컴퓨터비전특론 3rd week summary

[Lec 04. Loss]

Analytic Gradient

- Hinge loss

1) Binary hinge loss

$$L = \max(0, 1-y\cdot s) \quad \begin{pmatrix} s = w^T x + b \\ y = \pm 1 \end{pmatrix}$$

$$= \begin{cases} 1-y\cdot s & \text{if } 1-y\cdot s>0 \\ 0 & \text{o.w} \end{cases}$$

$$\Rightarrow \frac{\partial L}{\partial s} = \begin{cases} -y & \text{if } 1-y\cdot s>0 \\ 0 & \text{o.w} \end{cases}$$

2) hinge loss (multiclass)

$$L = \sum_{j=1, j\neq y}^{n} \max(0, s_j - s_y + 1)$$

$s = wx+b$, $W = \begin{pmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_n^T \end{pmatrix}$, $s = \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix}$

x: image, y: class label $(1 \le y \le n)$

$$\Rightarrow \frac{\partial L}{\partial s_y} = -\sum_{j=1, j\neq y}^{n} 1(s_j - s_y + 1 > 0) \quad \text{for } j=y$$

$$\frac{\partial L}{\partial s_j} = 1(s_j - s_y + 1 > 0) \quad \text{for } j\neq y$$

$$* \ 1(F) = \begin{cases} 1 & \text{if } F \text{ is true} \\ 0 & \text{o.w} \end{cases}$$

- Log likelihood loss

$L = -\log p_y$ where $z_y = 1$,

- $p = \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix}$: Prob. for i^{th} image
- y : class label for i^{th} image, $1 \le y \le n$
- z : class probability $= (z_1, \cdots, z_n)^T$, $z_y = 1$, $z_{k\neq y} = 0$

$$\Rightarrow \frac{\partial L}{\partial p} = -\begin{bmatrix} 0 \\ \vdots \\ 1/p_y \\ \vdots \\ 0 \end{bmatrix}$$

정답 label만 0이 아님

Cross - Entropy loss

$$L = -\sum_{j=1}^{n} (z_j \log p_j + (1-z_j)\log(1-p_j)),$$

- $p = \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix}$: Prob. for i^{th} image
- y : class label for i^{th} image, $1 \le y \le n$
- z : class probability $= (z_1, \cdots, z_n)^T$, $z_y = 1$, $z_{k\neq y} = 0$

$$\Rightarrow \frac{\partial L}{\partial p} = \begin{bmatrix} 1/(1-p_1) \\ \vdots \\ -1/p_y \\ \vdots \\ 1/(1-p_n) \end{bmatrix}$$

정답 label

- Regression loss

$y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$, $s = \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix}$ 일 때

- $L = (y-s)^2 \Rightarrow \frac{\partial L}{\partial s} = -2(y-s)$
- $L = |y-s| \Rightarrow \frac{\partial L}{\partial s} = \begin{cases} -1 & \text{for } y-s>0 \\ 1 & \text{for } y-s<0 \end{cases}$

* 0일 때는 미분 불가능!

Image features vs ConvNet

- Image features : SIFT, HoG, BoW 등의 feature transformation 방법으로 이미지를 vector로 표현

  Handcrafted feature : 변환 사이즈 지정, edge 분할 등

  인간이 정해준 규칙에 따라 feature extraction

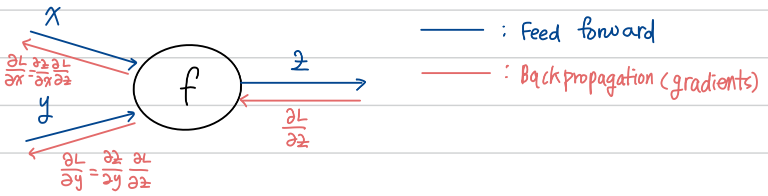
  → Feature extraction 후 classifier training (SVM, RF 등)

- ConvNets : Conv를 통한 feature representation 과정과 softmax + log likelihood 과정이 모두 training에 포함됨

  ⇒ 딥러닝이 더 좋은 성능을 보인다

[Lec 05. Backpropagation and Neural Networks]

Backpropagation

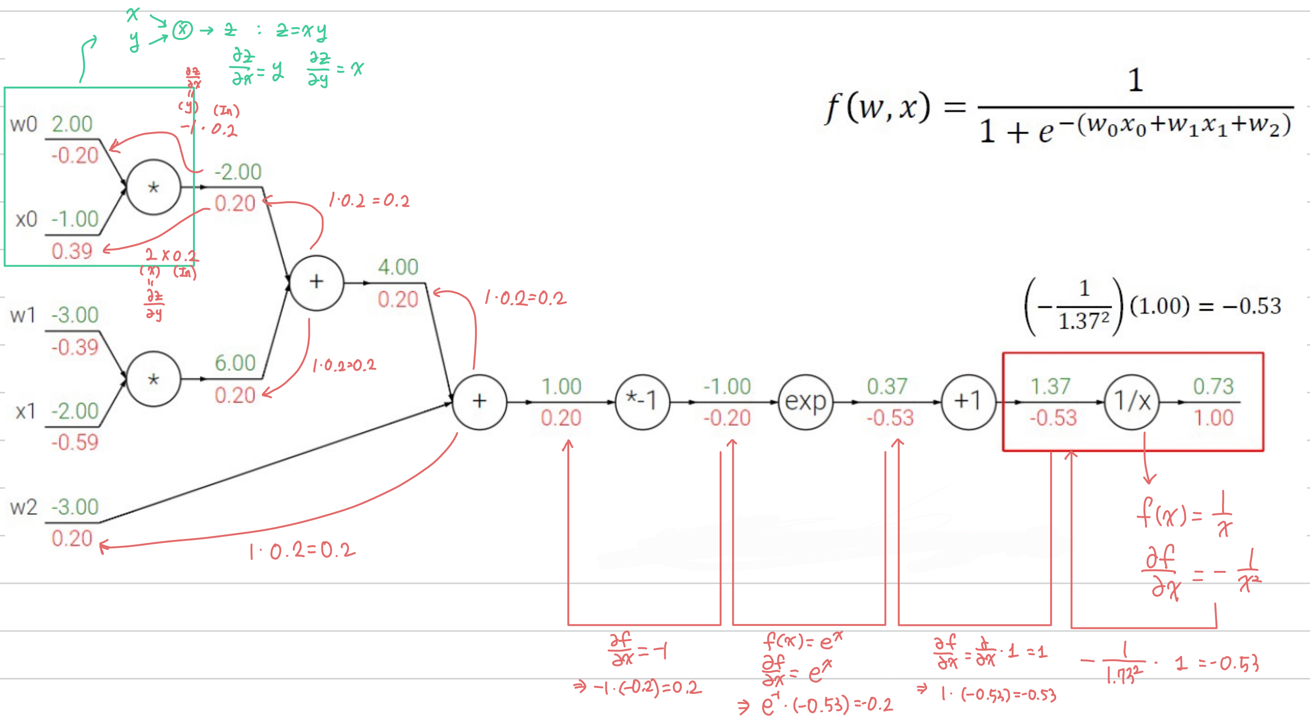


Chain rule

$p = \binom{x}{y}$, $\frac{\partial L}{\partial z}$: 이전 단계의 outgoing gradient일 때

$$\underbrace{\overset{2\times1}{\frac{\partial L}{\partial p}}}_{\text{Outgoing gradient}} = \underbrace{\overset{2\times1}{\frac{\partial z}{\partial p}}}_{\text{Jaccobian matrix}} \cdot \underbrace{\overset{\text{scalar}}{\frac{\partial L}{\partial z}}}_{\text{Ingoing gradient}}$$

– scalar example



$$f(w,x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2)}}$$

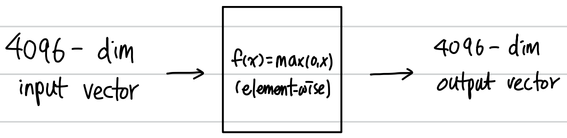
* Derivative of sigmoid function

Sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$

$$\rightarrow \frac{\partial \sigma(x)}{\partial x} = \frac{1+e^{-x}-1}{(1+e^{-x})^2} = \frac{e^{-x}}{1+e^{-x}} \cdot \frac{1}{1+e^{-x}} = ((1-\sigma(x))\sigma(x)$$

⤳ Sigmoid function의 backpropagation을 한번에 계산가능

– vectorized operations



$$\Rightarrow \overset{\text{Outgoing}}{\underset{4096\times1}{\frac{\partial L}{\partial x}}} = \underbrace{\frac{\partial f}{\partial x}}_{\text{Jaccobian matrix : } 4096 \times 4096} \overset{\text{Incoming}}{\underset{4096\times1}{\frac{\partial L}{\partial f}}}$$

* Minibatch = 100 인 경우

: 4096 차원 vector가 100번 들어옴

⇒ 409600 × 409600 matrix

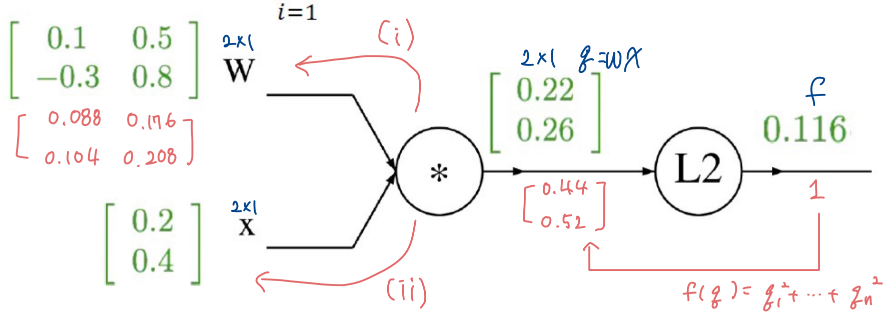
(= 4096 × 4096 × 100 3D Tensor)

- vectorized example

$q_i = (Wx)_i$: i^{th} element
q : $n\times1$ vector

$$f(\mathbf{x},\mathbf{W}) = |\mathbf{W}\mathbf{x}|^2 = \sum_{i=1}^{n}(\mathbf{W}\mathbf{x})_i^2 \qquad \mathbf{x}\in\Re^d,\ \mathbf{W}\in\Re^{n\times d}$$

$$\frac{\partial f}{\partial \mathbf{x}}, \frac{\partial f}{\partial \mathbf{W}} = ?$$



$f(q) = q_1^2 + \cdots + q_n^2$

$\frac{\partial f}{\partial q_i} = 2q$

$\Rightarrow \frac{\partial f}{\partial q} = 2q$

$$\mathbf{q} = \mathbf{W}\mathbf{x} = \begin{pmatrix} W_{1,1}x_1 + & \cdots & +W_{1,d}x_d \\ & \vdots & \\ W_{n,1}x_1 + & \cdots & +W_{n,d}x_d \end{pmatrix} = \begin{pmatrix} q_1 \\ \vdots \\ q_n \end{pmatrix}$$

$$f(\mathbf{q}) = |\mathbf{q}|^2 = q_1^2 + \cdots + q_n^2$$

(ii) $\frac{\partial q_k}{\partial x_j} = W_{k,j}$

$$\frac{\partial f}{\partial x_j} = \sum_k \frac{\partial f}{\partial q_k}\frac{\partial q_k}{\partial x_j} = \sum_k 2q_k \cdot W_{k,j}$$

$\rightarrow$ $\boxed{\frac{\partial f}{\partial x} = 2W^Tq = 2W^TWx}$

(i) $\frac{\partial f}{\partial W_{i,j}} = \sum_k \frac{\partial f}{\partial q_k}\frac{\partial q_k}{\partial W_{ij}}$

· $q = \begin{pmatrix} q_1 \\ \vdots \\ q_n \end{pmatrix}$, $q_k = W_{k,1}x_1 + \cdots + W_{k,d}x_d$

$\rightarrow \frac{\partial q_k}{\partial W_{i,j}} = \underbrace{1_{k=i}}_{i=k\text{이면 }1,\text{ 아니면 }0}x_j$

$\rightarrow \frac{\partial f}{\partial W_{i,j}} = \sum_k \frac{\partial f}{\partial q_k}\cdot\frac{\partial q_k}{\partial W_{i,j}}$

$= \sum_k 2q_k(1_{k=i}x_j) = 2q_ix_j$

$$\boxed{\underbrace{\frac{\partial f}{\partial W}}_{n\times d} = 2\underbrace{q}_{Wx}x^T = 2\underbrace{W}_{n\times d}\underbrace{xx^T}_{d\times d}}$$

☆

$$\therefore \begin{cases} \frac{\partial f}{\partial W_{i}} = \underbrace{\frac{\partial q}{\partial W_i}}_{x}\underbrace{\frac{\partial f}{\partial q}}_{2q_i = 2W_i^Tx} = 2(W_i^Tx)x \longrightarrow \frac{\partial f}{\partial W} = 2Wxx^T \\ \frac{\partial f}{\partial x} = \underbrace{\frac{\partial q}{\partial x}}_{W^T}\underbrace{\frac{\partial f}{\partial q}}_{2q} = 2W^Tq \longrightarrow \frac{\partial f}{\partial x} = 2W^TWx \end{cases}$$

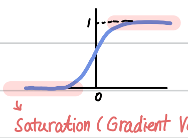
Neural Networks

- Linear vs NN

Linear : layer가 여러개 있어도 한번 쓴것과 동일한 결과
NN : Nonlinear operation을 사용 → Layer별로 다른 weight가 추정됨

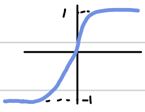
- Activation functions

[sigmoid] $\sigma(x) = \frac{1}{1+e^{-x}}$

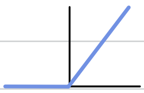


Saturation (Gradient Vanishing 발생 가능)

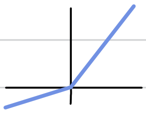
[tanh] $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$



[ReLU] $\max(0, x)$

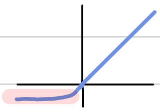


[Leaky ReLU] $\max(0.1x, x)$

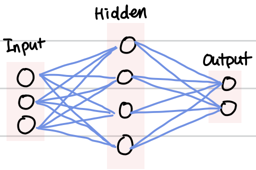


[Maxout] $\max(w_1^Tx+b_1, w_2^Tx+b_2)$

[ELU] $\begin{cases} x & , x\geq0 \\ \alpha(e^x-1) & , x<0 \end{cases}$ * α : hyperparameter



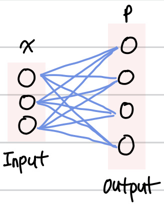
- Fully connected layers (= Multi layer Perceptron)



: 하나의 시냅스가 다음 layer의 모든 시냅스와 연결됨

- Derivative of NN using chain rules

1) 1-layer NN (L2 regression loss)



. $S = w x + b$

. activation function $p = \sigma(s) = \frac{1}{1+e^{-s}}$

. Loss $L = (\underset{\text{ground truth}}{z} - p)^2$

$S = \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix}$ $w = \begin{pmatrix} w_1^T \\ \vdots \\ w_n^T \end{pmatrix} = \begin{pmatrix} w_{11} & \cdots & w_{1d} \\ \vdots & & \\ w_{n1} & \cdots & w_{nd} \end{pmatrix}$ $b = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$ $x = \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix}$

$\Rightarrow \frac{\partial L}{\partial p}, \frac{\partial L}{\partial s}, \frac{\partial L}{\partial w}, \frac{\partial L}{\partial b}$ 을 계산해야 함

. $\frac{\partial L}{\partial p} = (z-p)^2 \cdot \frac{\partial}{\partial p} = -2(z-p)$

. $\underset{\text{out}}{\frac{\partial L}{\partial s}} = \underset{J}{\frac{\partial p}{\partial s}} \underset{\text{In}}{\frac{\partial L}{\partial p}} = \text{diag}((1-\sigma(s_j))\sigma(s_j)) \cdot \frac{\partial L}{\partial p} = -2 \begin{bmatrix} (1-\sigma(s_1))\sigma(s_1)(z_1-p_1) \\ \vdots \\ (1-\sigma(s_n))\sigma(s_n)(z_n-p_n) \end{bmatrix} = (1-\sigma(s)) \otimes \sigma(s) \otimes \frac{\partial L}{\partial p}$

Element-wise multiplication

. $\frac{\partial L}{\partial w_j} = \frac{\partial s}{\partial w_j} \frac{\partial L}{\partial s} = X_j \frac{\partial L}{\partial s} = [0\ 0 \cdots \underset{\text{jth column}}{x} \cdots 0] \frac{\partial L}{\partial s} = \left(\frac{\partial L}{\partial s}\right)_j x$

$\rightarrow \frac{\partial L}{\partial w} = \frac{\partial L}{\partial w} x^T$

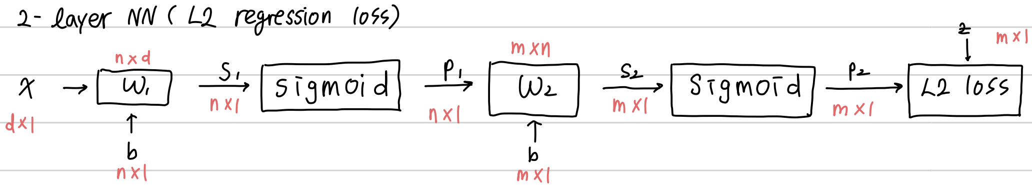
. $\frac{\partial L}{\partial b} = \underset{\text{Identity}}{\frac{\partial s}{\partial b}} \frac{\partial L}{\partial s} = \frac{\partial L}{\partial s}$

* $S = Wx + b$

$\Rightarrow \frac{\partial L}{\partial W} = \frac{\partial L}{\partial s} \cdot x^T \qquad \frac{\partial L}{\partial b} = \frac{\partial L}{\partial s} \cdot 1 = \frac{\partial L}{\partial s}$

$\frac{\partial L}{\partial x} = \frac{\partial s}{\partial x} \cdot \frac{\partial L}{\partial s} = W^T \cdot \frac{\partial L}{\partial s}$

2) 2-layer NN (L2 regression loss)



. $\frac{\partial L}{\partial p_2} = -2(z - p_2)$

. $\frac{\partial L}{\partial s_2} = \frac{\partial p_2}{\partial s_2} \frac{\partial L}{\partial p_2} = \text{diag}((1-\sigma(s_{2,j}))\sigma(s_{2,j})) \frac{\partial L}{\partial p_2}$

. $\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial s_2} p_1^T$

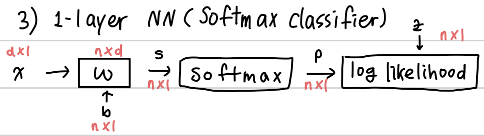
. $\frac{\partial L}{\partial b_2} = \frac{L}{\partial s_2}$

. $\frac{\partial L}{\partial p_1} = \frac{\partial s_2}{\partial p_1} \cdot \frac{\partial L}{\partial s_2} = W_2^T \cdot \frac{\partial L}{\partial s_2}$

두번째 layer의 input data와 같음!

. $\frac{\partial L}{\partial s_1} = \text{diag}((1-\sigma(s_1))\sigma(s_1) \cdot \frac{\partial L}{\partial p_1}$ $\longrightarrow \left[\begin{array}{l} \frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial s_1} x^T \\ \frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial s_1} \end{array} \right.$

3) 1-layer NN (Softmax classifier)



. $s = Wx + b$, $s_j = w_j^T x + b_j$

. Activation function $P = \frac{e^s}{\Sigma_{j=1}^{n} e^{s_j}}$

. Loss $L = -\log P_y$ where $z_y = 1$, $z = (z_1, \cdots, z_n)^T$, $z_y = 1$, $z_{k \neq y} = 0$

$S = \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix}$ $w = \begin{pmatrix} w_1^T \\ \vdots \\ w_n^T \end{pmatrix} = \begin{pmatrix} w_{11} & \cdots & w_{1d} \\ \vdots & & \\ w_{n1} & \cdots & w_{nd} \end{pmatrix}$ $b = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$ $x = \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix}$

$\Rightarrow$

$D_{ab} = P_a(\delta_{ab} - P_b)$, $\delta_{ab} = \begin{cases} 1 & \text{if } a=b \\ 0 & \text{o.w} \end{cases}$

. $\frac{\partial L}{\partial p} = \begin{bmatrix} 0 \\ \vdots \\ -1/P_y \\ \vdots \\ 0 \end{bmatrix}$

. $\frac{\partial L}{\partial s} = \frac{\partial p}{\partial s} \frac{\partial L}{\partial p} = -\frac{1}{P_y} \begin{bmatrix} D_{1y} \\ \vdots \\ D_{ny} \end{bmatrix} = -\frac{1}{P_y} \begin{bmatrix} -P_1 P_y \\ \vdots \\ P_y(1-P_y) \\ \vdots \\ -P_n P_y \end{bmatrix} = \begin{bmatrix} P_1 \\ \vdots \\ P_y - 1 \\ P_n \end{bmatrix} = P - z$

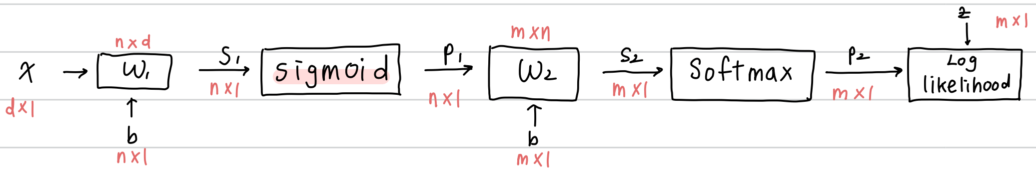
. $\frac{\partial L}{\partial w_j} = \frac{\partial s}{\partial w_j} \frac{\partial L}{\partial s} = X_j \frac{\partial L}{\partial s} = [0 \cdots \underset{\text{jth column}}{x} \cdots 0] \frac{\partial L}{\partial s} = \left(\frac{\partial L}{\partial s}\right)_j x$

$\rightarrow \frac{\partial L}{\partial w} = \left(\frac{\partial L}{\partial w_1} \cdots \frac{\partial L}{\partial w_n}\right)^T = \frac{\partial L}{\partial s} x^T$

. $\frac{\partial L}{\partial b} = \frac{\partial s}{\partial b} \frac{\partial L}{\partial s} = \frac{\partial L}{\partial s}$

* $\frac{\partial L}{\partial x} = \frac{\partial s}{\partial x} \frac{\partial L}{\partial s} = W^T \frac{\partial L}{\partial s}$

4) 2- layer NN (Softmax classifier)



- $\frac{\partial L}{\partial P_2} = \begin{bmatrix} 0 \\ \vdots \\ -1/P_y \\ \vdots \\ 0 \end{bmatrix}$
- $\frac{\partial L}{S_2} = D \frac{\partial L}{\partial P_2}$, $D_{ab} = P_a(\delta_{ab} - P_b)$, $\delta_{ab} = \begin{cases} 1 & \text{if } a=b \\ 0 & \text{o.w} \end{cases}$
- $\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial S_2} P_1^T$
- $\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial S_2}$
- $\frac{\partial L}{\partial P_1} = W_2^T \frac{\partial L}{\partial S_2}$

- $\frac{\partial L}{\partial S_1} = \text{diag}\ ((1 - \sigma(S_{1,j})) \sigma(S_{1,j}) \frac{\partial L}{\partial P_1}$
- $\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial S_1} x^T$
- $\frac{\partial L}{\partial b} = \frac{\partial L}{\partial S_1}$