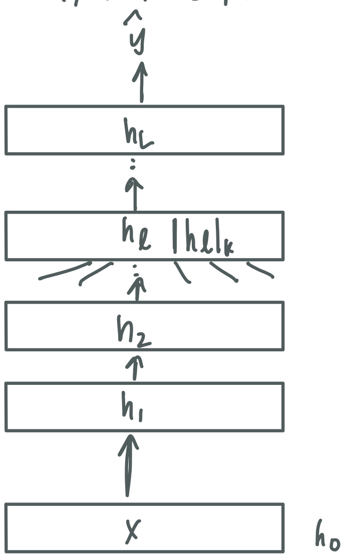
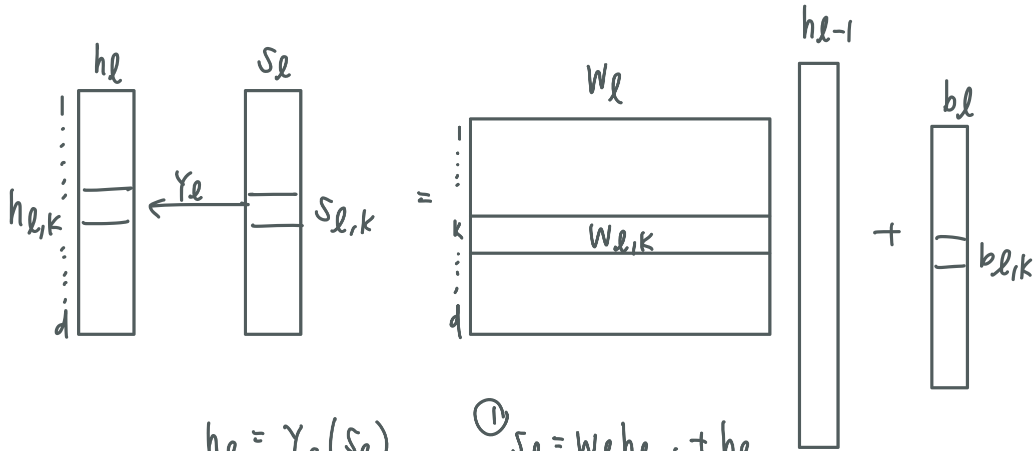
Multi-layer perceptron





$h_l = \gamma_l(s_l)$ elementwise (coordinatewise)

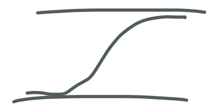
① $s_l = W_l h_{l-1} + b_l$

$l = 1, \dots, L$

① $s_{l,k} = \langle w_{l,k}, h_{l-1} \rangle + b_{l,k}$

② $h_{l,k} = \text{sigmoid}(s_{l,k}) = \dfrac{e^{s_{l,k}}}{1 + e^{s_{l,k}}}$

or $\text{ReLU}(s_{l,k}) = \max(0, s_{l,k})$



leaky ReLU

$\theta = (W_l, b_l,\ l = 1, \dots, L)$

Forward process · how to predict y based on x

$X \longrightarrow h_1 \longrightarrow h_2 \longrightarrow \dots \longrightarrow h_{l-1} \longrightarrow h_l \longrightarrow \dots \longrightarrow h_L \longrightarrow \hat{y}$

$\uparrow W_1, b_1 \qquad \uparrow W_2, b_2 \qquad \uparrow W_{l-1}, b_{l-1} \qquad \uparrow W_l, b_l \qquad \uparrow W_L, b_L$

Backward process · error back propagation via chain rule
enables training of NN by gradient descent, calc derivative

$\longleftarrow h_1 \longleftarrow \dots \longleftarrow h_{l-1} \xleftarrow{①} h_l \xleftarrow{①} \dots \xleftarrow{①} h_L \xleftarrow{①} e$

$h_1 \downarrow W_1, b_1 \qquad h_{l-1} \downarrow② W_{l-1}, b_{l-1} \qquad h_l \downarrow② W_l, b_l \qquad h_L \downarrow② W_L, b_L$

$$\frac{\partial Loss}{\partial h_{l-1}} \xleftarrow{①} \frac{\partial Loss}{\partial h_l}$$

$$\frac{\partial Loss}{\partial h_l} \xrightarrow{②} \frac{\partial Loss}{\partial W_l}$$

$h_{\ell-1}$ col vector

$h_{\ell-1}^T$ row vector

$$\frac{\partial L}{\partial h_{\ell-1}^T} = \sum_{k=1}^{d} \frac{\partial L}{\partial h_{\ell,k}} \frac{\partial h_{\ell,k}}{\partial s_{\ell,k}} \frac{\partial s_{\ell,k}}{\partial h_{\ell-1}^T}$$

$$= \sum_{k=1}^{d} \frac{\partial L}{\partial h_{\ell,k}} r'_{\ell}(s_{\ell,k})\, w_{\ell,k}$$

$$= \begin{array}{c} \left(\begin{array}{ccccccc} & & & \frac{\partial L}{\partial h_{\ell,k}} r'_{\ell}(s_{\ell,k}) & & & \end{array} \right) \\ \begin{array}{ccccccc} 1, & 2, & \ldots & k, & & \ldots & d \end{array} \end{array} \begin{array}{cc} \left(\begin{array}{c} \\ w_{\ell,k} \\ \\ \end{array} \right) & \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array} \end{array}$$

$$= \begin{array}{c} \left(\begin{array}{ccccc} & & \frac{\partial L}{\partial h_{\ell,k}} & & \end{array} \right) \\ \begin{array}{ccccc} 1, & 2, \ldots & k & \ldots & d \end{array} \end{array} \begin{array}{cc} \left(\begin{array}{c} \\ r'_{\ell}(s_{\ell,k})\, w_{\ell,k} \\ \\ \end{array} \right) & \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array} \end{array}$$

row wise multiplication

$$\frac{\partial L}{\partial h_{\ell-1}^T} = \frac{\partial L}{\partial h_{\ell}^T}\, r'_{\ell} \odot w_{\ell} \qquad (1)$$

$$r'_{\ell} = \begin{array}{cc} \left(\begin{array}{c} \\ r'_{\ell}(s_{\ell,k}) \\ \\ \end{array} \right) & \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array} \end{array}$$

$$\frac{\partial L}{\partial h_{\ell-1}^T} = \frac{\partial L}{\partial h_{\ell}^T} \operatorname{diag}(r'_{\ell})\, w_{\ell}$$

(2)
$$\frac{\partial L}{\partial w_{\ell,k}} = \frac{\partial L}{\partial h_{\ell,k}} \frac{\partial h_{\ell,k}}{\partial s_{\ell,k}} \frac{\partial s_{\ell,k}}{\partial w_{\ell,k}}$$

$$= \frac{\partial L}{\partial h_{\ell,k}} r'_{\ell}(s_{\ell,k})\, h_{\ell-1}^T$$

$$\frac{\partial L}{\partial w_{\ell}} = \begin{array}{cc} \left(\begin{array}{c} \\ \frac{\partial L}{\partial w_{\ell,k}} \\ \\ \end{array} \right) & \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array} \end{array} = \begin{array}{cc} \left(\begin{array}{c} \\ \frac{\partial L}{\partial w_{\ell,k}} r'_{\ell}(s_{\ell,k}) \\ \\ \end{array} \right) & \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array} \end{array} h_{\ell-1}^T = r'_{\ell} \odot \frac{\partial L}{\partial h_{\ell}} h_{\ell-1} = \operatorname{diag}(r'_{\ell}) \frac{\partial L}{\partial h_{\ell}} h_{\ell-1}$$

Top layer

()

linear model

$s = x^T\beta$

neural network

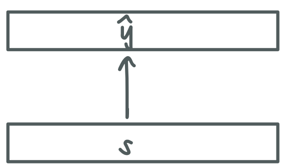
$x \rightarrow h_1 \rightarrow \ldots \rightarrow h_{l-1} \rightarrow \underset{(h_L)}{s} \rightarrow \hat{y}$

linear regression

$y \sim N(s, \sigma^2)$ $\quad$ $\text{Loss} = \frac{1}{2}(y-s)^2$ $\quad$ $\frac{dL}{ds} = -(y-s) = -e$

logistic regression

$y \sim \text{Bernoulli}(p = \text{sigmoid}(s))$ $\quad$ $\text{Loss} = -(y_i s_i - \log(1+e^{s}))$ $\quad$ $\frac{dL}{ds} = -(y-p) = -e$



regression:

$$\text{Loss} = \frac{1}{2}|y-s|^2$$

$$\frac{dL}{ds} = -(y-s) = -e$$

logistic : multiclass classification

C classes $\quad (1, 2, \ldots, c, \ldots, C)$

$$y = \text{one-hot} = \begin{array}{|c|c} 0 & 1 \\ 0 & 2 \\ & \vdots \\ 1 & c \\ & \vdots \\ 0 & C \end{array}$$

soft max

$$P(y=c|s) = \frac{e^{s_c}}{\sum_{c'=1}^{C} e^{s_{c'}}}$$

$$s = \text{logit } s = \begin{array}{|c|c} s_1 & 1 \\ s_2 & 2 \\ & \vdots \\ s_c & c \\ & \vdots \\ s_C & C \end{array}$$