

# Lecture 4: Backpropagation and Neural Networks

# Administrative

**Assignment 1 due Thursday April 20, 11:59pm on Canvas**

# Administrative

**Project:** TA specialities and some project ideas are posted on Piazza

# Administrative

**Google Cloud:** All registered students will receive an email this week with instructions on how to redeem \$100 in credits

# Where we are...

$$s = f(x; W) = Wx$$

scores function

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

SVM loss

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \sum_k W_k^2$$

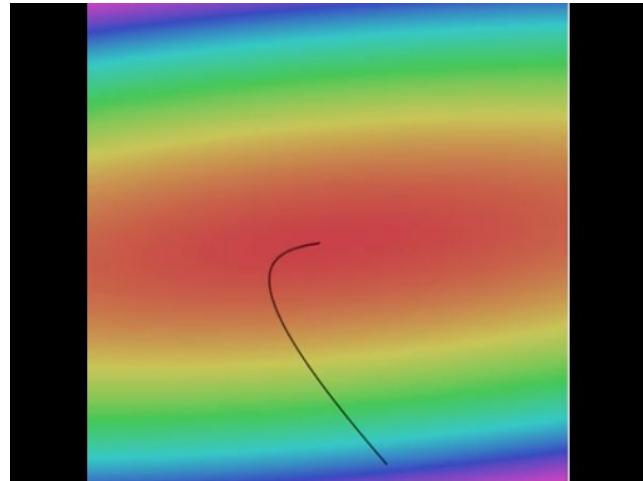
data loss + regularization



want  $\nabla_W L$

# Optimization

Loss  $\equiv$  玄函数



```
# Vanilla Gradient Descent

while True:
    weights_grad = evaluate_gradient(loss_fun, data, weights)
    weights += - step_size * weights_grad # perform parameter update
```

Landscape image is [CC0 1.0](#) public domain  
Walking man image is [CC0 1.0](#) public domain

# Gradient descent

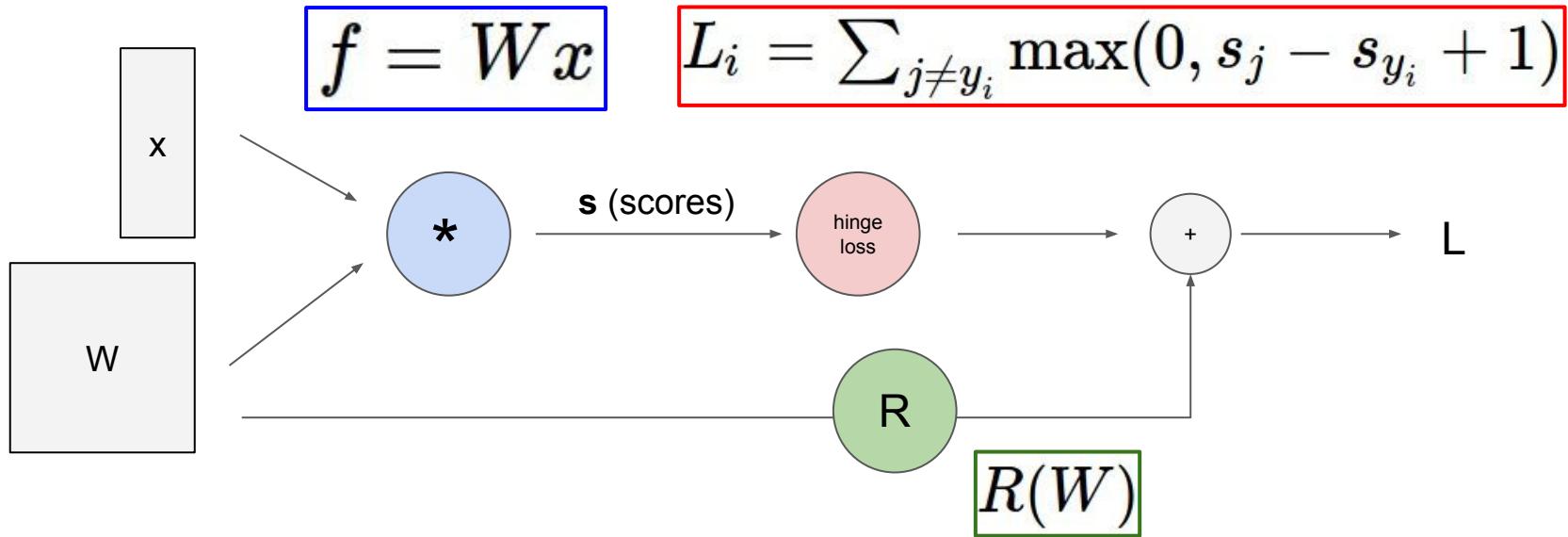
$$\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h}$$

**Numerical gradient:** slow :, approximate :, easy to write :)

**Analytic gradient:** fast :), exact :), error-prone :(

In practice: Derive analytic gradient, check your implementation with numerical gradient

# Computational graphs



# Convolutional network (AlexNet)

input image

weights

loss

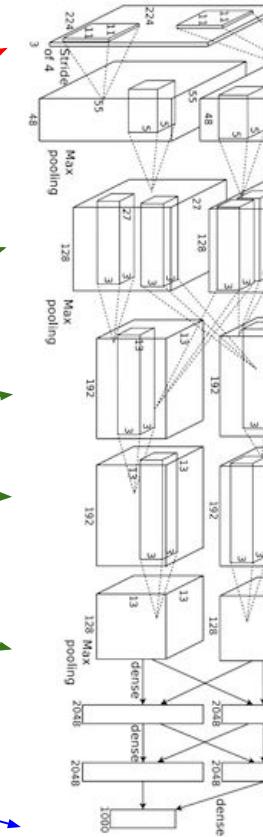


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

# Neural Turing Machine

input image

loss

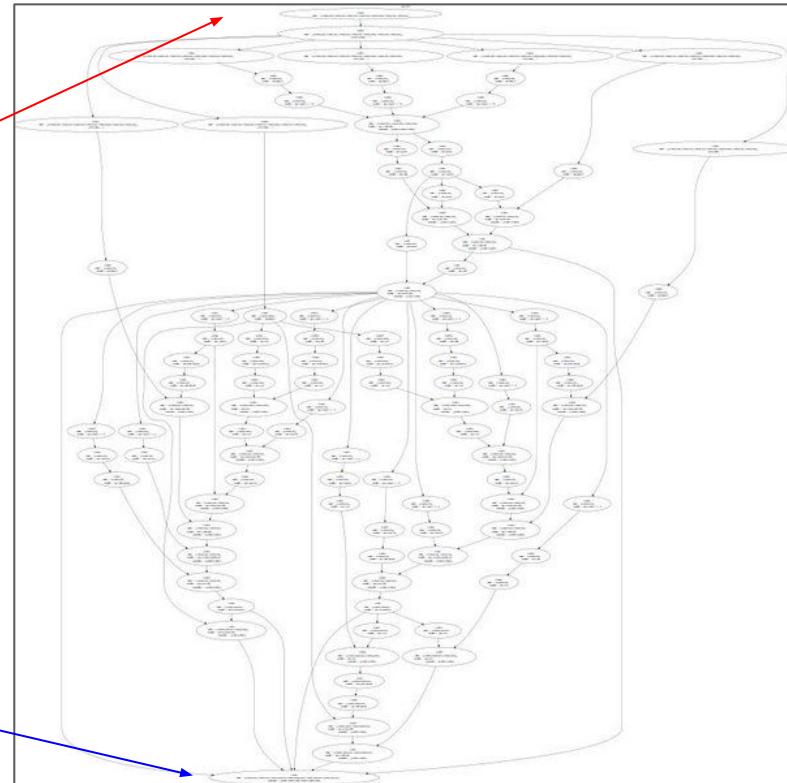


Figure reproduced with permission from a [Twitter post](#) by Andrej Karpathy.

# Neural Turing Machine

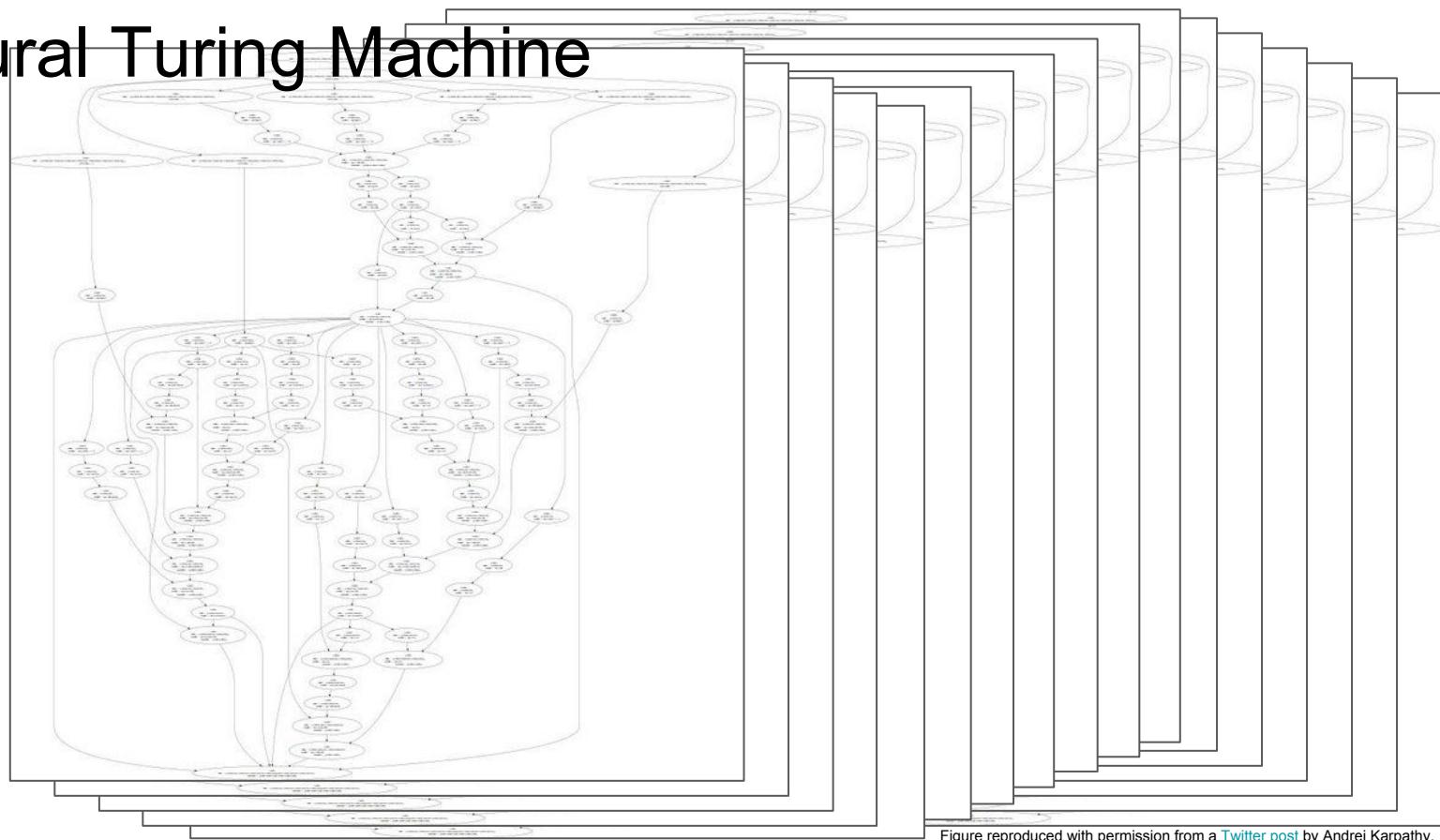
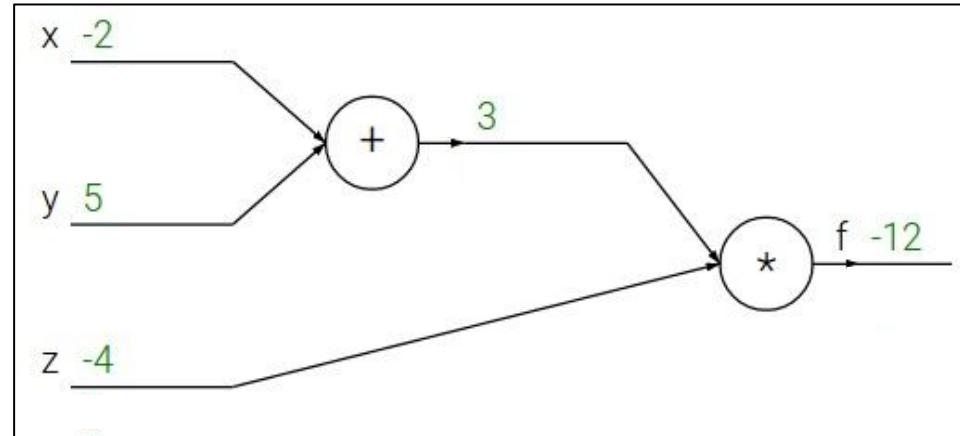


Figure reproduced with permission from a [Twitter post](#) by Andrej Karpathy.

## Backpropagation: a simple example

$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$



## Backpropagation: a simple example

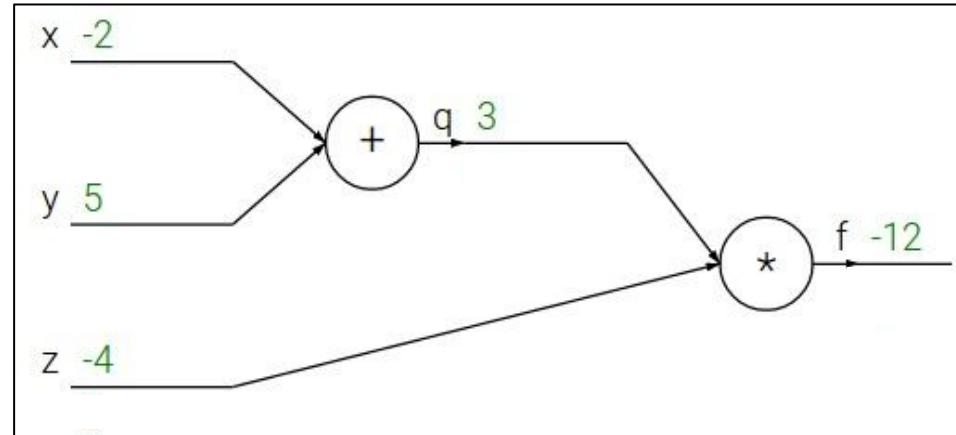
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



## Backpropagation: a simple example

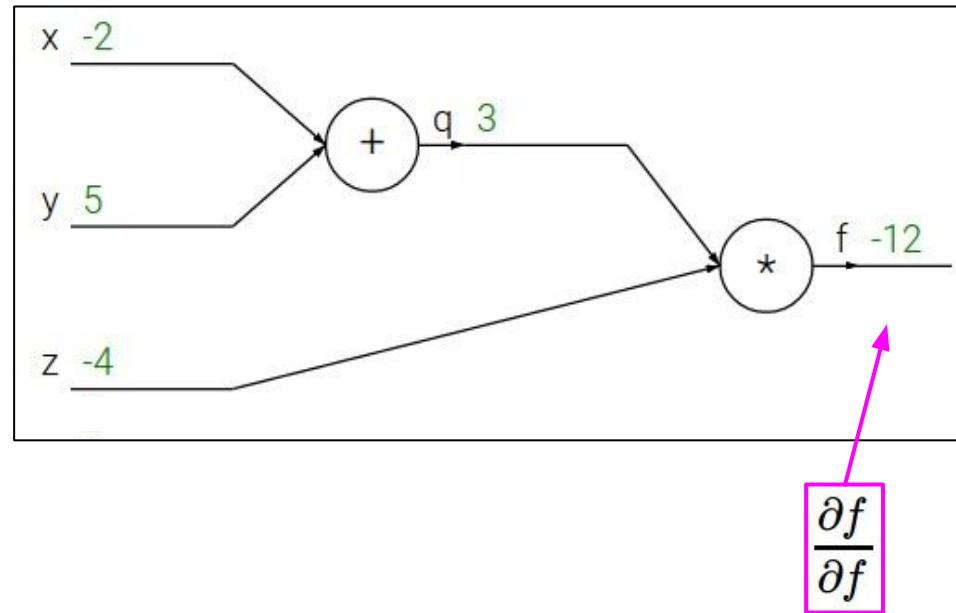
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



## Backpropagation: a simple example

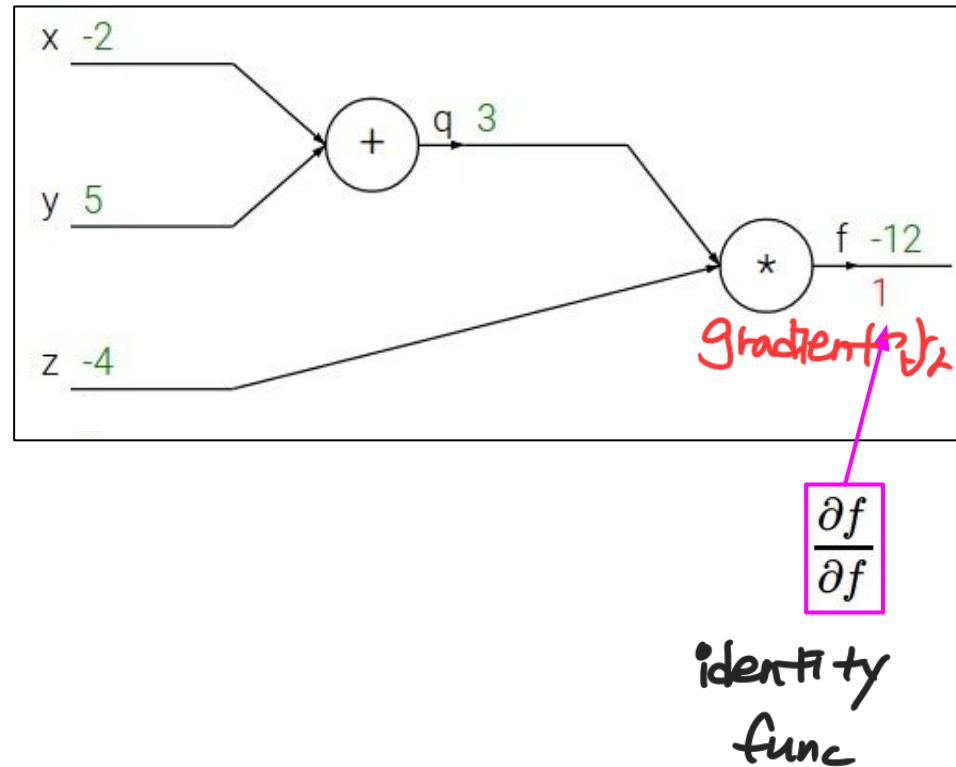
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



identity  
func

## Backpropagation: a simple example

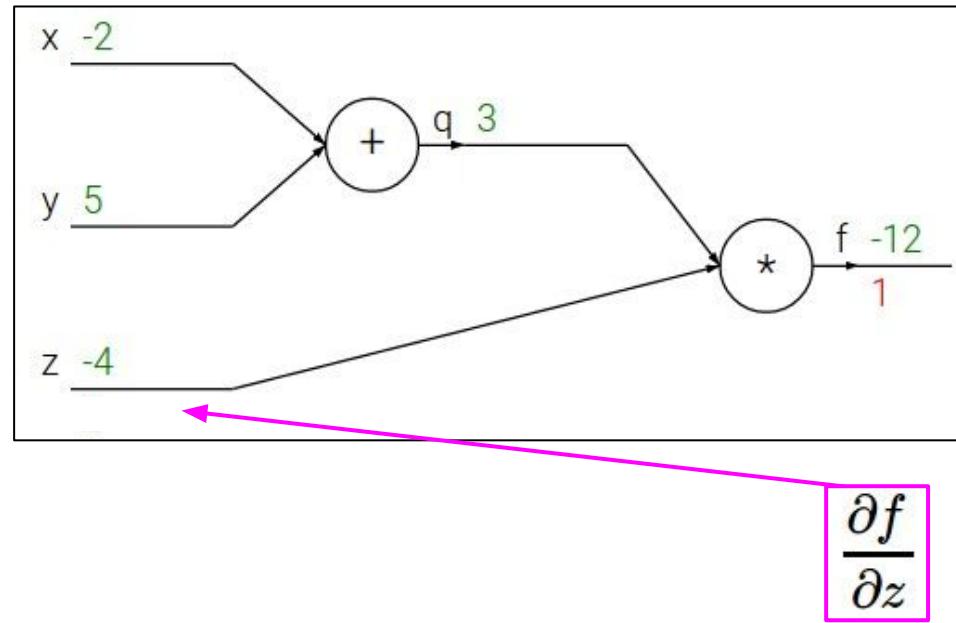
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial z}$$

## Backpropagation: a simple example

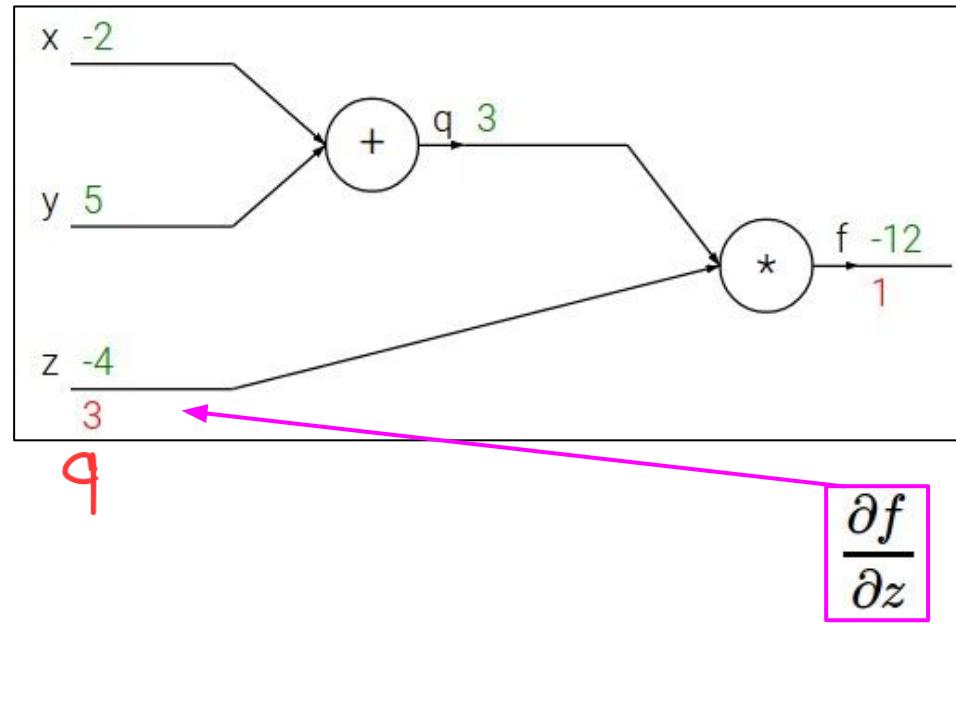
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



## Backpropagation: a simple example

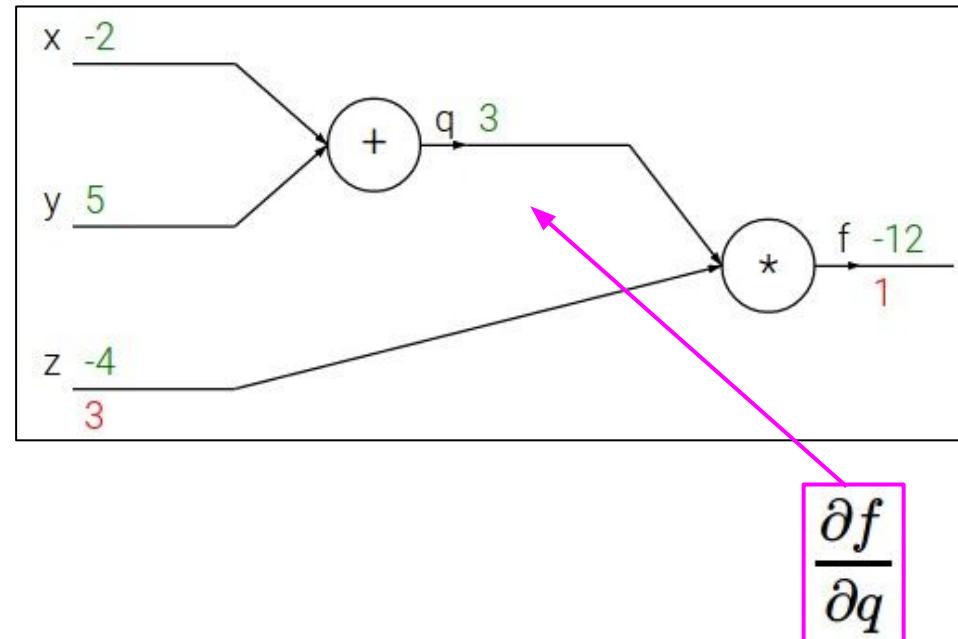
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



## Backpropagation: a simple example

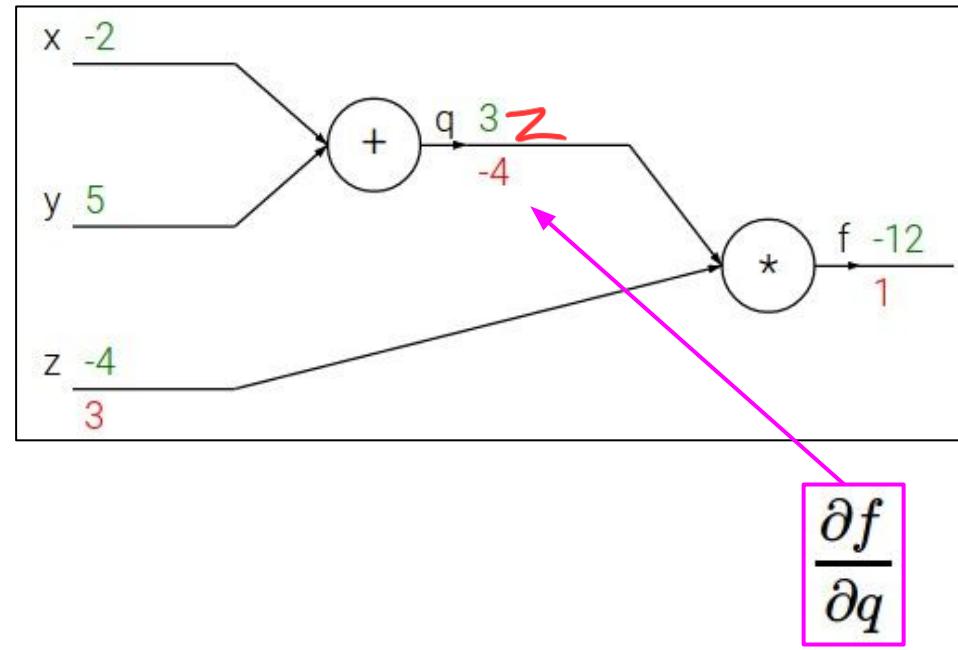
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



## Backpropagation: a simple example

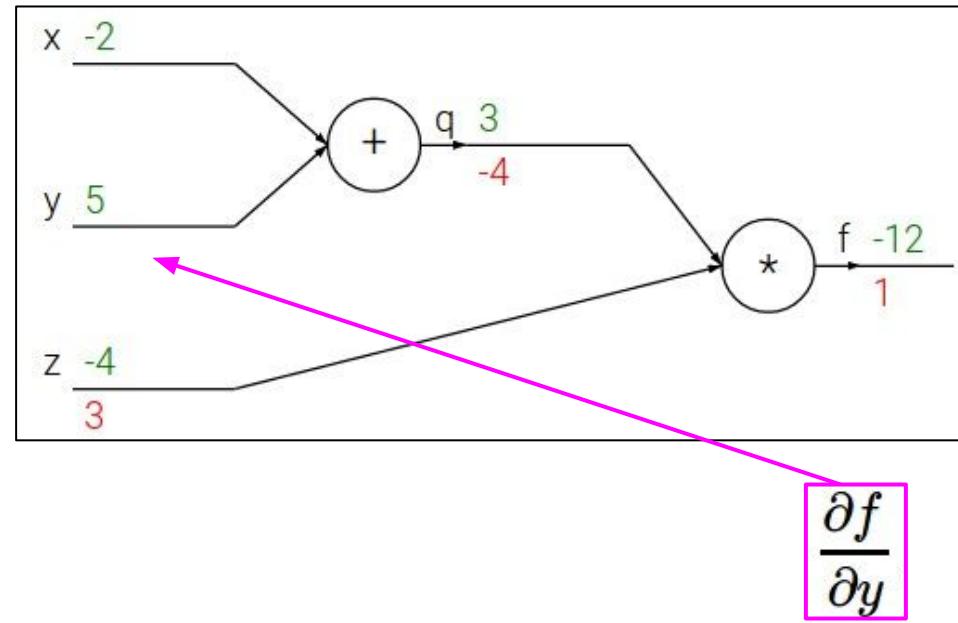
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



## Backpropagation: a simple example

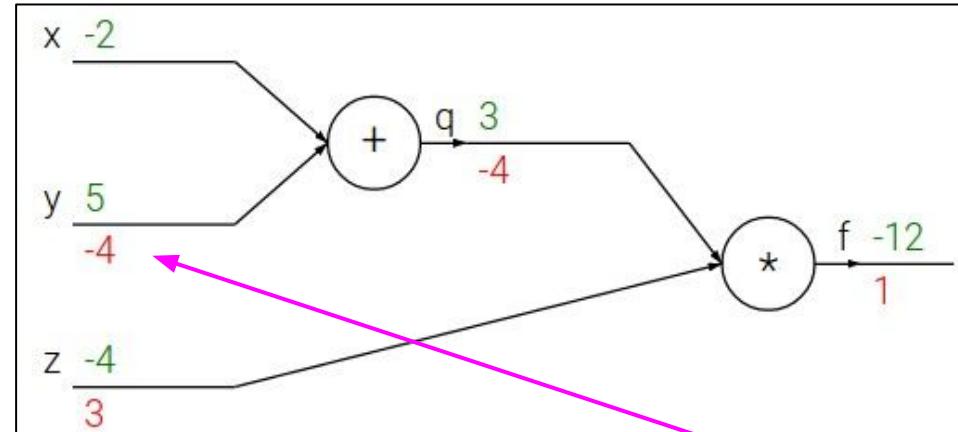
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Chain rule:

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial y}$$

$$\frac{\partial f}{\partial y}$$

## Backpropagation: a simple example

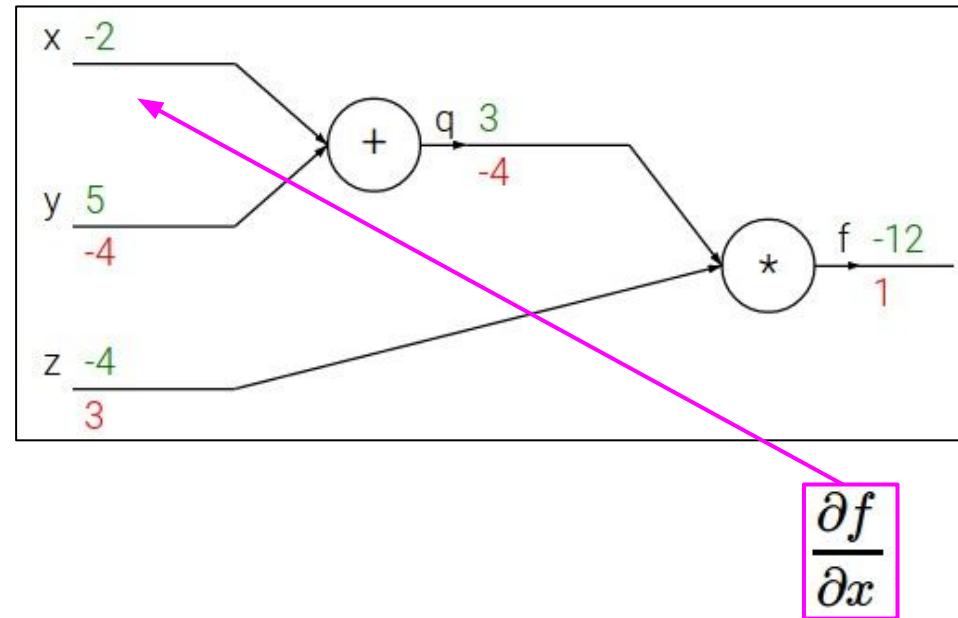
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



## Backpropagation: a simple example

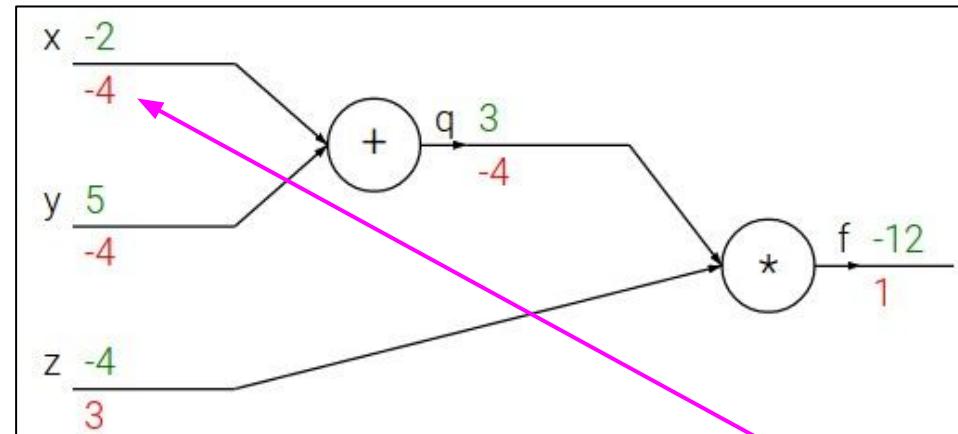
$$f(x, y, z) = (x + y)z$$

e.g.  $x = -2$ ,  $y = 5$ ,  $z = -4$

$$q = x + y \quad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

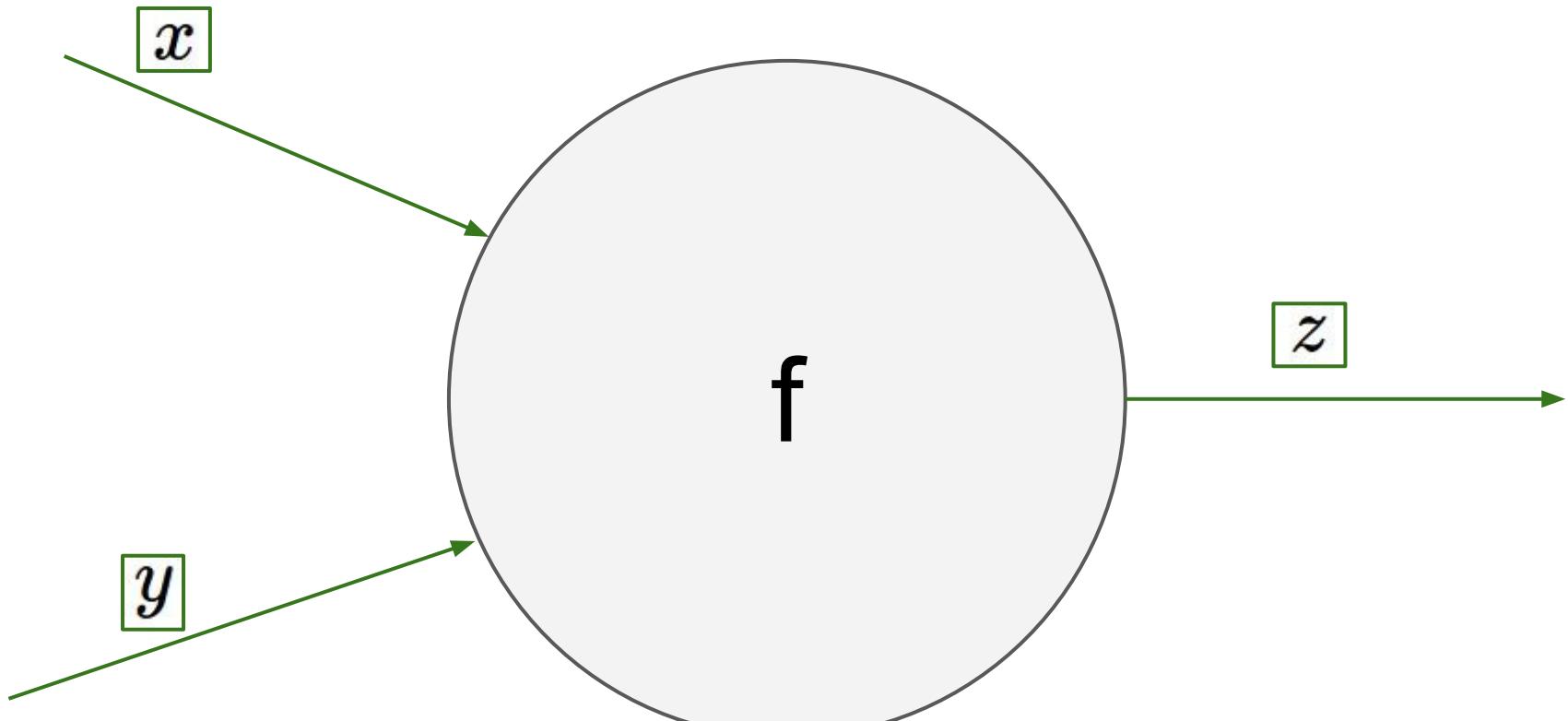
$$f = qz \quad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

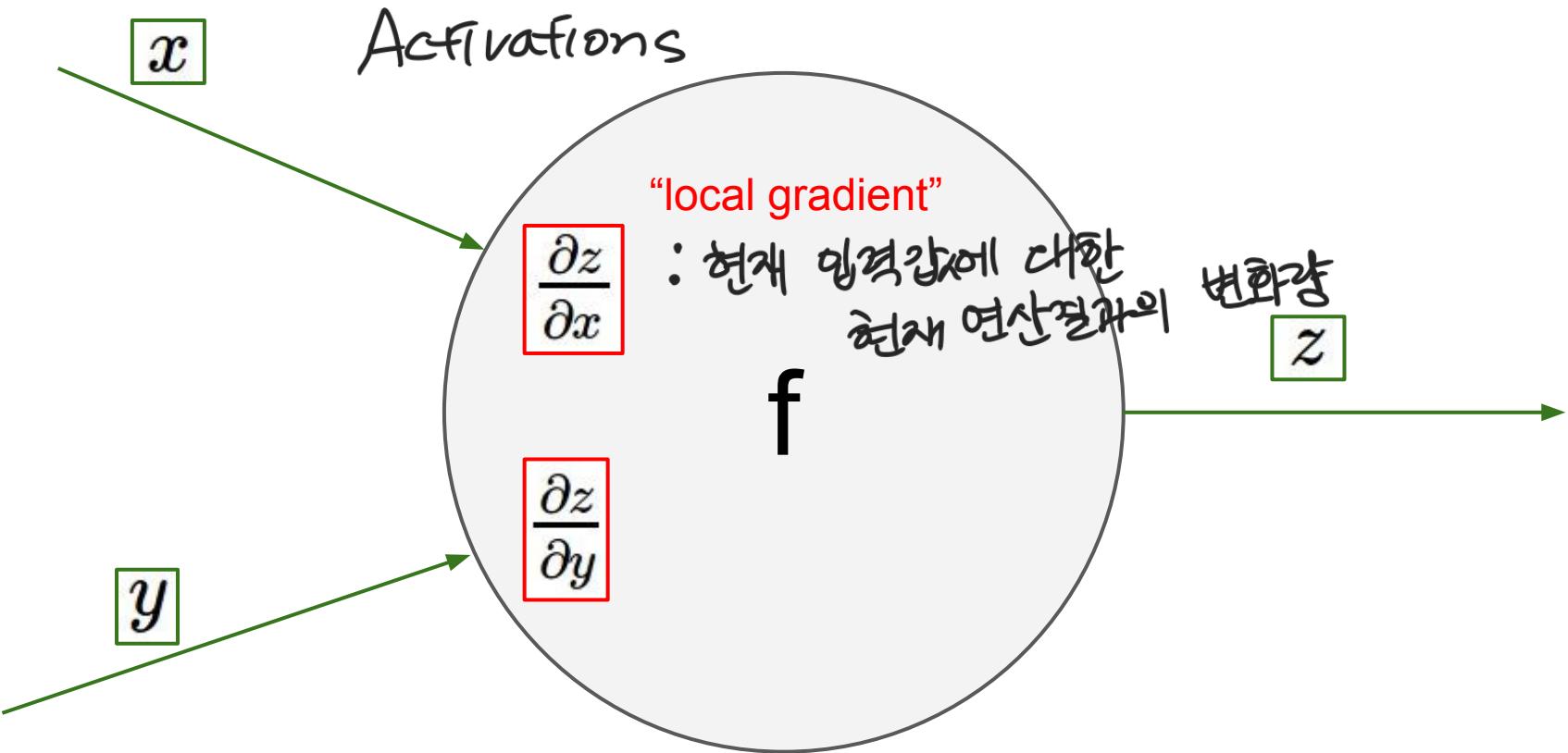
Want:  $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

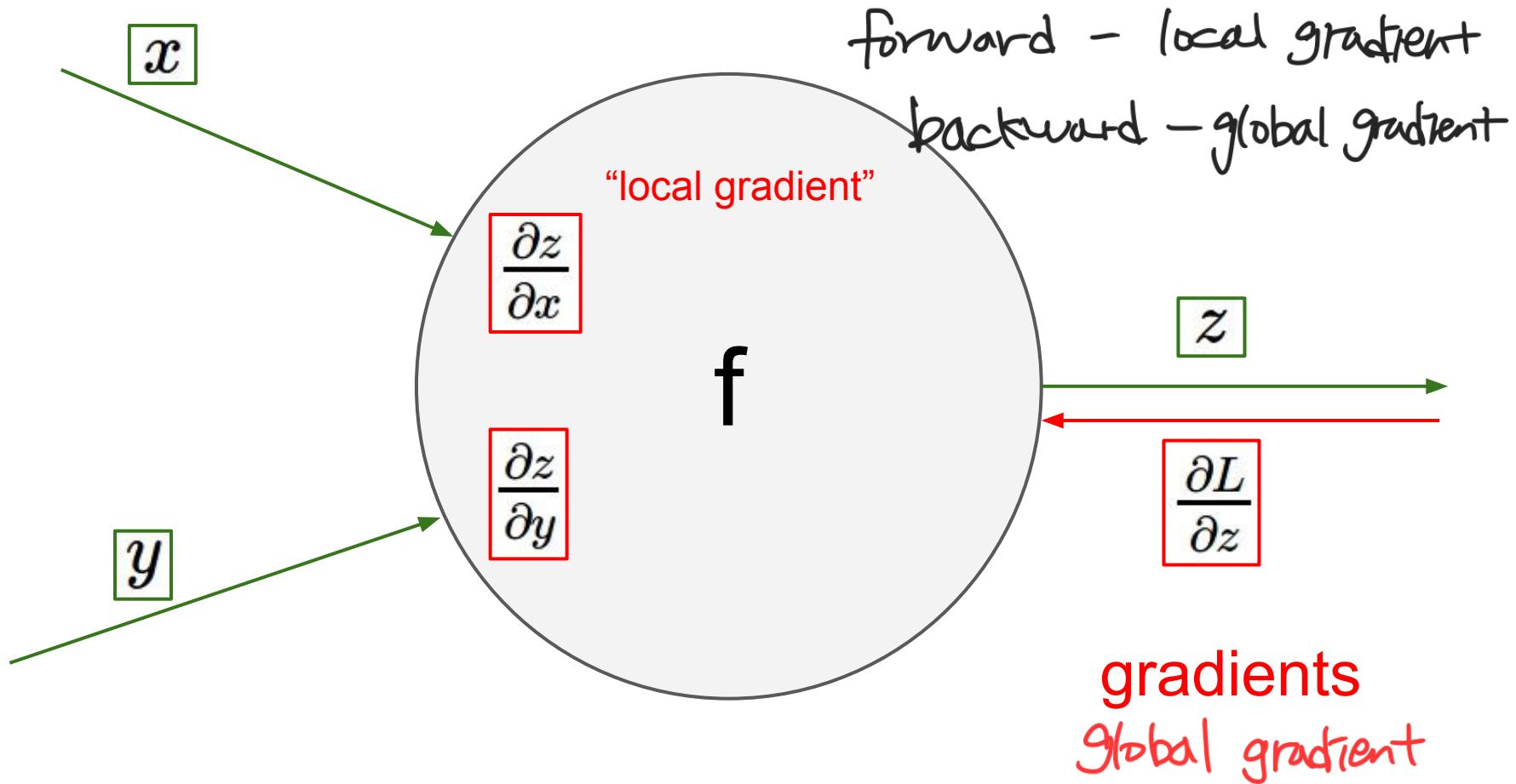


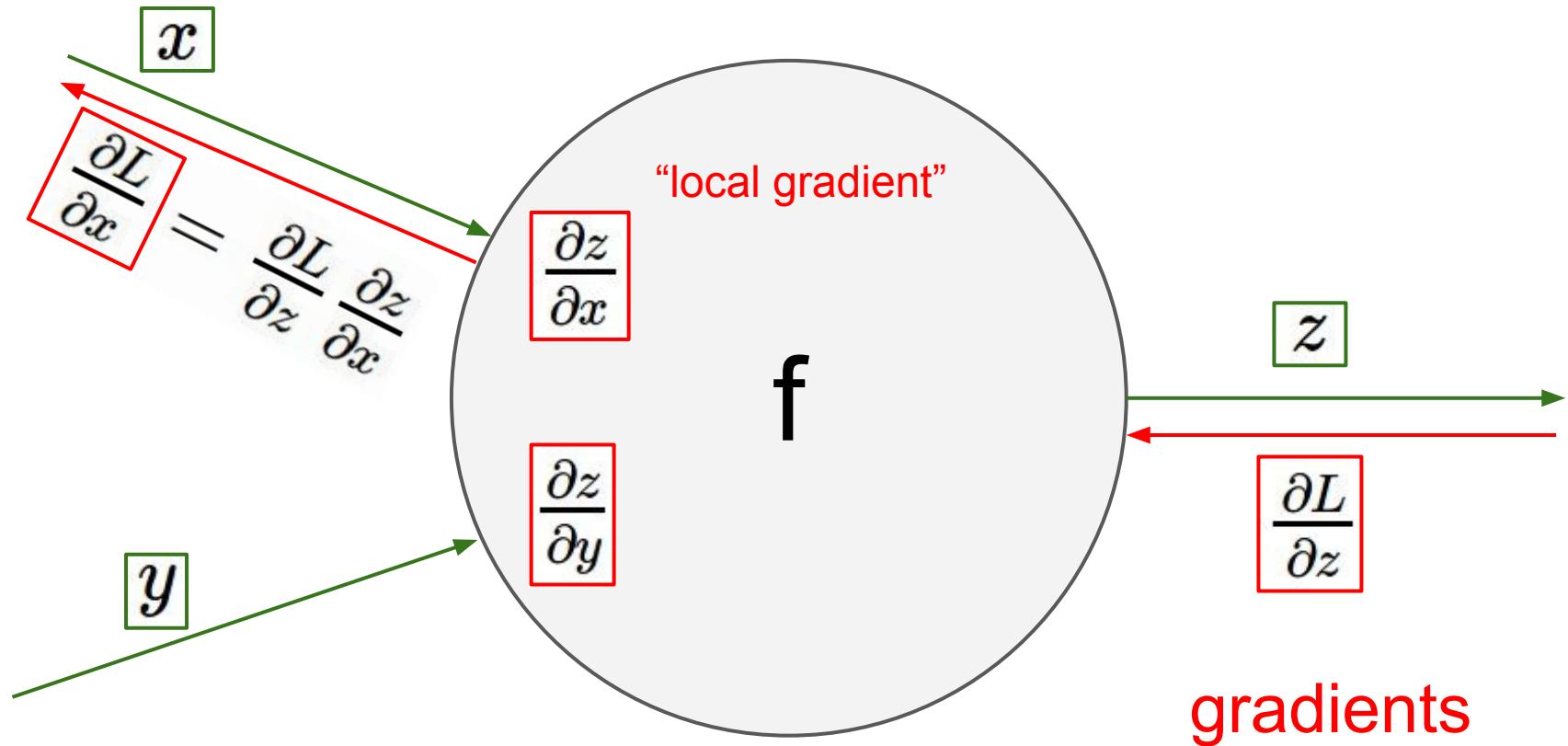
Chain rule:

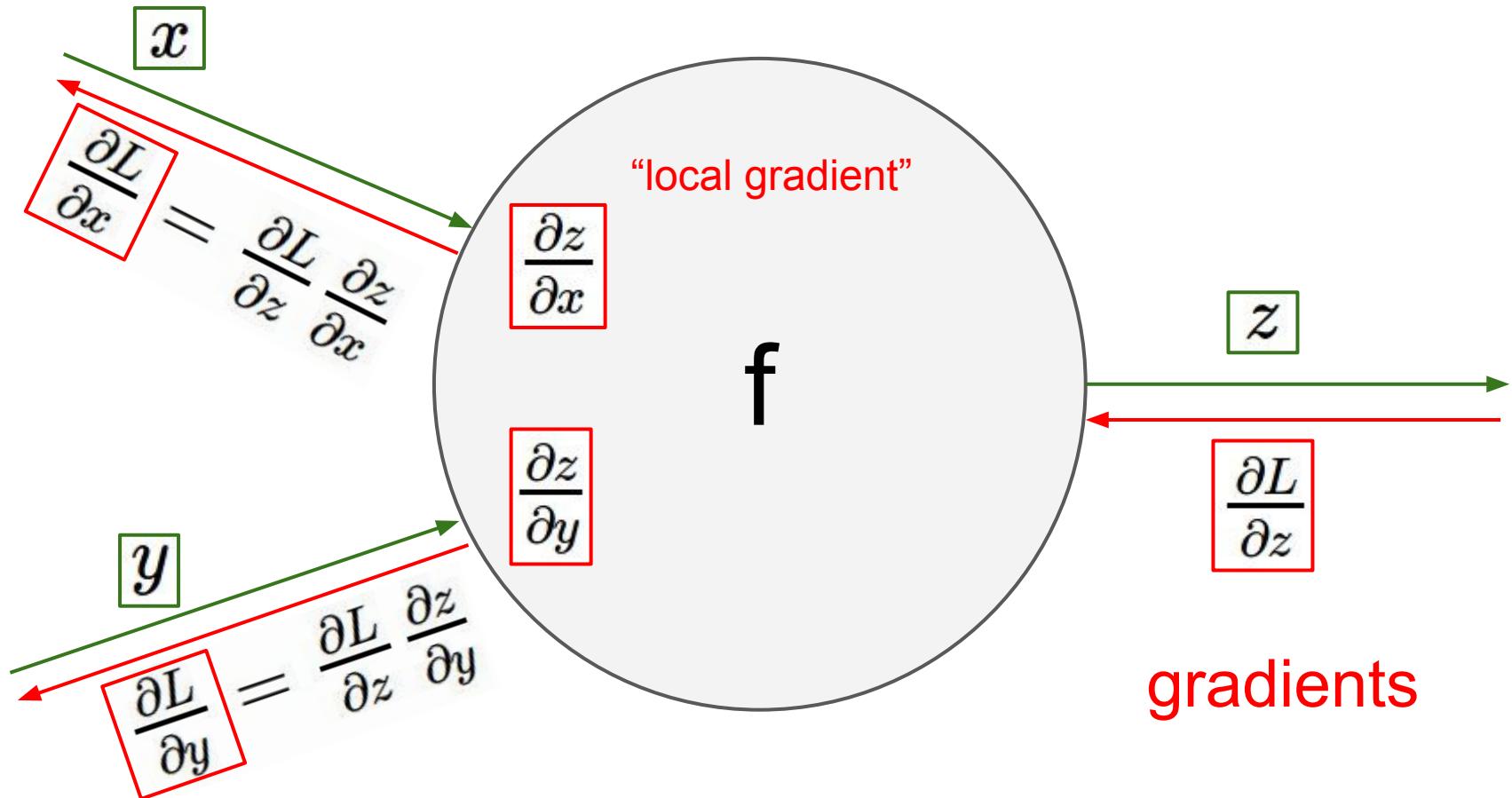
$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$$

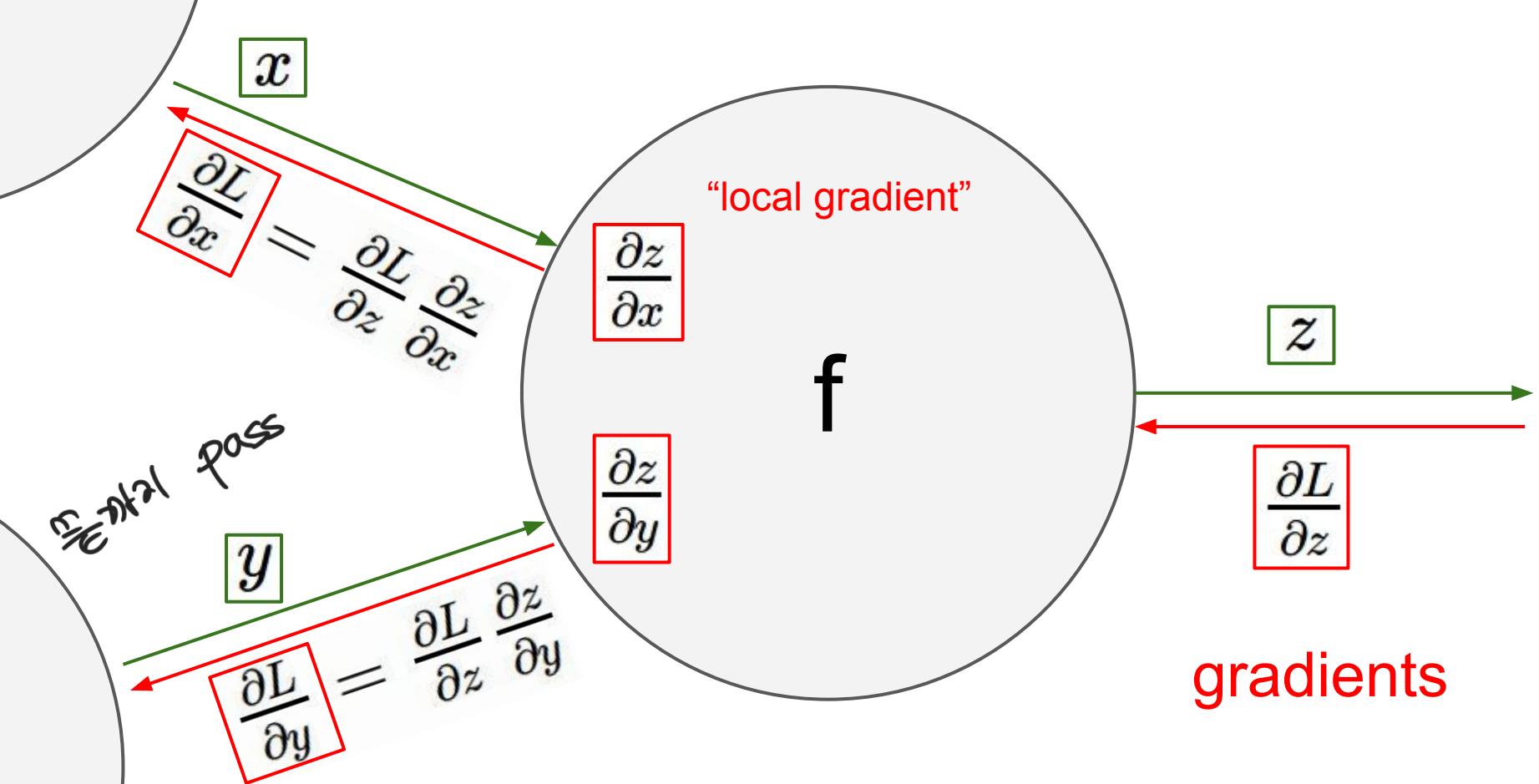






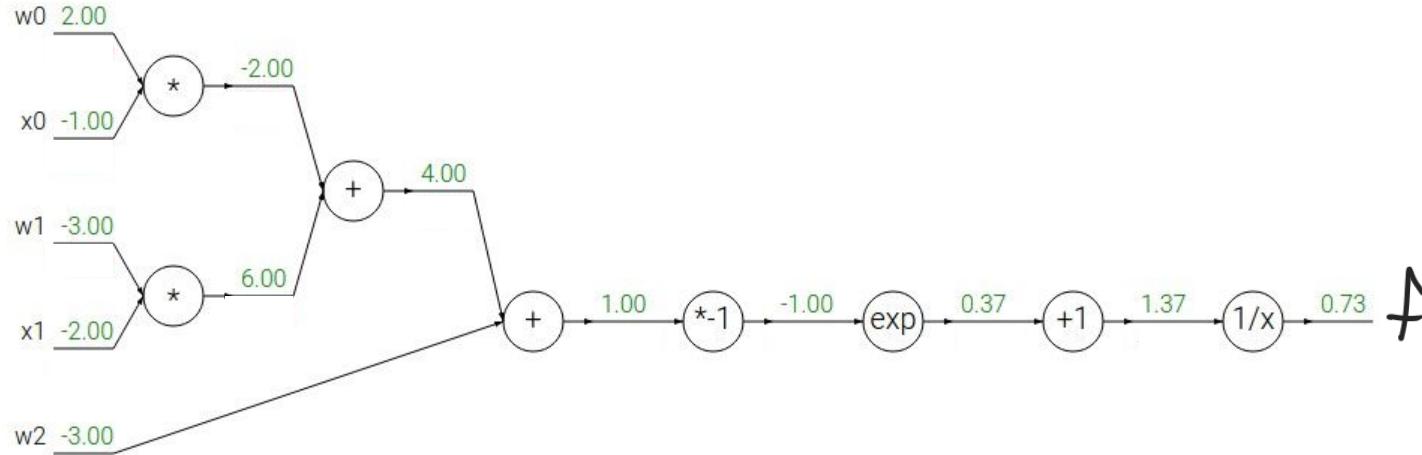






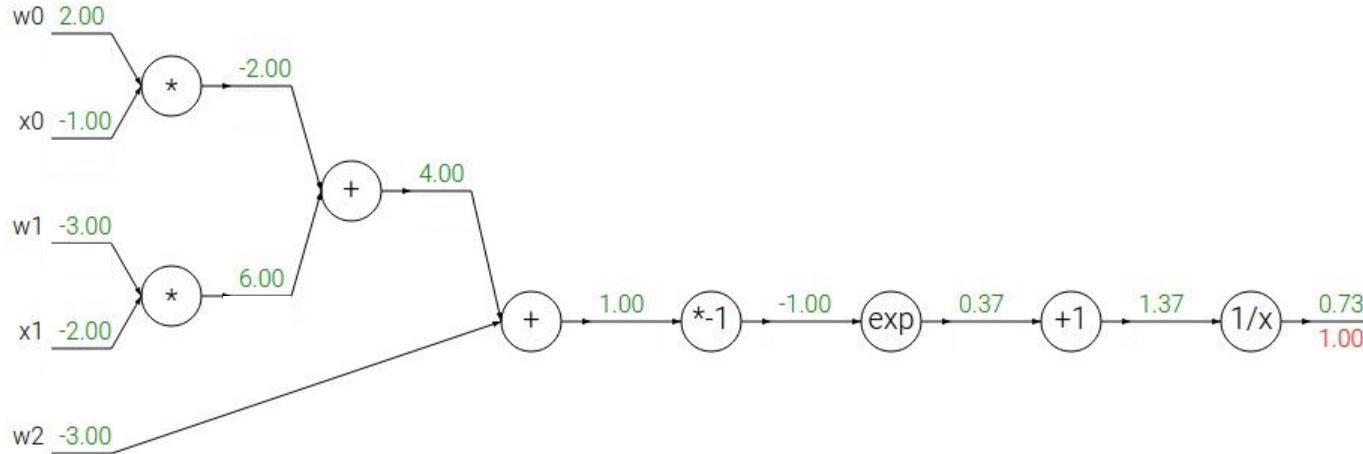
Another example:

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



Another example:

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



$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$$f_c(x) = c + x$$

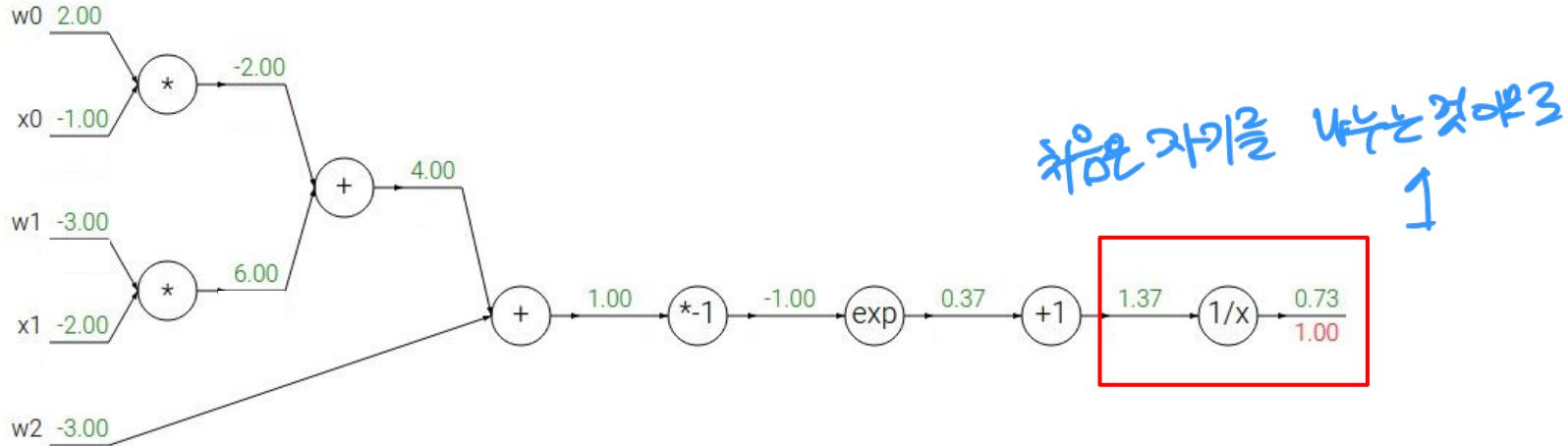
$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

→

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

→

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

→

$$\frac{df}{dx} = -1/x^2$$

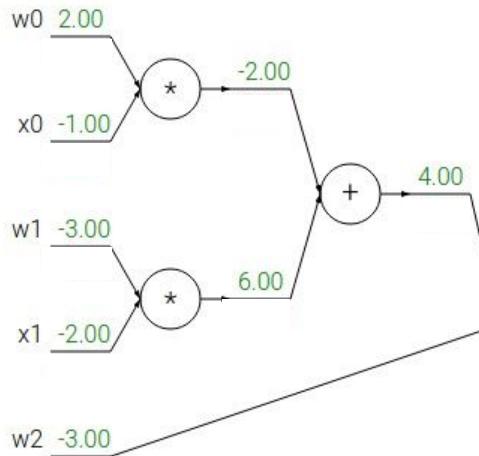
$$f_c(x) = c + x$$

→

$$\frac{df}{dx} = 1$$

Another example:

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



$$\frac{dl}{db} \times \frac{db}{da}$$

global grad local grad  
 $(\frac{-1}{1.37^2})(1.00) = -0.53$

$$(\frac{-1}{1.37^2}) \times \text{global } \nabla$$

$da$        $dl$   
 $1.37$        $-0.53$   
 $0.37$        $0.73$   
 $1.00$        $1.00$

$$f(x) = e^x$$

→

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

→

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

→

$$\frac{df}{dx} = -1/x^2$$

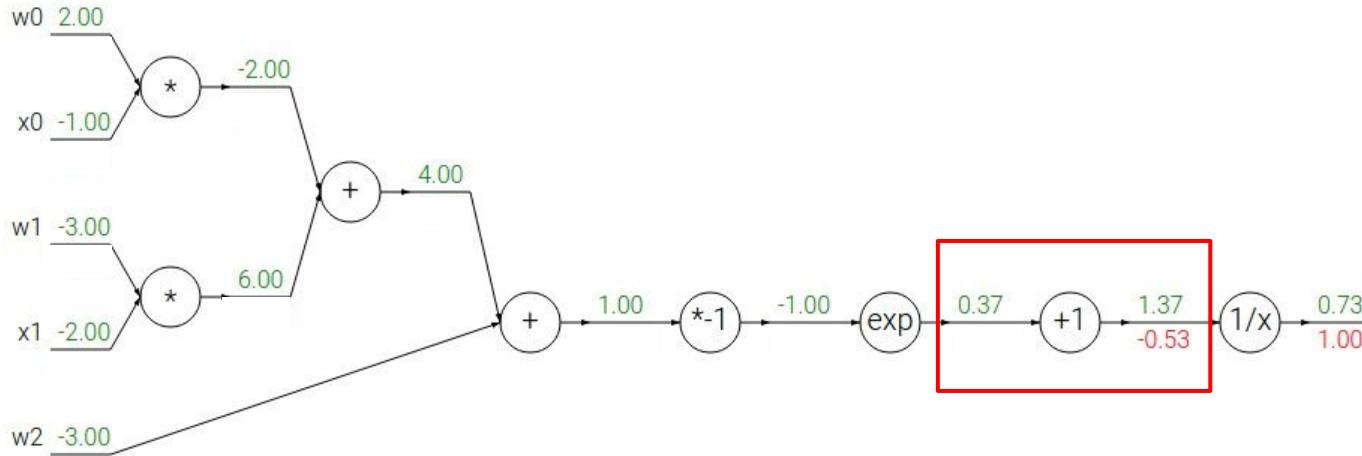
$$f_c(x) = c + x$$

→

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

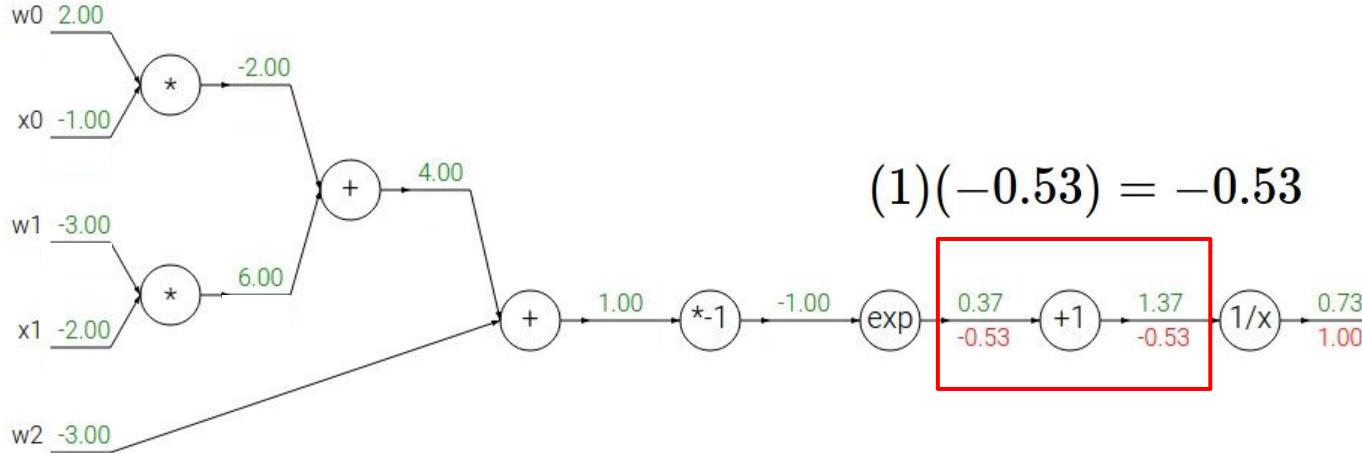
$$f_c(x) = c + x$$

$\rightarrow$

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

→

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

→

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

→

$$\frac{df}{dx} = -1/x^2$$

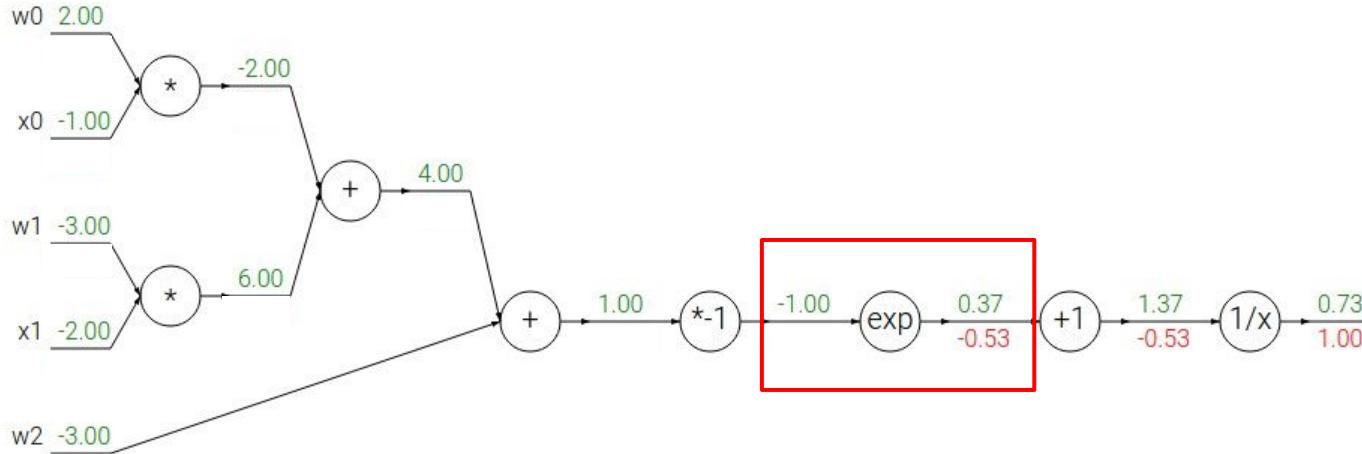
$$f_c(x) = c + x$$

→

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

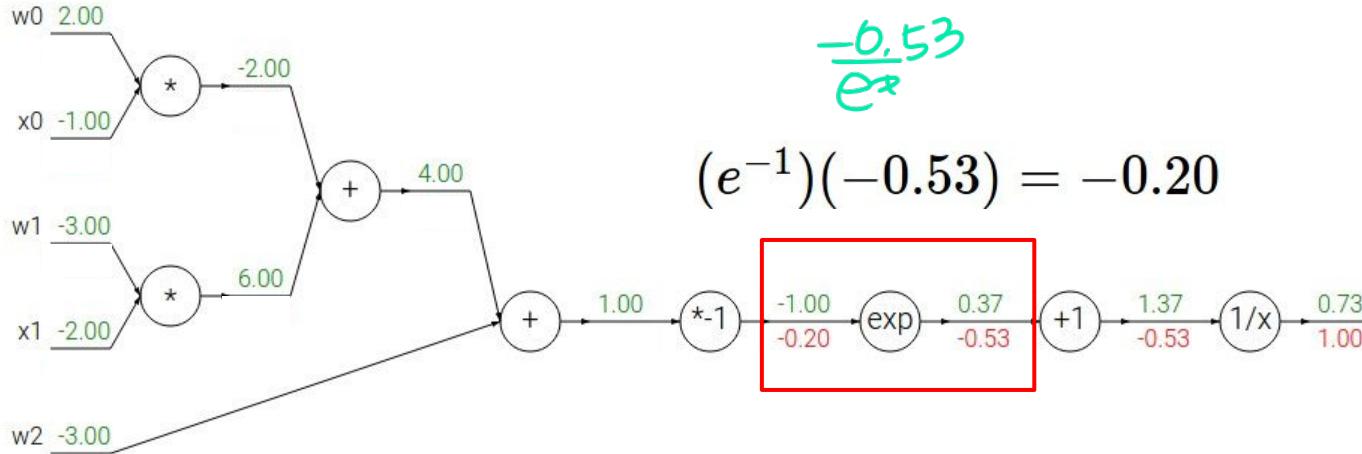
$$f_c(x) = c + x$$

$\rightarrow$

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

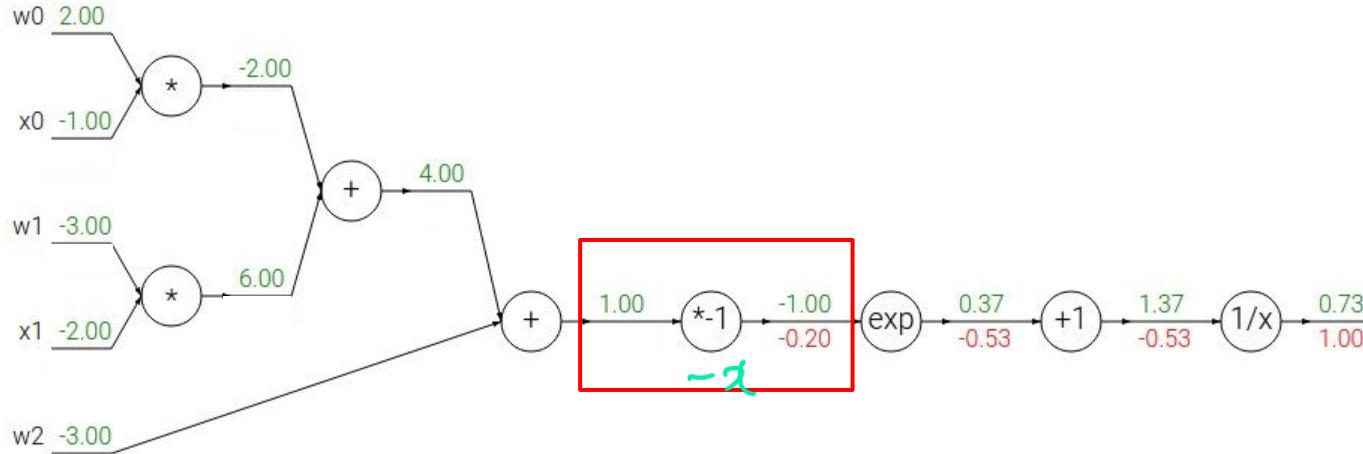
$$f_c(x) = c + x$$

$\rightarrow$

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$$f_c(x) = c + x$$

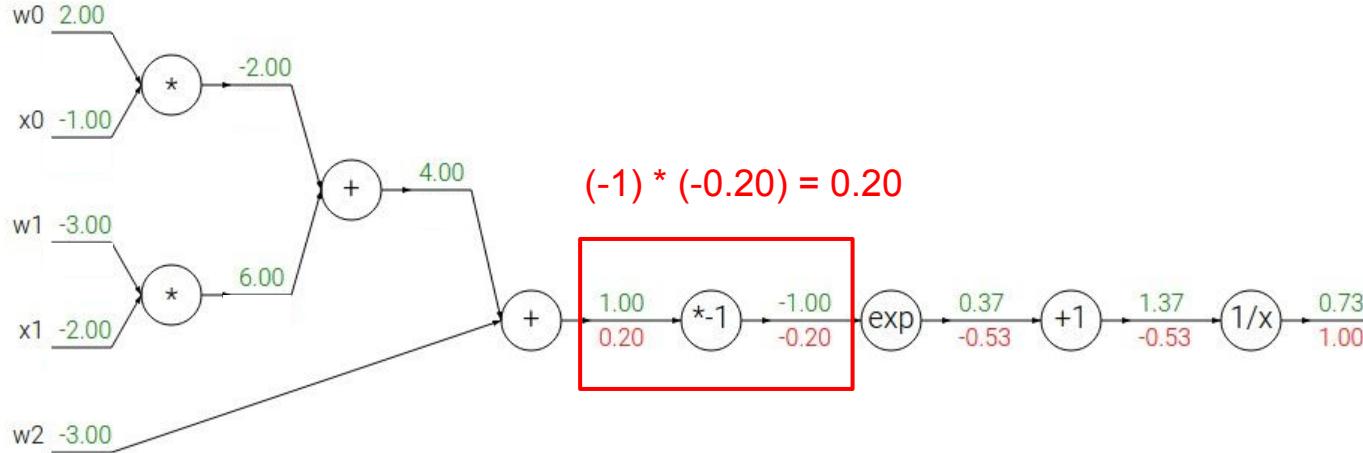
$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

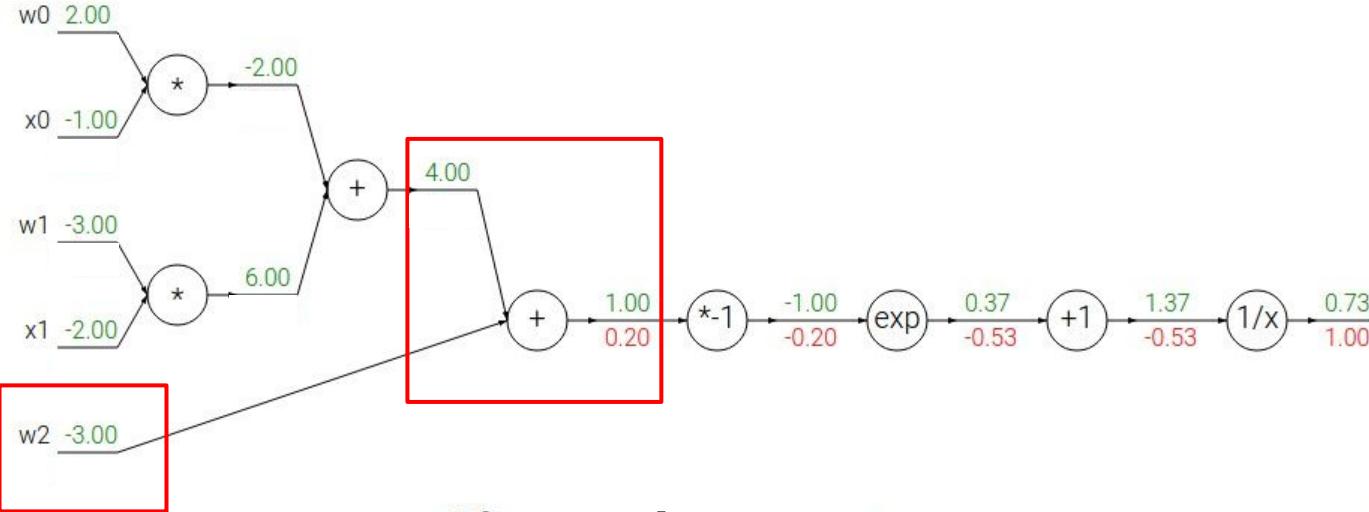
$$f_c(x) = c + x$$

$\rightarrow$

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$$f_c(x) = c + x$$

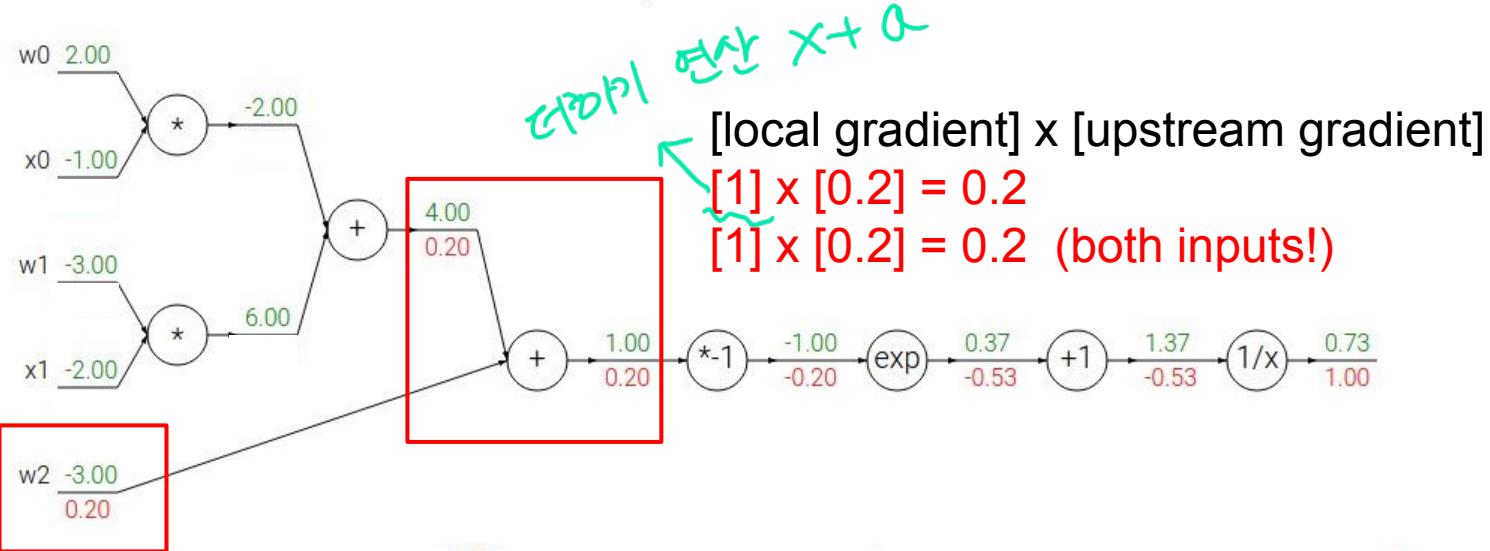
$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

→

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

→

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

→

$$\frac{df}{dx} = -1/x^2$$

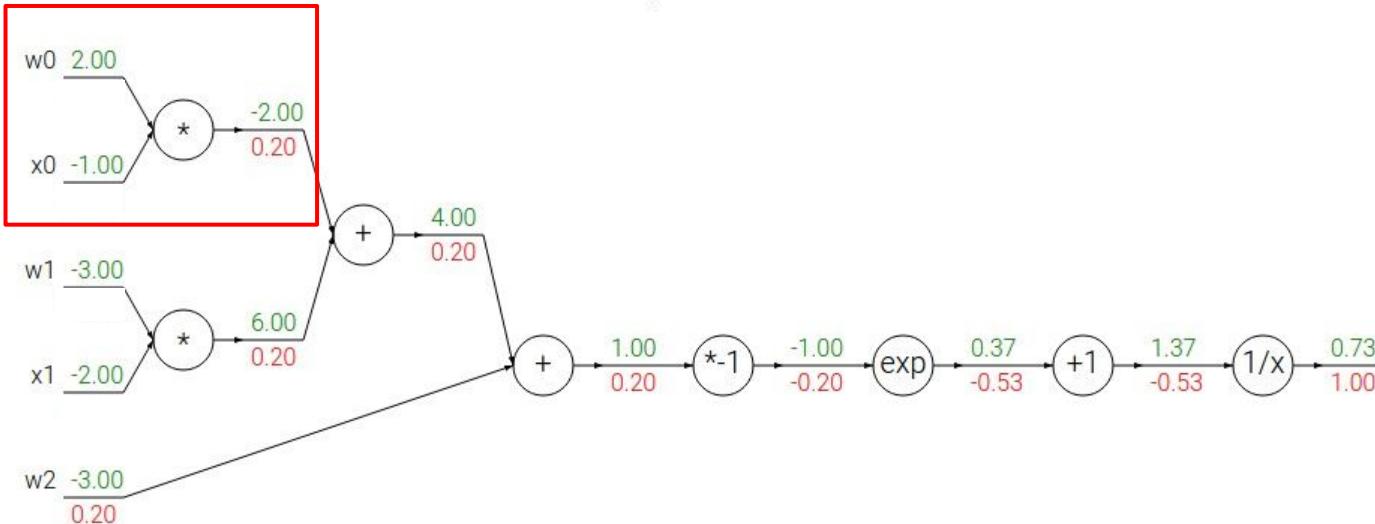
$$f_c(x) = c + x$$

→

$$\frac{df}{dx} = 1$$

Another example:

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



$$f(x) = e^x$$

→

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

→

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$$f_c(x) = c + x$$

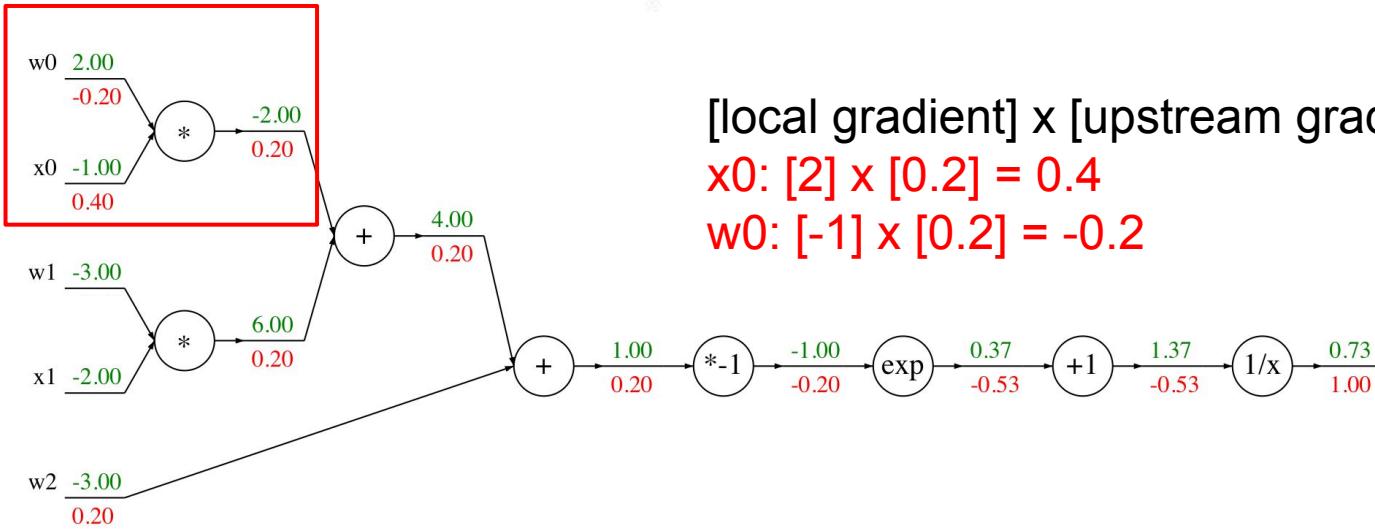
→

$$\frac{df}{dx} = -1/x^2$$

$$\frac{df}{dx} = 1$$

Another example:

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



[local gradient] x [upstream gradient]  
 $x_0: [2] \times [0.2] = 0.4$   
 $w_0: [-1] \times [0.2] = -0.2$

$$f(x) = e^x$$

$\rightarrow$

$$\frac{df}{dx} = e^x$$

$$f_a(x) = ax$$

$\rightarrow$

$$\frac{df}{dx} = a$$

$$f(x) = \frac{1}{x}$$

$$f_c(x) = c + x$$

$\rightarrow$

$$\frac{df}{dx} = -1/x^2$$

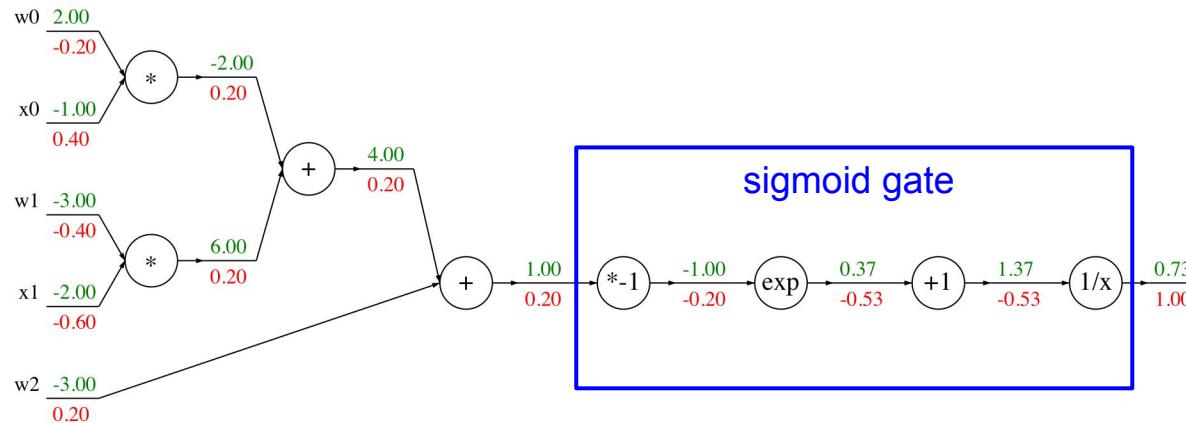
$$\frac{df}{dx} = 1$$

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

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

sigmoid function

$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2} = \left( \frac{1 + e^{-x} - 1}{1 + e^{-x}} \right) \left( \frac{1}{1 + e^{-x}} \right) = (1 - \sigma(x))\sigma(x)$$

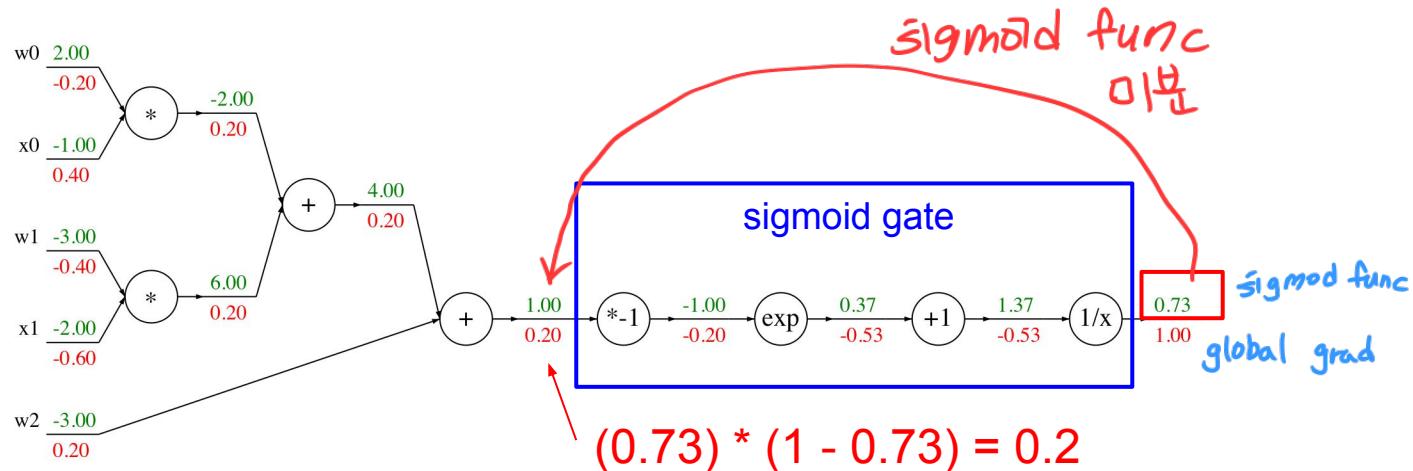


$$f(w, x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2)}}$$

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

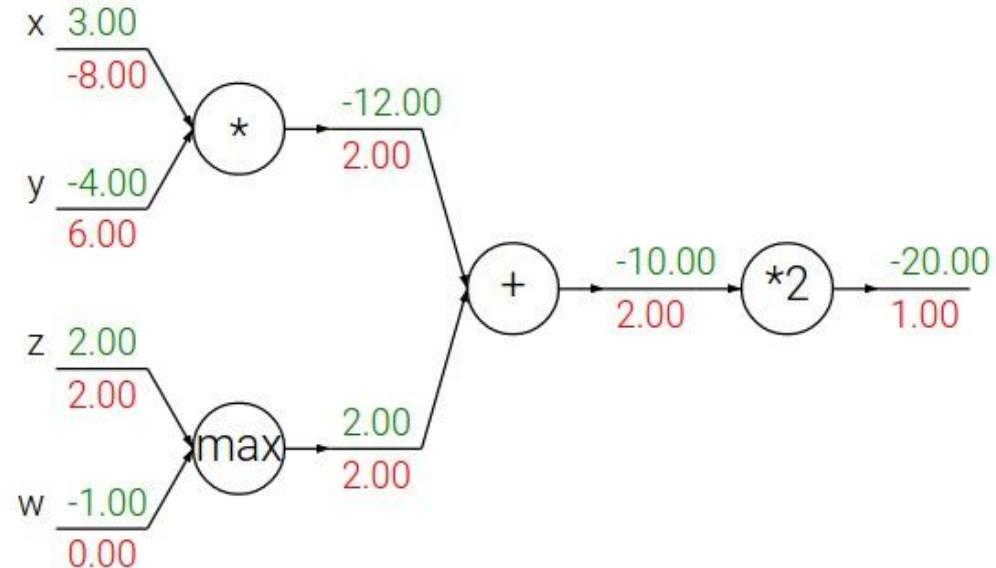
sigmoid function

$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2} = \left( \frac{1 + e^{-x} - 1}{1 + e^{-x}} \right) \left( \frac{1}{1 + e^{-x}} \right) = \boxed{(1 - \sigma(x)) \sigma(x)}$$



# Patterns in backward flow

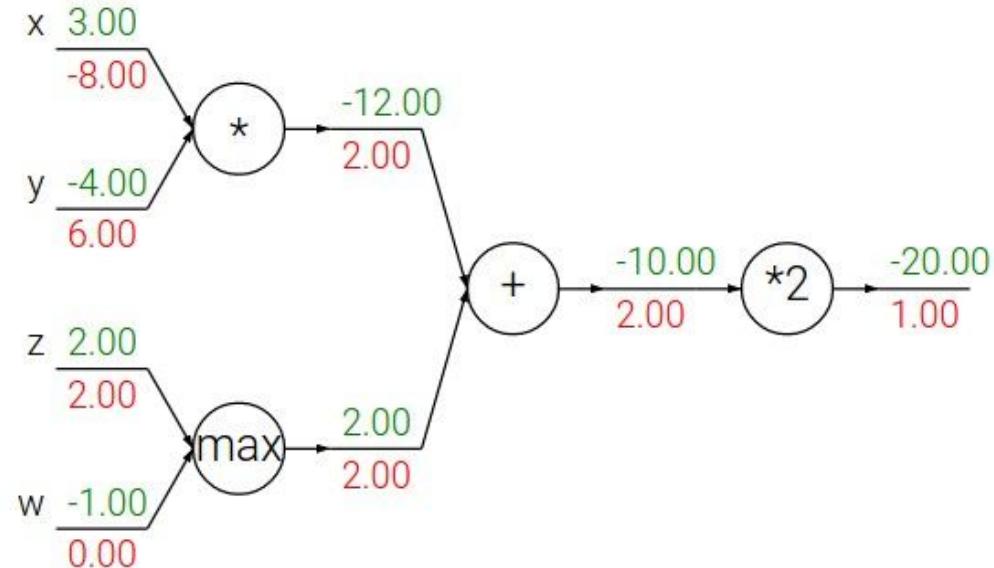
**add gate: gradient distributor**



# Patterns in backward flow

**add gate:** gradient distributor

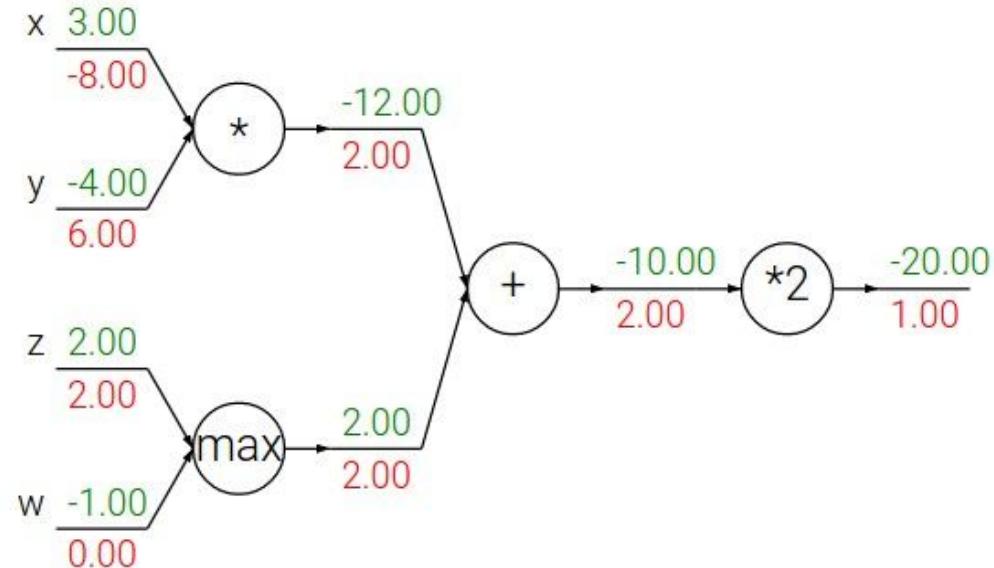
Q: What is a **max** gate?



# Patterns in backward flow

**add** gate: gradient distributor

**max** gate: gradient router

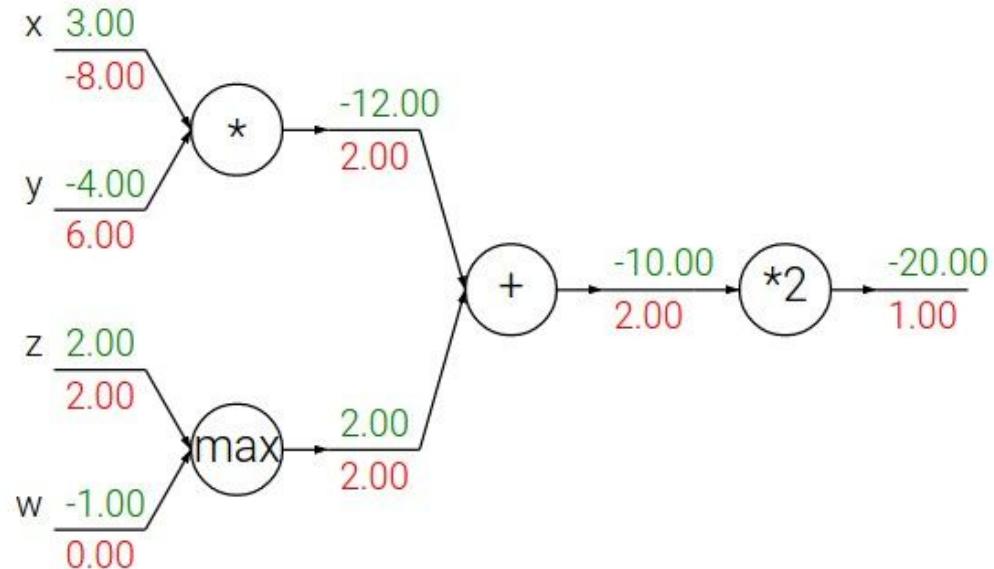


# Patterns in backward flow

**add** gate: gradient distributor

**max** gate: gradient router

Q: What is a **mul** gate?



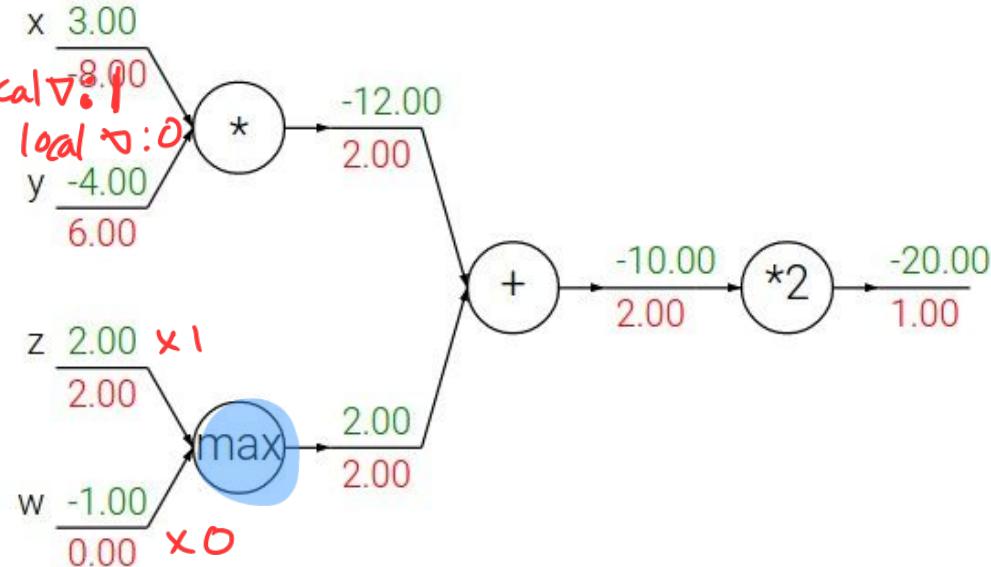
# Patterns in backward flow

**add gate:** gradient distributor

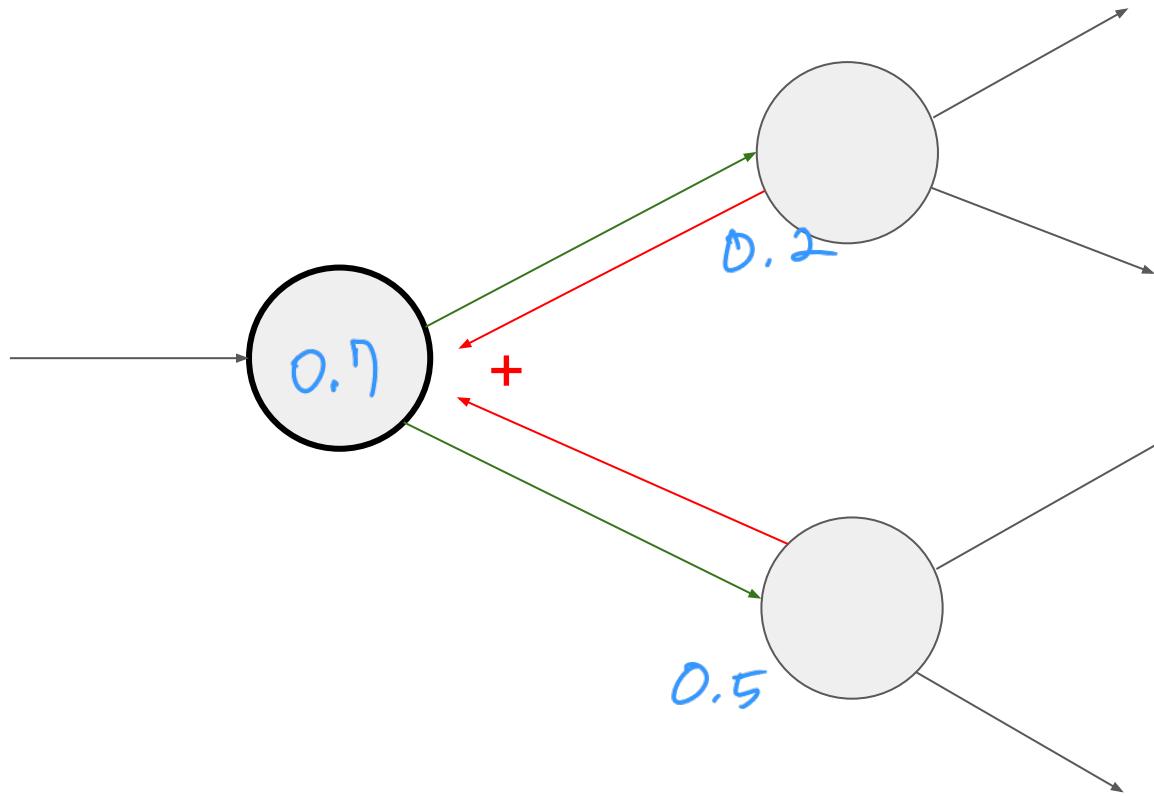
**max gate:** gradient router

**mul gate:** gradient switcher

반대로 더 주해짐



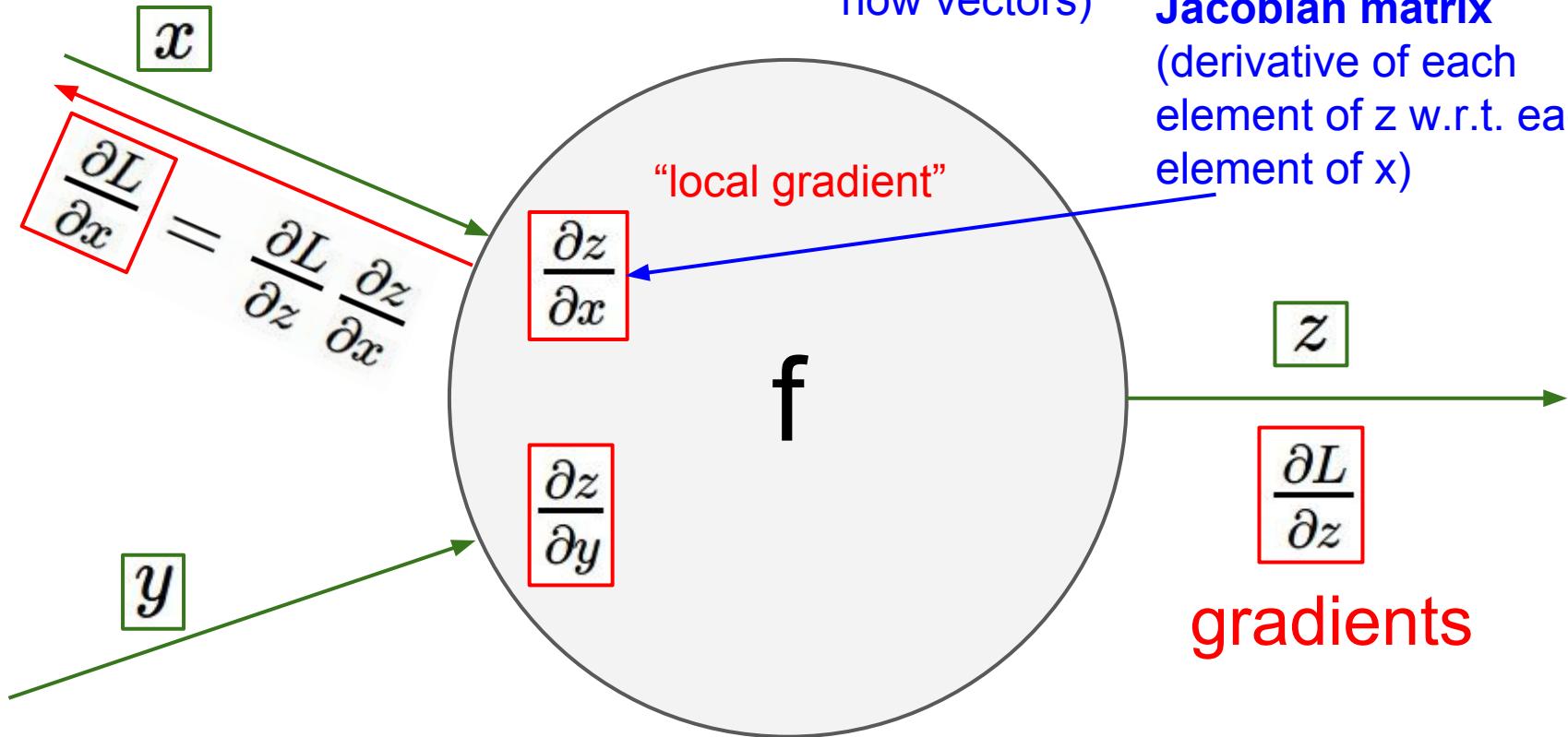
# Gradients add at branches



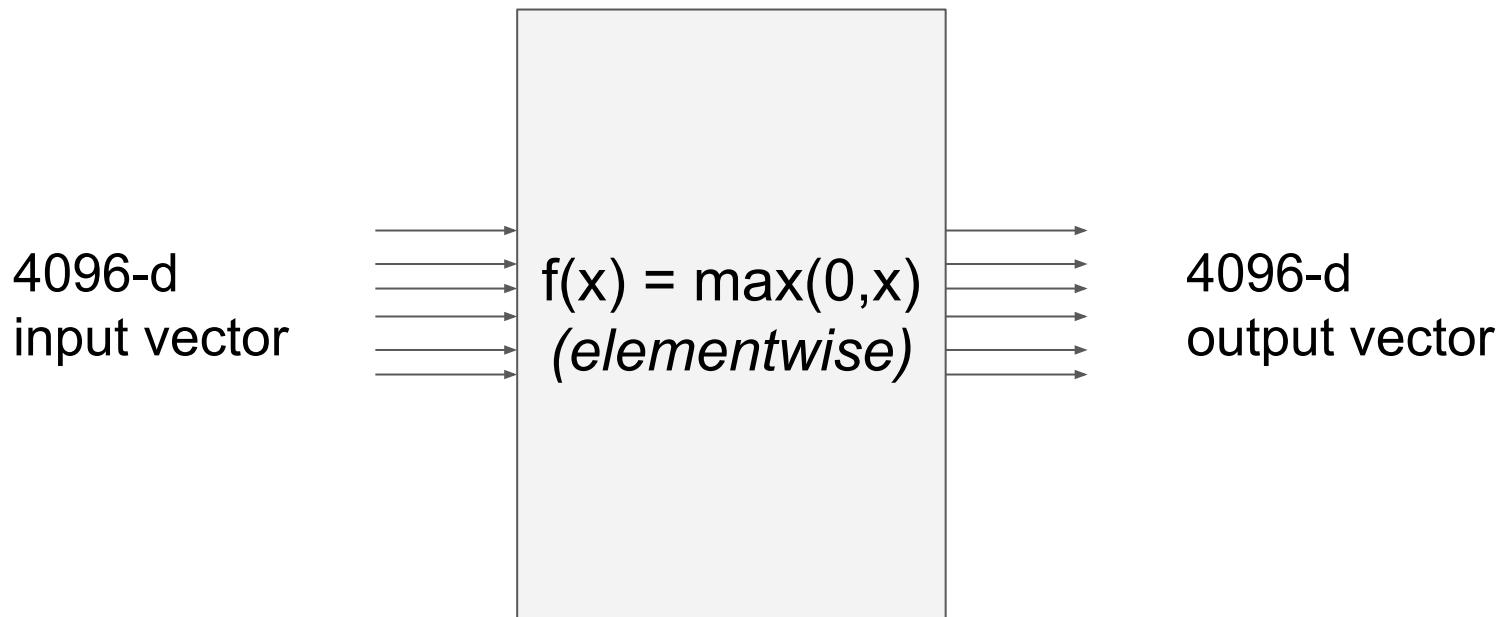
# Gradients for vectorized code

( $x, y, z$  are now vectors)

This is now the **Jacobian matrix** (derivative of each element of  $z$  w.r.t. each element of  $x$ )



# Vectorized operations

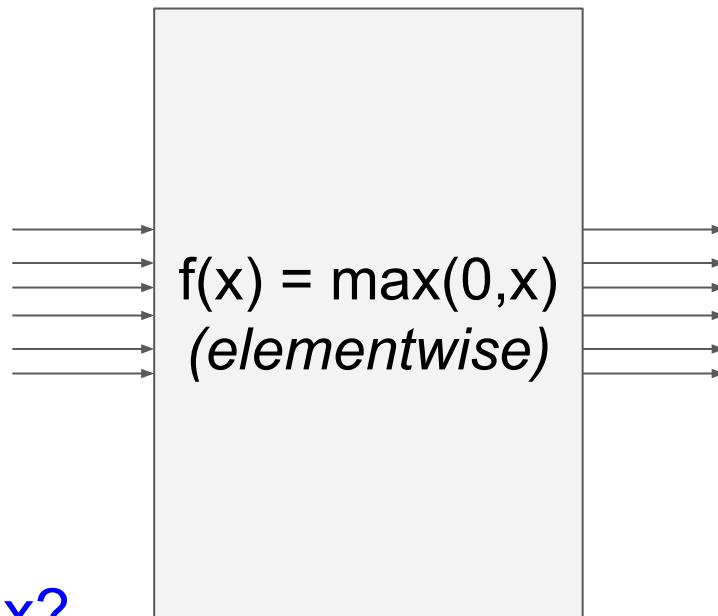


# Vectorized operations

$$\frac{\partial L}{\partial x} = \boxed{\frac{\partial f}{\partial x}} \frac{\partial L}{\partial f}$$

Jacobian matrix

4096-d  
input vector



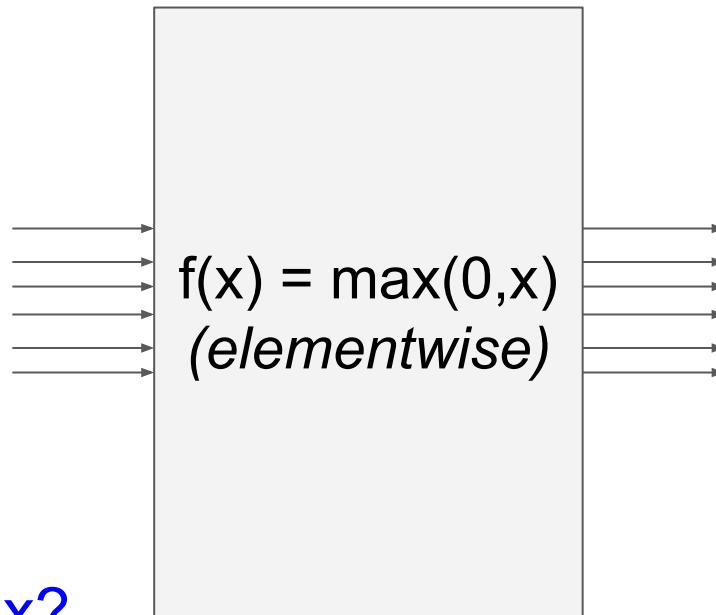
Q: what is the  
size of the  
Jacobian matrix?

# Vectorized operations

$$\frac{\partial L}{\partial x} = \boxed{\frac{\partial f}{\partial x}} \frac{\partial L}{\partial f}$$

Jacobian matrix

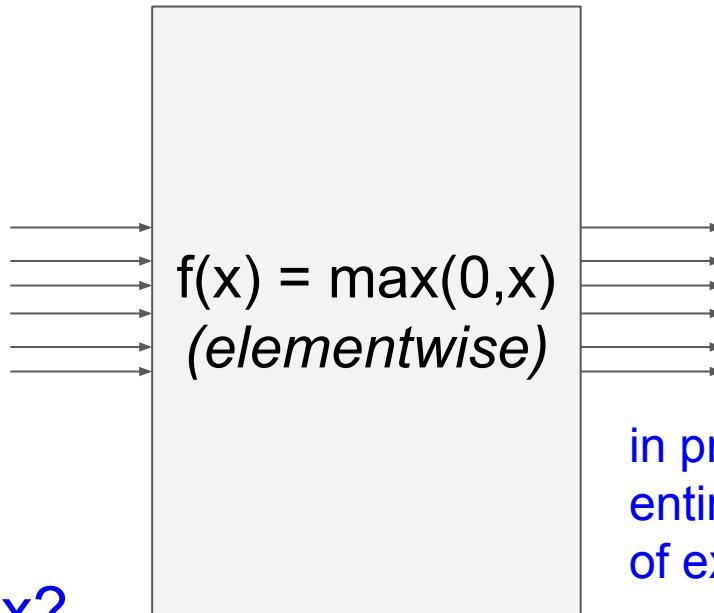
4096-d  
input vector



Q: what is the  
size of the  
Jacobian matrix?  
[4096 x 4096!]

# Vectorized operations

4096-d  
input vector



Q: what is the  
size of the  
Jacobian matrix?  
[4096 x 4096!]

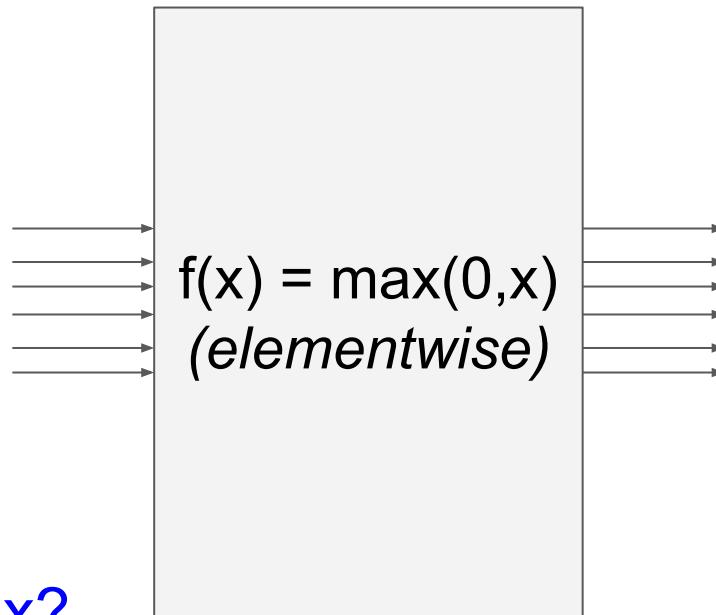
in practice we process an  
entire minibatch (e.g. 100)  
of examples at one time:  
i.e. Jacobian would technically be a  
[409,600 x 409,600] matrix :\  
\\

# Vectorized operations

$$\frac{\partial L}{\partial x} = \boxed{\frac{\partial f}{\partial x}} \frac{\partial L}{\partial f}$$

Jacobian matrix

4096-d  
input vector



Q: what is the  
size of the  
Jacobian matrix?  
[4096 x 4096!]

4096-d  
output vector

Q2: what does it  
look like?

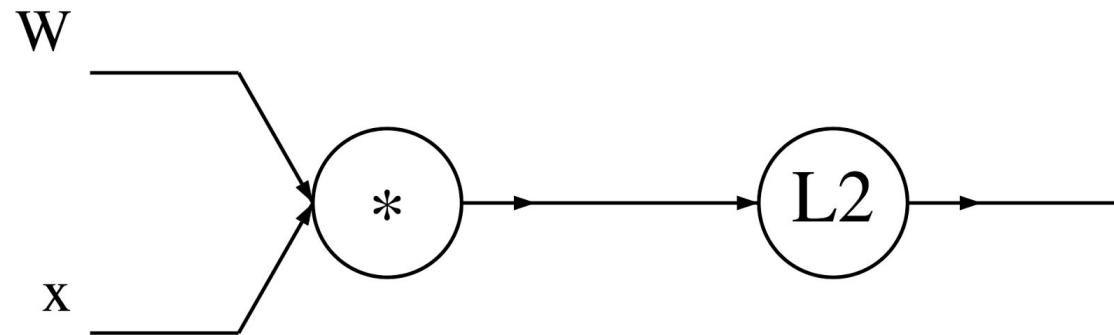
Identity matrix  
비슷하게 됨

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

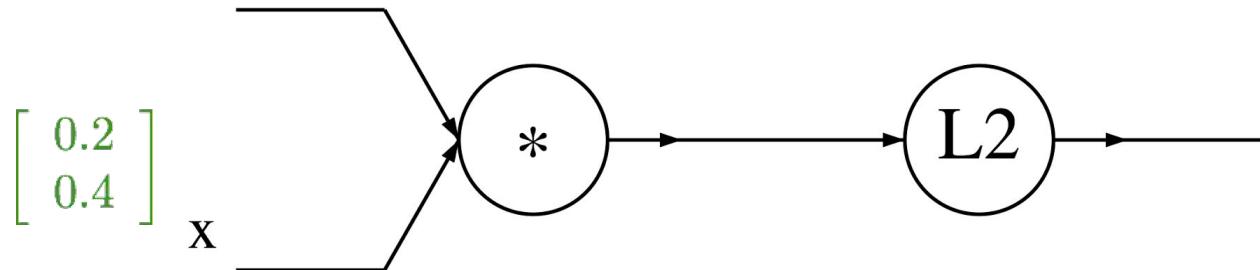
$\downarrow$        $\downarrow$   
 $\in \mathbb{R}^n$     $\in \mathbb{R}^{n \times n}$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$



A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix} W$$

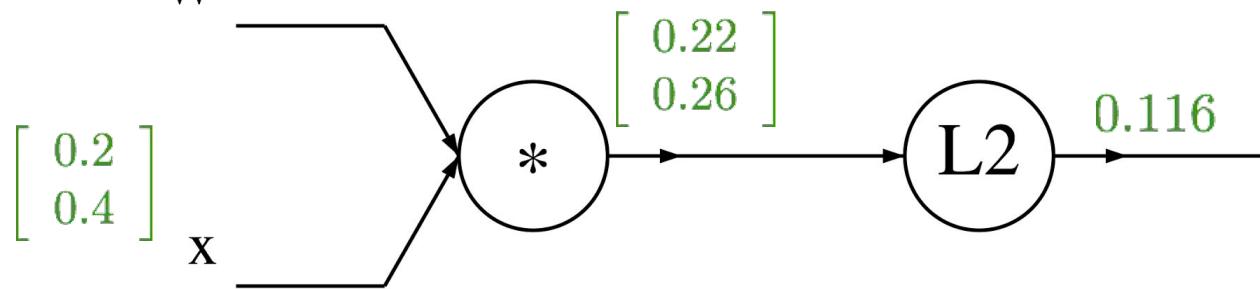


$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix} W$$

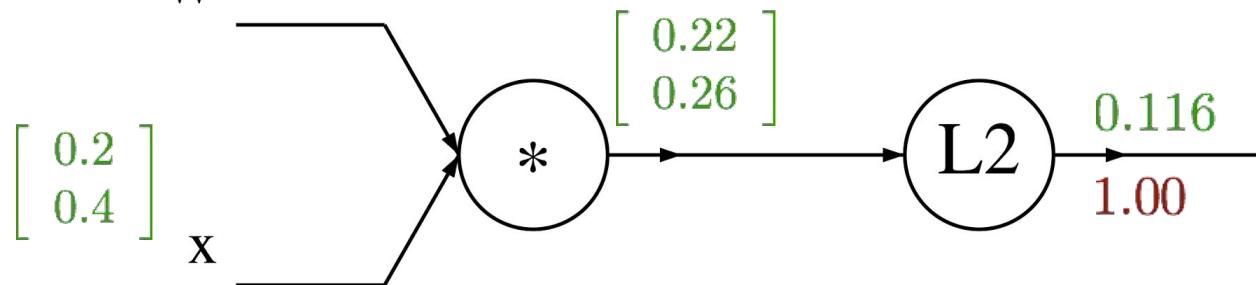


$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix} W$$

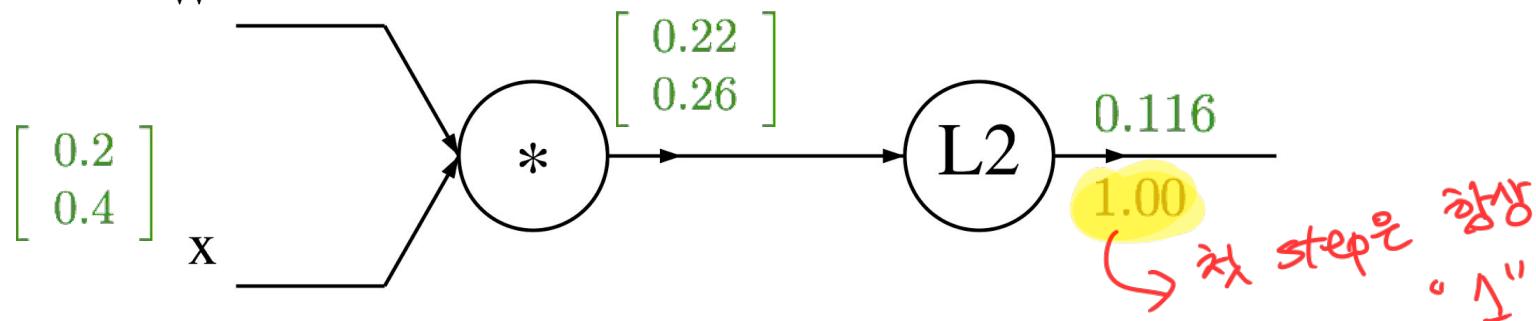


$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix} W$$



$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

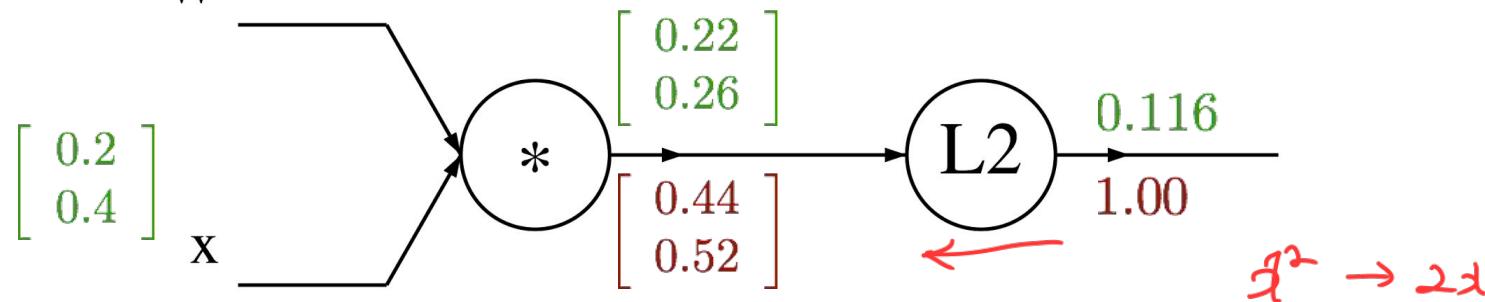
$$f(q) = ||q||^2 = q_1^2 + \cdots + q_n^2$$

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

$$\nabla_q f = 2q$$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix} W$$



$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \dots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \dots + W_{n,n}x_n \end{pmatrix}$$

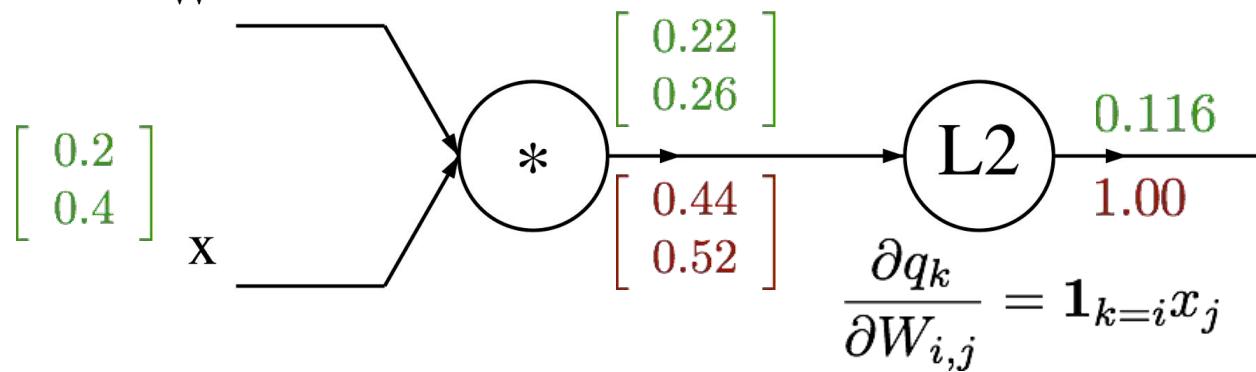
$$f(q) = ||q||^2 = q_1^2 + \dots + q_n^2$$

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

$$\nabla_q f = 2q$$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix} W$$



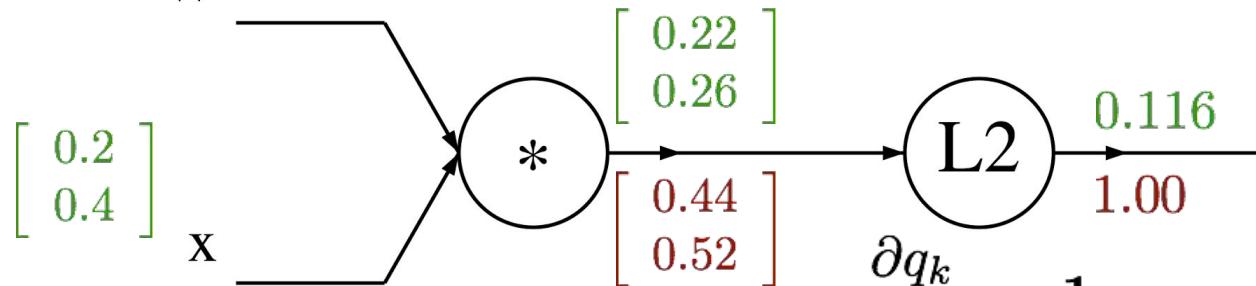
$$\frac{\partial q_k}{\partial W_{i,j}} = \mathbf{1}_{k=i} x_j$$

$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix} W$$



$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

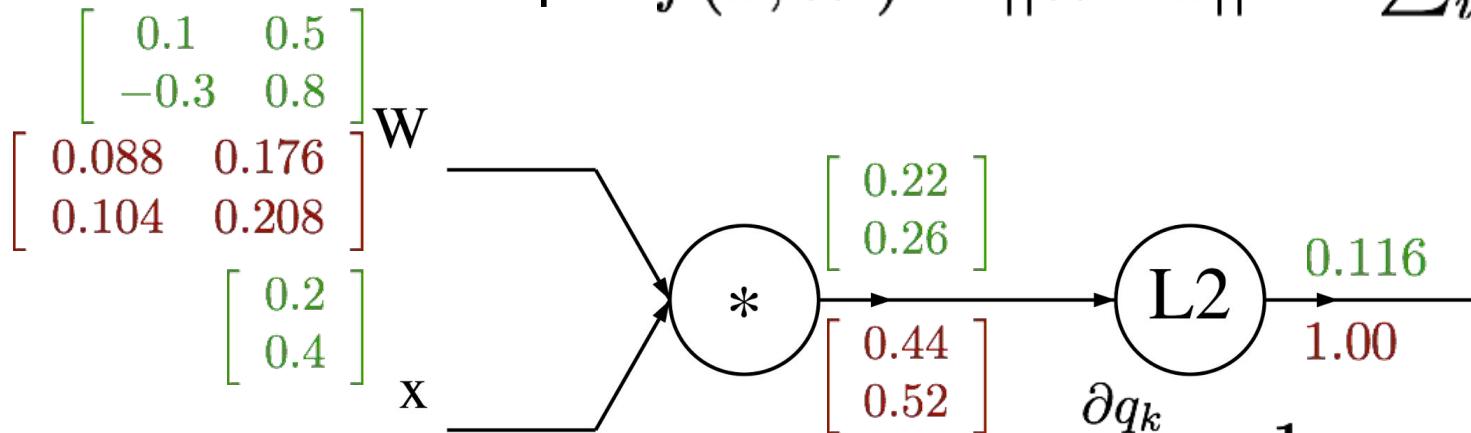
$$\frac{\partial q_k}{\partial W_{i,j}} = \mathbf{1}_{k=i} x_j$$

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

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

$$= 2q_i x_j$$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$



$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

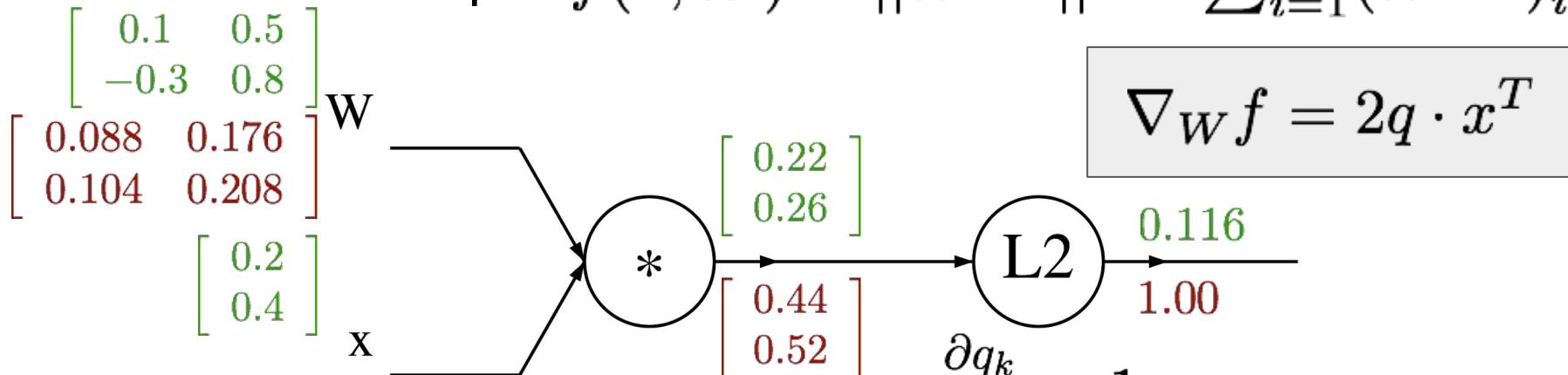
$$\frac{\partial q_k}{\partial W_{i,j}} = \mathbf{1}_{k=i} x_j$$

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

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

$$= 2q_i x_j$$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$



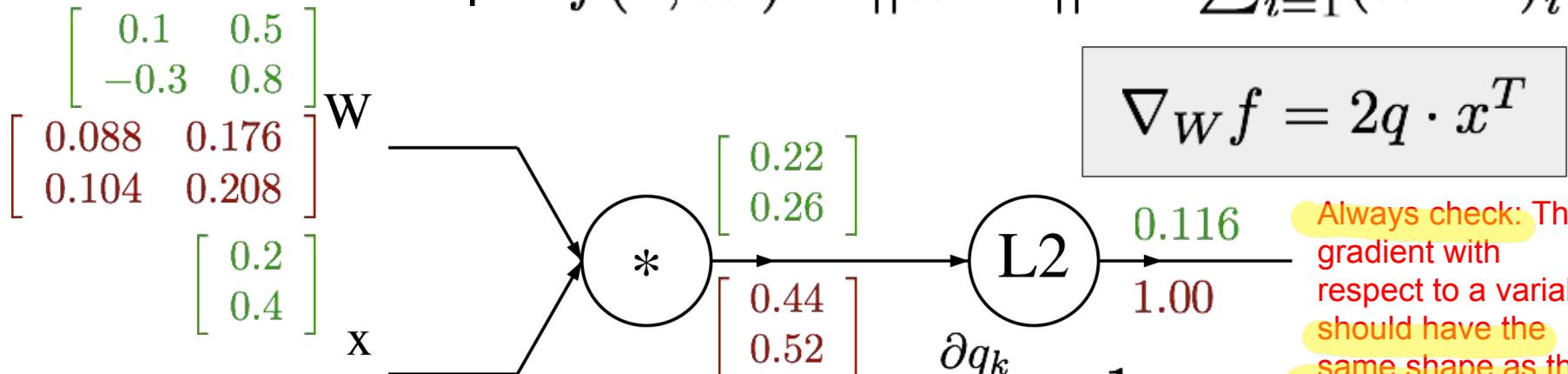
$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

$$\frac{\partial q_k}{\partial W_{i,j}} = \mathbf{1}_{k=i} x_j$$

$$\begin{aligned} \frac{\partial f}{\partial W_{i,j}} &= \sum_k \frac{\partial f}{\partial q_k} \frac{\partial q_k}{\partial W_{i,j}} \\ &= \sum_k (2q_k)(\mathbf{1}_{k=i} x_j) \\ &= 2q_i x_j \end{aligned}$$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$



Always check: The gradient with respect to a variable should have the same shape as the variable

$$\frac{\partial q_k}{\partial W_{i,j}} = \mathbf{1}_{k=i} x_j$$

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

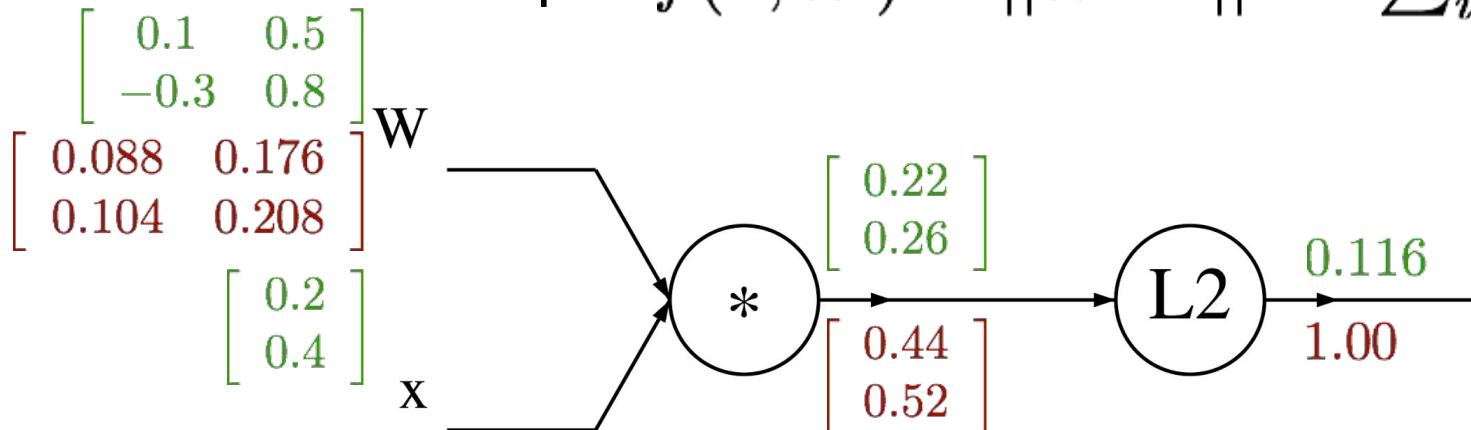
$$= \sum_k (2q_k)(\mathbf{1}_{k=i} x_j)$$

$$= 2q_i x_j$$

$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

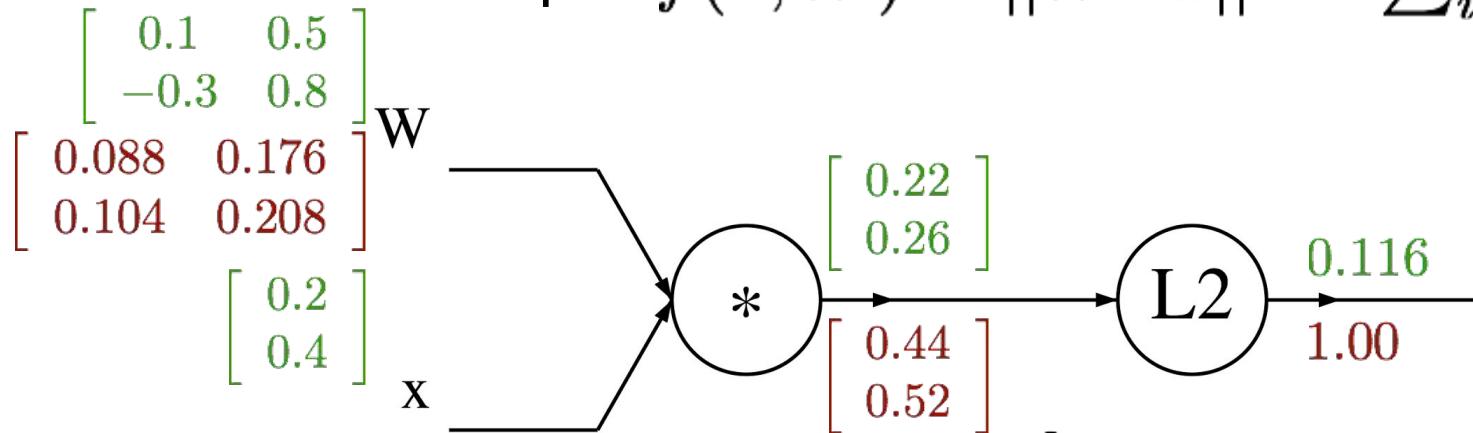


$$\frac{\partial q_k}{\partial x_i} = W_{k,i}$$

$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

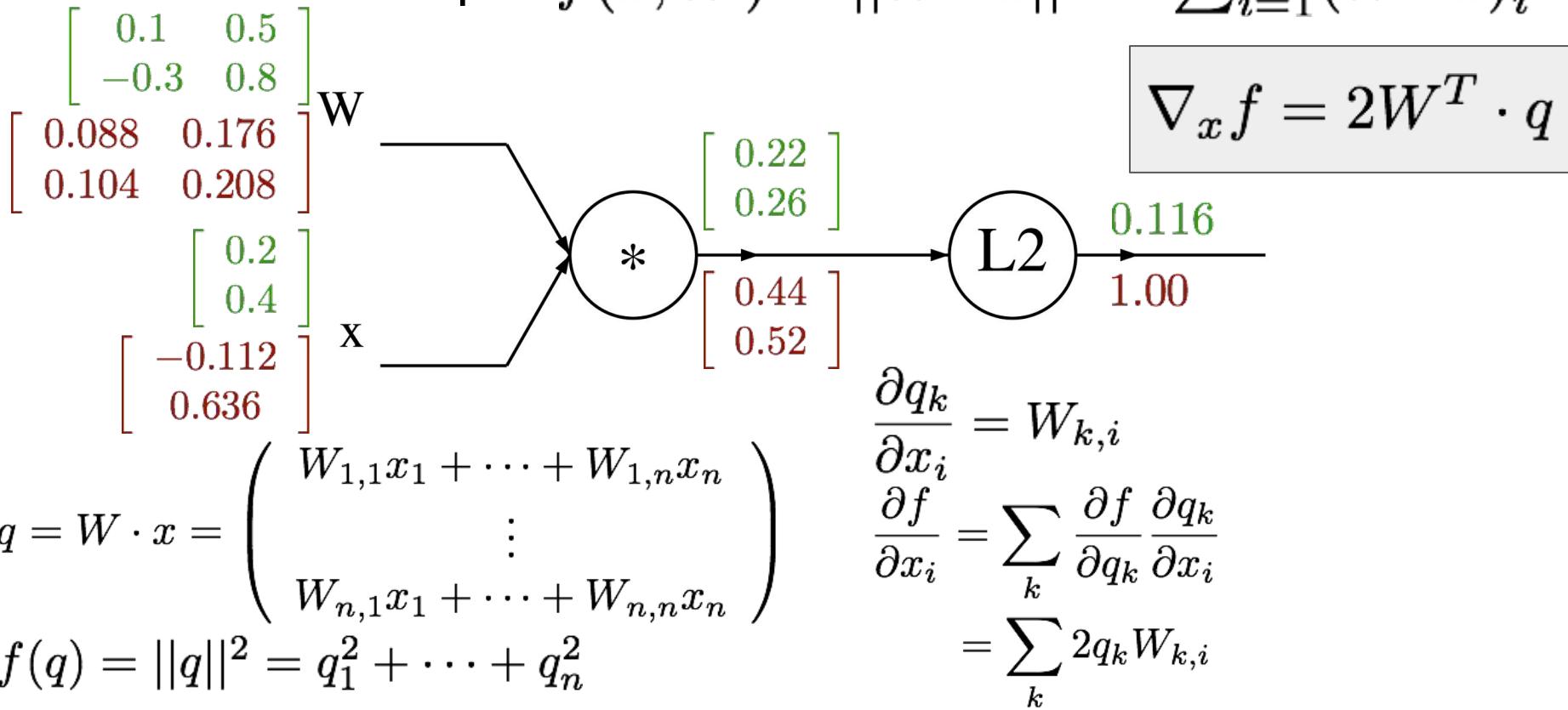


$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

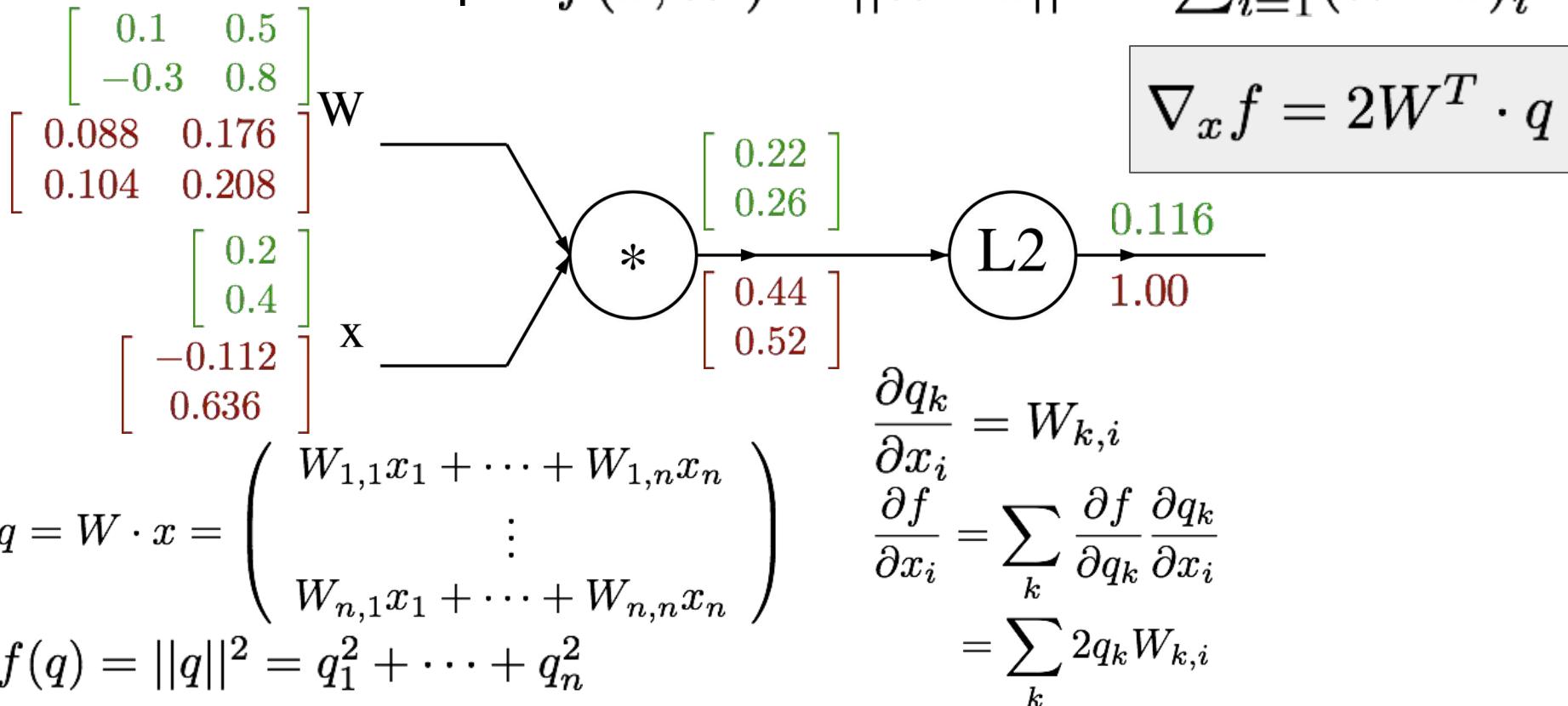
$$f(q) = ||q||^2 = q_1^2 + \cdots + q_n^2$$

$$\begin{aligned}\frac{\partial q_k}{\partial x_i} &= W_{k,i} \\ \frac{\partial f}{\partial x_i} &= \sum_k \frac{\partial f}{\partial q_k} \frac{\partial q_k}{\partial x_i} \\ &= \sum_k 2q_k W_{k,i}\end{aligned}$$

A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$

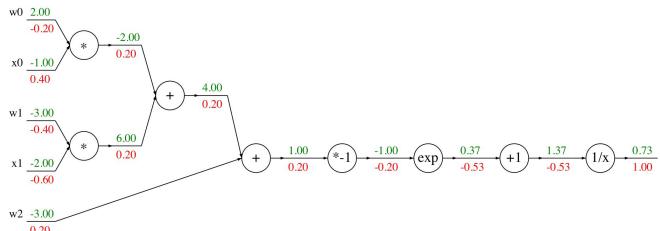


A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$



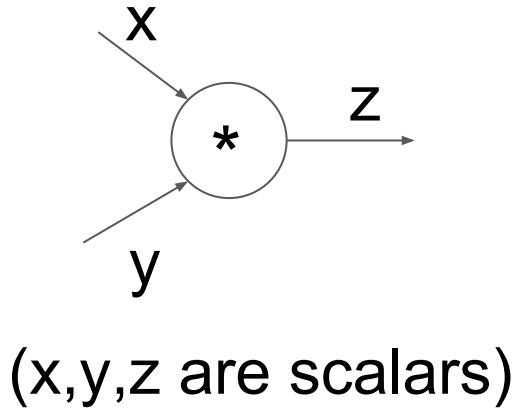
# Modularized implementation: forward / backward API

Graph (or Net) object (*rough psuedo code*)



```
class ComputationalGraph(object):  
    ...  
    def forward(self, inputs):  
        # 1. [pass inputs to input gates...]  
        # 2. forward the computational graph:  
        for gate in self.graph.nodes_topologically_sorted():  
            gate.forward()  
        return loss # the final gate in the graph outputs the loss  
    def backward(self):  
        for gate in reversed(self.graph.nodes_topologically_sorted()):  
            gate.backward() # little piece of backprop (chain rule applied)  
        return inputs_gradients
```

# Modularized implementation: forward / backward API

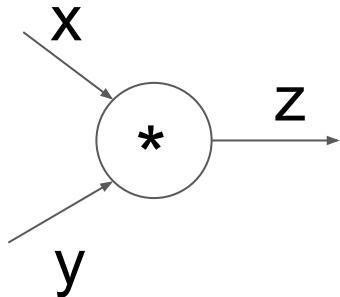


```
class MultiplyGate(object):  
    def forward(x,y):  
        z = x*y  
        return z  
    def backward(dz):  
        # dx = ... #todo  
        # dy = ... #todo  
        return [dx, dy]
```

$$\frac{\partial L}{\partial z}$$

$$\frac{\partial L}{\partial x}$$

# Modularized implementation: forward / backward API



$(x,y,z$  are scalars)

```
class MultiplyGate(object):

    def forward(x,y):
        z = x*y
        self.x = x # must keep these around!
        self.y = y
        return z

    def backward(dz):
        dx = self.y * dz # [dz/dx * dL/dz]
        dy = self.x * dz # [dz/dy * dL/dz]
        return [dx, dy]
```

# Example: Caffe layers

Branch: master / <a href="#">cafe</a> / <a href="#">src</a> / <a href="#">cafe</a> / <a href="#">layers</a> /		
		<a href="#">Create new file</a> <a href="#">Upload files</a> <a href="#">Find file</a> <a href="#">History</a>
 shelhamer committed on GitHub Merge pull request #4630 from BiGene/load_hdf5_fix	Latest commit e687a71 21 days ago	
..		
 <a href="#">absval_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">absval_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">accuracy_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">argmax_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">base_conv_layer.cpp</a>	enable dilated deconvolution	a year ago
 <a href="#">base_data_layer.cpp</a>	Using default from proto for prefetch	3 months ago
 <a href="#">base_data_layer.cu</a>	Switched multi-GPU to NCCL	3 months ago
 <a href="#">batch_norm_layer.cpp</a>	Add missing spaces besides equal signs in batch_norm_layer.cpp	4 months ago
 <a href="#">batch_norm_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">batch_reindex_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">batch_reindex_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">bias_layer.cpp</a>	Remove incorrect cast of gemm int arg to Dtype in BiasLayer	a year ago
 <a href="#">bias_layer.cu</a>	Separation and generalization of ChannelwiseAffineLayer into BiasLayer	a year ago
 <a href="#">bnll_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">bnll_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">concat_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">concat_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">contrastive_loss_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">contrastive_loss_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">conv_layer.cpp</a>	add support for 2D dilated convolution	a year ago
 <a href="#">conv_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">crop_layer.cpp</a>	remove redundant operations in Crop layer (#5138)	2 months ago
 <a href="#">crop_layer.cu</a>	remove redundant operations in Crop layer (#5138)	2 months ago
 <a href="#">cudnn_conv_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_conv_layer.cu</a>	Add cuDNN v5 support, drop cuDNN v3 support	11 months ago

 <a href="#">cudnn_lcn_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_lcn_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_lrn_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_lrn_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_pooling_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_pooling_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_relu_layer.cpp</a>	Add cuDNN v5 support, drop cuDNN v3 support	11 months ago
 <a href="#">cudnn_relu_layer.cu</a>	Add cuDNN v5 support, drop cuDNN v3 support	11 months ago
 <a href="#">cudnn_sigmoid_layer.cpp</a>	Add cuDNN v5 support, drop cuDNN v3 support	11 months ago
 <a href="#">cudnn_sigmoid_layer.cu</a>	Add cuDNN v5 support, drop cuDNN v3 support	11 months ago
 <a href="#">cudnn_softmax_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_softmax_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">cudnn_tanh_layer.cpp</a>	Add cuDNN v5 support, drop cuDNN v3 support	11 months ago
 <a href="#">cudnn_tanh_layer.cu</a>	Add cuDNN v5 support, drop cuDNN v3 support	11 months ago
 <a href="#">data_layer.cpp</a>	Switched multi-GPU to NCCL	3 months ago
 <a href="#">deconv_layer.cpp</a>	enable dilated deconvolution	a year ago
 <a href="#">deconv_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">dropout_layer.cpp</a>	supporting N-D Blobs in Dropout layer Reshape	a year ago
 <a href="#">dropout_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">dummy_data_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">eltwise_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">eltwise_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">elu_layer.cpp</a>	ELU layer with basic tests	a year ago
 <a href="#">elu_layer.cu</a>	ELU layer with basic tests	a year ago
 <a href="#">embed_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">embed_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">euclidean_loss_layer.cpp</a>	dismantle layer headers	a year ago
 <a href="#">euclidean_loss_layer.cu</a>	dismantle layer headers	a year ago
 <a href="#">exp_layer.cpp</a>	Solving issue with exp layer with base e	a year ago
 <a href="#">exp_layer.cu</a>	dismantle layer headers	a year ago

blob = tensor  
diff = gradient

Caffe is licensed under [BSD 2-Clause](#)

# Caffe Sigmoid Layer

```
1 #include <cmath>
2 #include <vector>
3
4 #include "caffe/layers/sigmoid_layer.hpp"
5
6 namespace caffe {
7
8     template <typename Dtype>
9     inline Dtype sigmoid(Dtype x) {
10         return 1. / (1. + exp(-x));
11     }
12
13     template <typename Dtype>
14     void SigmoidLayer<Dtype>::Forward_cpu(const vector<Blob<Dtype>>& bottom,
15                                             const vector<Blob<Dtype>>& top) {
16         const Dtype* bottom_data = bottom[0]->cpu_data();
17         Dtype* top_data = top[0]->mutable_cpu_data();
18         const int count = bottom[0]->count();
19         for (int i = 0; i < count; ++i) {
20             top_data[i] = sigmoid(bottom_data[i]);
21         }
22     }
23
24     template <typename Dtype>
25     void SigmoidLayer<Dtype>::Backward_cpu(const vector<Blob<Dtype>>& top,
26                                              const vector<bool>& propagate_down,
27                                              const vector<Blob<Dtype>>& bottom) {
28         if (propagate_down[0]) {
29             const Dtype* top_data = top[0]->cpu_data();
30             const Dtype* top_diff = top[0]->cpu_diff();
31             Dtype* bottom_diff = bottom[0]->mutable_cpu_diff();
32             const int count = bottom[0]->count();
33             for (int i = 0; i < count; ++i) {
34                 const Dtype sigmoid_x = top_data[i];
35                 bottom_diff[i] = top_diff[i] * sigmoid_x * (1. - sigmoid_x);
36             }
37         }
38     }
39
40 #ifdef CPU_ONLY
41 STUB_GPU(SigmoidLayer);
42#endif
43
44 INSTANTIATE_CLASS(SigmoidLayer);
45
46
47 } // namespace caffe
```

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

forward pass

$$(1 - \sigma(x)) \sigma(x)$$

backward pass

\* top\_diff (chain rule)

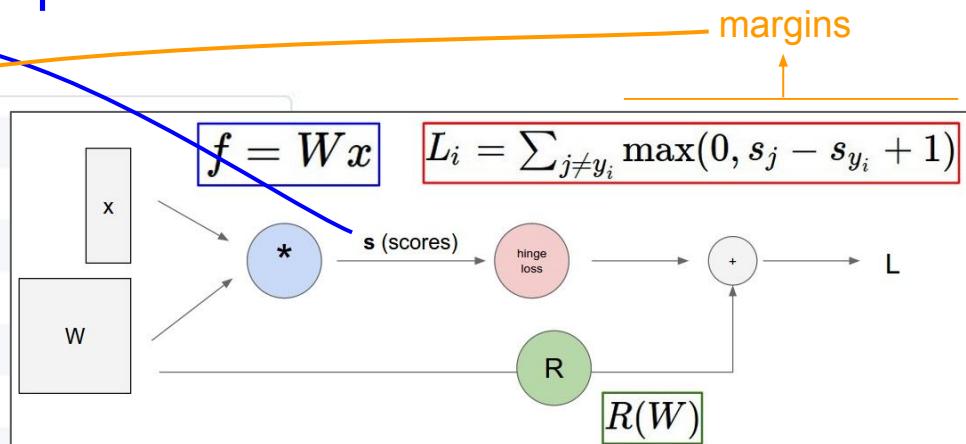
Caffe is licensed under [BSD 2-Clause](#)

# In Assignment 1: Writing SVM / Softmax

Stage your forward/backward computation!

E.g. for the SVM:

```
# receive W (weights), X (data)
# forward pass (we have 6 lines)
scores = #...
margins = #... ←
data_loss = #...
reg_loss = #...
loss = data_loss + reg_loss
# backward pass (we have 5 lines)
dmargins = # ... (optionally, we go direct to dscores)
dscores = #...
dW = #...
```



# Summary so far...

- neural nets will be very large: impractical to write down gradient formula by hand for all parameters
- **backpropagation** = recursive application of the chain rule along a computational graph to compute the gradients of all inputs/parameters/intermediates
- implementations maintain a graph structure, where the nodes implement the **forward()** / **backward()** API
- **forward**: compute result of an operation and save any intermediates needed for gradient computation in memory
- **backward**: apply the chain rule to compute the gradient of the loss function with respect to the inputs

# Next: Neural Networks

# Neural networks: without the brain stuff

**(Before)** Linear score function:  $f = Wx$

# Neural networks: without the brain stuff

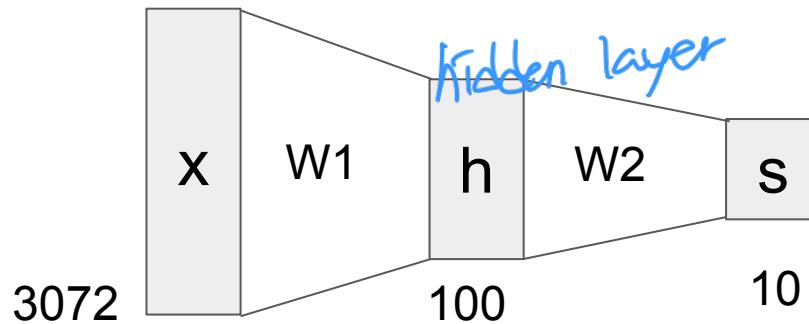
(Before) Linear score function:  $f = Wx$

(Now) 2-layer Neural Network  $f = W_2 \underbrace{\max(0, W_1 x)}_{\text{Ex) ReLU func}}$

# Neural networks: without the brain stuff

(Before) Linear score function:  $f = Wx$

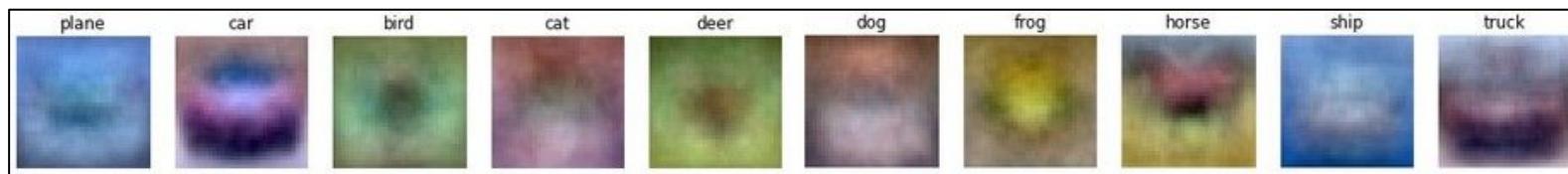
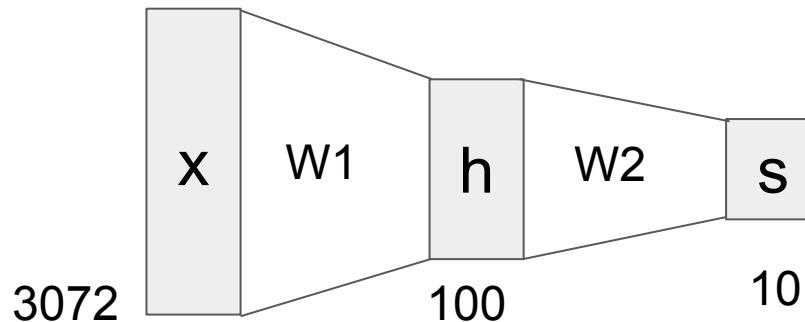
(Now) 2-layer Neural Network  $f = W_2 \max(0, W_1 x)$



# Neural networks: without the brain stuff

(Before) Linear score function:  $f = Wx$

(Now) 2-layer Neural Network  $f = W_2 \max(0, W_1 x)$



# Neural networks: without the brain stuff

- (Before) Linear score function:  $f = Wx$
- (Now) 2-layer Neural Network  $f = W_2 \max(0, W_1 x)$   
or 3-layer Neural Network  
                                   $f = W_3 \max(0, W_2 \max(0, W_1 x))$

# Full implementation of training a 2-layer Neural Network needs ~20 lines:

```
1 import numpy as np
2 from numpy.random import randn
3
4 N, D_in, H, D_out = 64, 1000, 100, 10
5 x, y = randn(N, D_in), randn(N, D_out)
6 w1, w2 = randn(D_in, H), randn(H, D_out)
7
8 for t in range(2000):
9     h = 1 / (1 + np.exp(-x.dot(w1)))
10    y_pred = h.dot(w2)
11    loss = np.square(y_pred - y).sum()
12    print(t, loss)
13
14 grad_y_pred = 2.0 * (y_pred - y)
15 grad_w2 = h.T.dot(grad_y_pred)
16 grad_h = grad_y_pred.dot(w2.T)
17 grad_w1 = x.T.dot(grad_h * h * (1 - h))
18
19 w1 -= 1e-4 * grad_w1
20 w2 -= 1e-4 * grad_w2
```

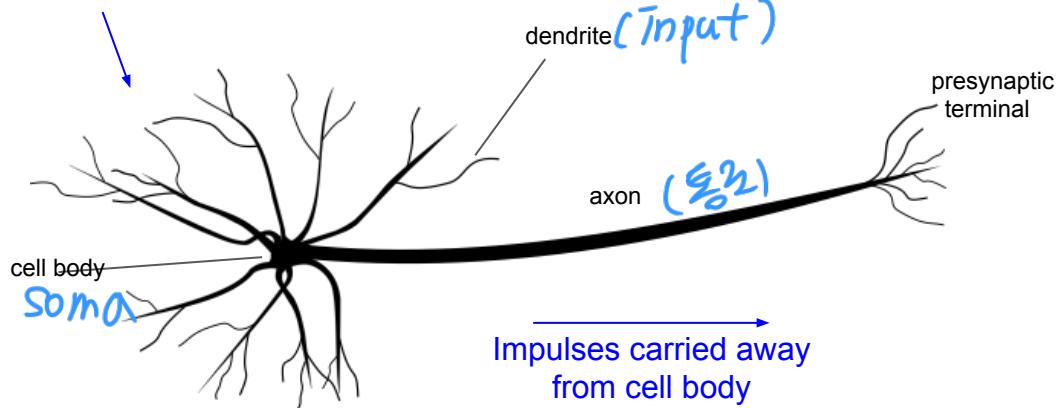
# In Assignment 2: Writing a 2-layer net

```
# receive W1,W2,b1,b2 (weights/biases), X (data)
# forward pass:
h1 = #... function of X,W1,b1
scores = #... function of h1,W2,b2
loss = #... (several lines of code to evaluate Softmax loss)
# backward pass:
dscores = #...
dh1,dW2,db2 = #...
dW1,db1 = #...
```



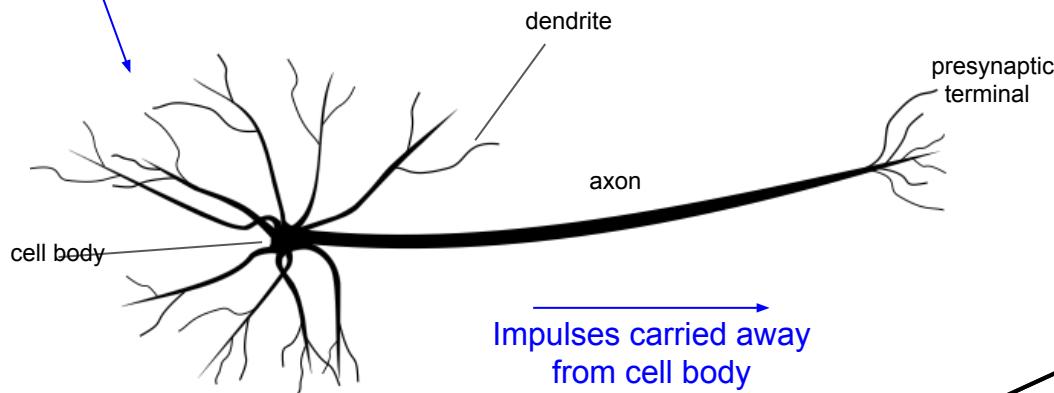
This image by [Fotis Bobolas](#) is  
licensed under [CC-BY 2.0](#)

Impulses carried toward cell body



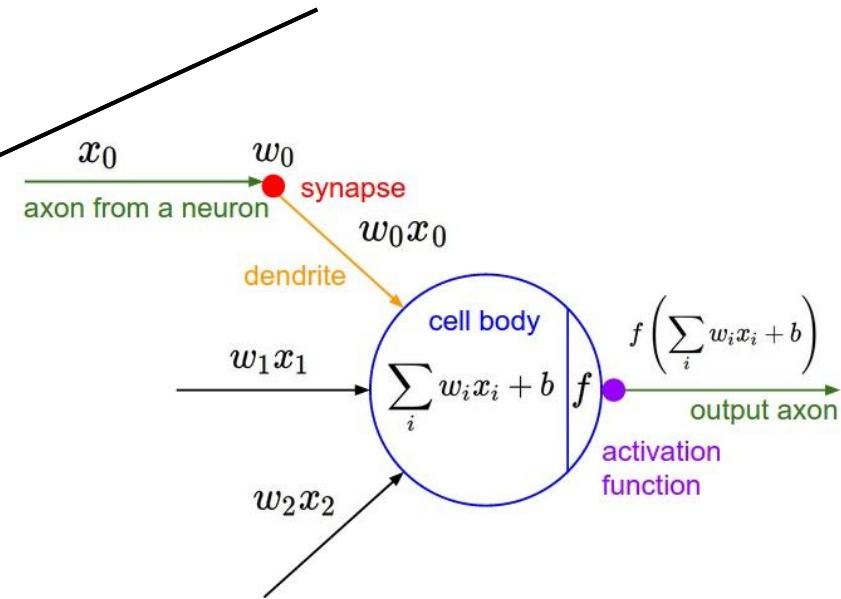
This image by Felipe Perucho  
is licensed under CC-BY 3.0

Impulses carried toward cell body

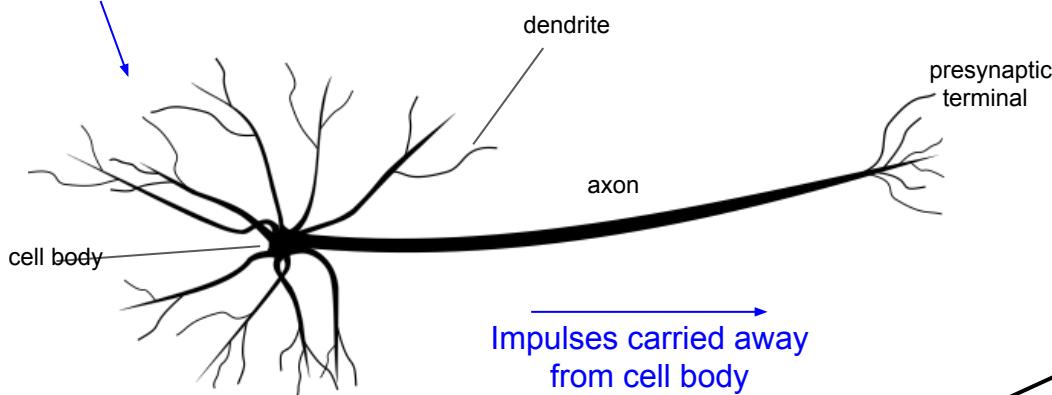


This image by Felipe Perucho  
is licensed under CC-BY 3.0

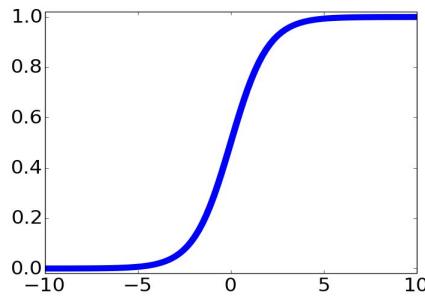
Impulses carried away  
from cell body



Impulses carried toward cell body



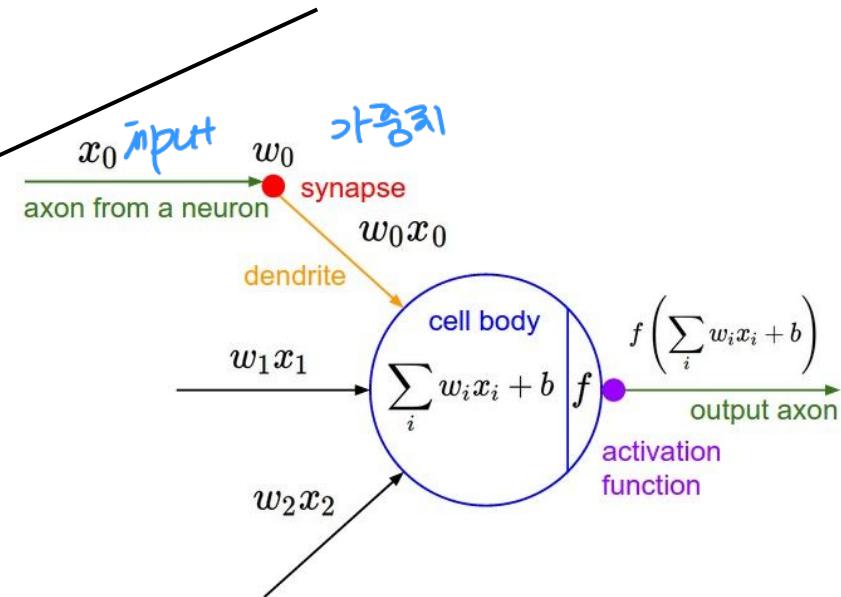
This image by Felipe Perucho  
is licensed under CC-BY 3.0



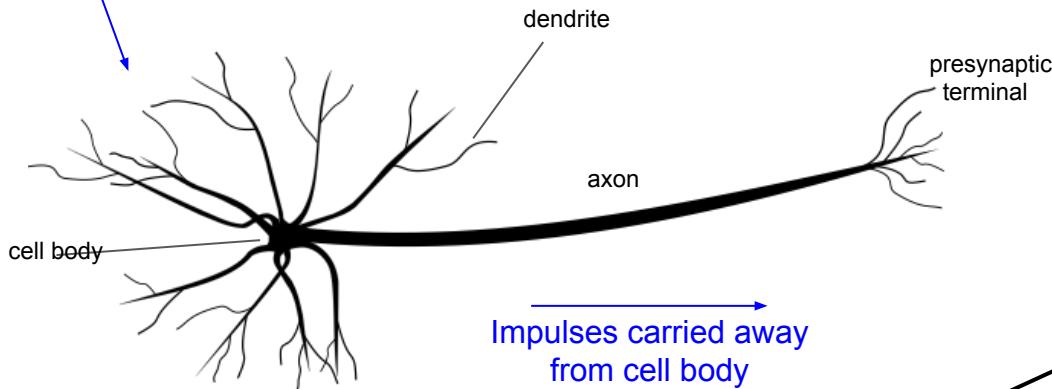
sigmoid activation function

$$\frac{1}{1 + e^{-x}}$$

Impulses carried away from cell body

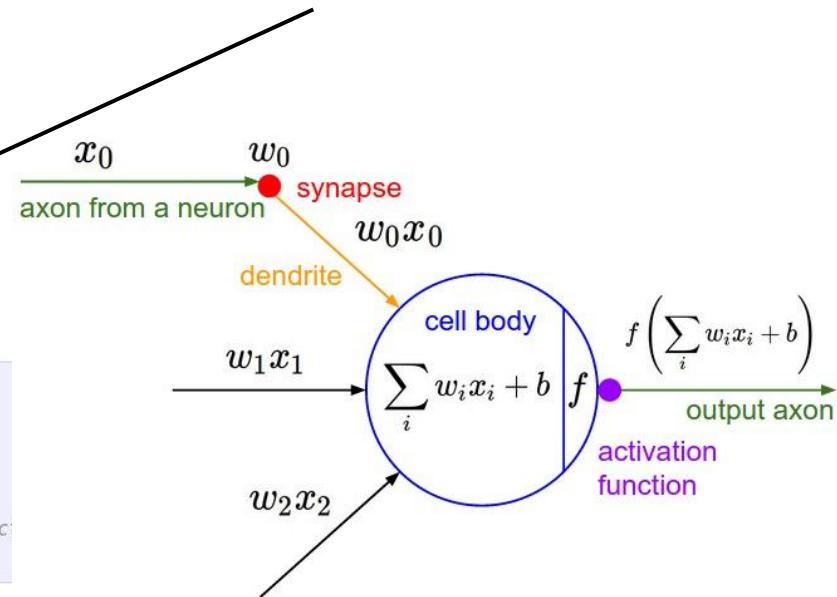


Impulses carried toward cell body



This image by Felipe Perucco  
is licensed under CC-BY 3.0

```
class Neuron:  
    ...  
    def neuron_tick(inputs):  
        """ assume inputs and weights are 1-D numpy arrays and bias is a number """  
        cell_body_sum = np.sum(inputs * self.weights) + self.bias  
        firing_rate = 1.0 / (1.0 + math.exp(-cell_body_sum)) # sigmoid activation function  
        return firing_rate
```



Be very careful with your brain analogies!

뉴런 생명망은 뇌장 완전히 닮을 건 X

### Biological Neurons:

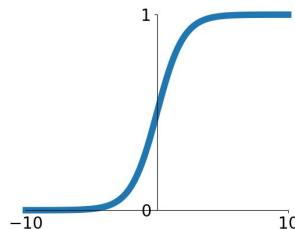
- Many different types
- Dendrites can perform complex non-linear computations
- Synapses are not a single weight but a complex non-linear dynamical system
- Rate code may not be adequate

[Dendritic Computation. London and Häusser]

# Activation functions

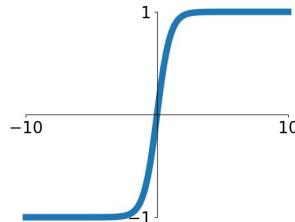
## Sigmoid

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



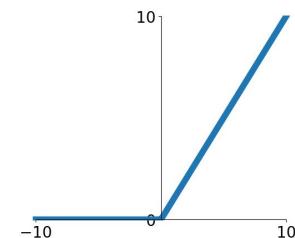
## tanh

$$\tanh(x)$$



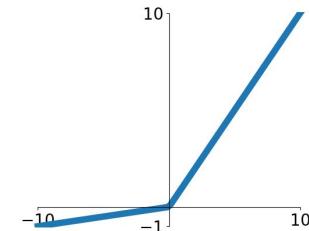
## ReLU

$$\max(0, x)$$



## Leaky ReLU

$$\max(0.1x, x)$$

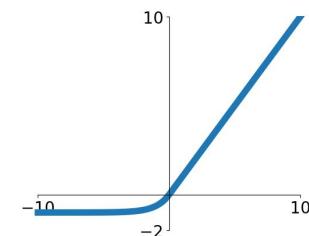


## Maxout

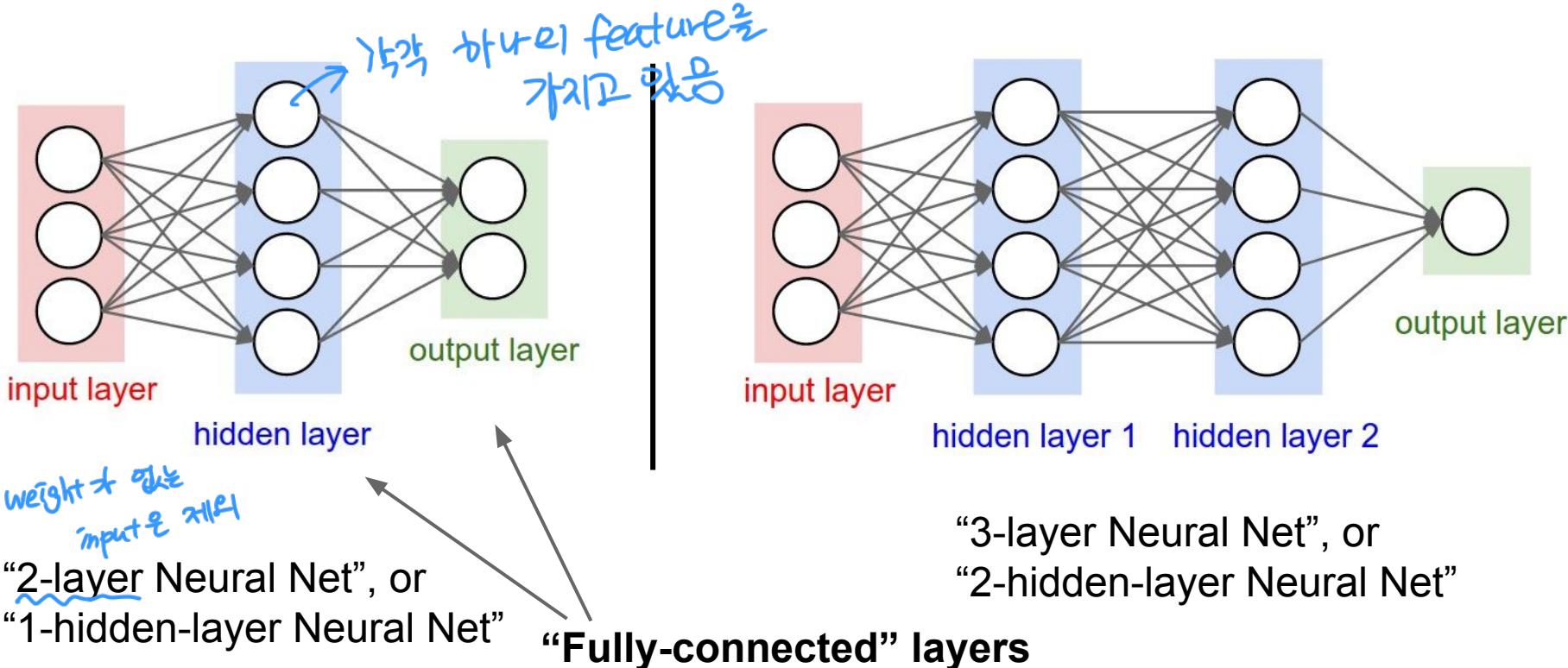
$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

## ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



# Neural networks: Architectures

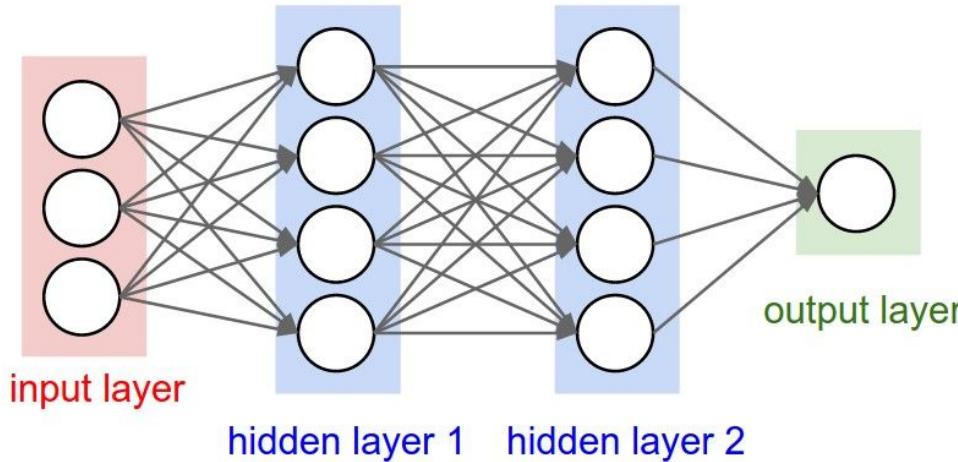


# Example feed-forward computation of a neural network

```
class Neuron:  
    # ...  
    def neuron_tick(inputs):  
        """ assume inputs and weights are 1-D numpy arrays and bias is a number """  
        cell_body_sum = np.sum(inputs * self.weights) + self.bias  
        firing_rate = 1.0 / (1.0 + math.exp(-cell_body_sum)) # sigmoid activation function  
        return firing_rate
```

We can efficiently evaluate an entire layer of neurons.

# Example feed-forward computation of a neural network



more neurons  
=  
more capacity  
+  
regularization

```
# forward-pass of a 3-layer neural network:  
f = lambda x: 1.0/(1.0 + np.exp(-x)) # activation function (use sigmoid)  
x = np.random.randn(3, 1) # random input vector of three numbers (3x1)  
h1 = f(np.dot(W1, x) + b1) # calculate first hidden layer activations (4x1)  
h2 = f(np.dot(W2, h1) + b2) # calculate second hidden layer activations (4x1)  
out = np.dot(W3, h2) + b3 # output neuron (1x1)
```

# Summary

- We arrange neurons into fully-connected layers
- The abstraction of a **layer** has the nice property that it allows us to use efficient vectorized code (e.g. matrix multiplies)
- Neural networks are not really *neural*
- Next time: Convolutional Neural Networks