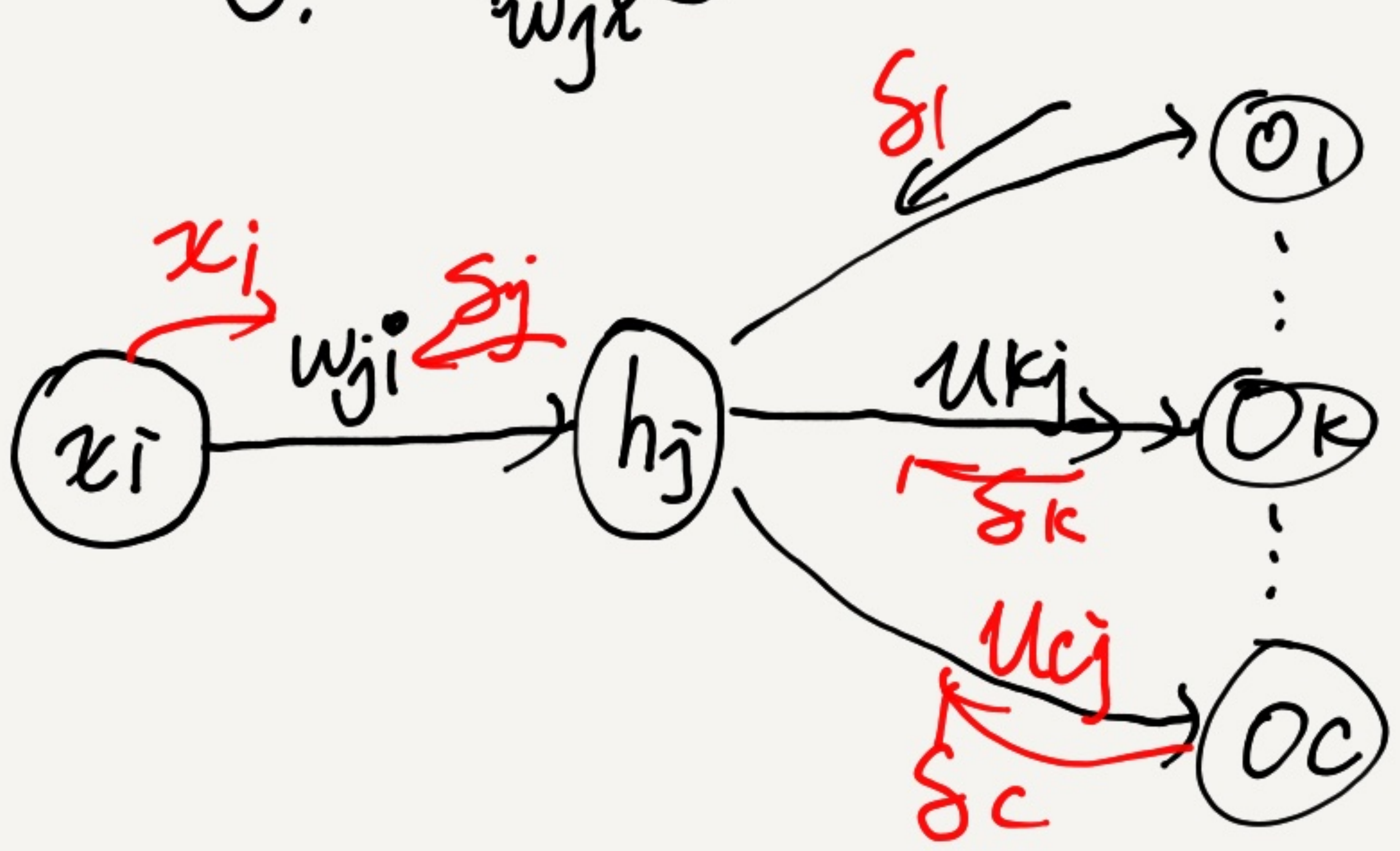
$$\nabla_{u_{kj}} \mathcal{L} = \frac{\partial L}{\partial O_k} \cdot \frac{\partial O_k}{\partial net_k} \cdot \frac{\partial net_k}{\partial u_{kj}}$$

$$= \underbrace{\frac{1}{2} \cdot 2}_{1} (O_k - y_k) \cdot g_o' \cdot h_j$$

$$= \boxed{(O_k - y_k) \cdot g_o'} \cdot h_j = \delta_k \cdot h_j$$

δ_k (feed back from the O_k)

3. $\nabla_{w_{ji}} \mathcal{L}$



$\boxed{\mathcal{L} \to g_o \to net_k \to g_h \to net_j}$

Loss: $\mathcal{L}(w, u) = \frac{1}{2} \sum_{k=1}^{c} (y_k - O_k)^2$

Model: $O_k = g_o(net_k)$, $net_k = \sum_{j=1}^{n_H} u_{kj} \cdot h_j$

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

$$\nabla_{w_{ji}} \mathcal{L} = \sum_{k=1}^{c} \left(\frac{\partial L}{\partial O_k} \cdot \frac{\partial O_k}{\partial net_k} \cdot \frac{\partial net_k}{\partial h_j} \right) \cdot \frac{\partial h_j}{\partial net_j} \cdot \frac{\partial net_j}{\partial w_{ji}}$$

$$= \left[\sum_{k=1}^{c} \overset{\delta_k}{\boxed{(O_k - y_k) \cdot g_o'}} \cdot u_{kj} \right] \cdot g_h' \cdot x_i = \overset{\delta_j}{\boxed{\sum_{k=1}^{c} (\delta_k \cdot u_{kj}) \cdot g_h'}} \cdot x_i$$

$\delta_j \cdot x_i$

$\|$

δ_j

4. summary of BP.

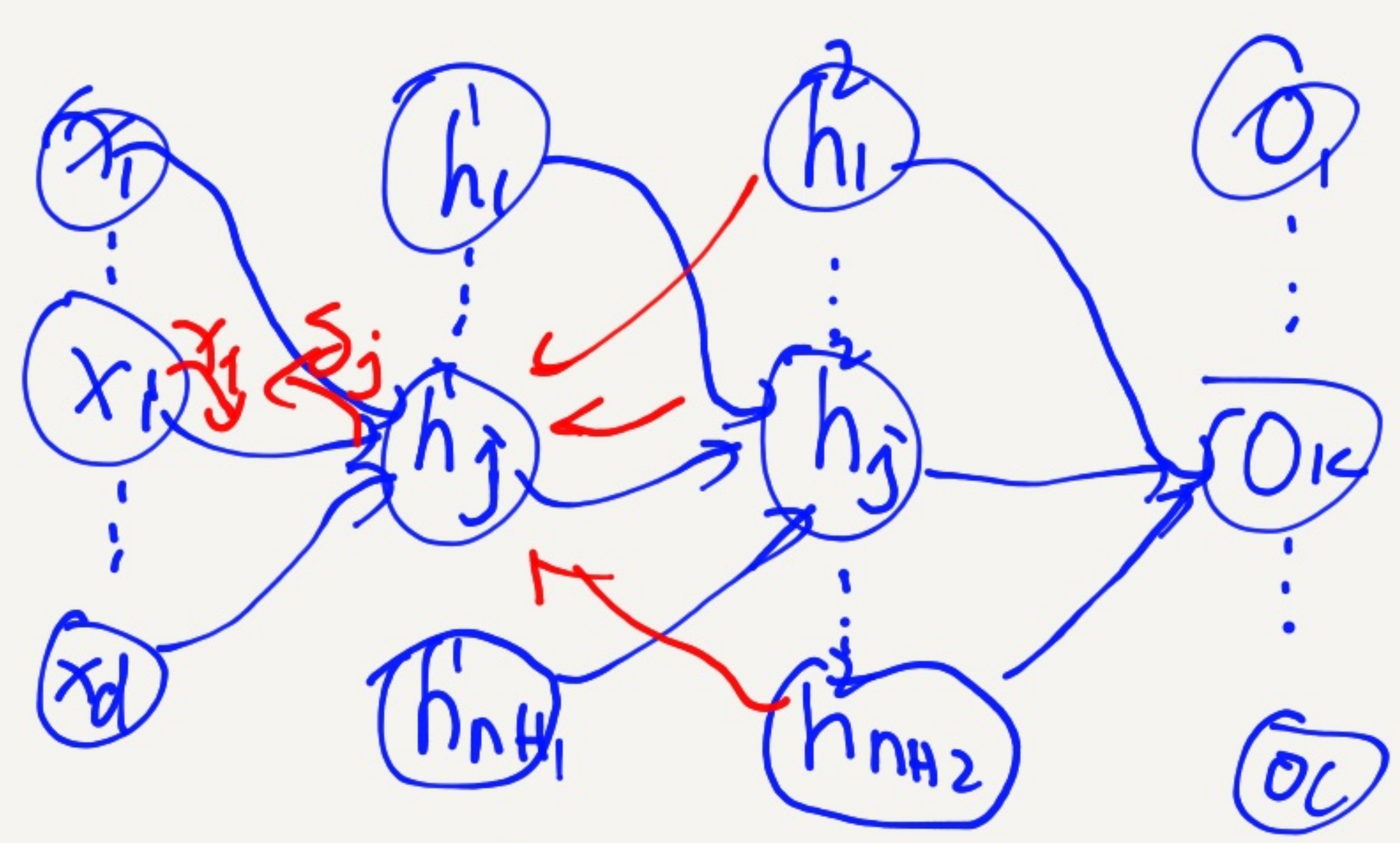
output – hidden : $\nabla_{u_{kj}} L = \delta_k \cdot h_j$

hidden – input layer: $\nabla_{w_{ji}} L = \delta_j \cdot x_i$

$$\delta_k = (o_k - y_k) \cdot g_o'$$

$$\delta_j = \sum_{k=1}^{c} \delta_k \cdot u_{kj}$$

If we have multiple hidden layers (more than 1), how can we extend the BP Algorithm?



4-layer NN.