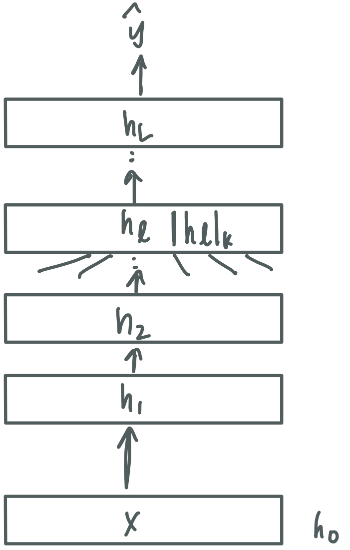
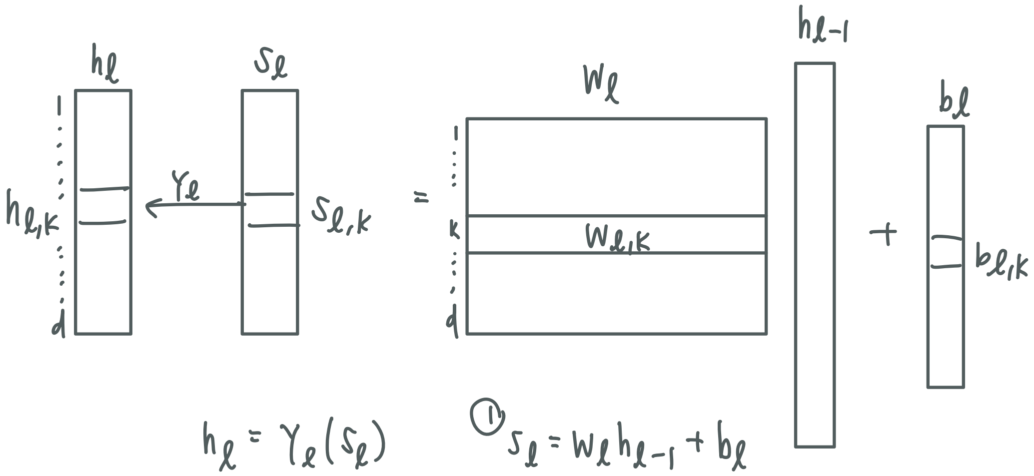
Multi-layer perceptron





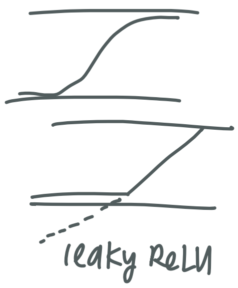
$h_l = \gamma_l(s_l)$ elementwise (coordinatewise)

① $s_l = W_l h_{l-1} + b_l$ $\quad l = 1, \dots, L$

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



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

Forward process · how to predict y based on x

$$\begin{tikzcd}
x \arrow[r] & h_1 \arrow[r] & h_2 \arrow[r] & \dots \arrow[r] & h_{l-1} \arrow[r] & h_l \arrow[r] & \dots \arrow[r] & h_L \arrow[r] & \hat{y} \\
& W_1, b_1 \arrow[u] & W_2, b_2 \arrow[u] & & W_{l-1}, b_{l-1} \arrow[u] & W_l, b_l \arrow[u] & & W_L, b_L \arrow[u] &
\end{tikzcd}$$

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

$$\begin{tikzcd}
{} & h_1 \arrow[l] \arrow[d] & \dots \arrow[l] & h_{l-1} \arrow[l] \arrow[d, "②"] & h_l \arrow[l, "①"'] \arrow[d, "②"] & \dots \arrow[l, "①"'] & h_L \arrow[l, "①"'] \arrow[d, "②"] & e \arrow[l, "①"'] \\
& W_1, b_1 & & W_{l-1}, b_{l-1} & W_l, b_l & & W_L, b_L &
\end{tikzcd}$$

$$\begin{tikzcd}
\dfrac{\partial Loss}{\partial h_{l-1}} & \dfrac{\partial Loss}{\partial h_l} \arrow[l, "①"'] \arrow[d, "②"] \\
& \dfrac{\partial Loss}{\partial W_l}
\end{tikzcd}$$

$h_{\ell-1}$ col vector

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

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

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

$$= \begin{pmatrix} \frac{\partial L}{\partial h_{\ell,k}} r_\ell'(s_{\ell,k}) \end{pmatrix}_{1,2,\ldots k,\ldots d} \begin{pmatrix} w_{\ell,k} \end{pmatrix} \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array}$$

$$= \begin{pmatrix} \frac{\partial L}{\partial h_{\ell,k}} \end{pmatrix}_{1,2,\ldots k \ldots d} \begin{pmatrix} r_\ell'(s_{\ell,k}) w_{\ell,k} \end{pmatrix} \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array}$$

row wise multiplication

$$\frac{\partial L}{\partial h_{\ell-1}^T} = \frac{\partial L}{\partial h_\ell^T} r_\ell' \odot w_\ell \qquad ①$$

$$r_\ell' = \begin{pmatrix} r_\ell'(s_{\ell,k}) \end{pmatrix} \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array}$$

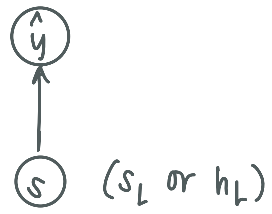
$$\frac{\partial L}{\partial h_{\ell-1}^T} = \frac{\partial L}{\partial h_\ell^T} \operatorname{diag}(r_\ell') w_\ell$$

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

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

$$\frac{\partial L}{\partial w_\ell} = \begin{pmatrix} \frac{\partial L}{\partial w_{\ell,k}} \end{pmatrix} \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array} = \begin{pmatrix} \frac{\partial L}{\partial w_{\ell,k}} r_\ell'(s_{\ell,k}) \end{pmatrix} \begin{array}{c} 1 \\ 2 \\ \vdots \\ k \\ \vdots \\ d \end{array} h_{\ell-1}^T = r_\ell' \odot \frac{\partial L}{\partial h_\ell} h_{\ell-1} = \operatorname{diag}(r_\ell') \frac{\partial L}{\partial h_\ell} h_{\ell-1}$$

Top layer



linear model

$s = x^T\beta$

neural network

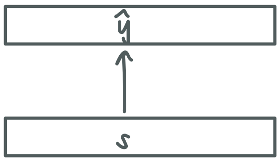
$$x \to h_1 \to \dots \to h_{l-1} \to \underset{(h_L)}{s} \to \hat{y}$$

linear regression

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

logistic regression

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



regression:

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

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

logistic: multiclass classification

C classes $(1, 2, \dots, c, \dots, C)$

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

soft max

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

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