Lecture4. Backpropagation & Neural Network

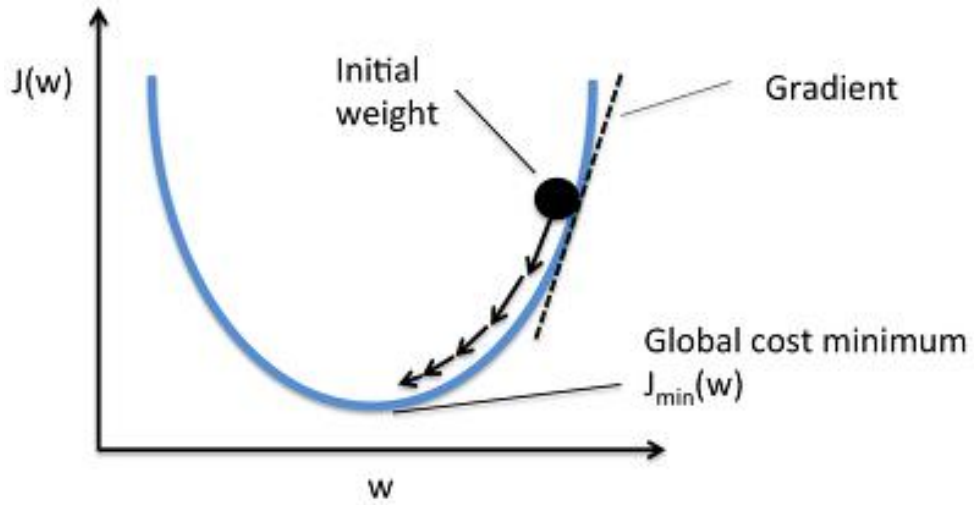
Kuggle Deep Learning Study week1

김효은

INDEX

INDEX

1. backpropagation(scalar)



$$W := W - \alpha \frac{\partial}{\partial W} cost(W)$$

잠시 review,

1. 우리는 gradient(편미분) 통해 weight가 업데이트 될 방향을 정하기로 했고,
2. 이 아이디어를 활용한 알고리즘이 gradient descent라는 것을 알고있다.

-> 이 아이디어를 실제 사용하기 위해서는 gradient를 알아야 한다! == 미분을 해야 한다!

1. backpropagation(scalar)

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

Numerical gradient: slow :(, approximate :(, easy to write :)
Analytic gradient: fast :), exact :), error-prone :(

바로 전 PPT에서,

Numerical gradient를 구하는 과정은 너무 느리기 때문에

Analytic gradient로 미분값을 구하기로 했다.

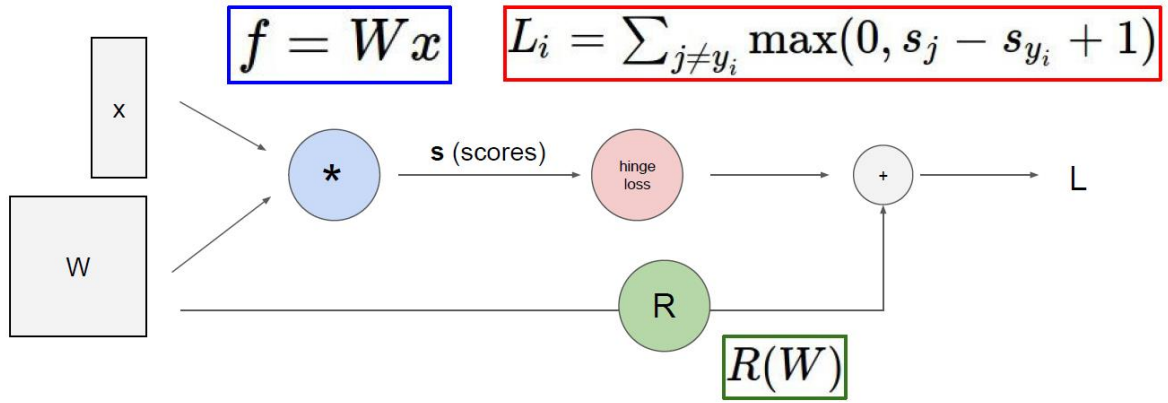
-> 어떻게 하면 쉽게 구할 수 있을까? -> backpropagation

Backpropagation에 대해 배우기 전, 우리가 알아야 할 사전지식

1. Computational Graph (계산그래프)
2. 합성함수의 미분

1. backpropagation(scalar)

본격적인 backpropagation에 대한 설명을 진행하기에 앞서, Computational graph에 대해서 설명을 하겠습니다.



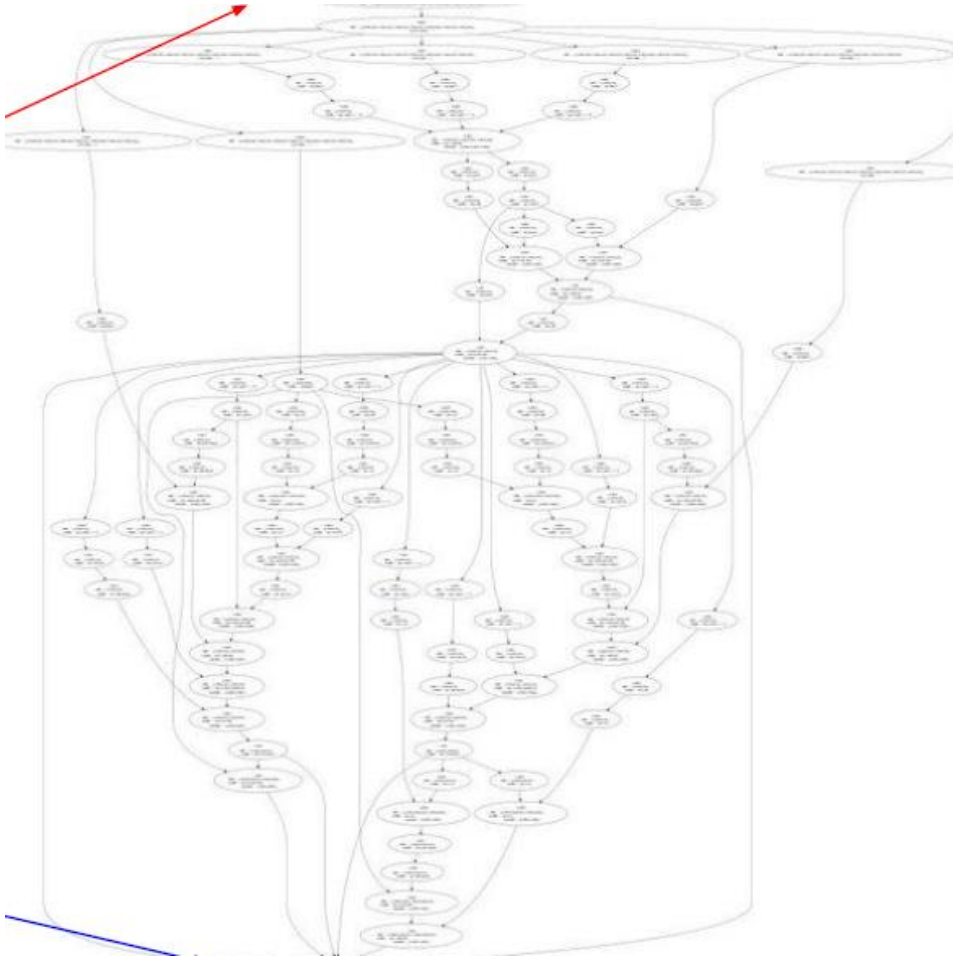
Computational graph는 계산과정을 그래프로 나타낸 것으로,

Node와 edge로 구성됩니다.

Node는 연산을 의미하고, Edge는 node를 연결하는 것을 의미하는 데, 여기서는 데이터가 흘러가는 것을 의미합니다.

이 계산그래프는 국소적 계산을 통해 결과를 얻는데요, 이 특징이 매우 중요해요! 이 특징이 미분을 간단하게 해줍니다.

1. backpropagation(scalar)



Neural network의 계산그래프

-> 국소적으로 바라봄

1. backpropagation(scalar)

Backpropagation에 대해 배우기 전, 우리가 알아야 할 사전지식

1. Computational Graph (계산그래프)
2. 합성함수의 미분

1. backpropagation(scalar)

$$F = g(h(x))$$

$$F' = g'(h(x)) * h'(x)$$

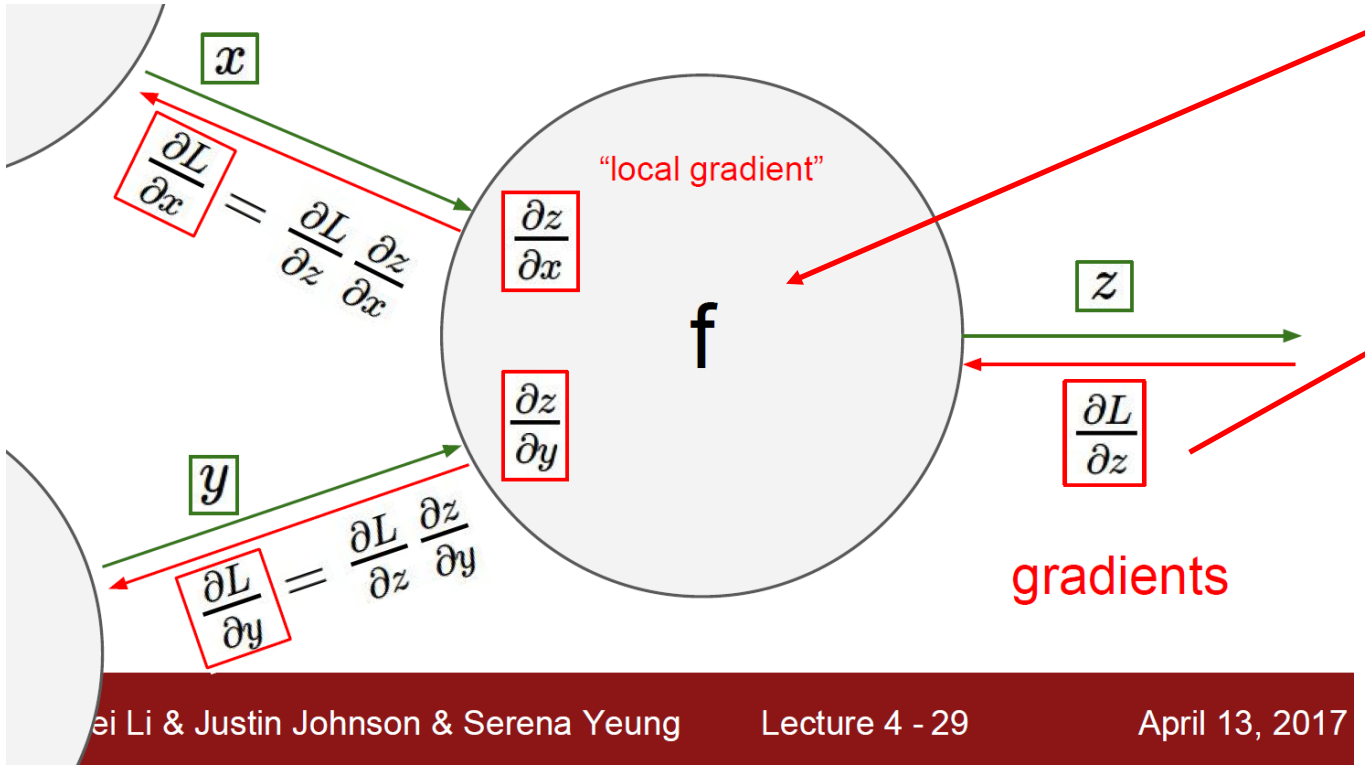
$$f(g(h(x)))$$

$$\frac{d}{dx}[f(w)] = \frac{df}{dw}\frac{dw}{dx} = \frac{df}{dw}\frac{dw}{dy}\frac{dy}{dx} = \frac{df}{dw}(w)\frac{dw}{dy}(y)\frac{dy}{dx}(x)$$

$$= \frac{df}{dw}(g(y))\frac{dw}{dy}(h(x))\frac{dy}{dx}(x) = \frac{df}{dw}(g(h(x)))\frac{dw}{dy}(h(x))\frac{dy}{dx}(x)$$

$$= f'(g(h(x)))g'(h(x))h'(x)$$

Backpropagtion 시작



Loss 계산하기 위해 적용한 함수 중 일부

$$= f'(g(h(x)))g'(h(x))h'(x)$$

L = a((…(f(x, y))…))

z = f(x, y)

1. backpropagation(scalar)

Backpropagation: a simple example

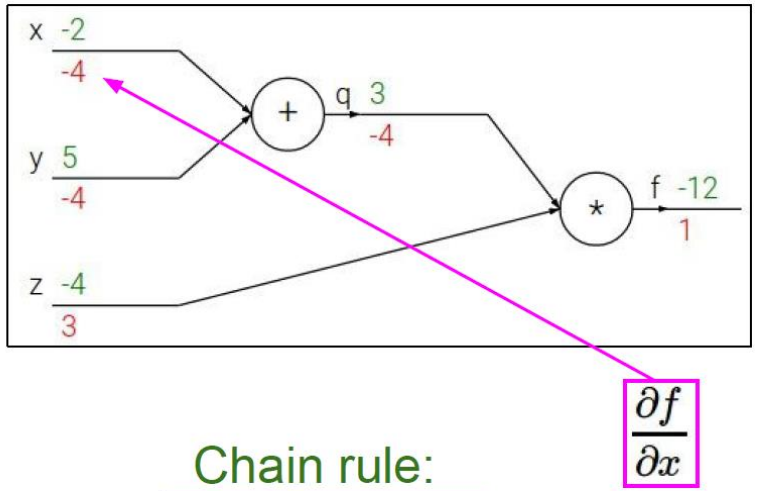
$$f(x, y, z) = (x + y)z$$

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

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

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

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

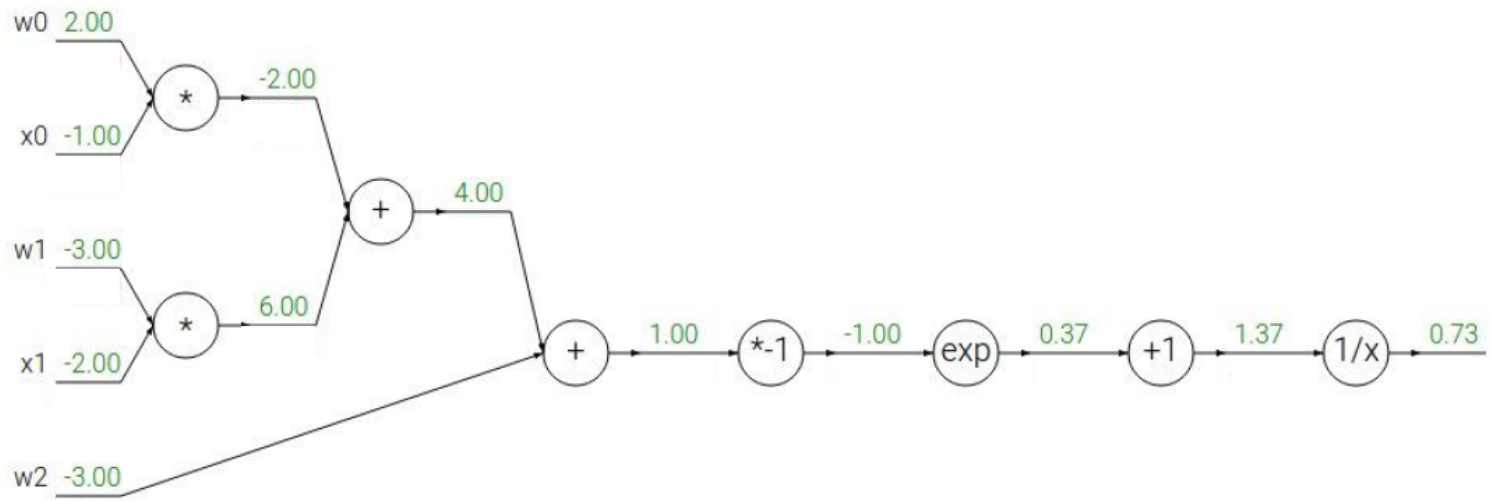


Chain rule:

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

이거는 비교적 쉬우니까 넘어갈게요 ☺

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



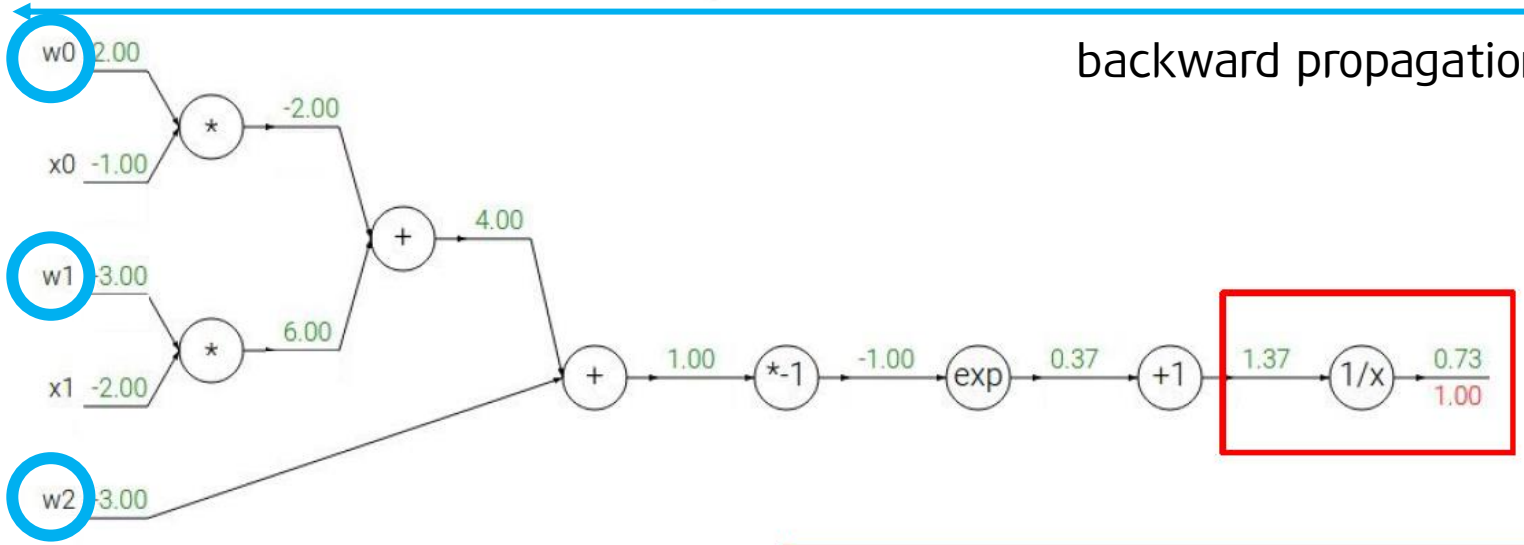
Forwad propagation

1. backpropagation(scalar)

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

backward propagation

우리가 gradient descent에 의해 업데이트 하고 싶은 parameter



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

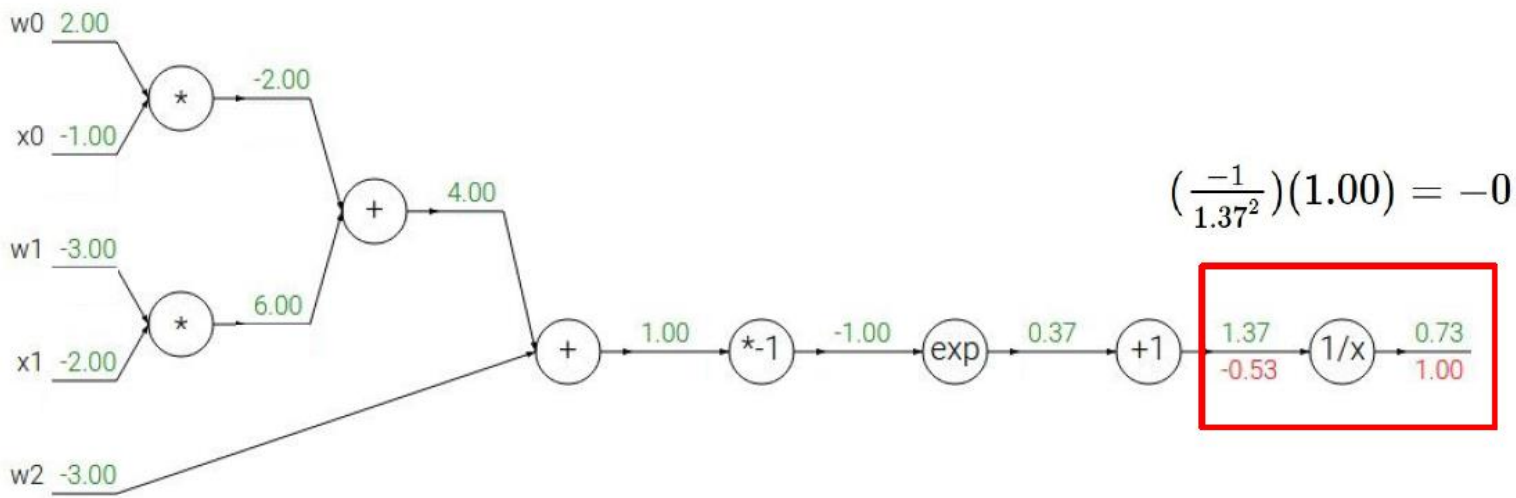
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

1. backpropagation(scalar)

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



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

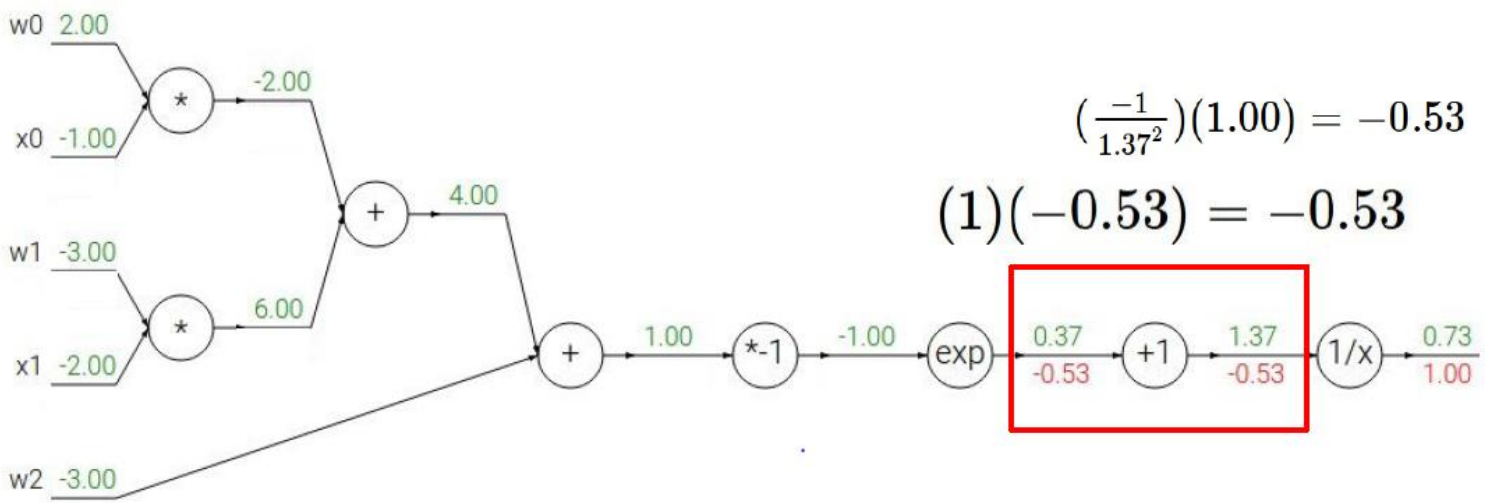
$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

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



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

$$(1)(-0.53) = -0.53$$

$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

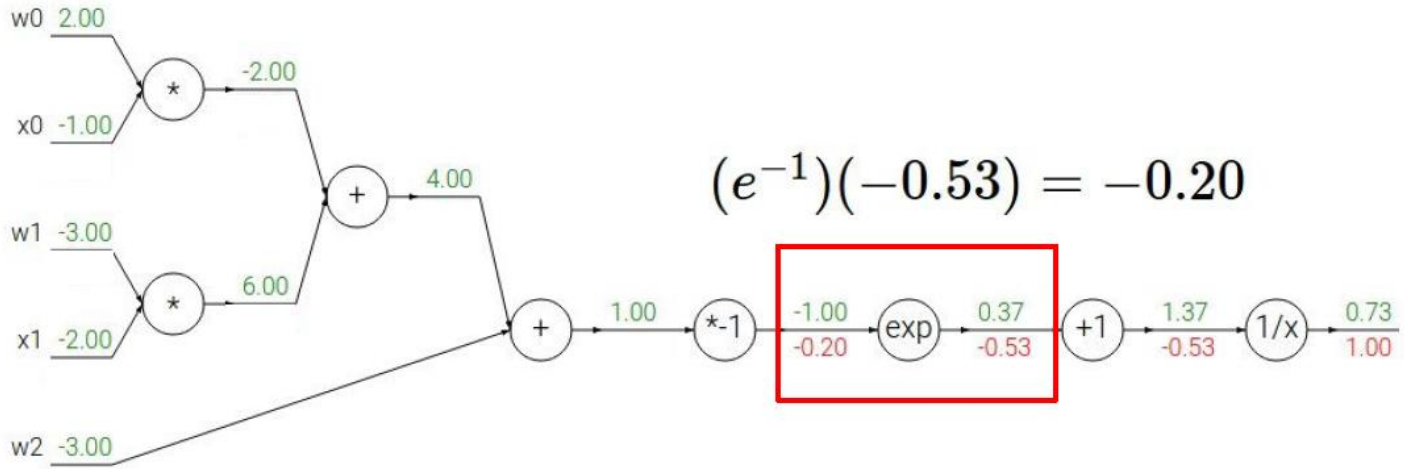
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

1. backpropagation(scalar)

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

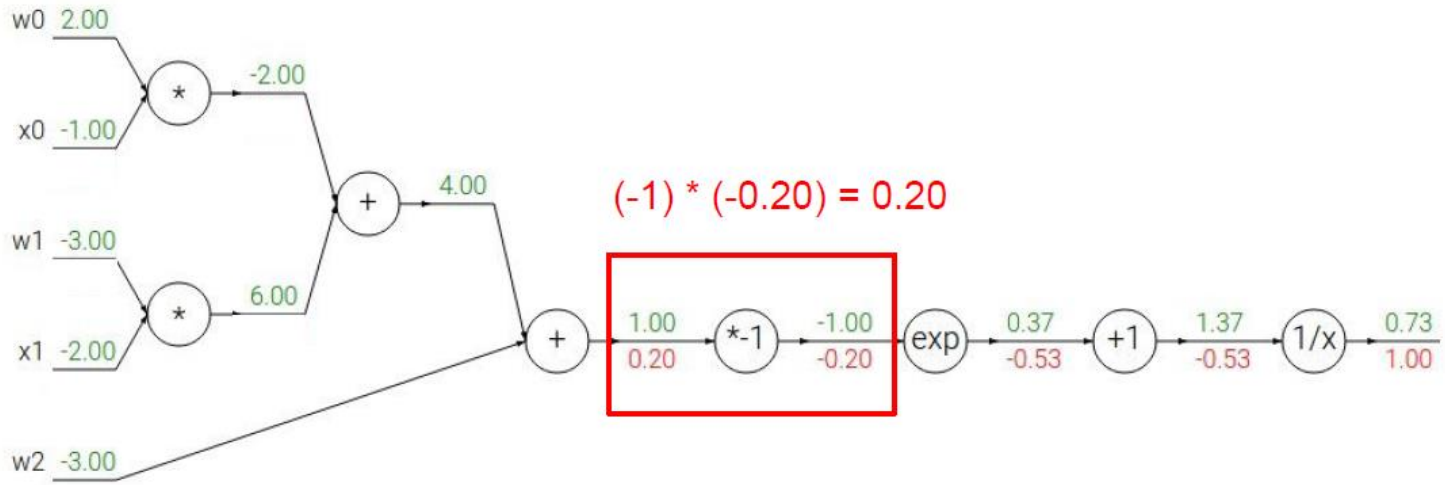


$$(e^{-1})(-0.53) = -0.20$$

$f(x) = e^x$	$\rightarrow$	$\frac{df}{dx} = e^x$
$f_a(x) = ax$	$\rightarrow$	$\frac{df}{dx} = a$

$f(x) = \frac{1}{x}$	$\rightarrow$	$\frac{df}{dx} = -1/x^2$
$f_c(x) = c + x$	$\rightarrow$	$\frac{df}{dx} = 1$

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



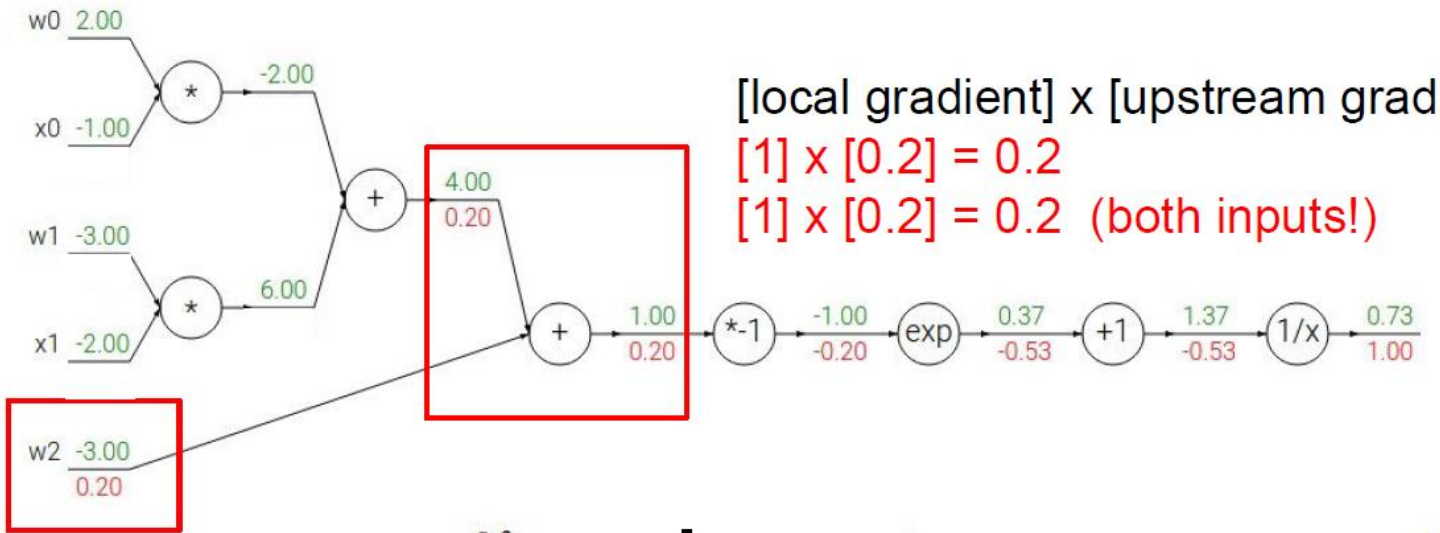
$f(x) = e^x \rightarrow \frac{df}{dx} = e^x$

$f_a(x) = ax \rightarrow \frac{df}{dx} = a$

$f(x) = \frac{1}{x} \rightarrow \frac{df}{dx} = -1/x^2$

$f_c(x) = c + x \rightarrow \frac{df}{dx} = 1$

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



[local gradient] x [upstream gradient]

[1] x [0.2] = 0.2

[1] x [0.2] = 0.2 (both inputs!)

$f(x) = e^x \rightarrow \frac{df}{dx} = e^x$

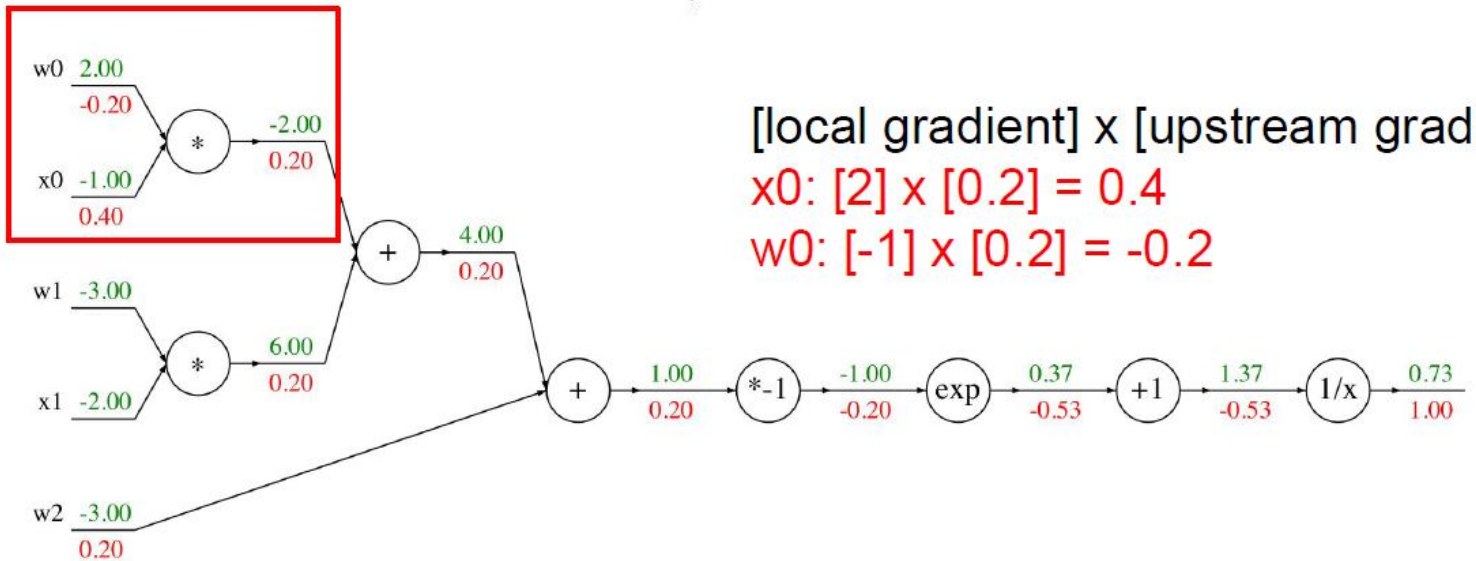
$f_a(x) = ax \rightarrow \frac{df}{dx} = a$

$f(x) = \frac{1}{x} \rightarrow \frac{df}{dx} = -1/x^2$

$f_c(x) = c + x \rightarrow \frac{df}{dx} = 1$

1. backpropagation(scalar)

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



[local gradient] x [upstream gradient]

x0: [2] x [0.2] = 0.4

w0: [-1] x [0.2] = -0.2

$f(x) = e^x \rightarrow \frac{df}{dx} = e^x$

$f_a(x) = ax \rightarrow \frac{df}{dx} = a$

$f(x) = \frac{1}{x} \rightarrow \frac{df}{dx} = -1/x^2$

$f_c(x) = c + x \rightarrow \frac{df}{dx} = 1$

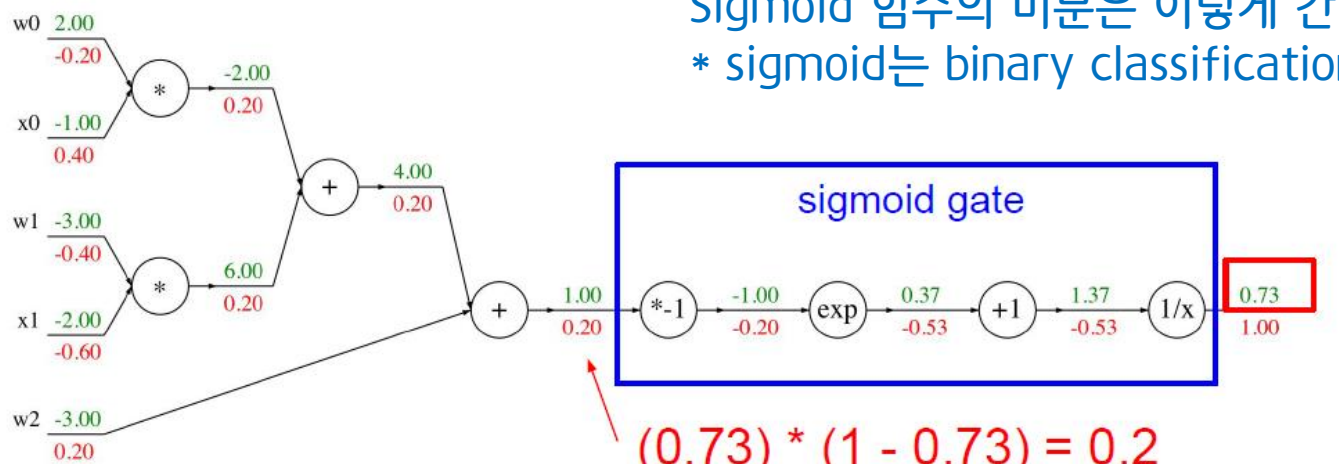
1. backpropagation(scalar)

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

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

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

Sigmoid 함수의 미분은 이렇게 간단하게도 가능!
* sigmoid는 binary classification 문제에서 마지막 함수로 사용

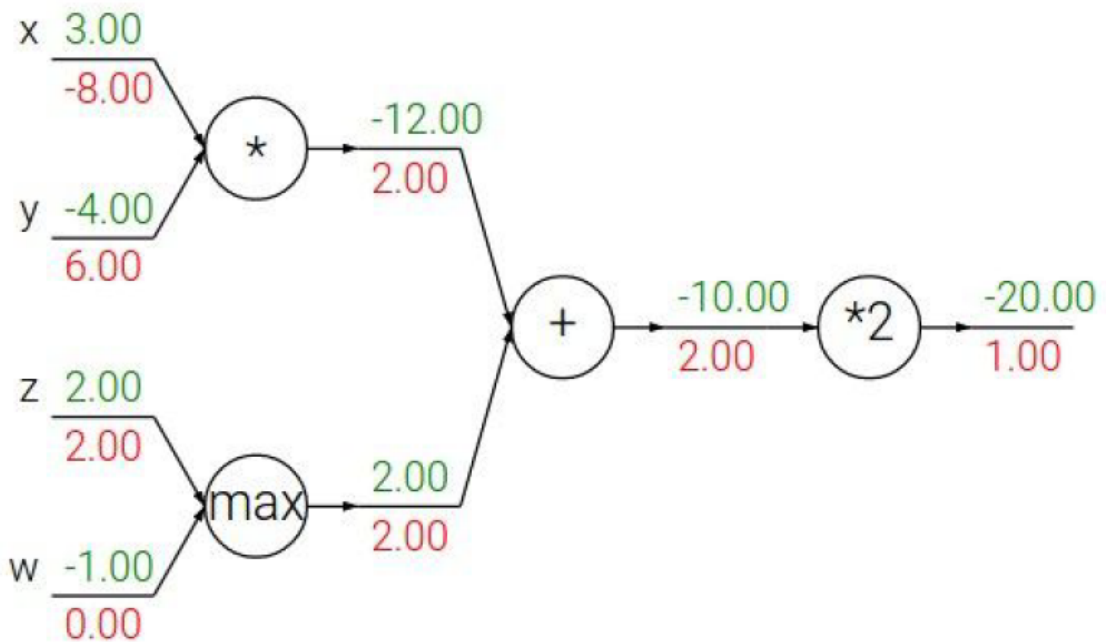


Patterns in backward flow

add gate: gradient distributor

max gate: gradient router

mul gate: gradient switcher



INDEX

2. backpropagation(vector)

함수의 유형

1. Univariate scalar function -> gradient
2. Univariate vector function -> jacobian
3. Multivariate scalar function -> gradient
4. Multivariate vector function -> jacobian

$$f(x) = 3x + 4$$

$$\nabla f(x) = 3$$

$$f(x) = \begin{bmatrix} 2x+3 \\ 3x^2+4x \end{bmatrix}$$

$$\nabla f(x) = \begin{bmatrix} 2 & 6x \end{bmatrix}$$

$$f(x_1, x_2) = 3x_1^2 + 4x_1x_2$$

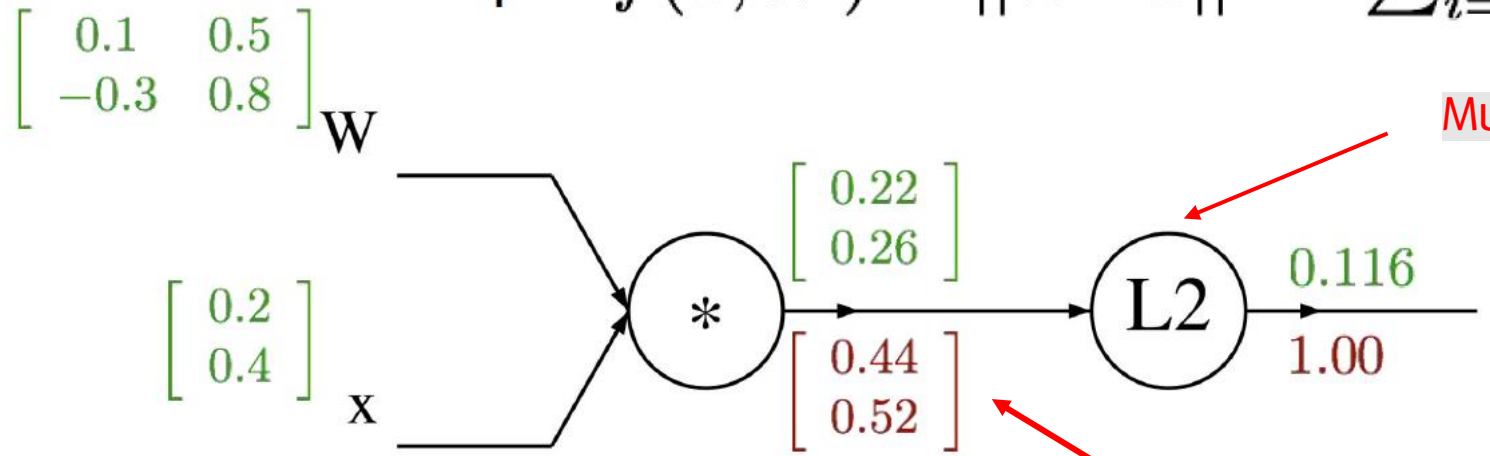
$$\nabla f = \begin{bmatrix} 6x_1 + 4x_2 \\ 4x_1 \end{bmatrix}$$

$$f(x_1, x_2) = \begin{bmatrix} 2x_1 + 4x_2^2 \\ 3x_1x_2 + 6x_2 \end{bmatrix}$$

$$\nabla f(x_1, x_2) = \begin{bmatrix} 2 & 3x_2 \\ 8x_2 & 3x_1 + 6 \end{bmatrix}$$

2. backpropagation(vector)

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



Multivariate scalar function

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

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

이거를 미분하면 어떻게 되게요?

2. backpropagation(vector)

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

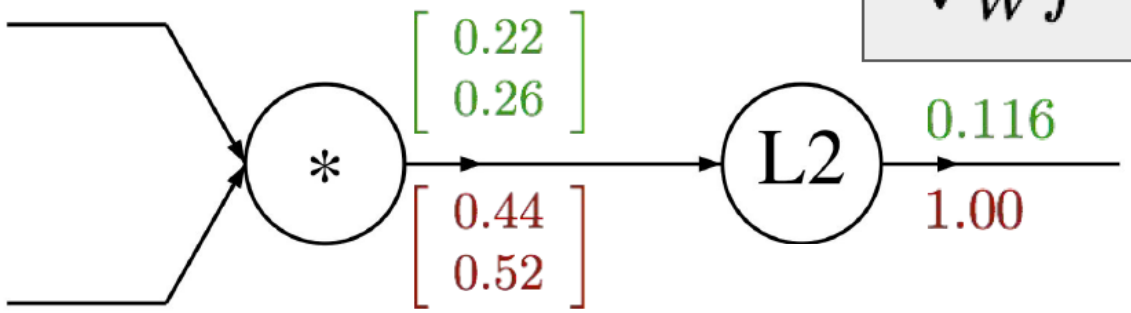
$$\begin{bmatrix} 0.1 & 0.5 \\ -0.3 & 0.8 \end{bmatrix}$$

W

$$\begin{bmatrix} 0.088 & 0.176 \\ 0.104 & 0.208 \end{bmatrix}$$

$$\begin{bmatrix} 0.2 \\ 0.4 \end{bmatrix}$$

x



$$\begin{bmatrix} 0.22 \\ 0.26 \end{bmatrix}$$

$$\begin{bmatrix} 0.44 \\ 0.52 \end{bmatrix}$$

0.116

1.00

$$\nabla_W f = 2q \cdot x^T$$

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

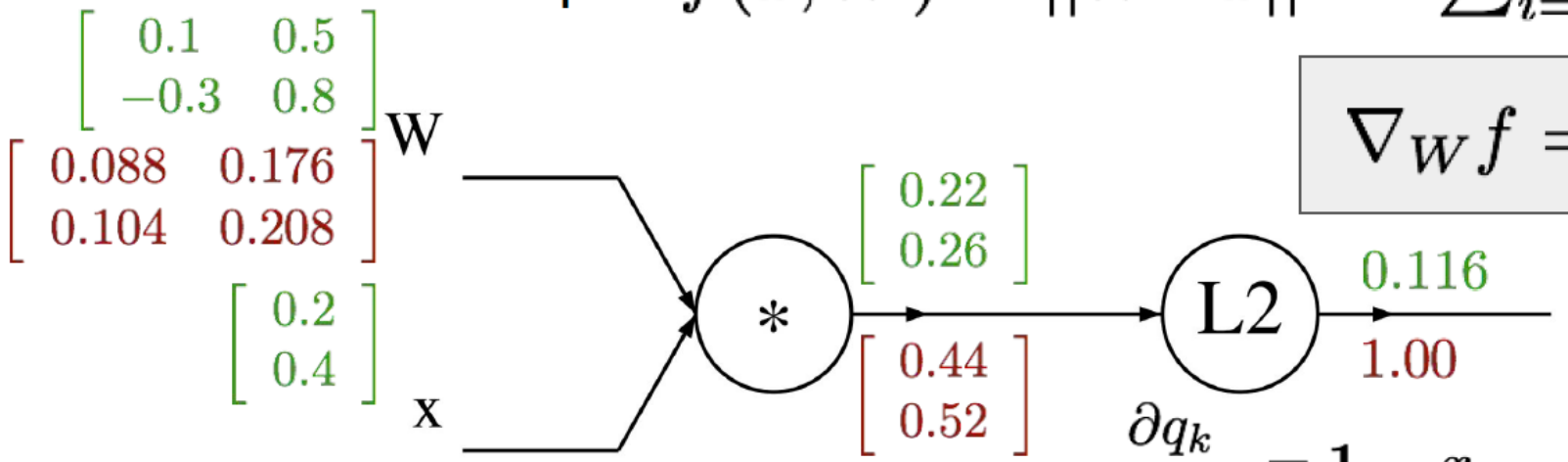
$$f(q) = ||q||^2 = q_1^2 + \cdots + q_n^2$$

$$q = Wx = \begin{bmatrix} w11 & w12 \\ w21 & w22 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \end{bmatrix} = \begin{bmatrix} w11 * x1 + w12 * x2 \\ w21 * x1 + w22 * x2 \end{bmatrix}$$

$$\nabla q = \begin{bmatrix} & \end{bmatrix}$$

2. backpropagation(vector)

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



$$\nabla_W f = 2q \cdot x^T$$

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

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

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

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

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

2. backpropagation(vector)

$$W=\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \quad x=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$g = Wx = \begin{bmatrix} w_{11}x_1 + w_{12}x_2 \\ w_{21}x_1 + w_{22}x_2 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} = g(W,x)$$

$$f(g) = \|g\|^2 = g_1^2 + g_2^2 = z$$

$$\longrightarrow \quad z = f(g(W,x))$$

$$\frac{\partial E}{\partial z} = \begin{bmatrix} \frac{\partial E}{\partial w_{11}} & \frac{\partial E}{\partial w_{12}} \\ \frac{\partial E}{\partial w_{21}} & \frac{\partial E}{\partial w_{22}} \end{bmatrix}$$ 이 행렬을 구하자!

$$\frac{\partial z}{\partial w_{11}} = \frac{\partial}{\partial w_{11}} f(g_1, g_2) = \frac{\partial z}{\partial g_1}\frac{\partial g_1}{\partial w_{11}}^{x_1} + \frac{\partial z}{\partial g_2}\frac{\partial g_2}{\partial w_{11}}^{0} = 2g_1x_1$$

$$\frac{\partial z}{\partial w_{12}} = \frac{\partial}{\partial w_{12}} f(g_1, g_2) = \frac{\partial z}{\partial g_1}\frac{\partial g_1}{\partial w_{12}}^{x_2} + \frac{\partial z}{\partial g_2}\frac{\partial g_2}{\partial w_{12}}^{0} = 2g_1x_2$$

$$\frac{\partial z}{\partial w_{21}} = \frac{\partial}{\partial w_{21}} f(g_1, g_2) = \frac{\partial z}{\partial g_1}\frac{\partial g_1}{\partial w_{21}}^{0} + \frac{\partial z}{\partial g_2}\frac{\partial g_2}{\partial w_{21}}^{x_1} = 2g_2x_1$$

$$\frac{\partial z}{\partial w_{22}} = \frac{\partial}{\partial w_{22}} f(g_1, g_2) = \frac{\partial z}{\partial g_1}\frac{\partial g_1}{\partial w_{22}}^{0} + \frac{\partial z}{\partial g_2}\frac{\partial g_2}{\partial w_{22}}^{x_2} = 2g_2x_2$$

$\frac{\partial z}{\partial g_1} = 2g_1$ ($\because z = g_1^2 + g_2^2$) $\qquad \frac{\partial z}{\partial g_2} = 2g_2$

$$\therefore \frac{\partial E}{\partial W} = 2\begin{bmatrix} g_1 \\ g_2 \end{bmatrix}\begin{bmatrix} x_1 & x_2 \end{bmatrix} = 2g \cdot x^T$$

2. backpropagation(vector)

1. Backpropagation?

합성함수는 합성함수를 구성하는 각 함수의 미분의 곱으로 나타낼 수 있다는 것을 이용하여

기울기(gradient)를 구한 것

(사실 그냥 합성함수의 미분이다.)

2. Backpropagation해서 뭐 어떻게 하라고?

$$W := W - \alpha \frac{\partial}{\partial W} cost(W)$$

여기에 구한 값을 넣어서 parameter update하면 된다

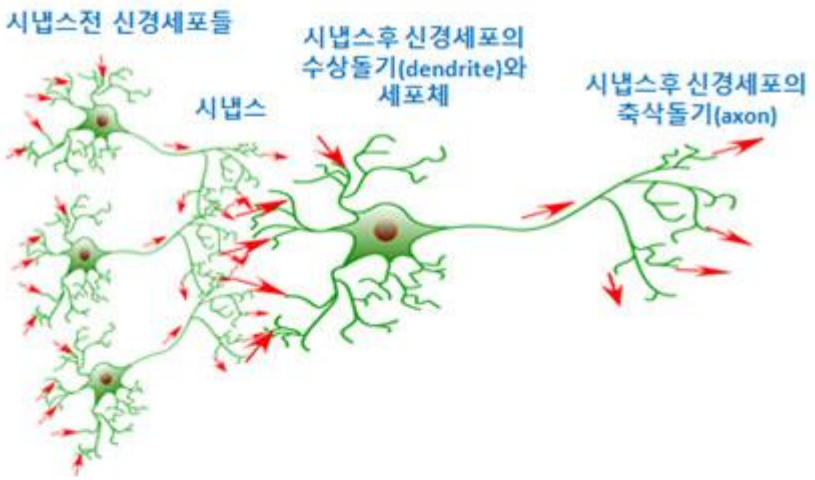
INDEX

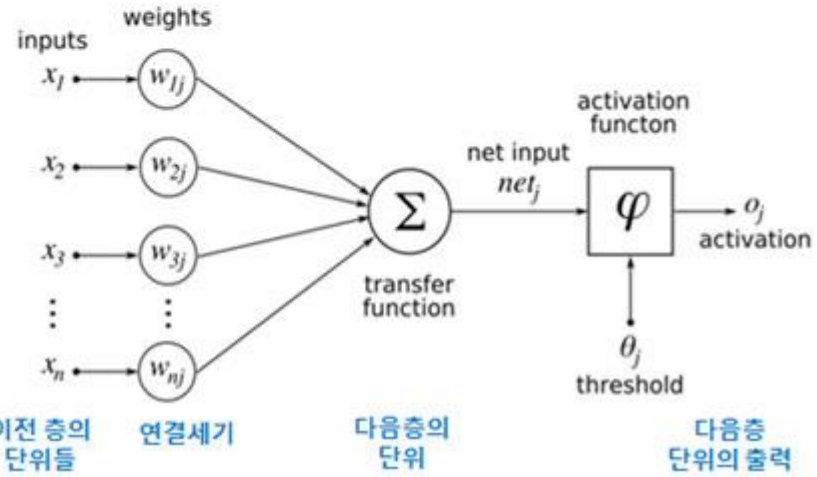
3. Idea of NN

뇌 속 신경망

수학적으로 모델링

인공 신경망





신경세포는 시냅스를 거쳐서 수상돌기로 받아들인
외부의 전달물질(=input)을 세포체에 저장하다가
자신의 용량을 넘어서면(=활성함수 값)
축색돌기를 통해 외부로 전달한다.

인공신경망은 생물학적인 뉴런을 수학적으로 모델링한 것

INDEX

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

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

or 3-layer Neural Network

$$f = W_3 \max(0, W_2 \max(0, W_1 x))$$

Neural Network?

Linear -> activation -> Linear -> activation -> ··· -> Linear -> 분류기(ex. Softmax)

4. Neural Network

Activation?

Linear만 깊게 쌓는 것은 의미가 X ->

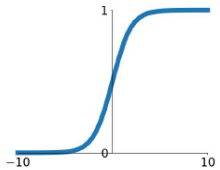
W3(W2(W1x+b1)+b2)+b3) = W3W2W1x+W3W2b1+W3b2+b3 = W4x1 + b4

그냥 하나의 Linear layer와 하다.

Activation functions

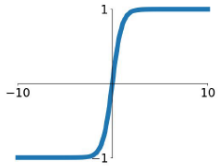
Sigmoid

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



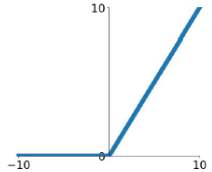
tanh

$\tanh(x)$



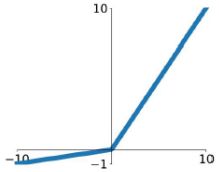
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

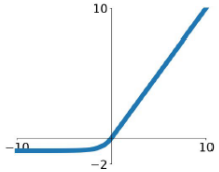


Maxout

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

ELU

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



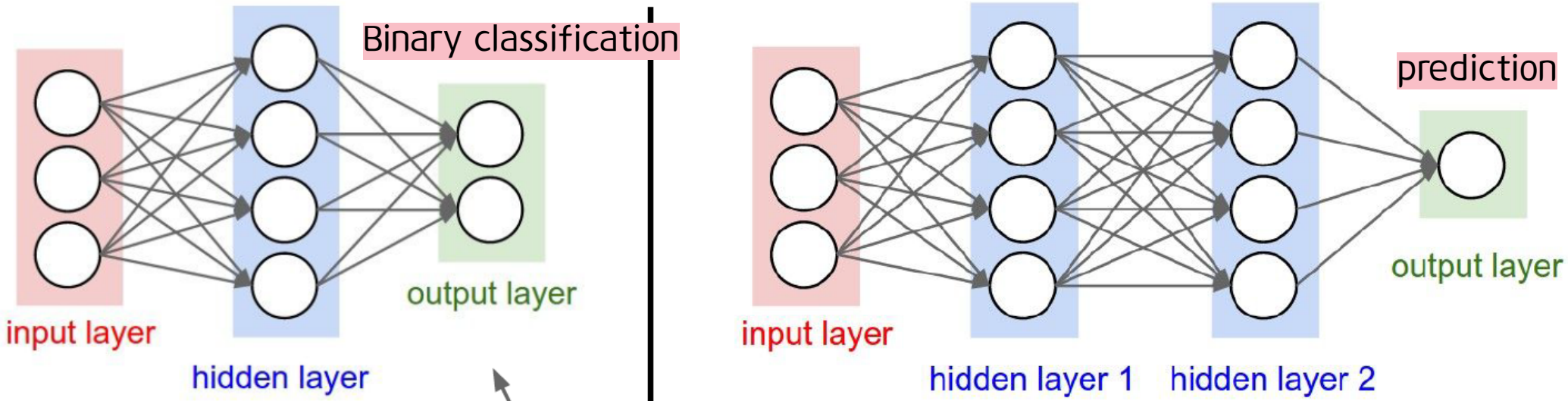
Linear에서 나온 값을

Activation function을 통과시켜

비선형성(non-linear)을 부여하는 것

-> layer를 깊게 쌓는 것이 의미있도록!

Neural networks: Architectures

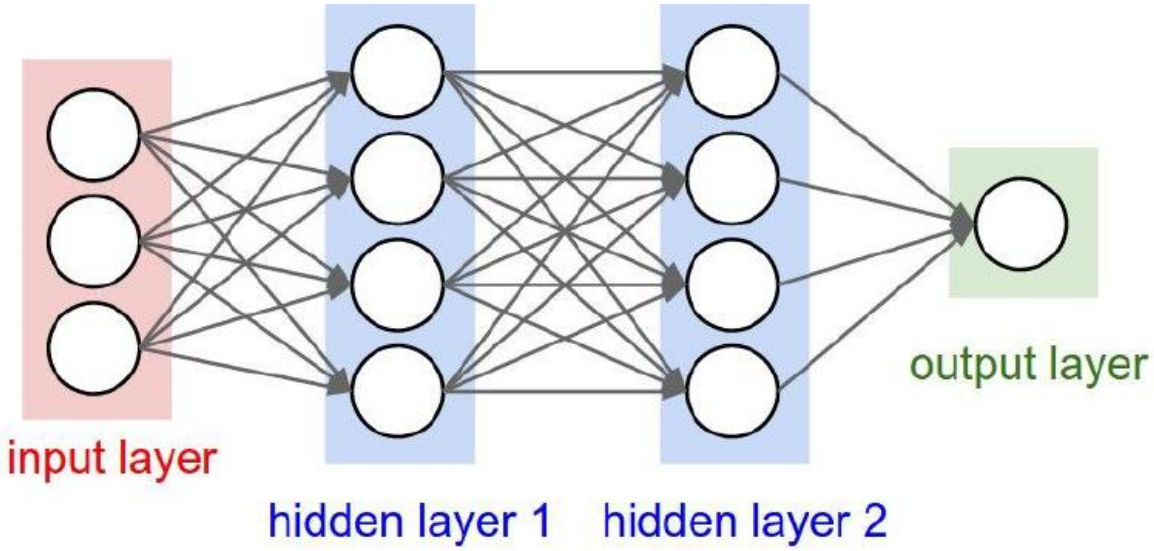


"2-layer Neural Net", or
"1-hidden-layer Neural Net"

"3-layer Neural Net", or
"2-hidden-layer Neural Net"

"Fully-connected" layers

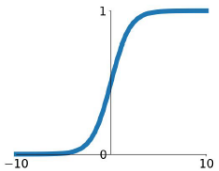
4. Neural Network




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

Sigmoid

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



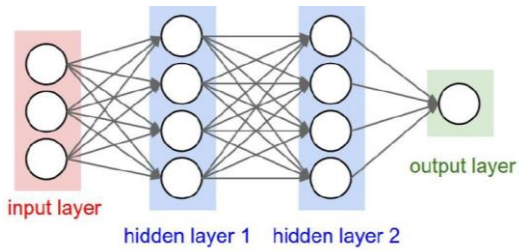
Linear -> sigmoid -> linear -> sigmoid -> linear (*여기서는 prediction 문제라서 분류기 사용X*)

4. Neural Network

Value

이게 얼마가 될거 같니?

output을 그냥 받는다.

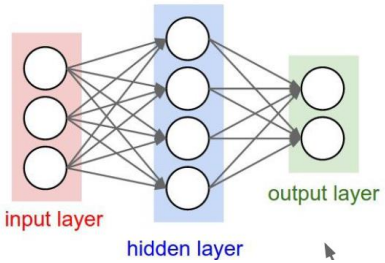


(실제값-예측값)^2의 합 / 2
= (평균제곱오차)을 Loss

O/X

기냐? 아니냐?

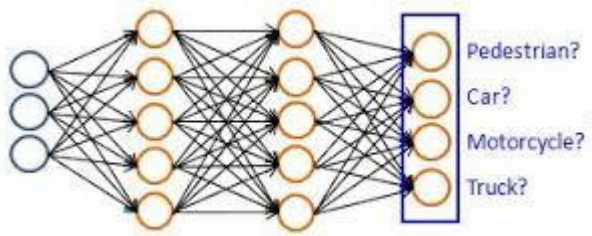
output에 sigmoid를 먹인다.



Category

종류중에 요건 뭐냐?

output에 softmax를 먹인다.



classification 문제에서는 cross entropy(=negative log likelihood)를 Loss로 두고 계산

https://ratsgo.github.io/deep%20learning/2017/09/24/loss/

4. Neural Network

[점검] 여기서 모르는 keyword 있다! -> 찾아보고 공부하기 필수 ☺

1. Back propagation
2. Forward propagation
3. Gradient descent algorithm
4. Fully-connected layer
5. Activation function
6. Sigmoid, softmax의 정의 + 두 개의 차이점
7. Cross entropy loss
8. Mean squared error loss
9. Prediction, classification(binary, multi-class)
10. Hidden layer

감 사 합 니 다 !

발표 들어주셔서 감사합니다