Learning From Data

Lecture 6: Backpropagation and Neural Networks

Yang Li

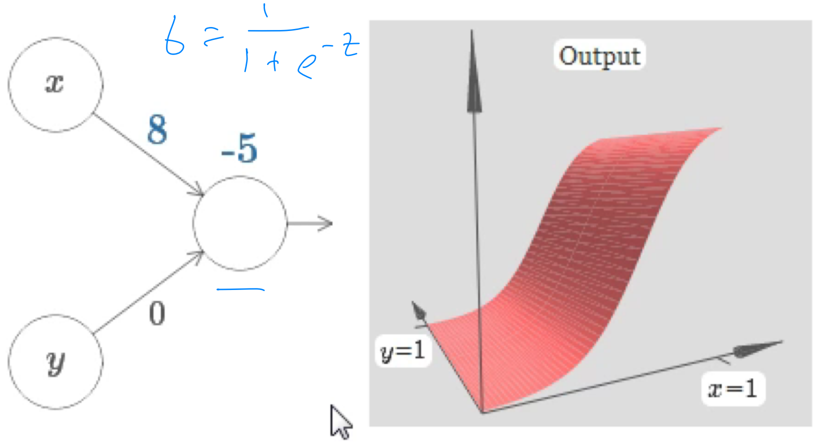
Slides by Lichen Wang

10/28/2022

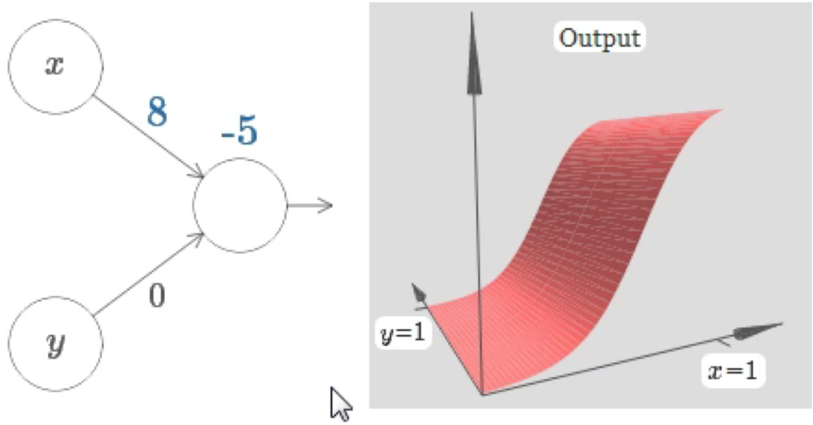
Learning From Data

Power of single neural

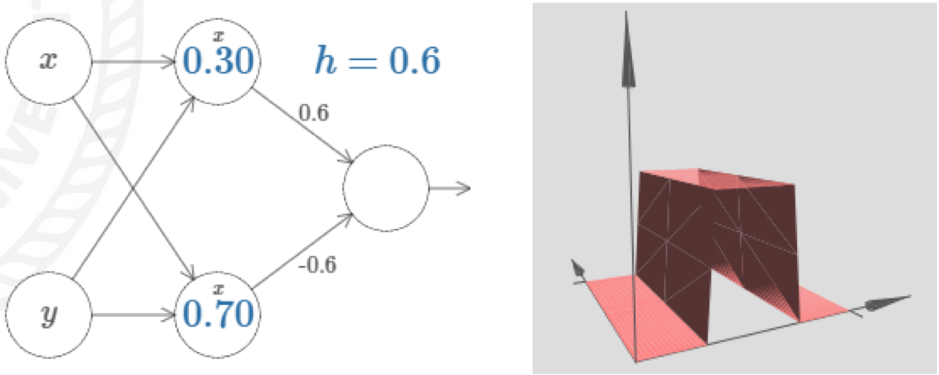
$\sigma = \frac{1}{1+e^{-z}}$



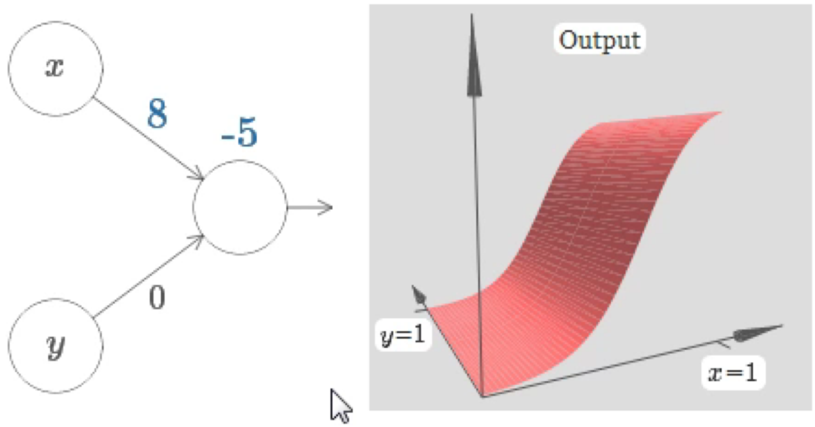
Power of single neural



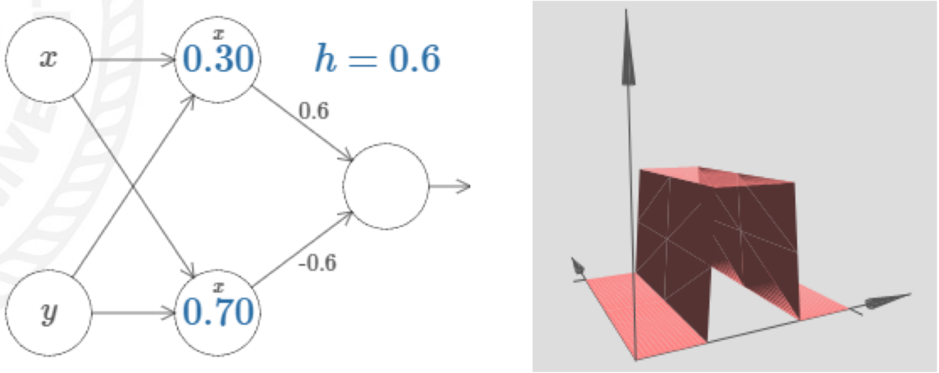
Two hidden units



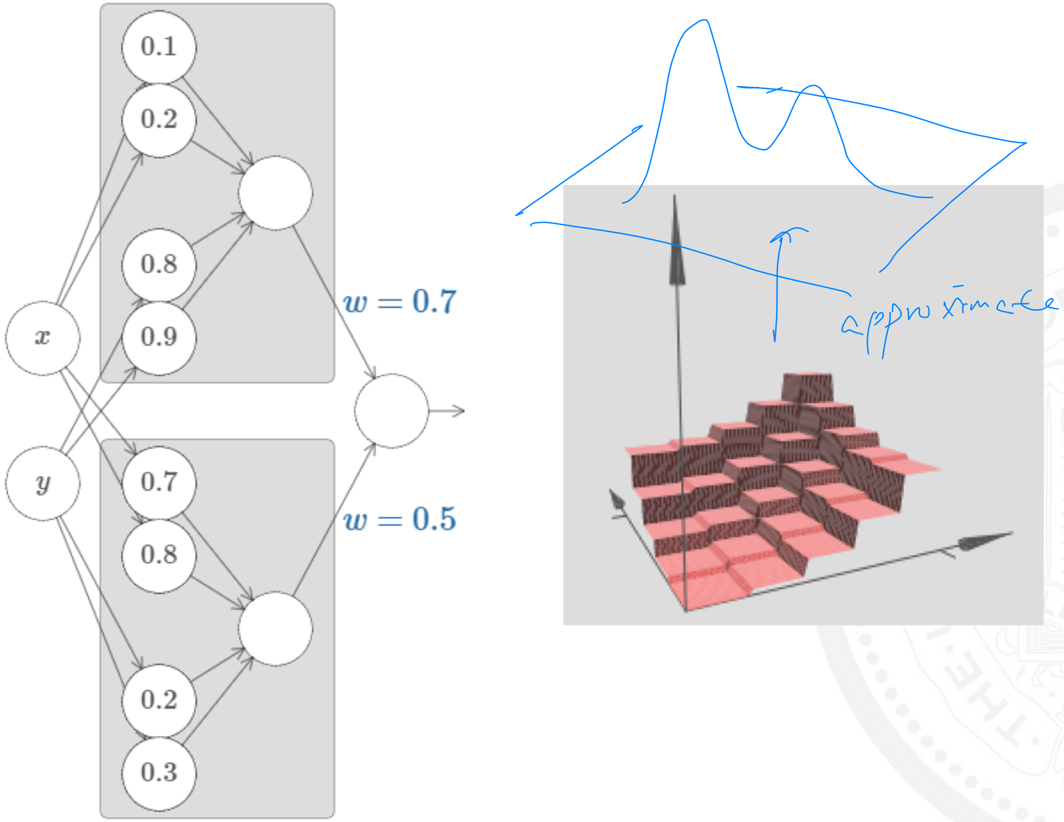
Power of single neural



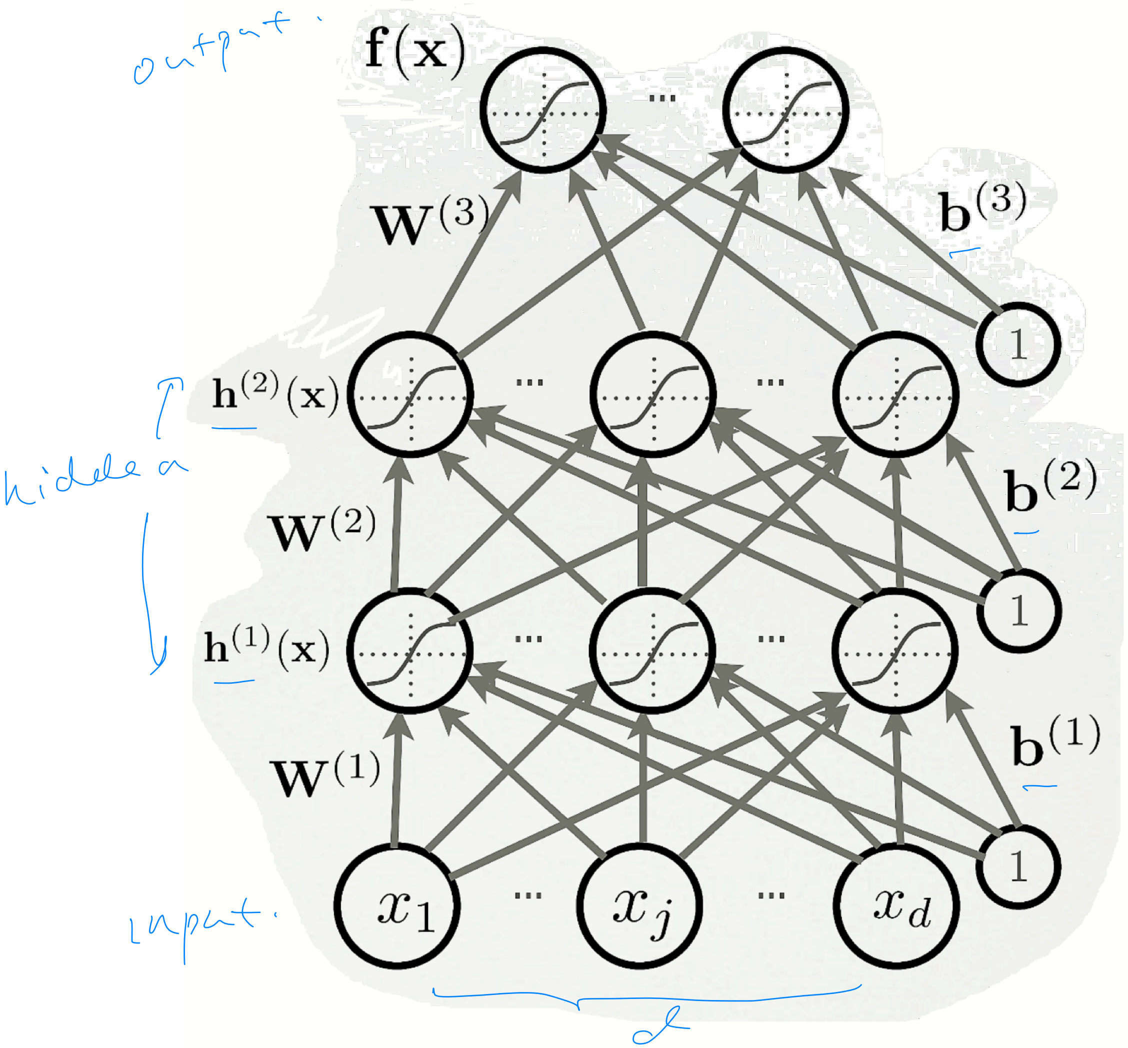
Two hidden units



Many hidden units



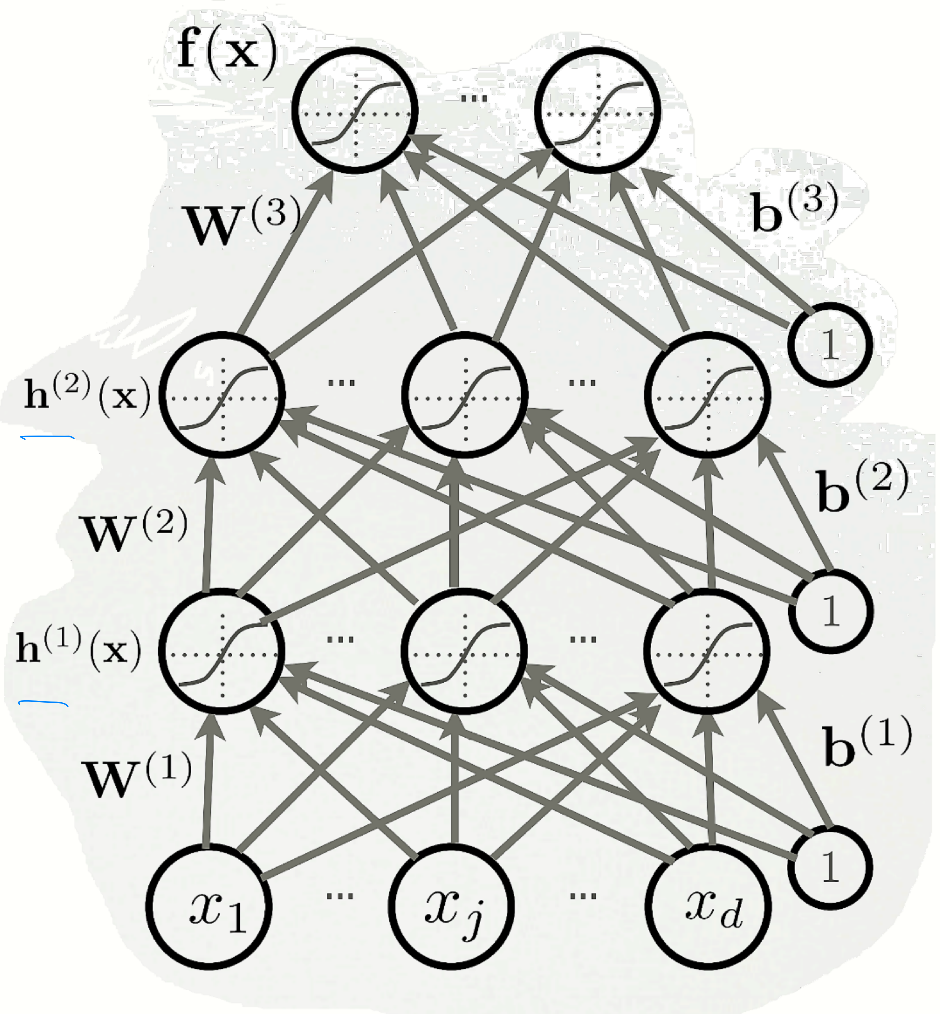
Could have L hidden layers



Could have L hidden layers

- layer input activation for $k>0$, $\mathbf{h}^{(0)}(\mathbf{x})=\mathbf{x}$

$$\mathbf{a}^{(k)}(\mathbf{x})=\mathbf{b}^{(k)}+\mathbf{W}^{(k)}\mathbf{h}^{(k-1)}(\mathbf{x})$$



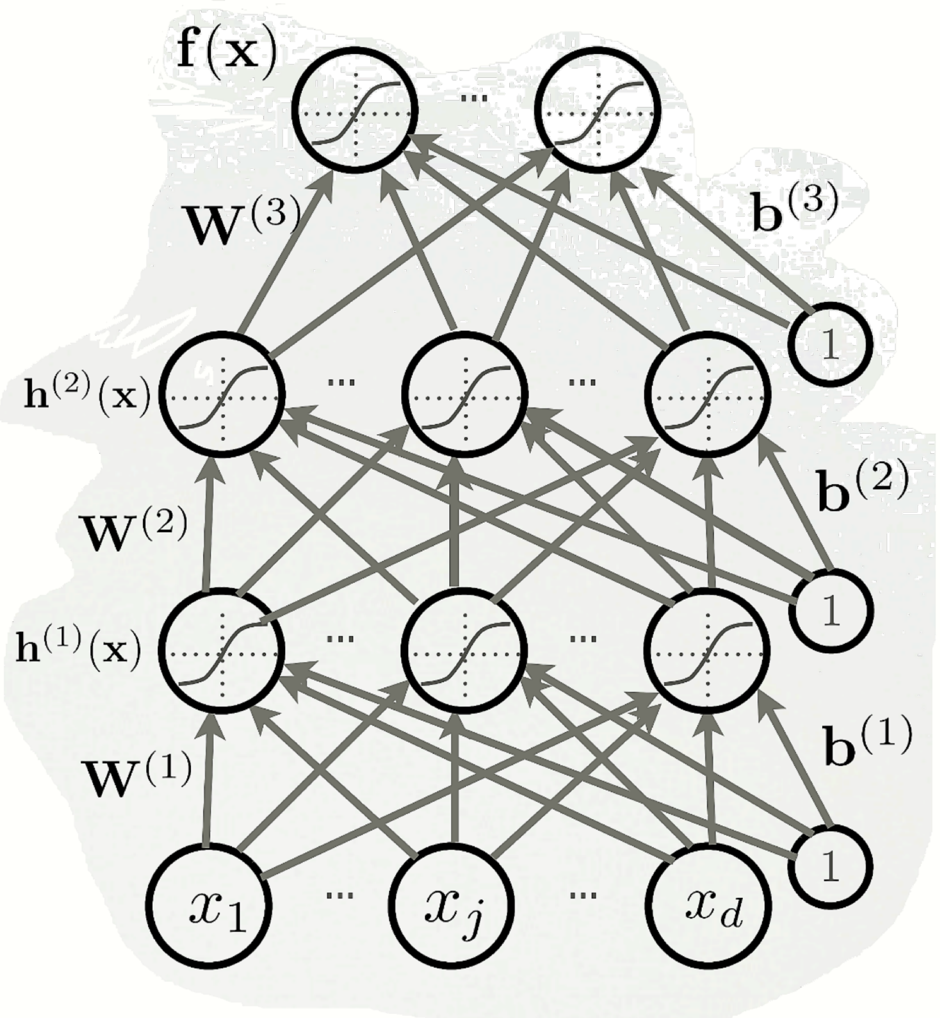
Could have L hidden layers

- layer input activation for $k > 0$, $\mathbf{h}^{(0)}(\mathbf{x}) = \mathbf{x}$

$$\mathbf{a}^{(k)}(\mathbf{x}) = \mathbf{b}^{(k)} + \mathbf{W}^{(k)}\mathbf{h}^{(k-1)}(\mathbf{x})$$

- hidden layer activation for $1 \le k \le L$

$$\mathbf{h}^{(k)}(\mathbf{x}) = \mathbf{g}(\mathbf{a}^{(k)}(\mathbf{x}))$$ e.g $h^{(1)}$ or $h^{(2)}$



Could have L hidden layers

- layer input activation for $k > 0$, $\mathbf{h}^{(0)}(\mathbf{x}) = \mathbf{x}$

$$\mathbf{a}^{(k)}(\mathbf{x}) = \mathbf{b}^{(k)} + \mathbf{W}^{(k)}\mathbf{h}^{(k-1)}(\mathbf{x})$$

- hidden layer activation for $1 \le k \le L$

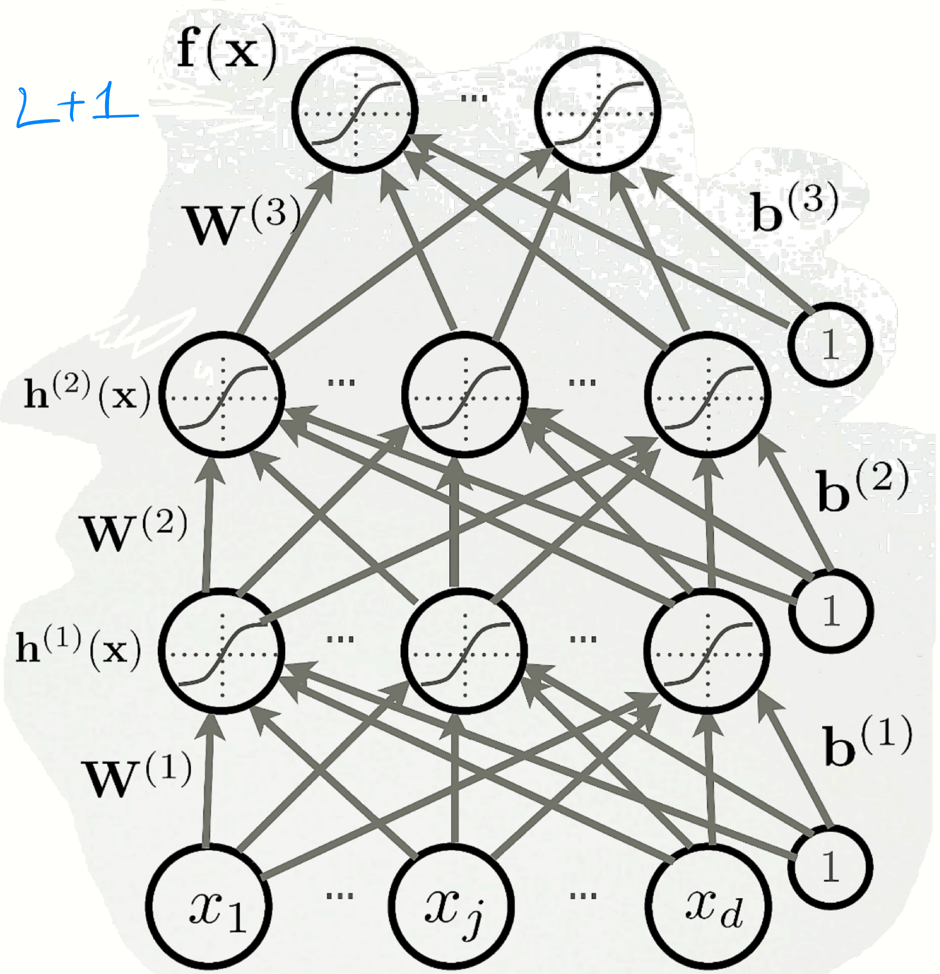
$$\mathbf{h}^{(k)}(\mathbf{x}) = \mathbf{g}(\mathbf{a}^{(k)}(\mathbf{x}))$$

- output layer activation for $k = L+1$

$$\mathbf{h}^{(L+1)}(\mathbf{x}) = \mathbf{g}(\mathbf{a}^{(L+1)}(\mathbf{x})) = \mathbf{f}(\mathbf{x})$$ output.

L

L+1



Empirical risk

$$\text{argmin}_W \frac{1}{m}\sum_{i=1}^{m} L\left(f\left(\mathbf{x}^{(i)}; W\right), y^{(i)}\right) + \lambda\Omega(W)$$

- $L\left(f\left(\mathbf{x}^{(i)}; W\right), y^{(i)}\right)$ is the loss function for sample $(\mathbf{x}^{(i)}, y^{(i)})$

  $\hat{y}^{(i)}$

- $\lambda\Omega(W)$ is the regularizer $p(y=i|x) = \frac{e^{\theta_i^T x}}{\sum_j e^{\theta_j^T x}}$

e. g. When L is the softmax loss

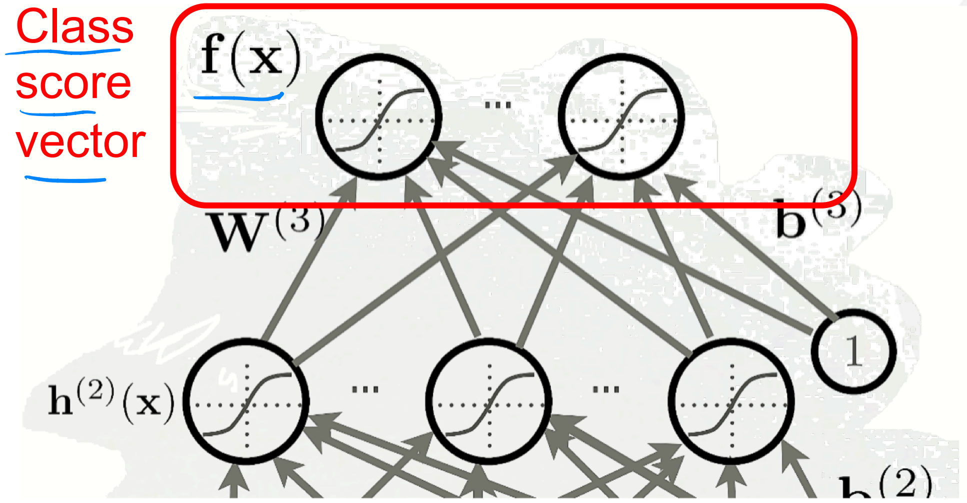
$$L\left(f\left(\mathbf{x}^{(i)}; W\right), y^{(i)}\right) = -\log\left(\frac{e^{f_{y^{(i)}}}}{\sum_{j=1}^{|Y|} e^{f_j}}\right)$$

f_j is the jth element of class score vector $f\left(\mathbf{x}^{(i)}; W\right)$

Softmax example:

Unnormalized class probability of |Y| classes

Class score vector



Optimization

- Find the optimal parameter

$$\operatorname{argmin}_W \frac{1}{m}\sum_{i=1}^{m} L\left(f\left(\mathbf{x}^{(i)}; W\right), y^{(i)}\right) + \lambda\Omega(W)$$

Optimization

- Find the optimal parameter

$$\operatorname{argmin}_W \frac{1}{m}\sum_{i=1}^{m} L\left(f\left(\mathbf{x}^{(i)}; W\right), y^{(i)}\right) + \lambda\Omega(W)$$

- Stochastic Gradient Descent (SGD)

Algorithm

1. Initialize $\mathbf{W}$

repeat: for each training example $(\mathbf{x}^{(t)}, y^{(t)})$

2a. $\Delta = -\nabla_{\mathbf{W}} L\left(f\left(\mathbf{x}^{(t)}; \mathbf{W}\right), y^{(t)}\right) - \lambda \nabla_{\mathbf{W}} \Omega(\mathbf{W})$

2b. $\mathbf{W} \leftarrow \mathbf{W} + \alpha\Delta$

Training epoch
=
Iterating over all examples

To apply this algorithm, we need:

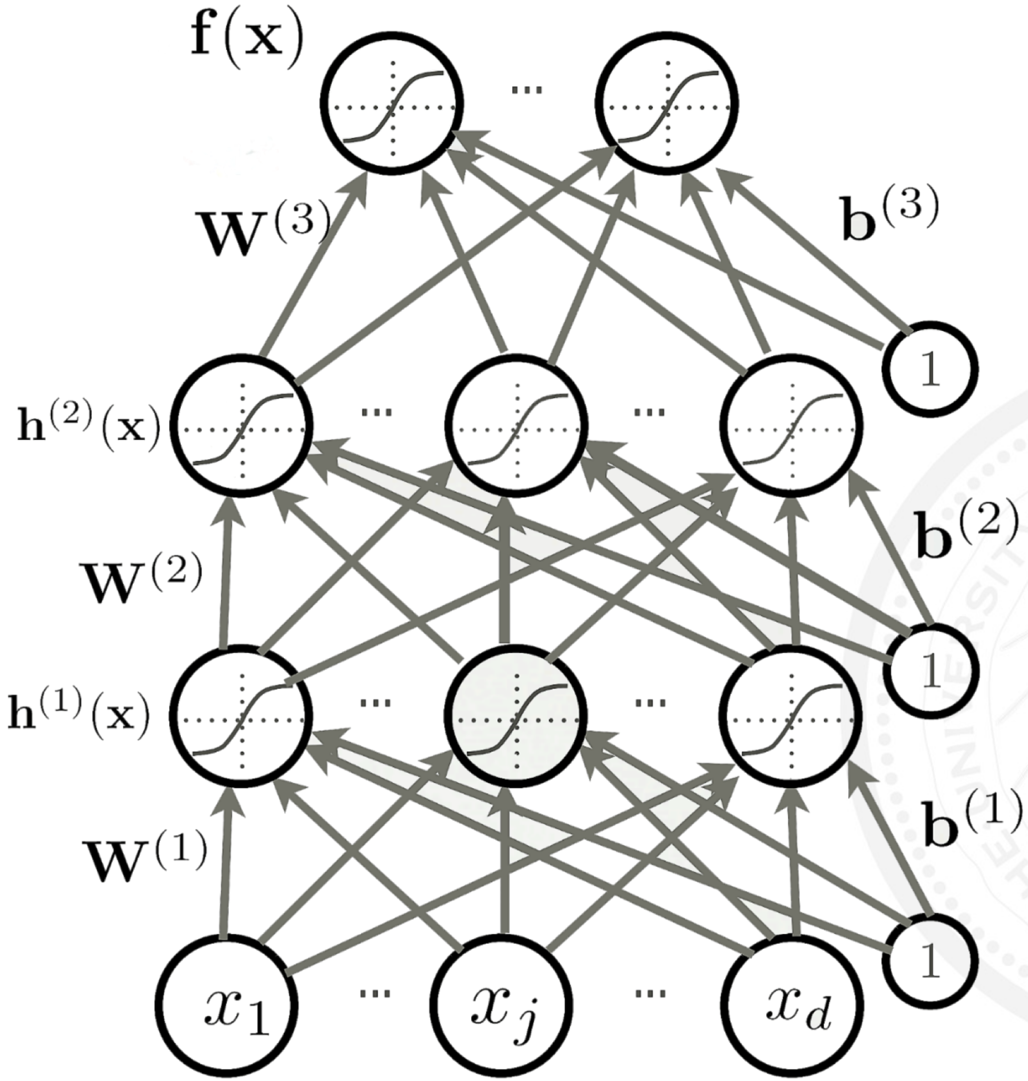
1. A procedure to compute the parameter gradient
2. The regularizer (and its gradient)
3. Updating rule
4. Initialization method

Animation:

http://vision.stanford.edu/teaching/cs231n-demos/linear-classify/

Follow the slope

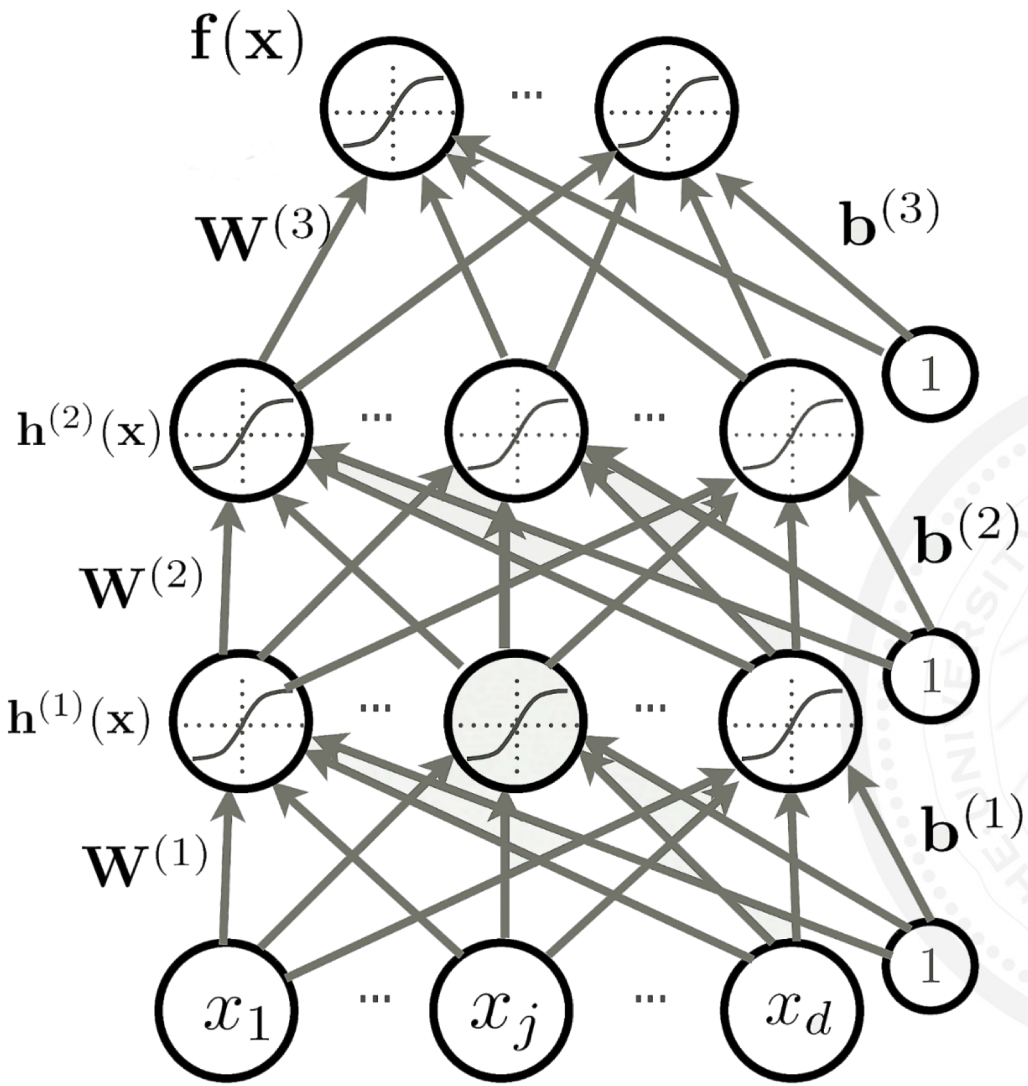
How many parameter do we have?



Follow the slope

How many parameter do we have?

VGGNet *[Simonyan and Zisserman, 2014]* used 138M parameters

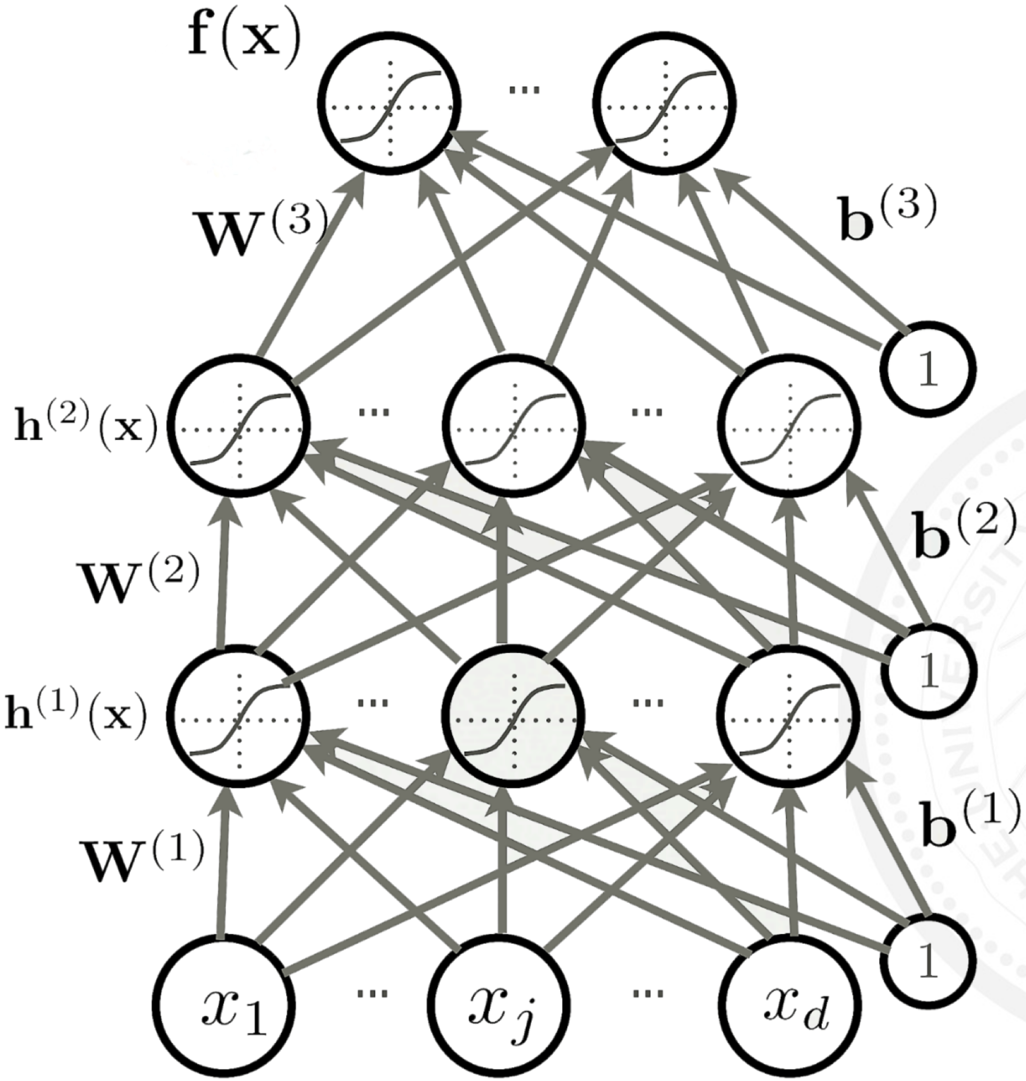


How many parameter do we have?

VGGNet *[Simonyan and Zisserman, 2014]* used 138M parameters

In 1-dimension, the derivative of a function:

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

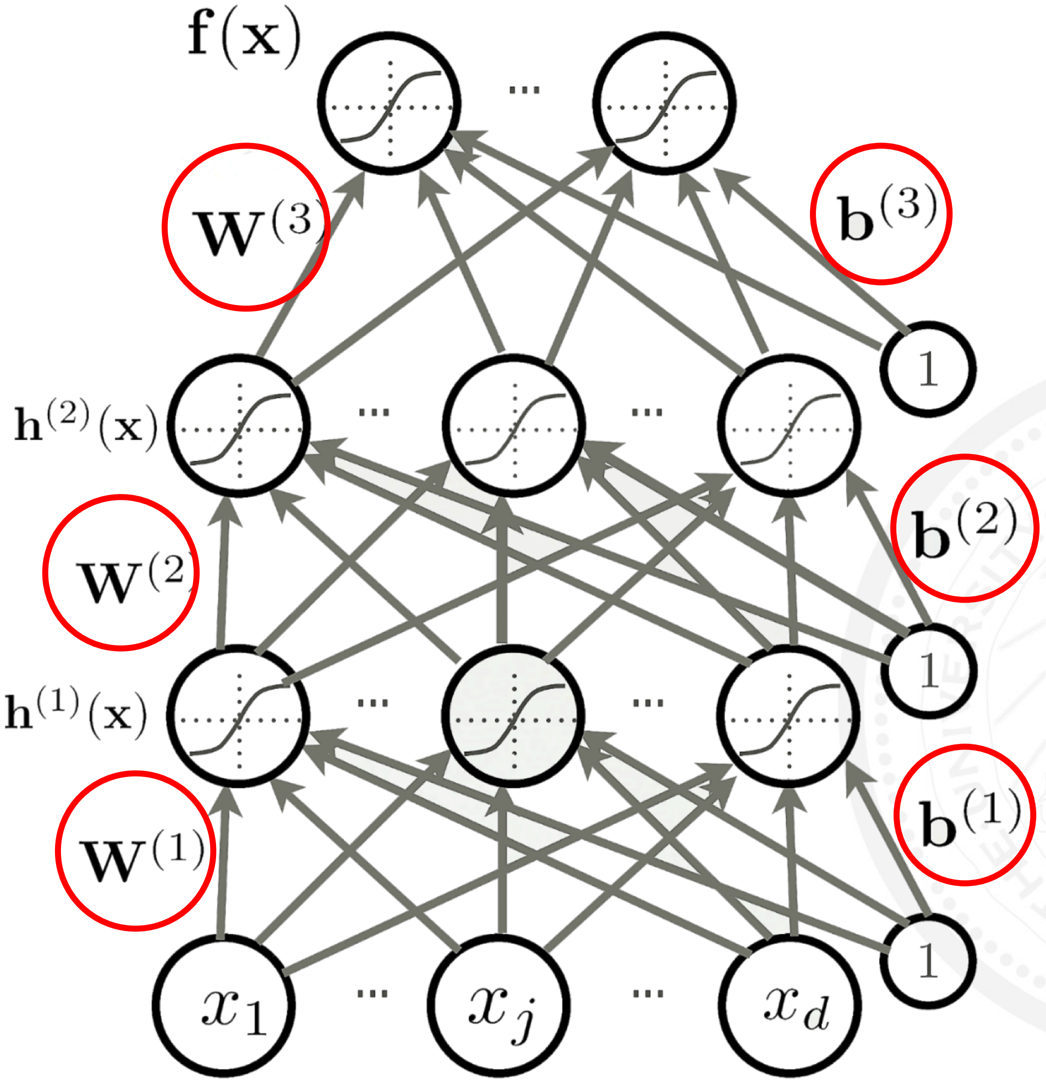


How many parameter do we have?

VGGNet *[Simonyan and Zisserman, 2014]* used 138M parameters

In 1-dimension, the derivative of a function:

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



Numerical Gradient

Current W:

Gradient dW:

Numerical Gradient

Current W:

[0.25,
-1.56,
0.55,
3.8,
0.98,
0.77,
-0.11,
-2.9,…]

Loss 1.25742

$$\partial w = \lim_{h \to 0} \frac{f(w+h) - f(w)}{h}$$

Gradient dW:

[? ,
?,
?,
?,
?,
?,
?,
?,…]

Numerical Gradient

Current W:	W + h (third dim):	Gradient dW:
[0.25,	[0.25 + 0.0001,	[? ,
-1.56,	-1.56,	?,
0.55,	0.55,	?,
3.8,	3.8,	?,
0.98,	0.98,	?,
0.77,	0.77,	?,
-0.11,	-0.11,	?,
-2.9,…]	-2.9,…]	?,…]
Loss 1.25742	**Loss 1.25763**	
$f(w)$	$f(w+h)$	

Current W:	W + h (third dim):	Gradient dW:
[0.25,	w_{01} [0.25 + 0.0001,	[2.1,
-1.56,	-1.56,	?,
0.55,	0.55,	?,
3.8,	3.8,	?,
0.98,	0.98,	?,
0.77,	0.77,	?,
-0.11,	-0.11,	?,
-2.9,…]	-2.9,…]	?,…]
Loss 1.25742	**Loss 1.25763**	

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

$$=\frac{(1.25763-1.25742)}{0.0001}$$

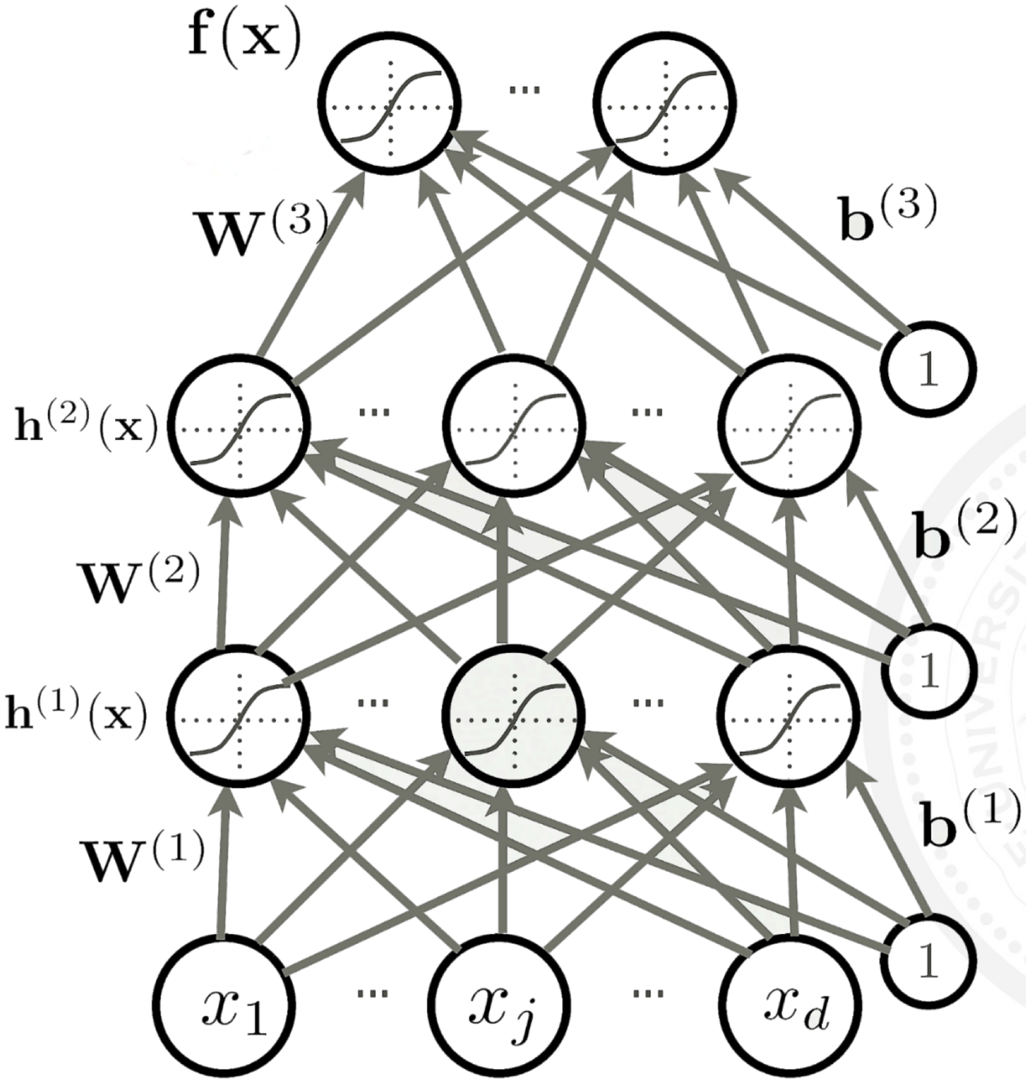
Follow the slope

How many parameter do we have?

VGGNet *[Simonyan and Zisserman, 2014]* used 138M parameters

In 1-dimension, the derivative of a function:

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



Follow the slope

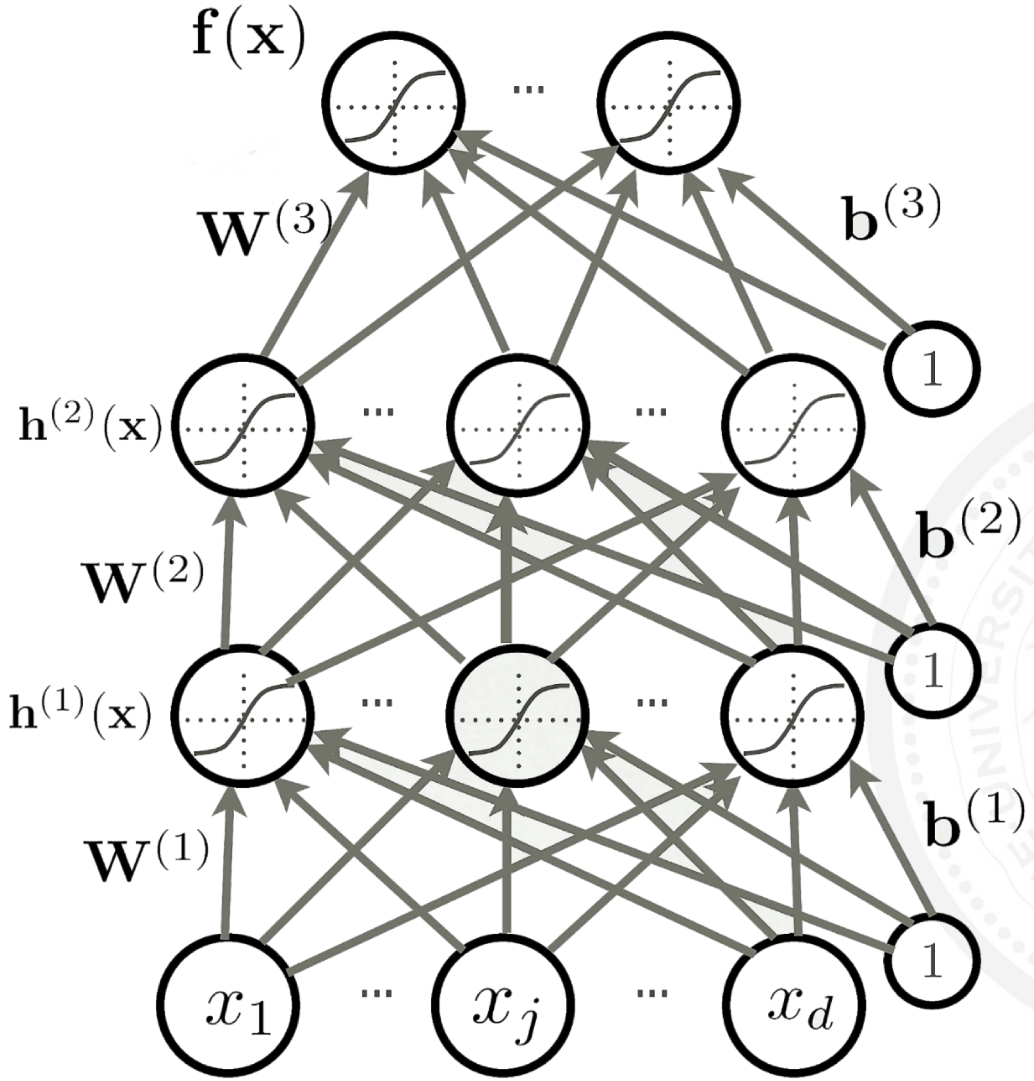
How many parameter do we have?

VGGNet *[Simonyan and Zisserman, 2014]* used 138M parameters

In 1-dimension, the derivative of a function:

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

Numerical gradient: approximate, slow, easy to write



Follow the slope

How many parameter do we have?

VGGNet *[Simonyan and Zisserman, 2014]* used 138M parameters

In 1-dimension, the derivative of a function:

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

Numerical gradient: approximate, slow, easy to write

Calculus!

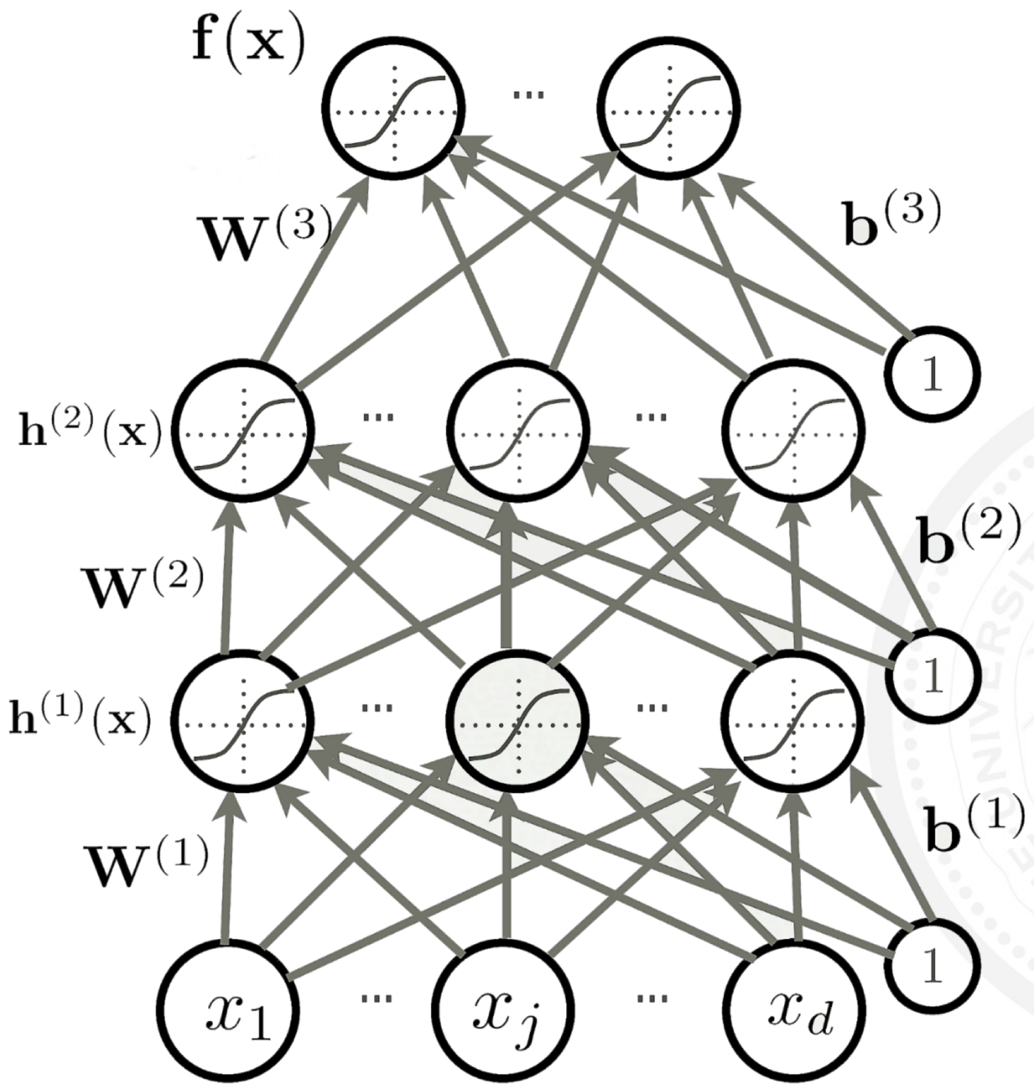
$$\underline{\operatorname{argmin}_W \frac{1}{m} \sum_{i=1}^{m} L\left(f\left(\mathbf{x}^{(i)}; W\right), y^{(i)}\right) + \lambda \Omega(W)}$$

$$\mathbf{h}^{(L+1)}(\mathbf{x}) = \mathbf{g}(\mathbf{a}^{(L+1)}(\mathbf{x})) = \mathbf{f}(\mathbf{x})$$

$$\mathbf{h}^{(k)}(\mathbf{x}) = \mathbf{g}(\mathbf{a}^{(k)}(\mathbf{x}))$$

$$\mathbf{a}^{(k)}(\mathbf{x}) = \mathbf{b}^{(k)} + \mathbf{W}^{(k)} \mathbf{h}^{(k-1)}(\mathbf{x})$$

Analytic gradient: exact, fast, error-prone

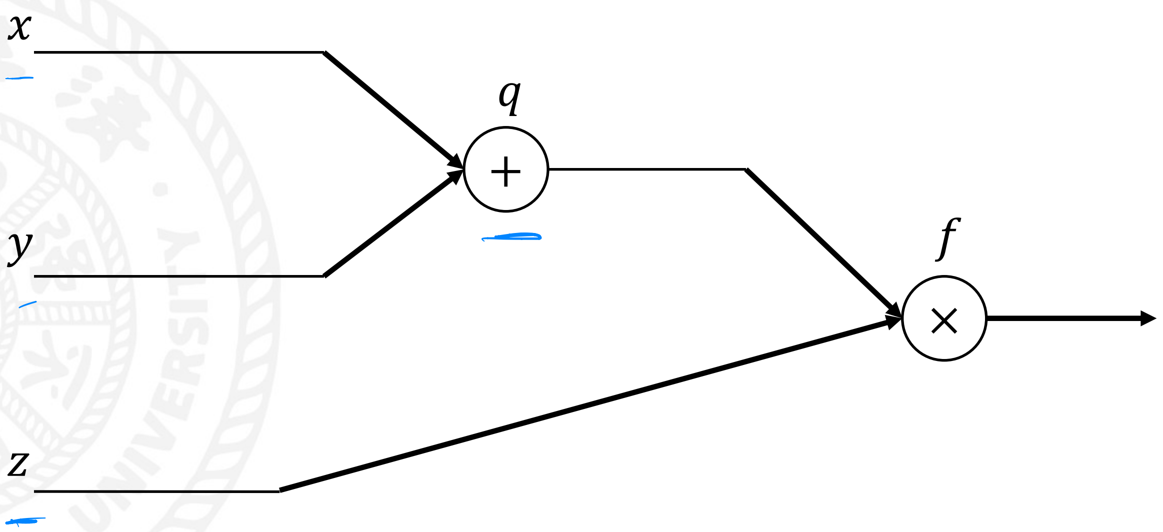


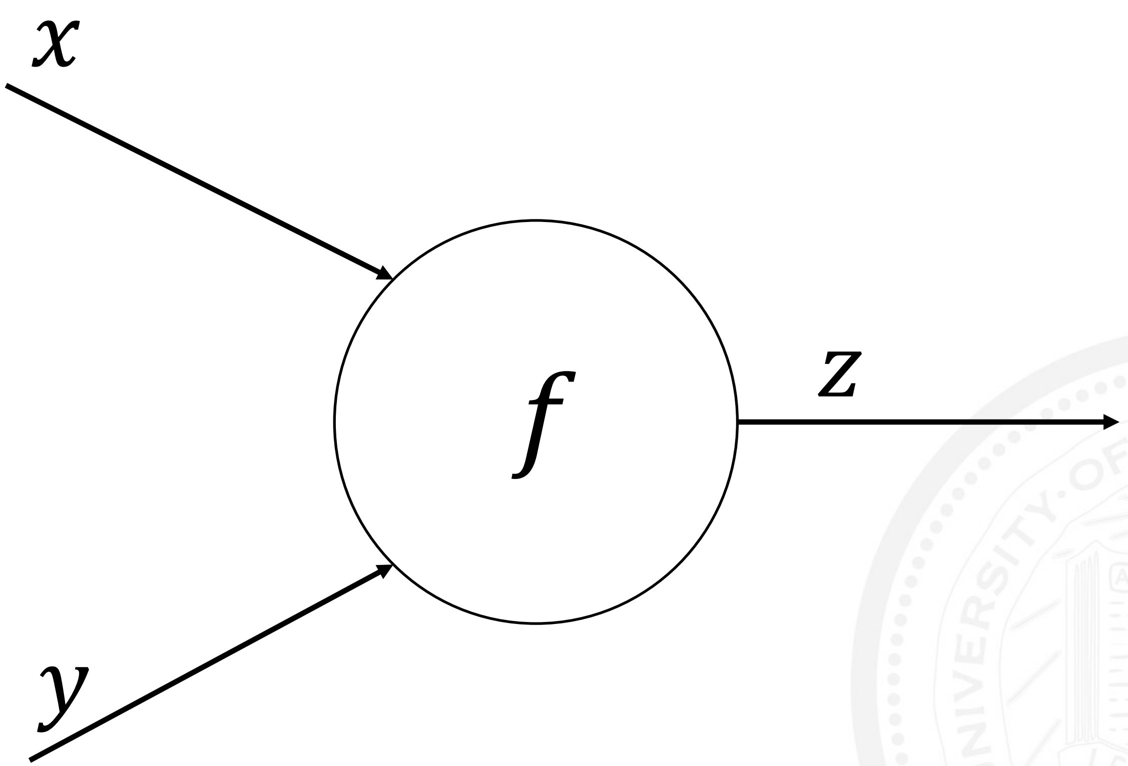
Backpropagation

computation graph

$$f(x,y,z) = \underline{(x+y)}z$$

We want $\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}$



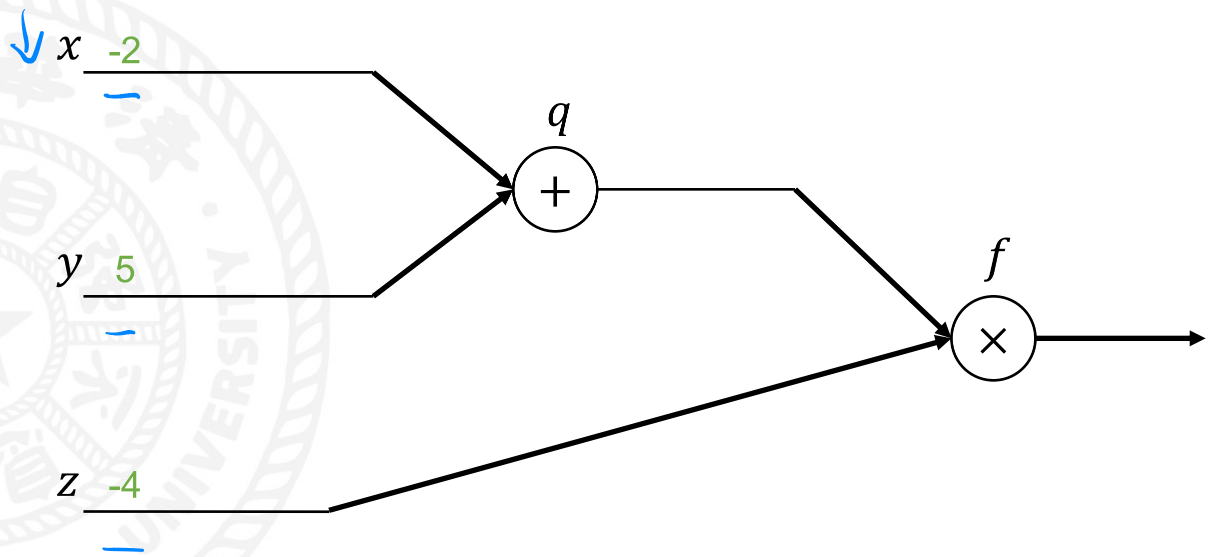


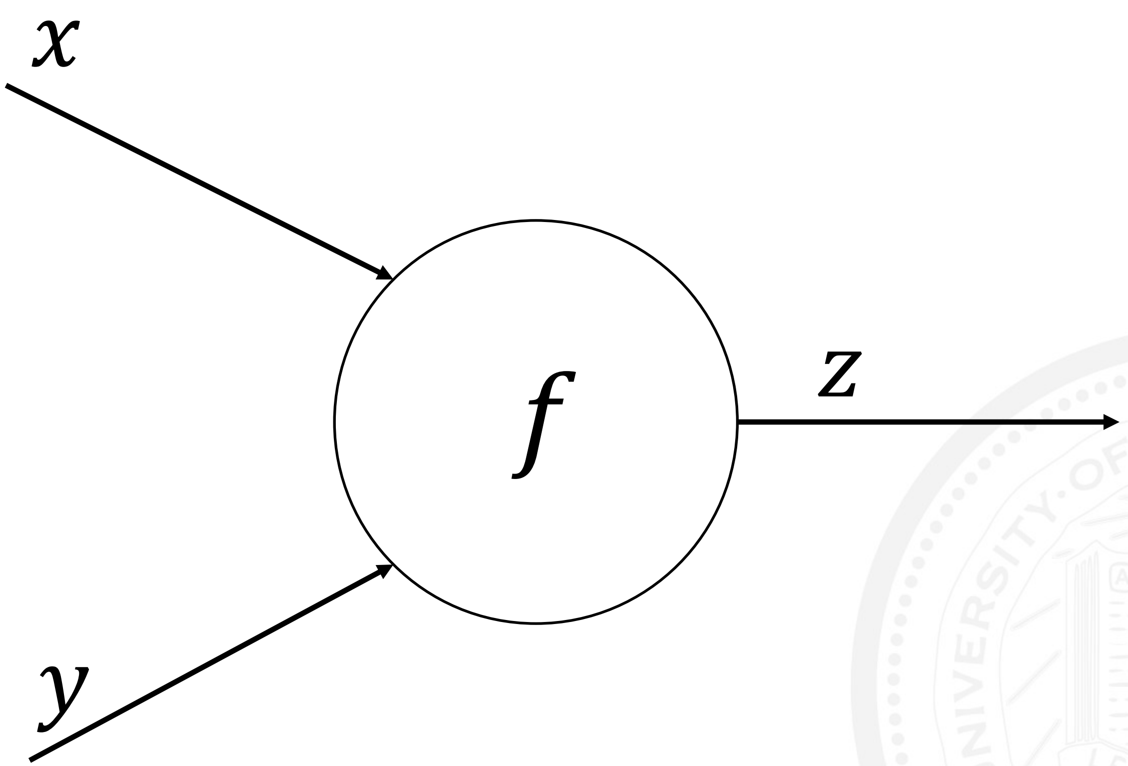
Backpropagation

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

We want $\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}$

Initialize

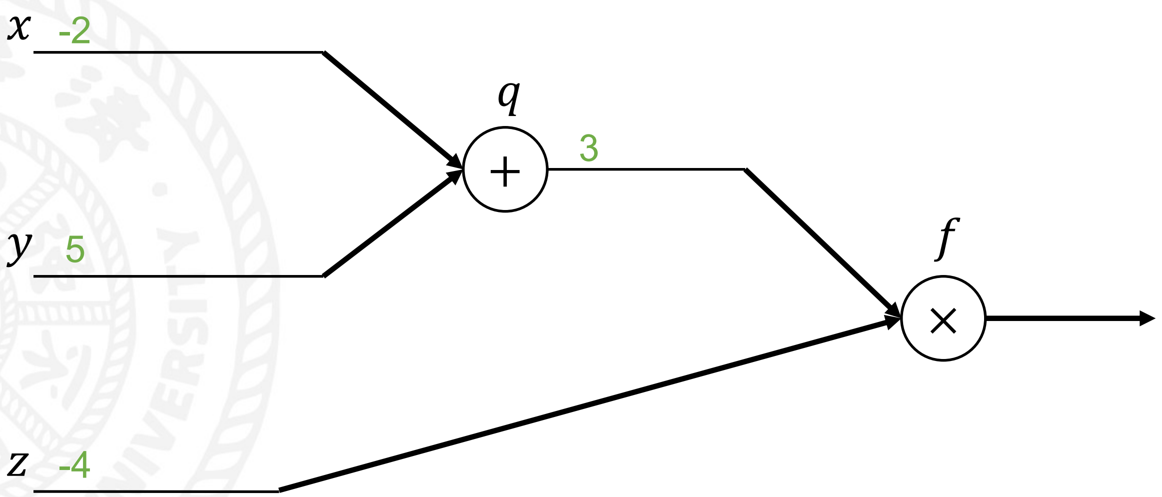


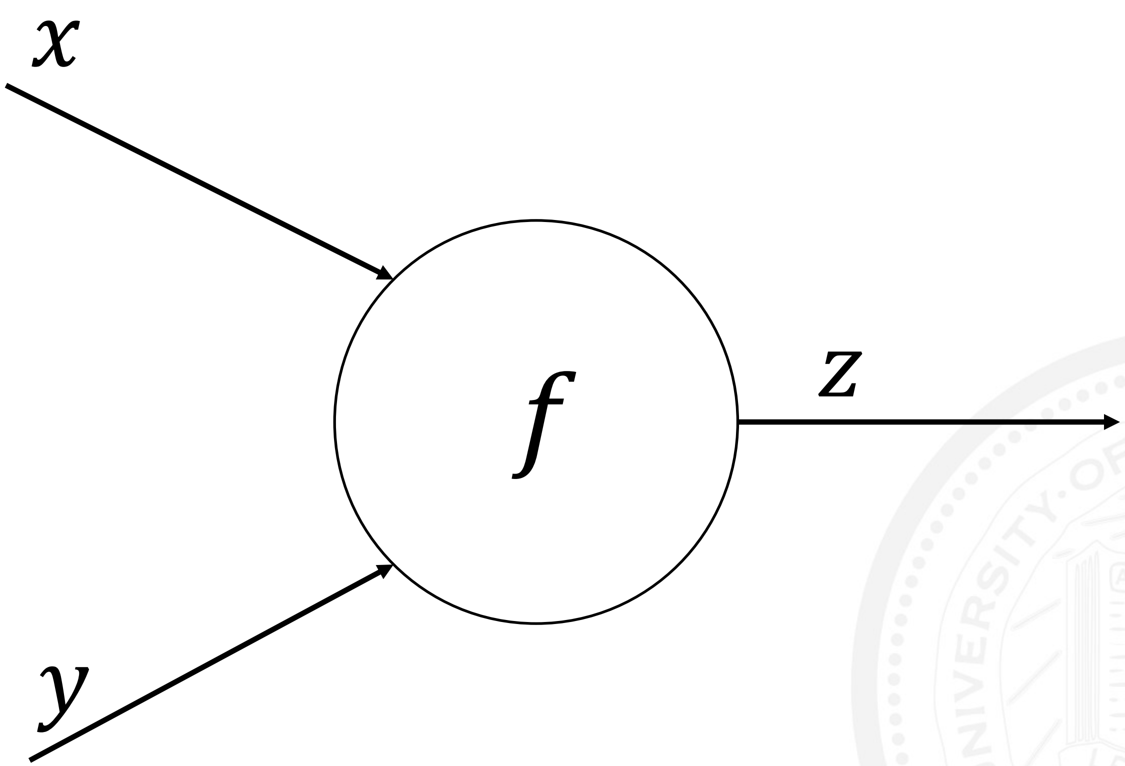


Backpropagation

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

We want $\dfrac{df}{dx}, \dfrac{df}{dy}, \dfrac{df}{dz}$

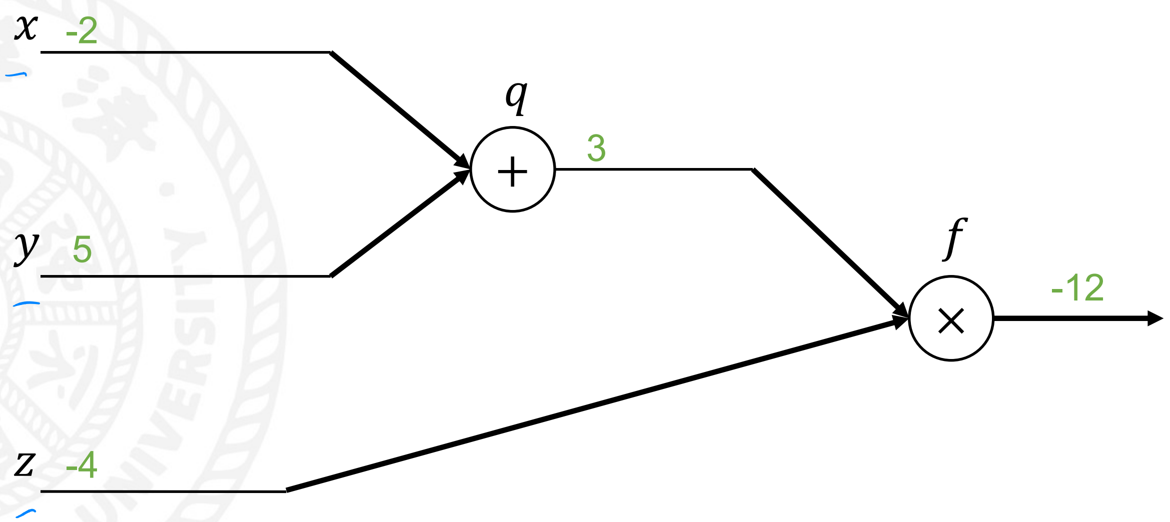


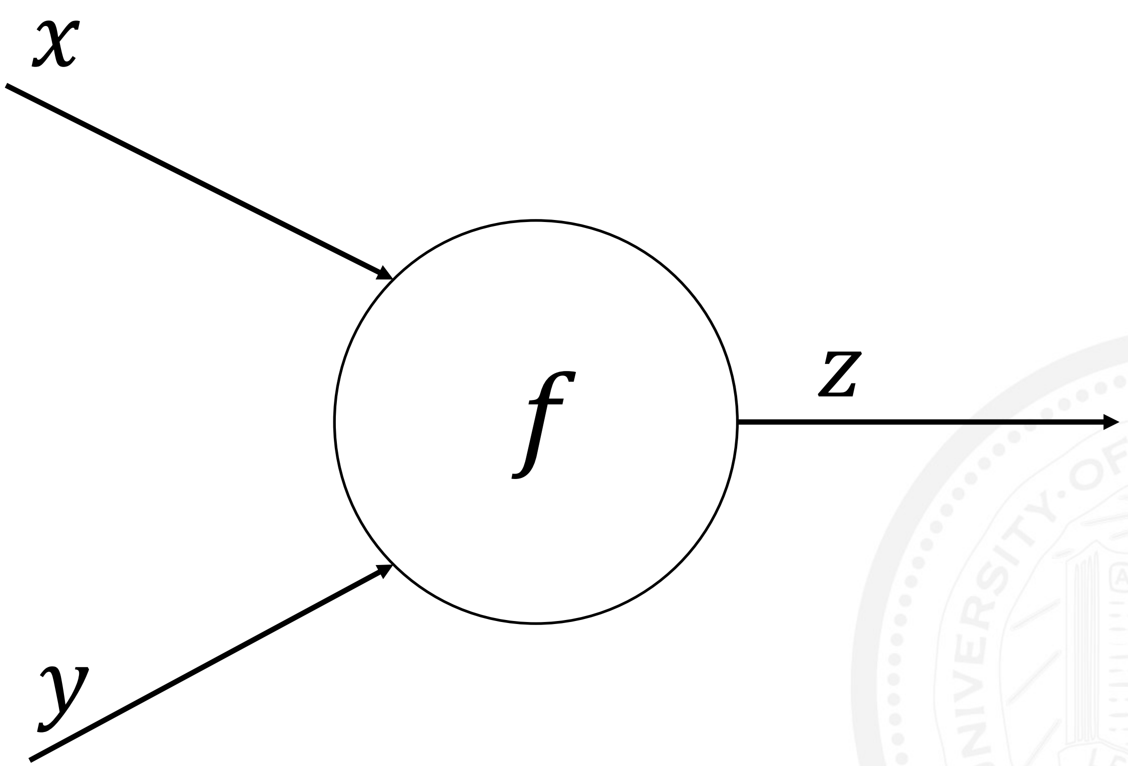


Backpropagation

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

We want $\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}$

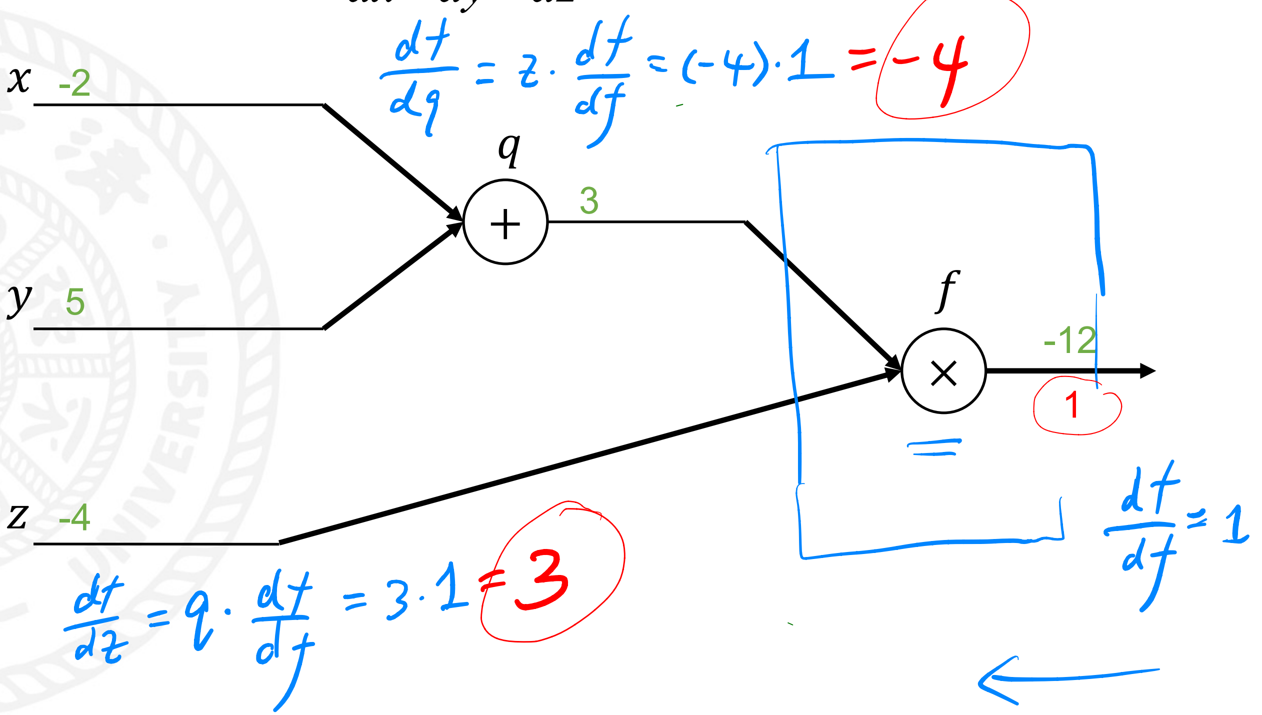


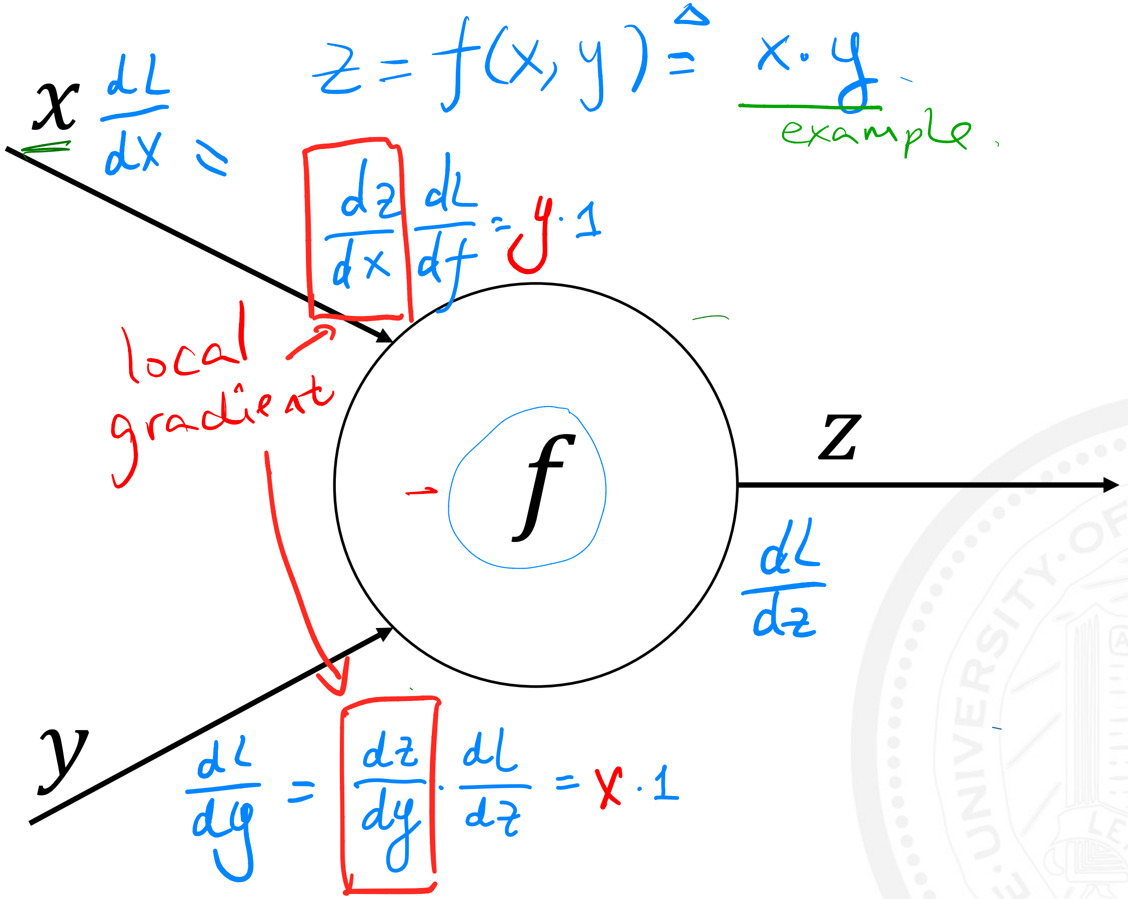


Backpropagation

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

We want $\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}$

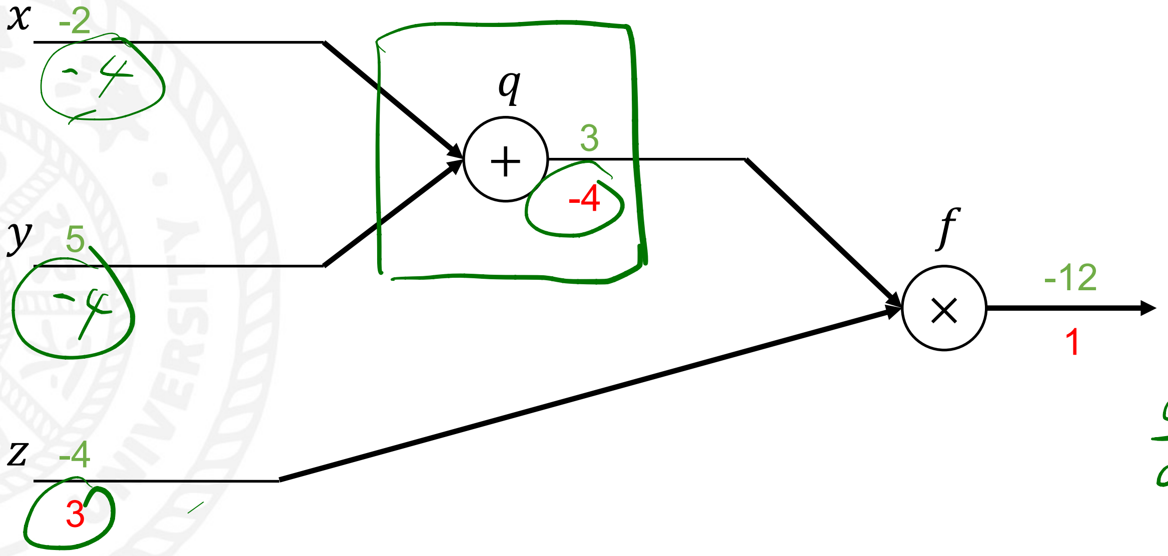


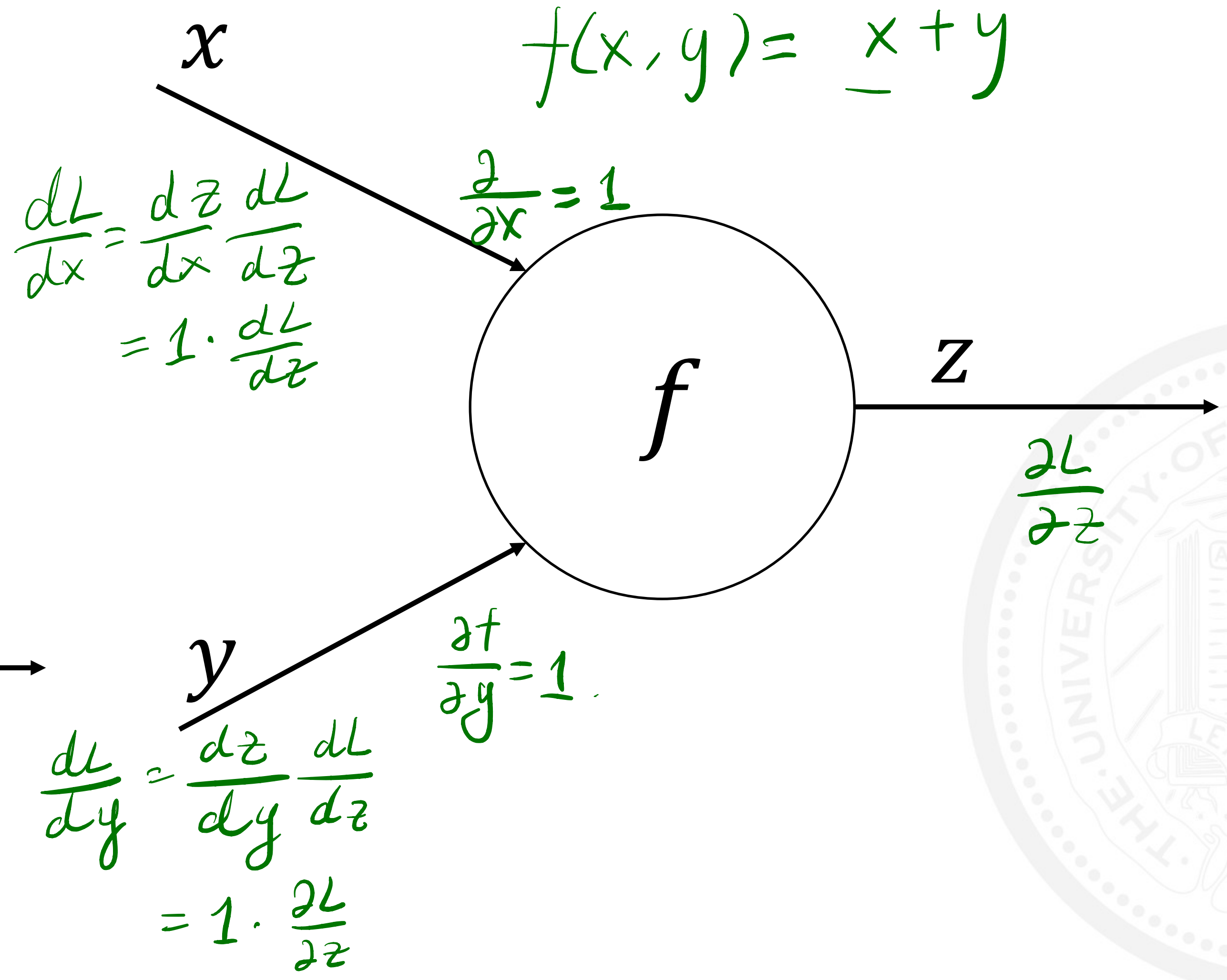


Backpropagation

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

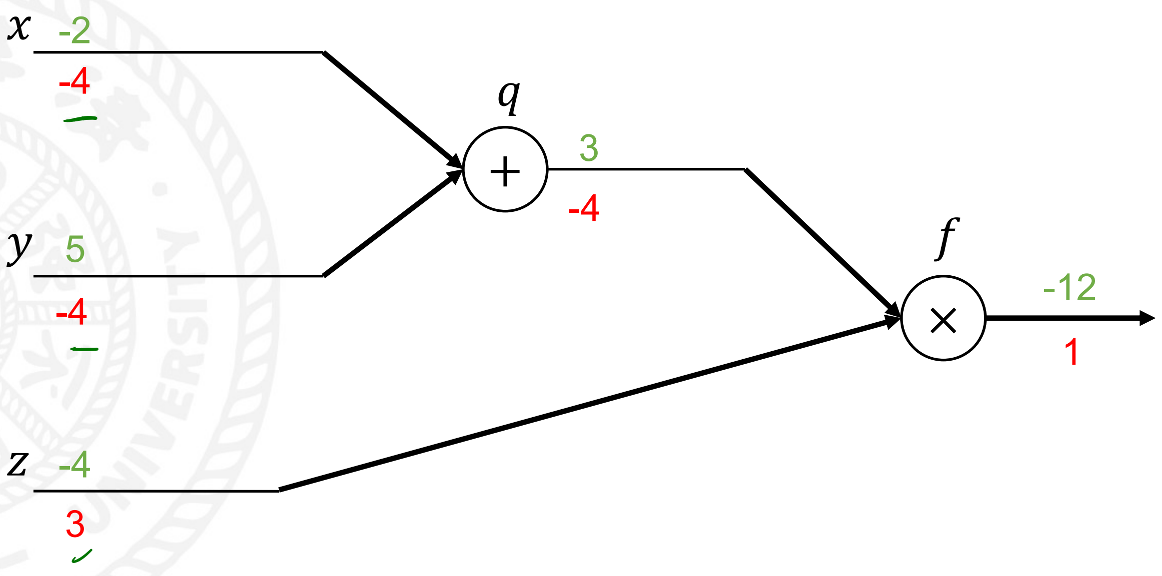
We want $\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}$

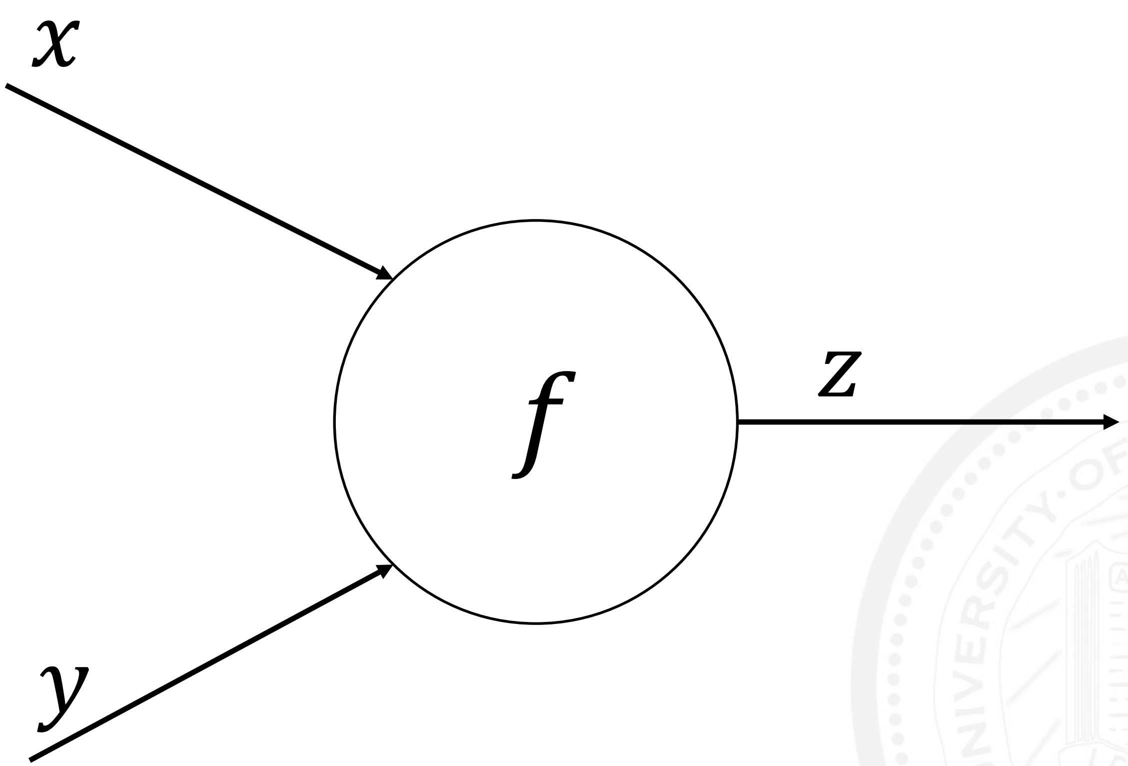




$$f(x,y,z)=(x+y)z$$

We want $\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}$

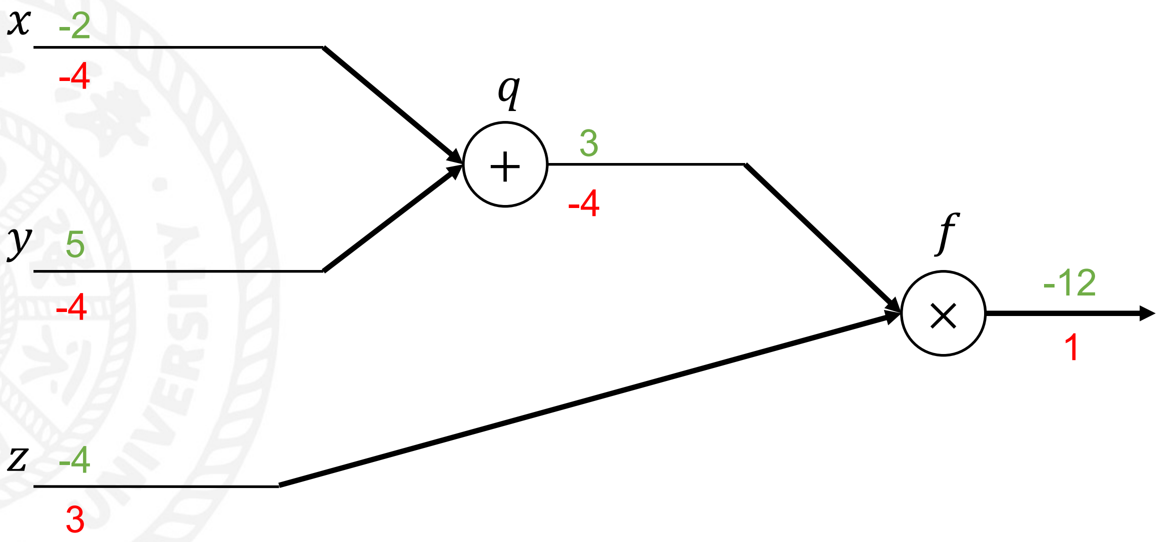


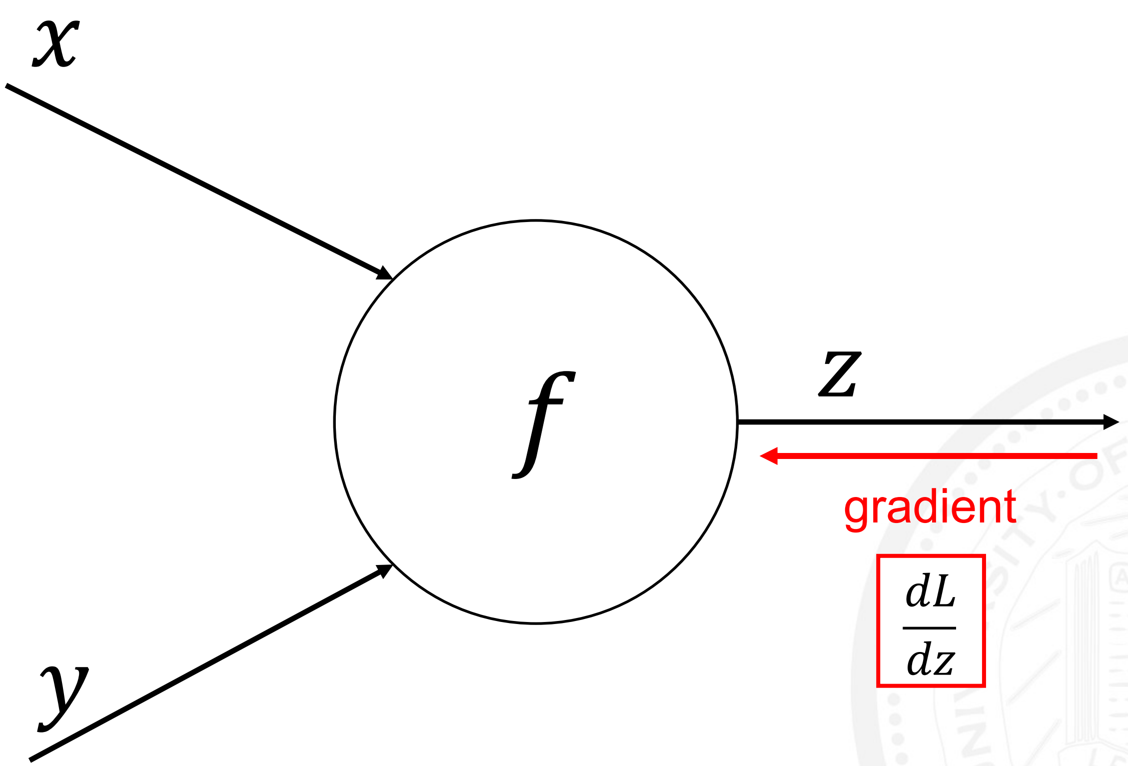


Backpropagation

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

We want $\dfrac{df}{dx}, \dfrac{df}{dy}, \dfrac{df}{dz}$

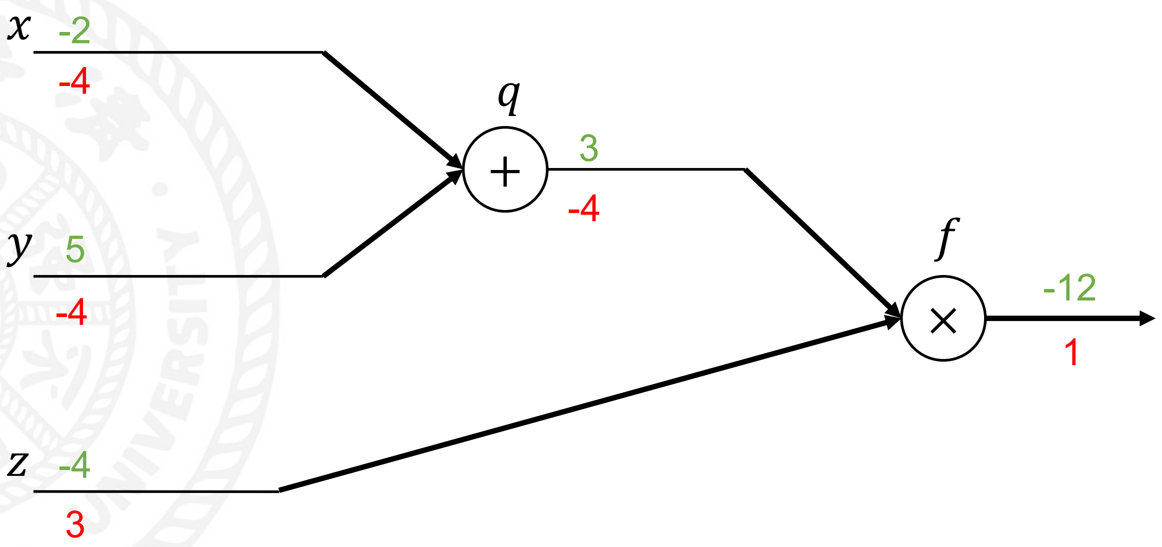


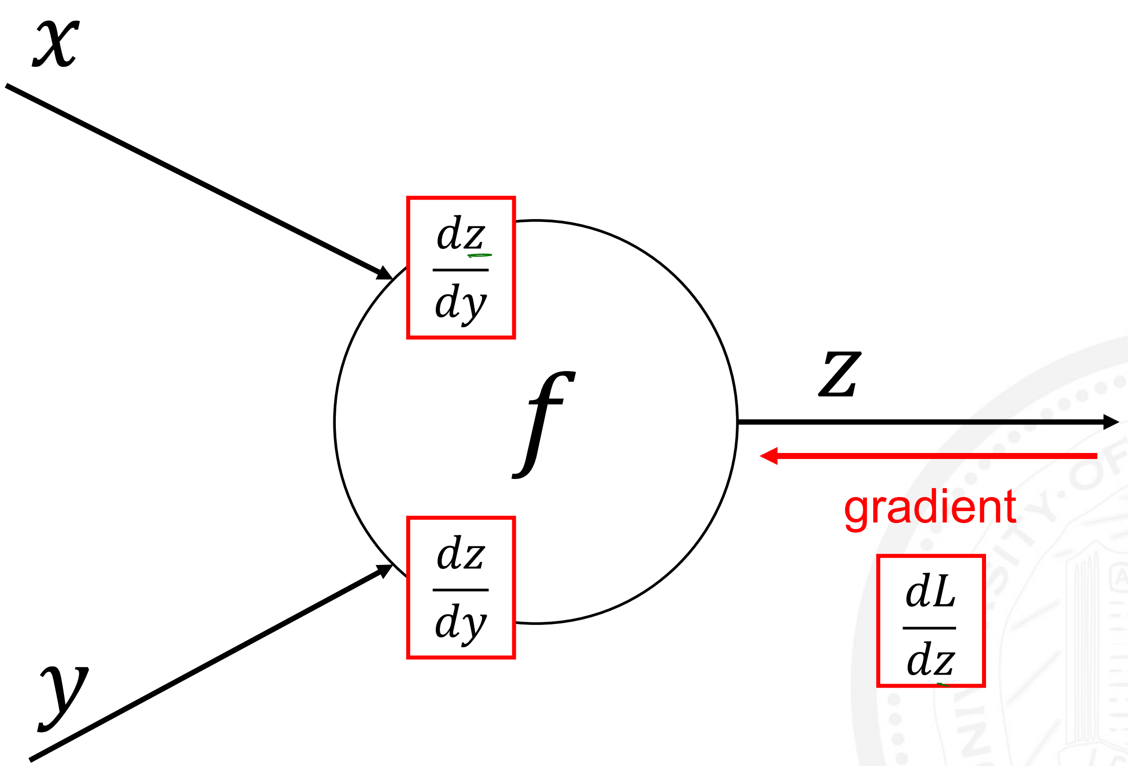


Backpropagation

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

We want $\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}$

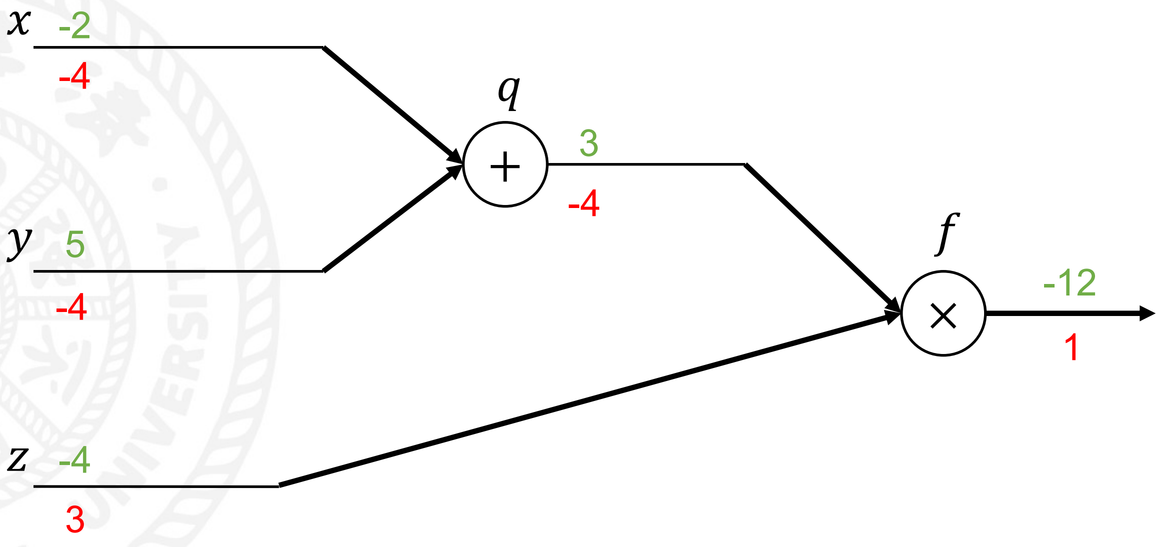


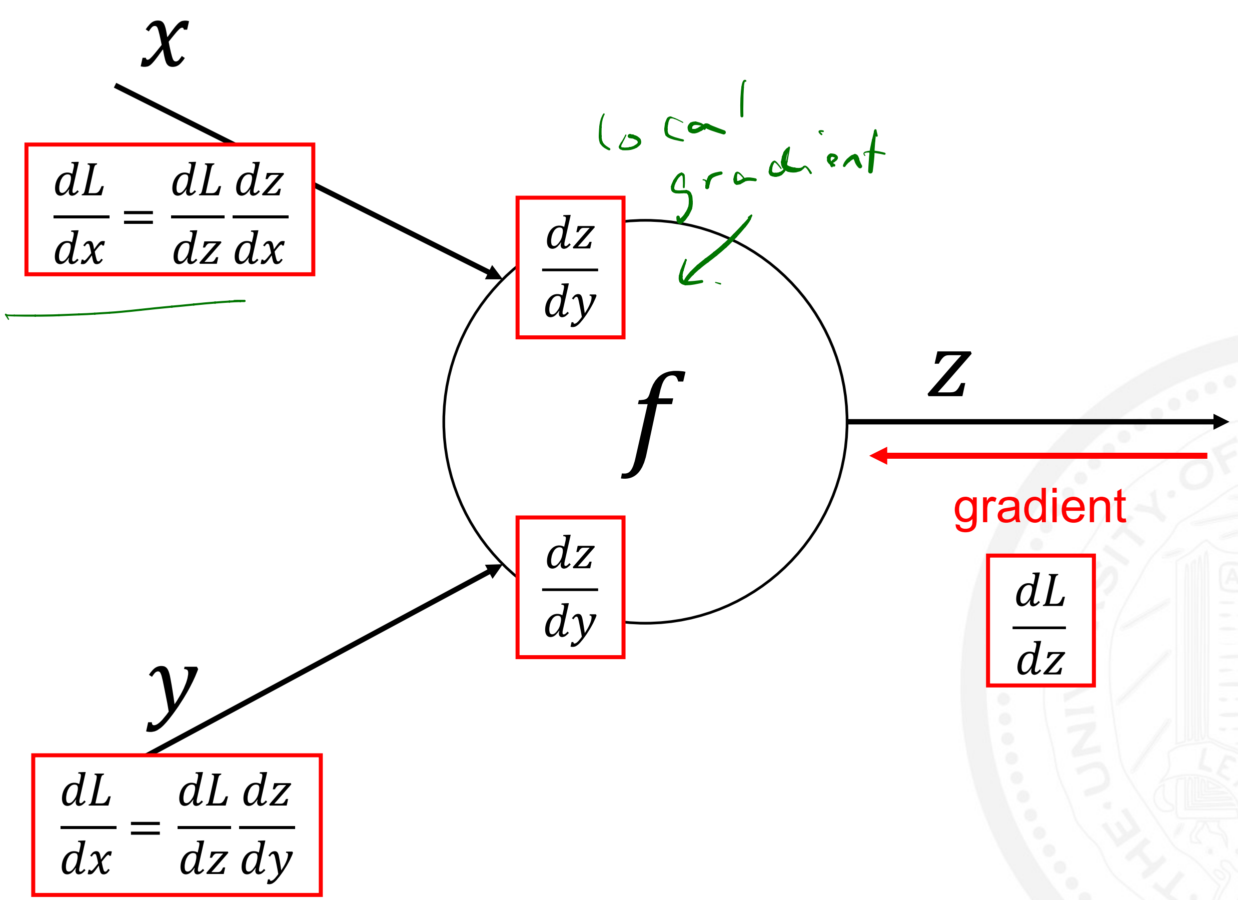


Backpropagation

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

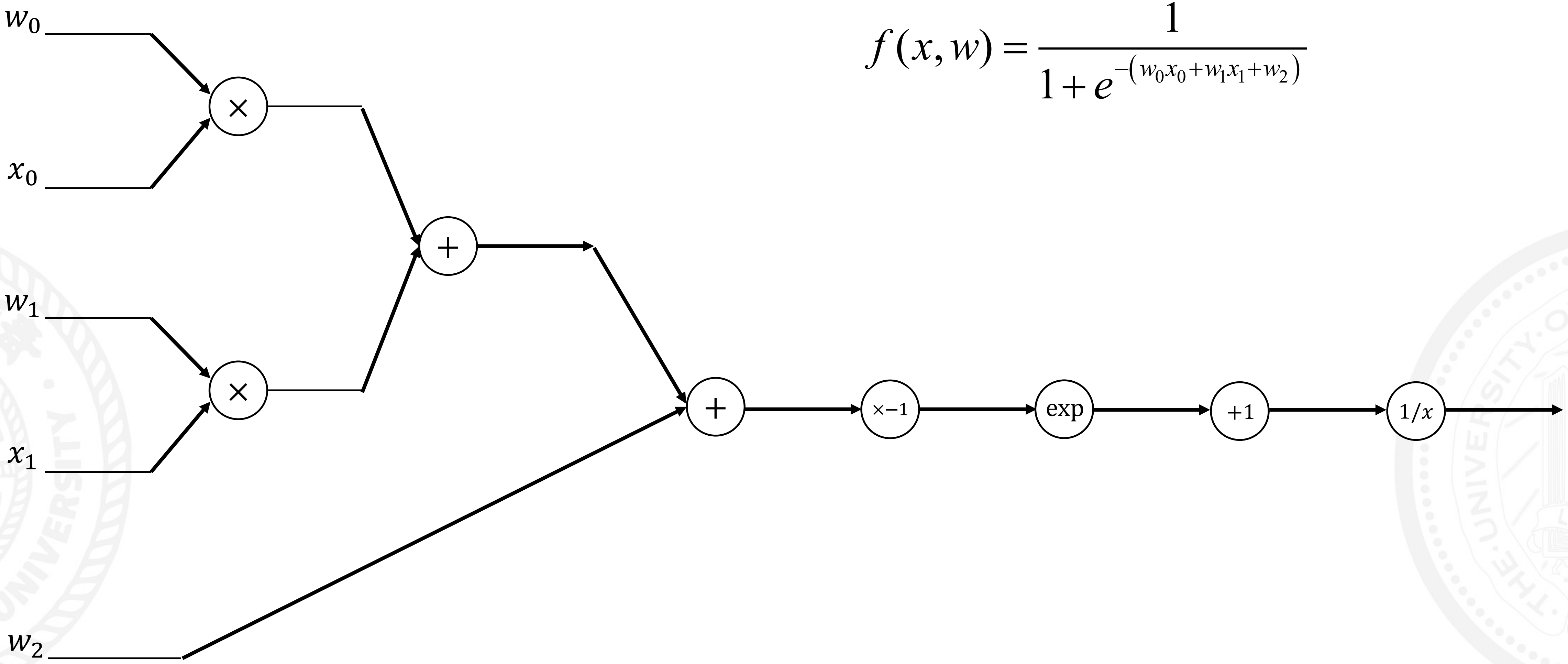
We want $\dfrac{df}{dx}, \dfrac{df}{dy}, \dfrac{df}{dz}$





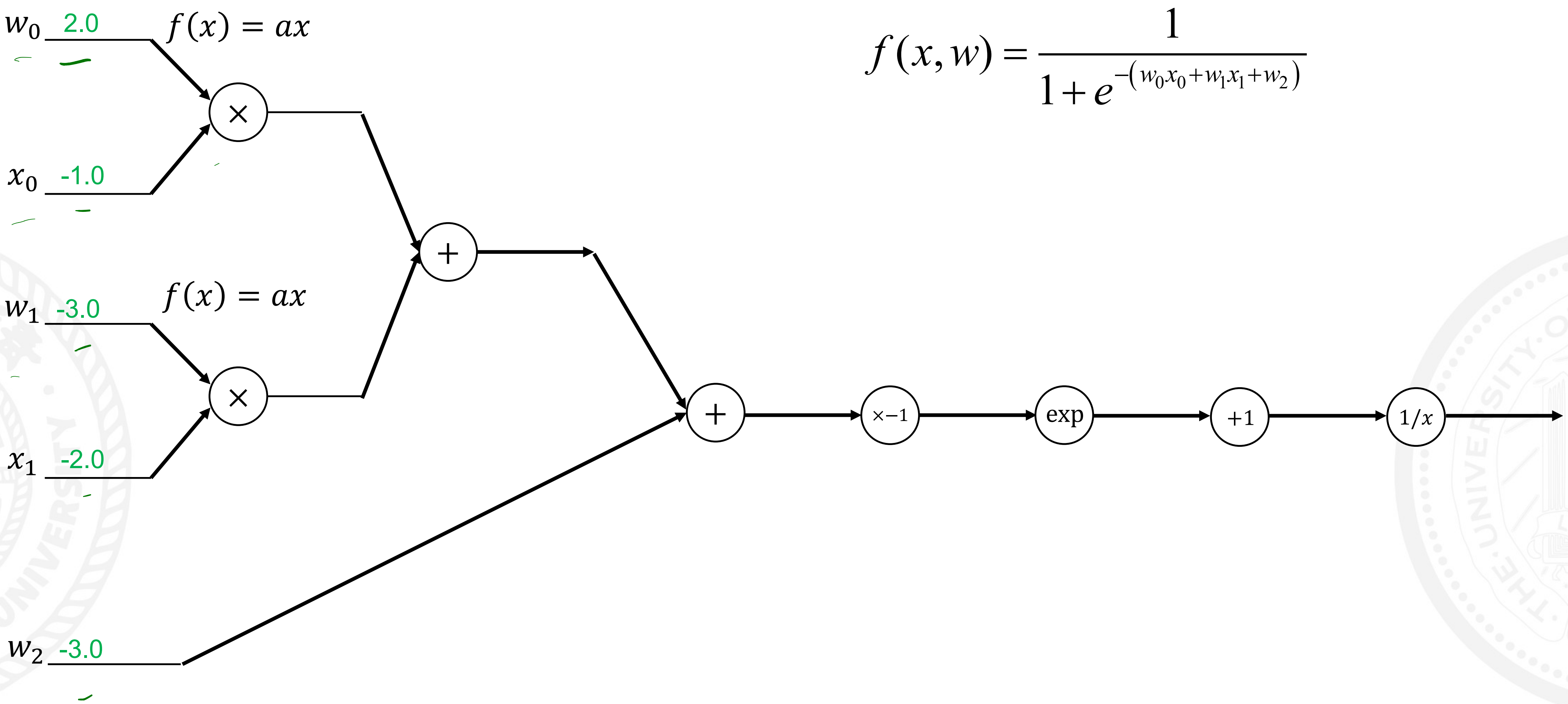
Backpropagation

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



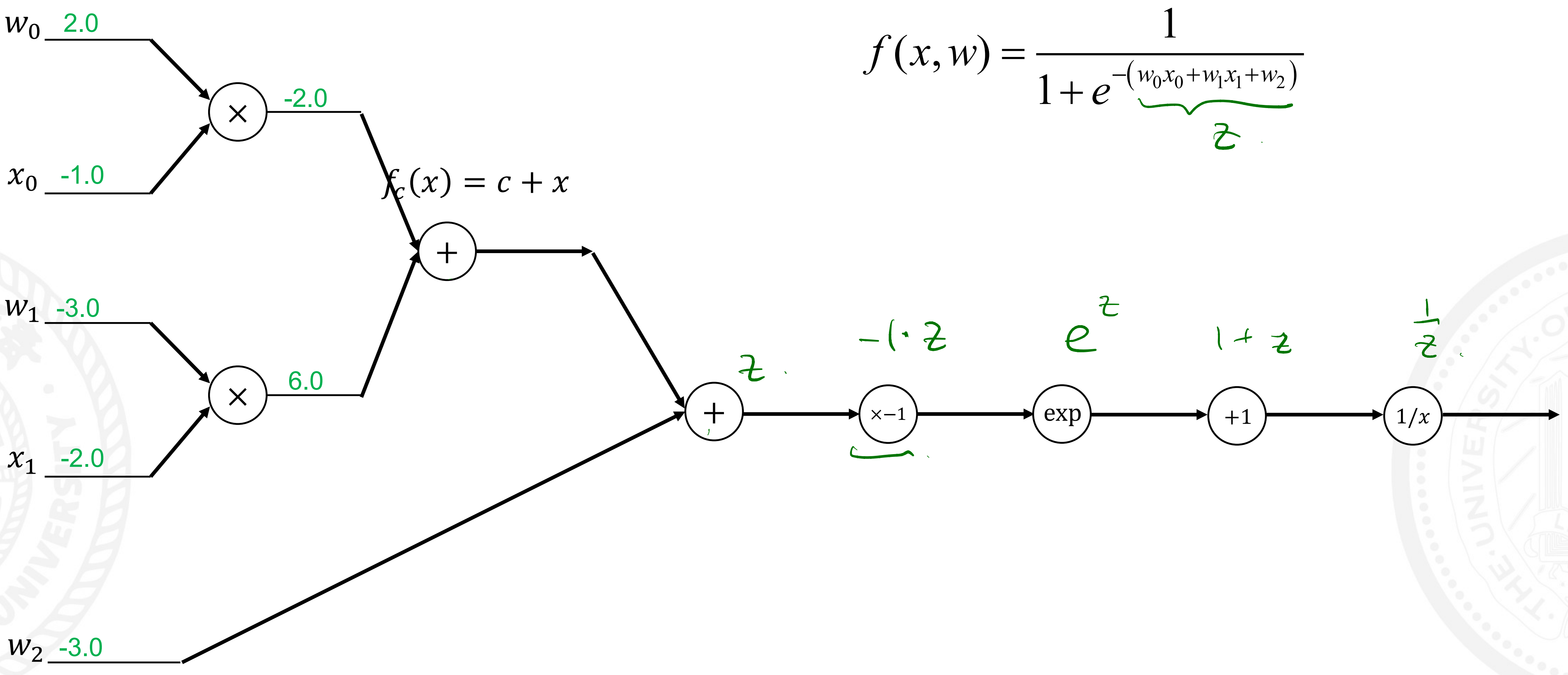
Backpropagation

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



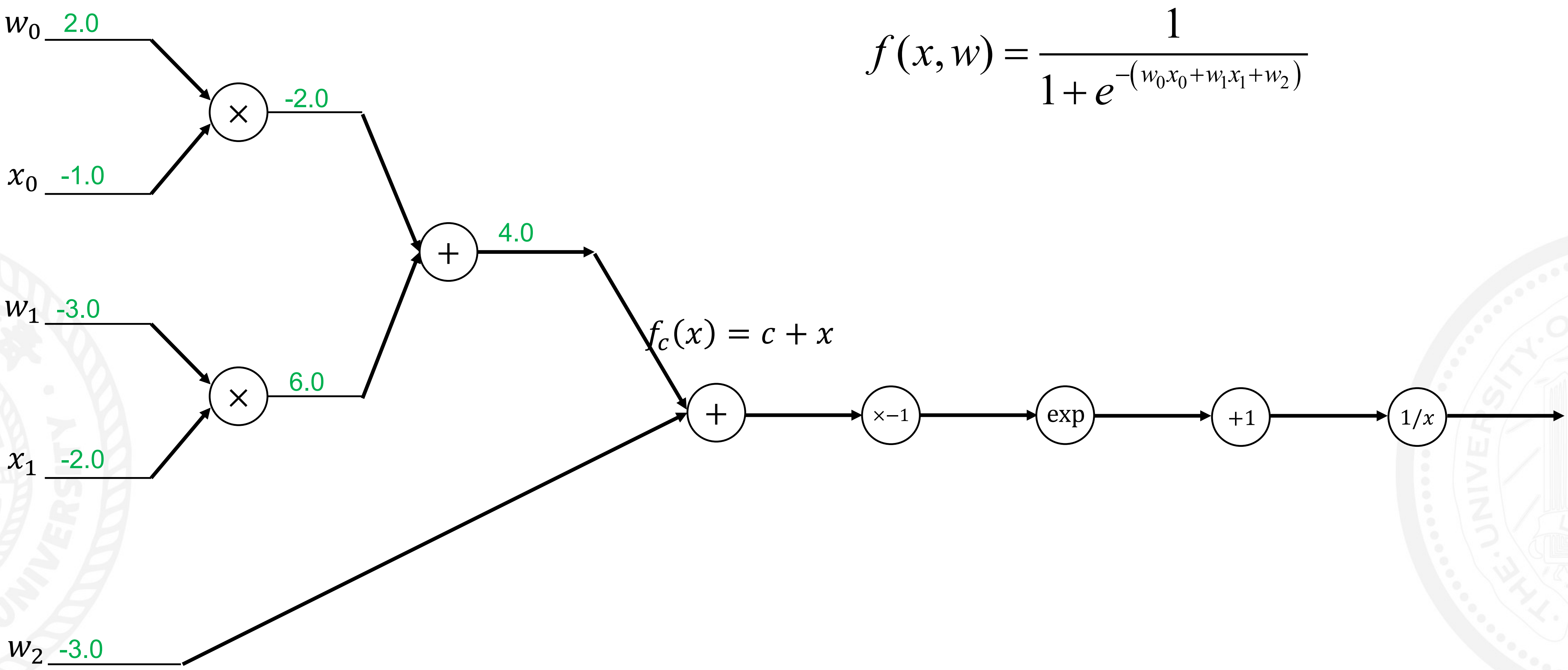
Backpropagation

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



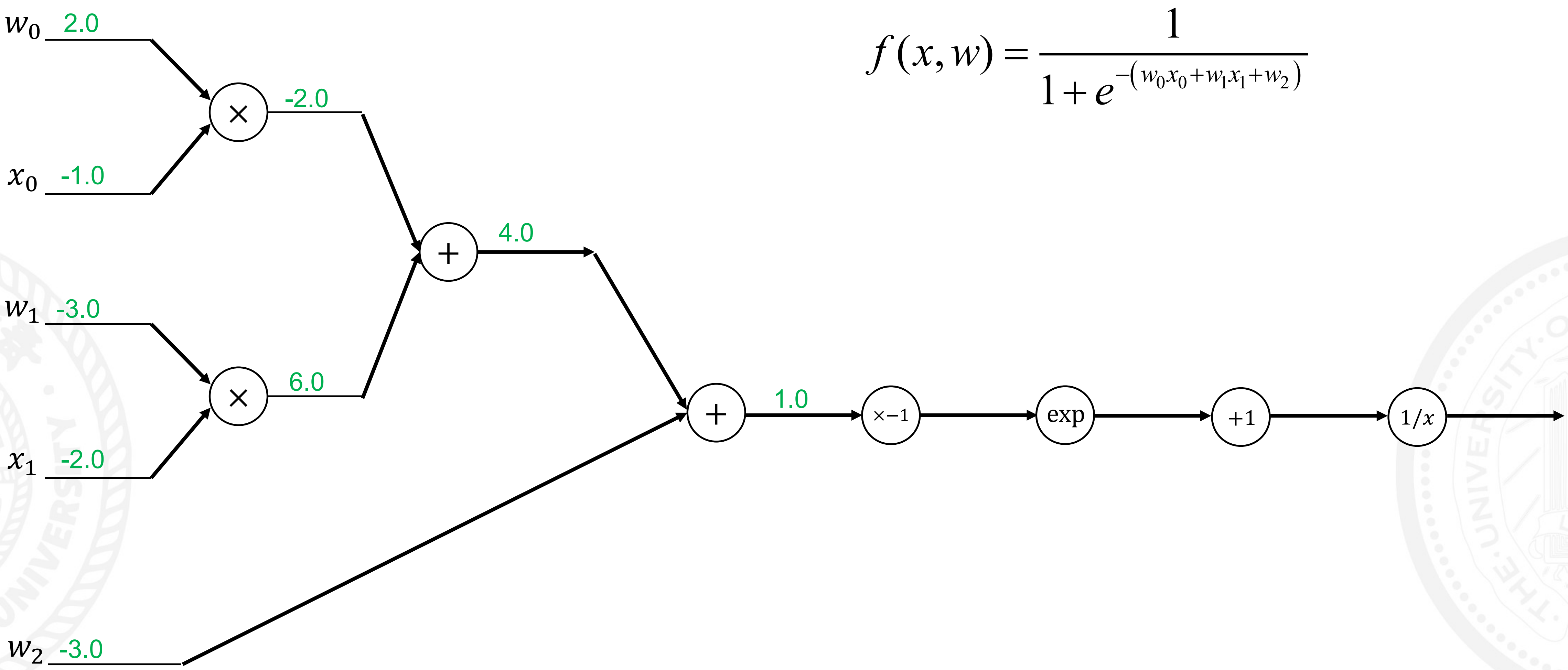
Backpropagation

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



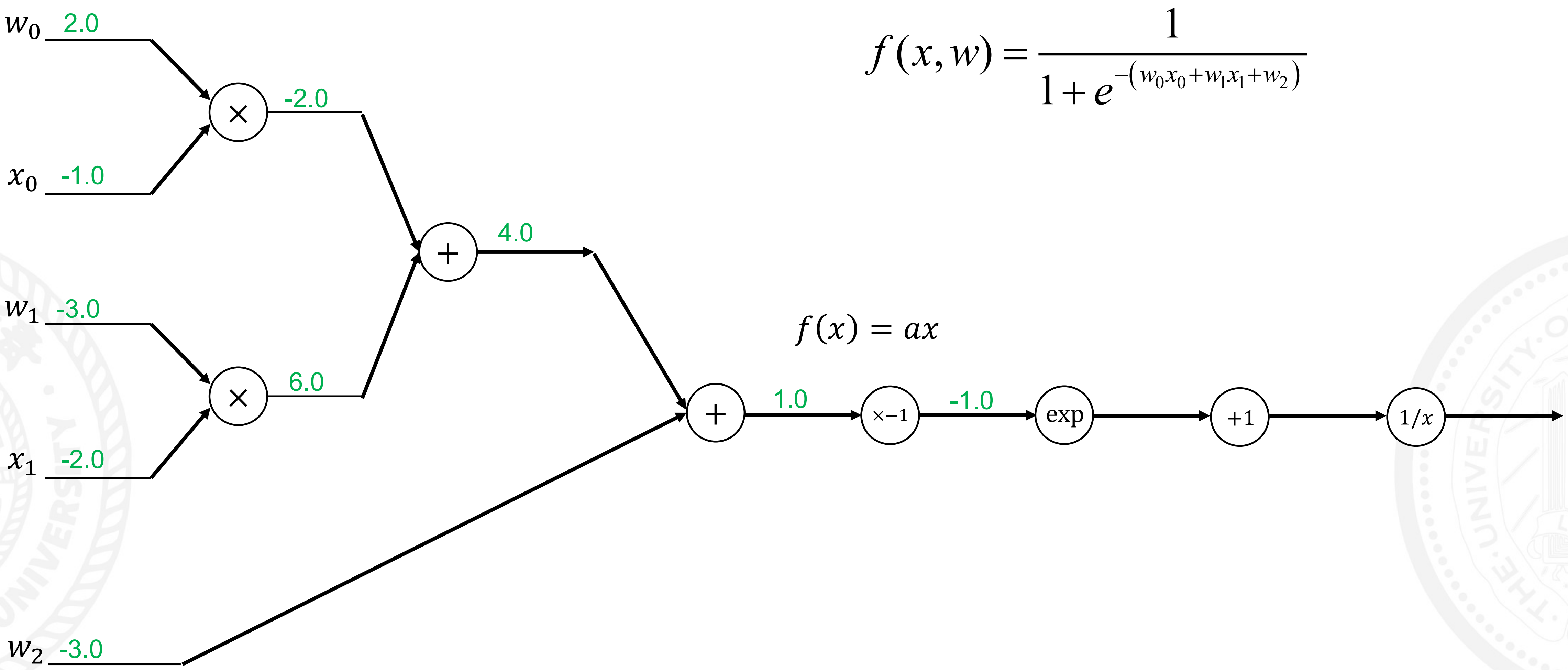
Backpropagation

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



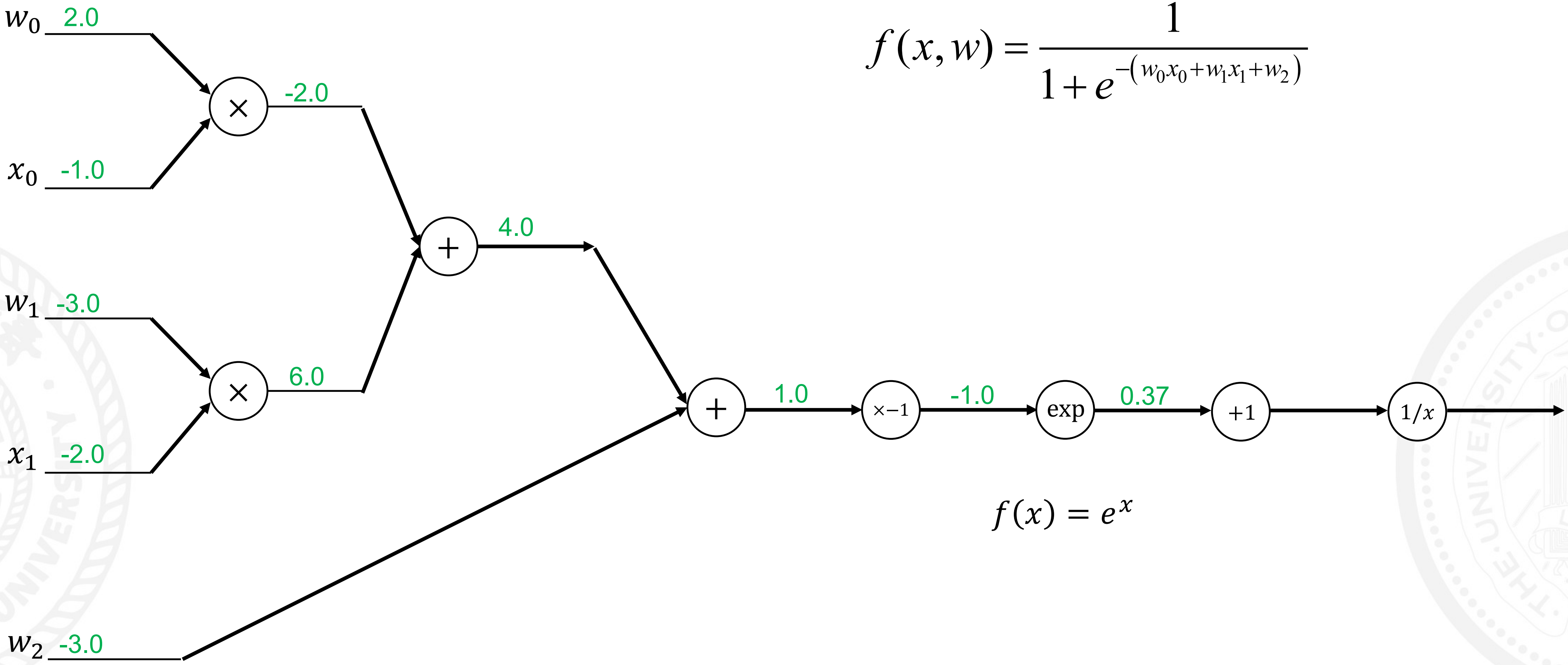
Backpropagation

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



Backpropagation

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

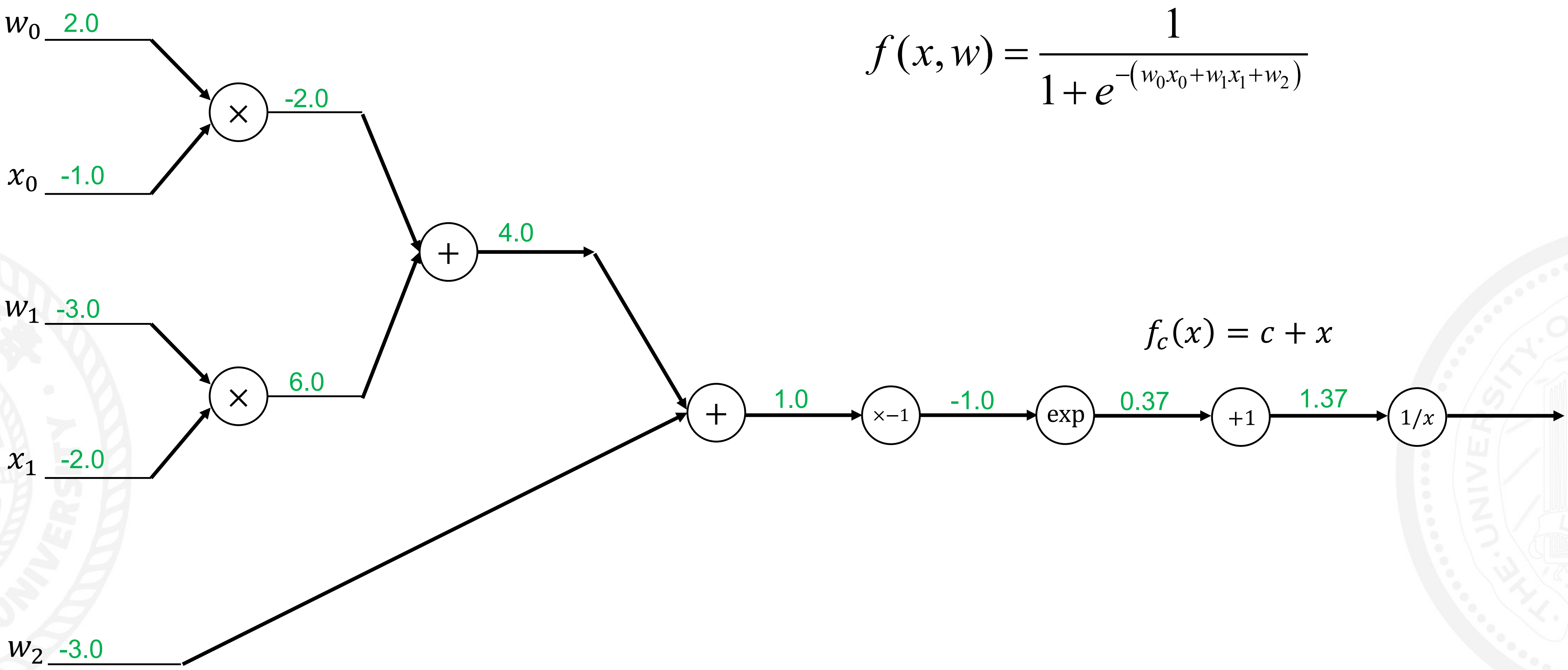


$$f(x) = e^x$$

Backpropagation

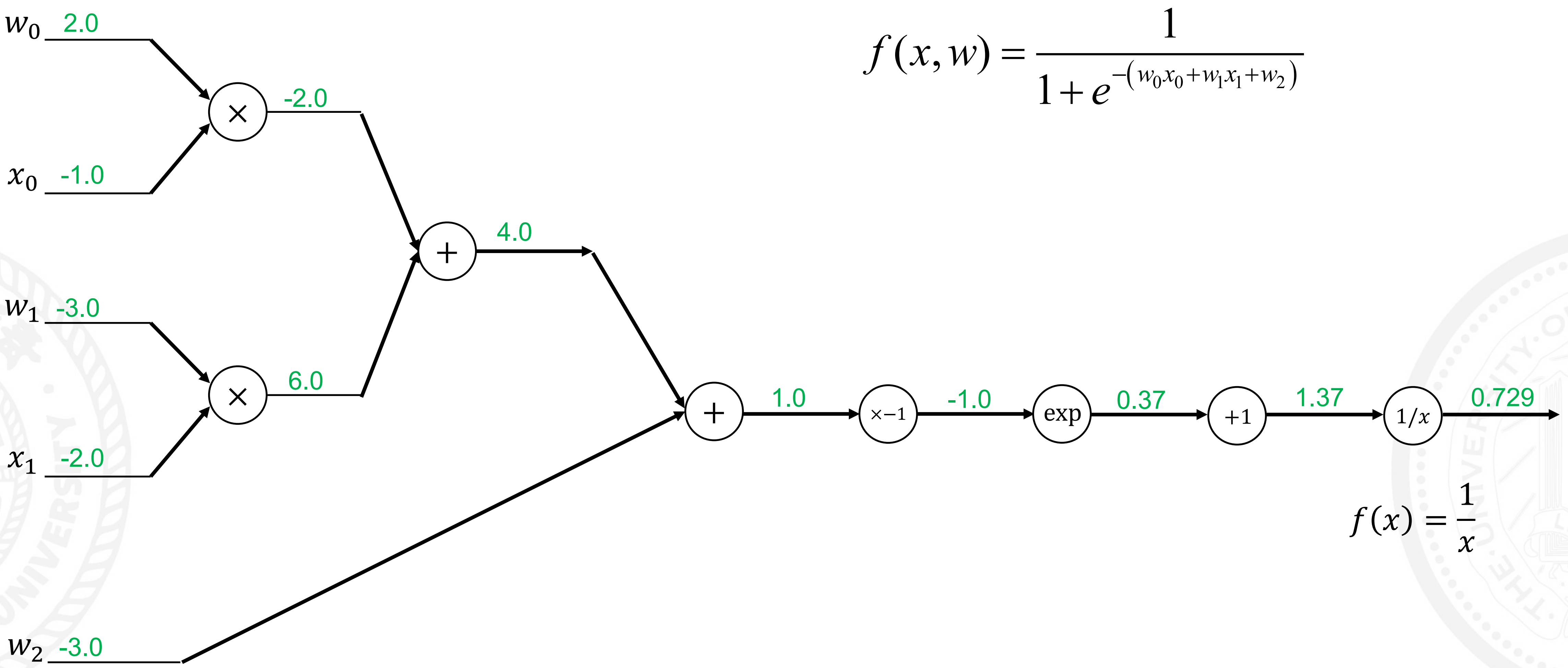
$$f(x,w) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$

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



Backpropagation

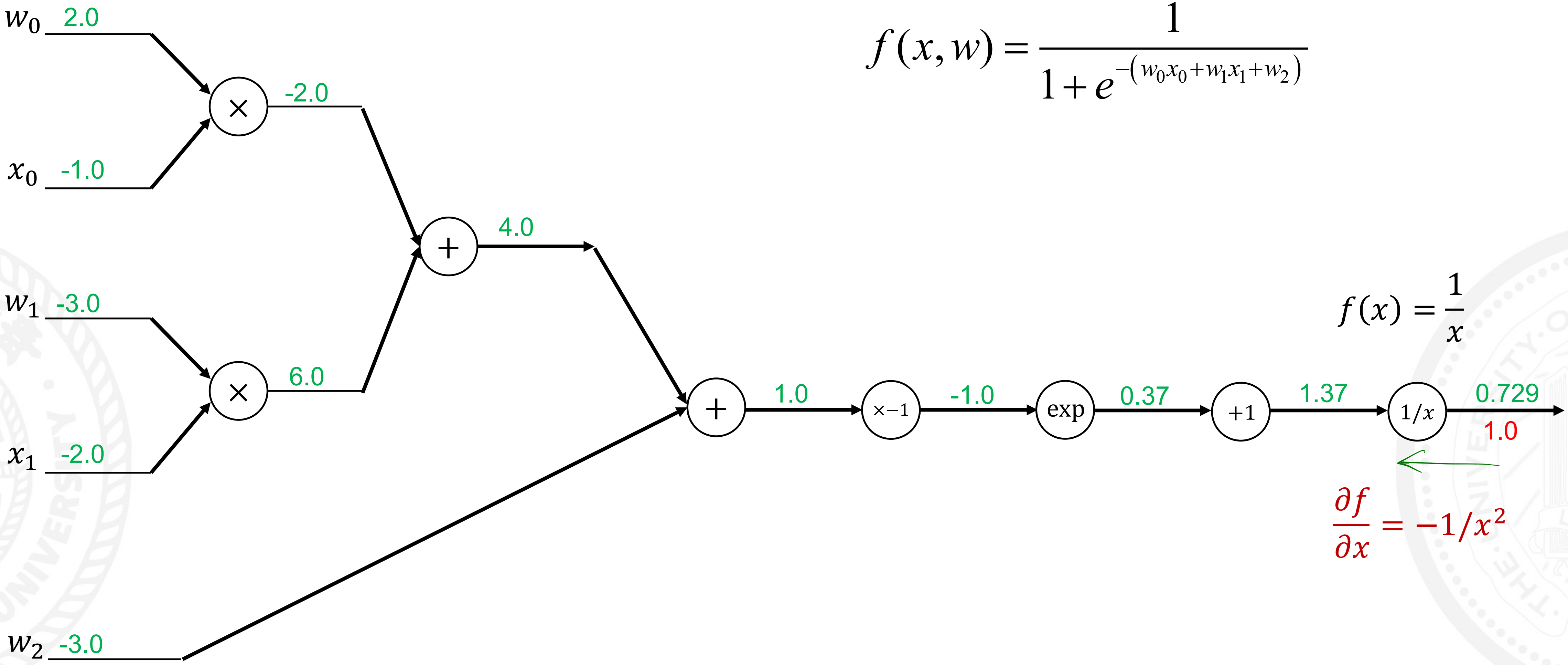
$$f(x,w)=\frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



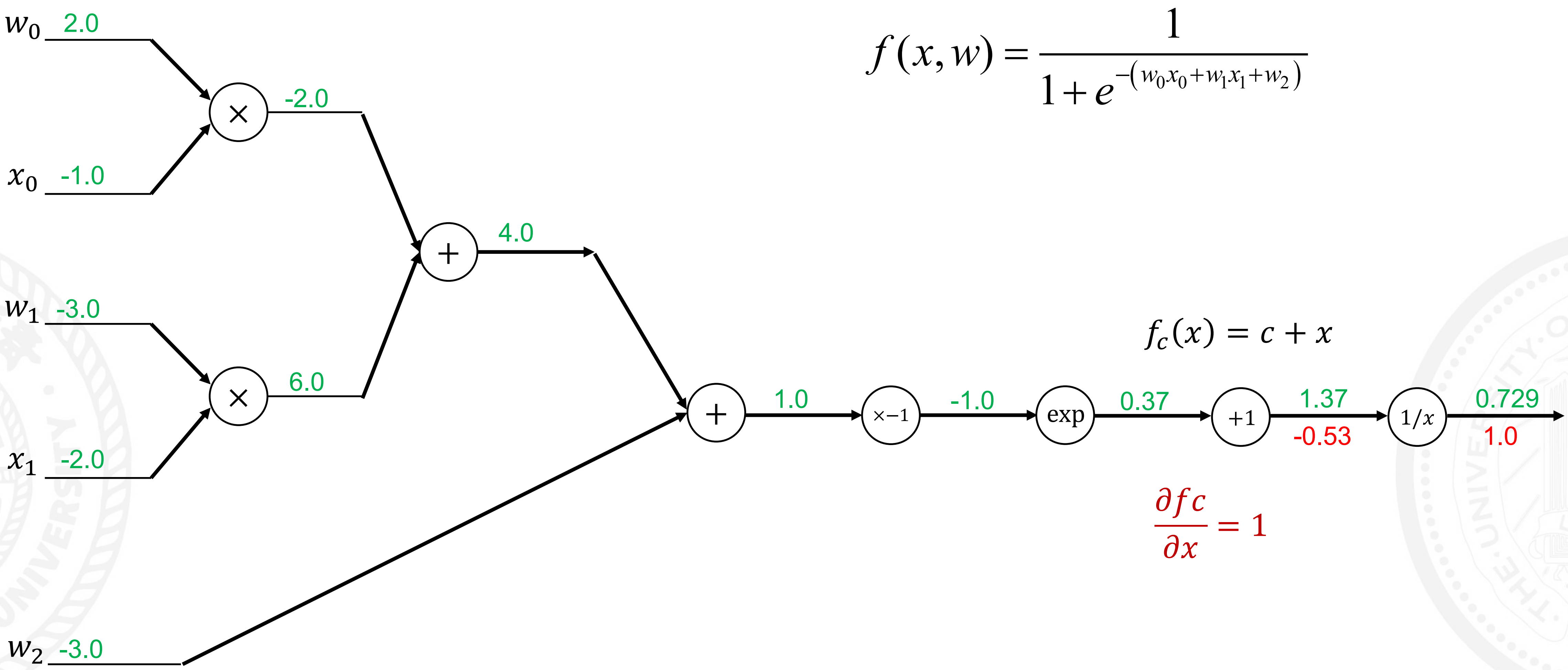
$$f(x)=\frac{1}{x}$$

Backpropagation

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



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

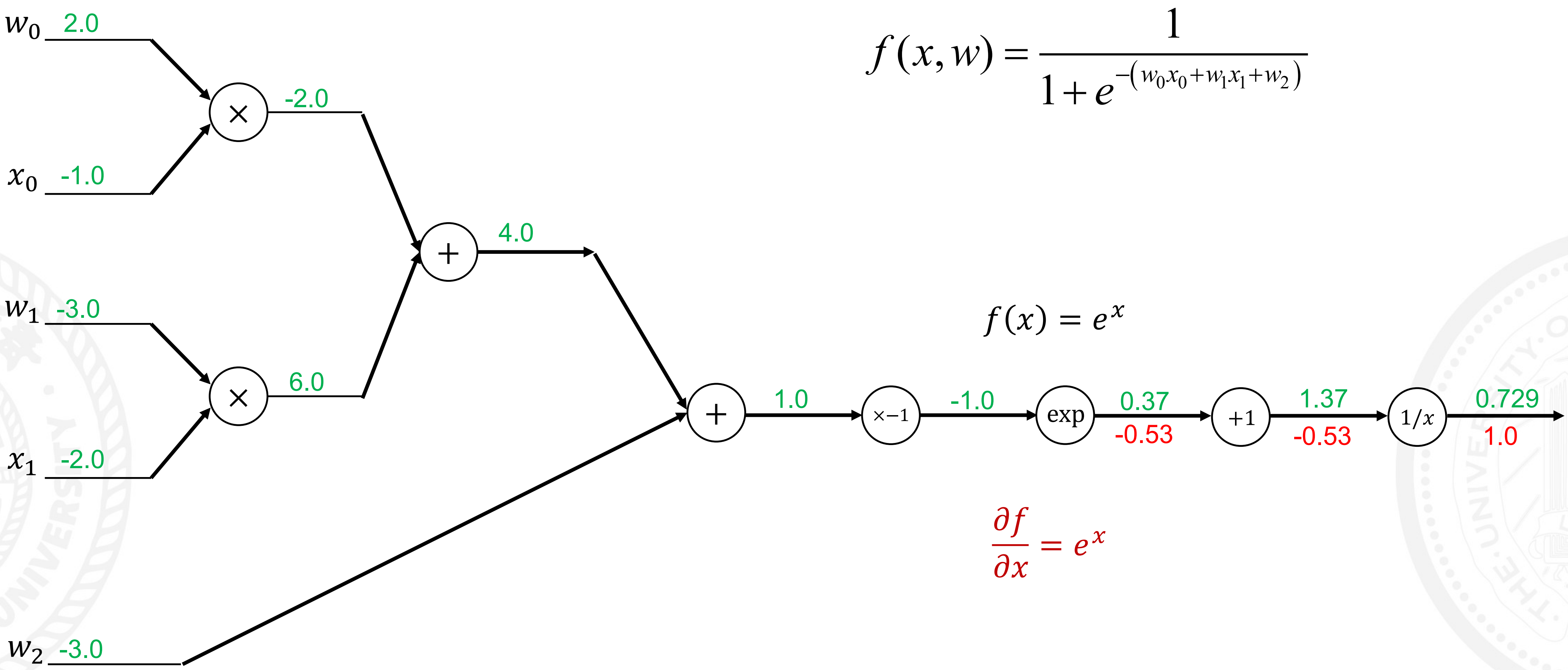


$$f_c(x) = c + x$$

$$\frac{\partial fc}{\partial x} = 1$$

Backpropagation

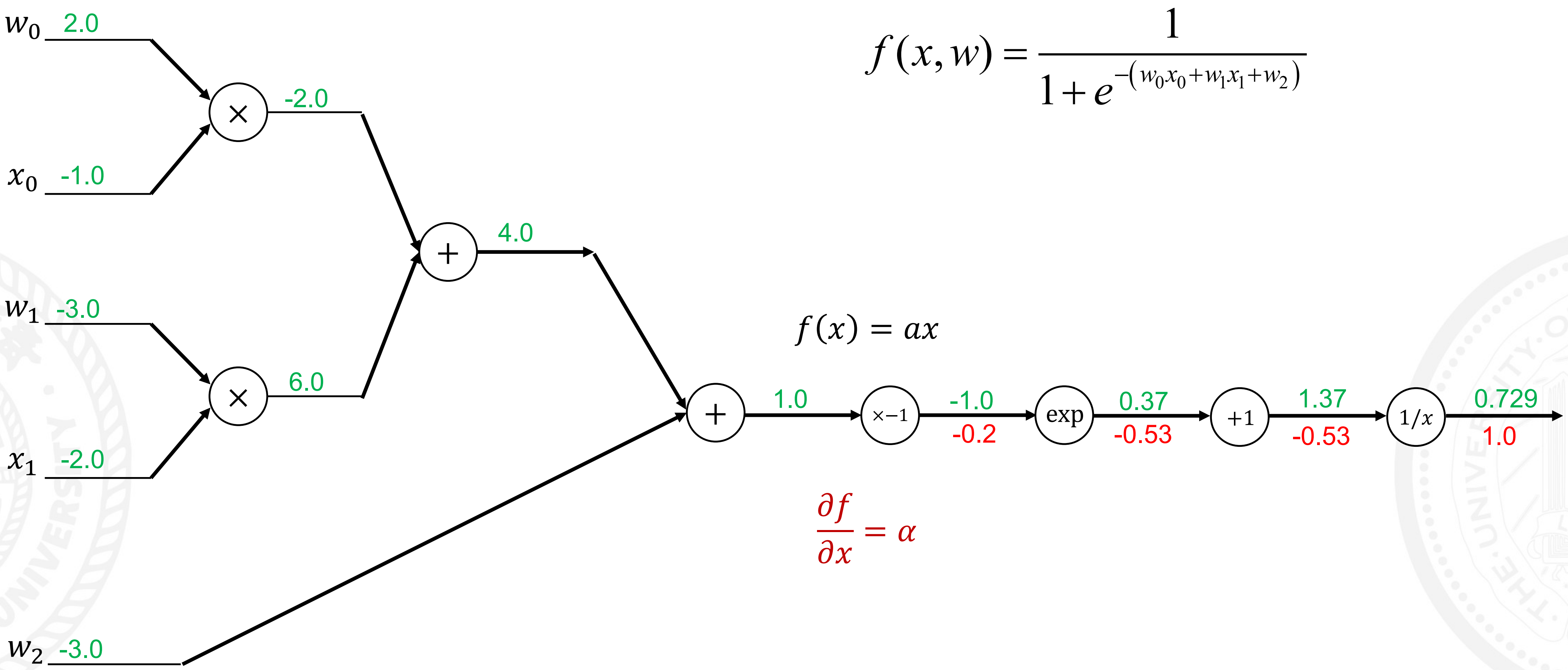
$$f(x,w) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



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

$$\frac{\partial f}{\partial x} = e^x$$

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

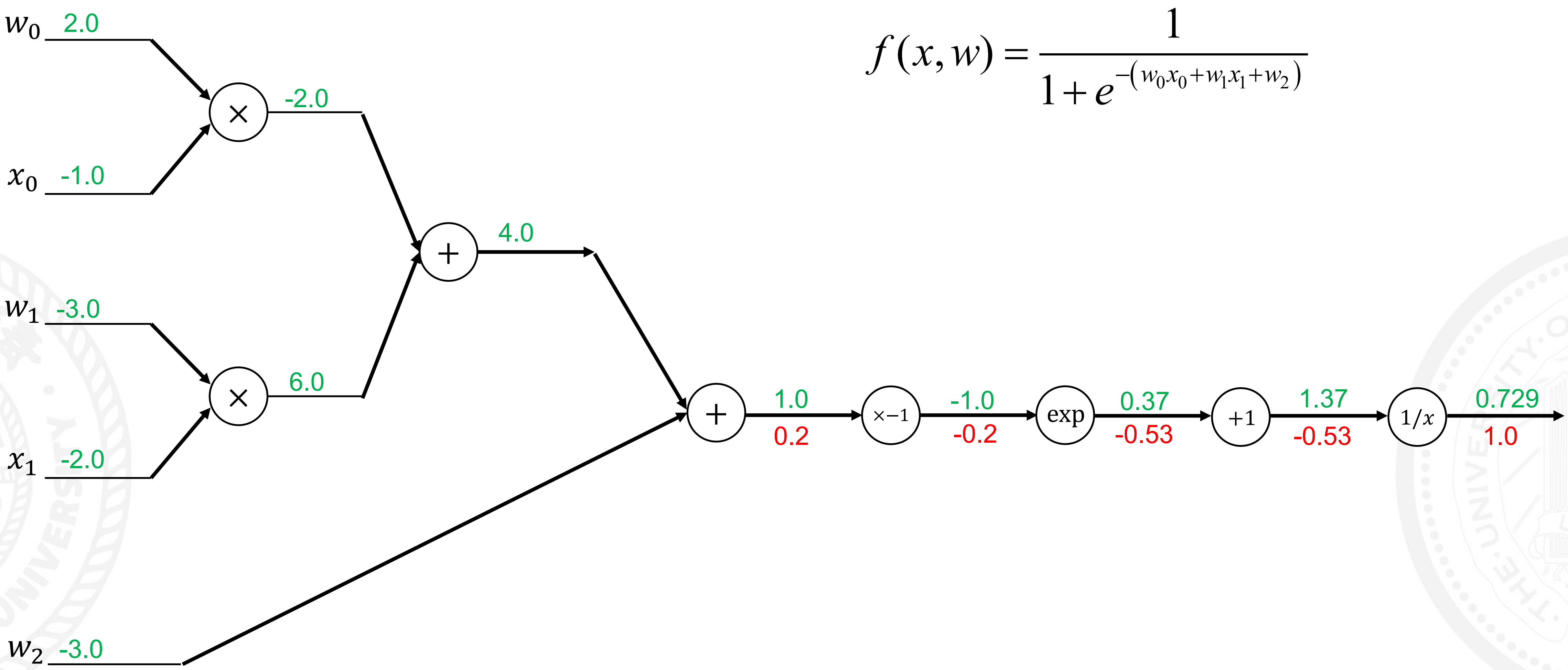


$f(x) = ax$

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

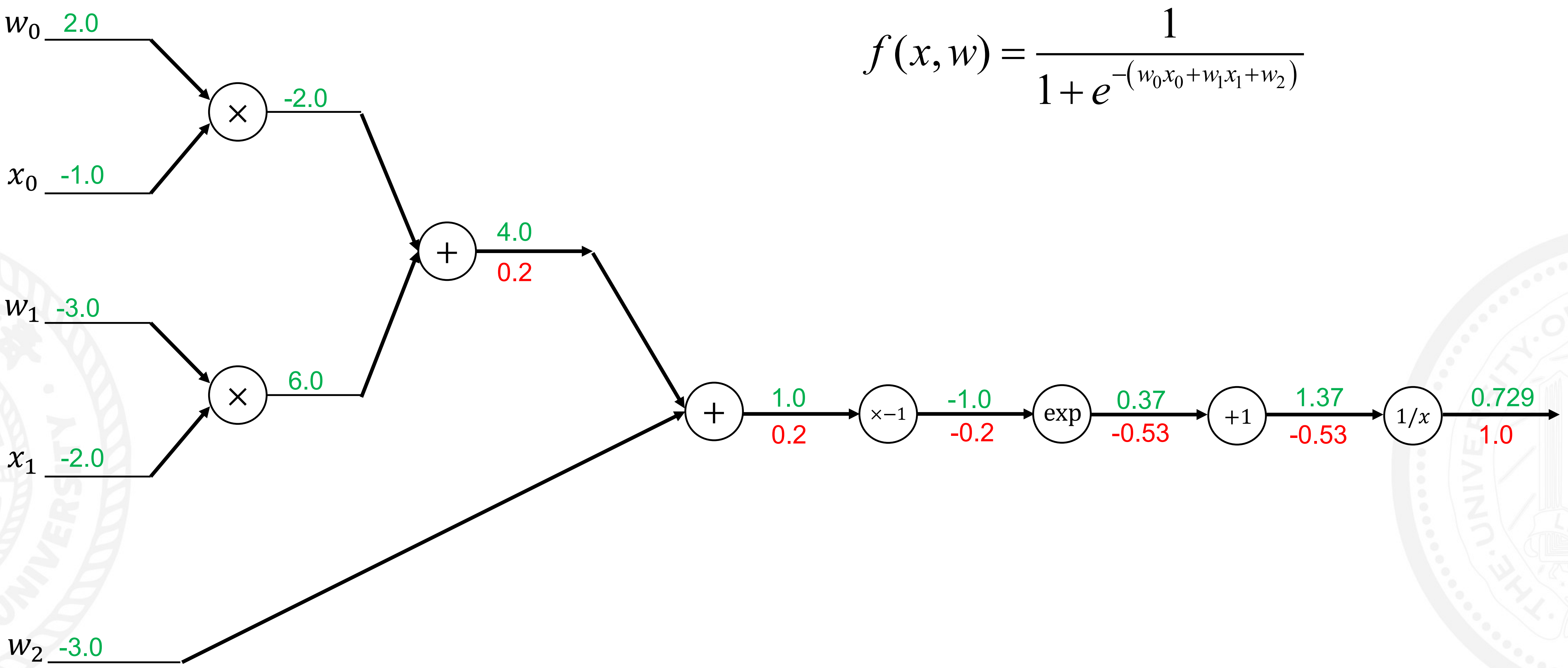
Backpropagation

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



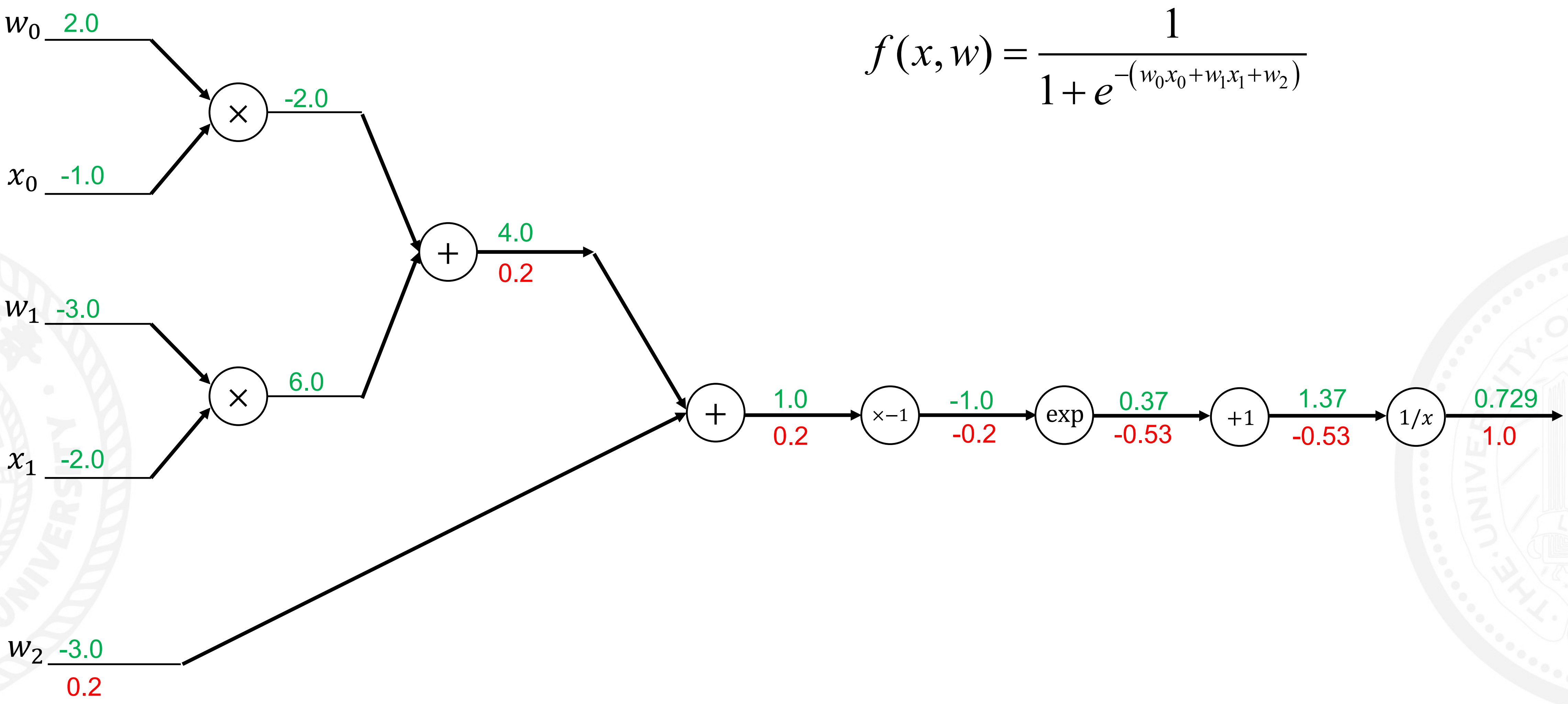
Backpropagation

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



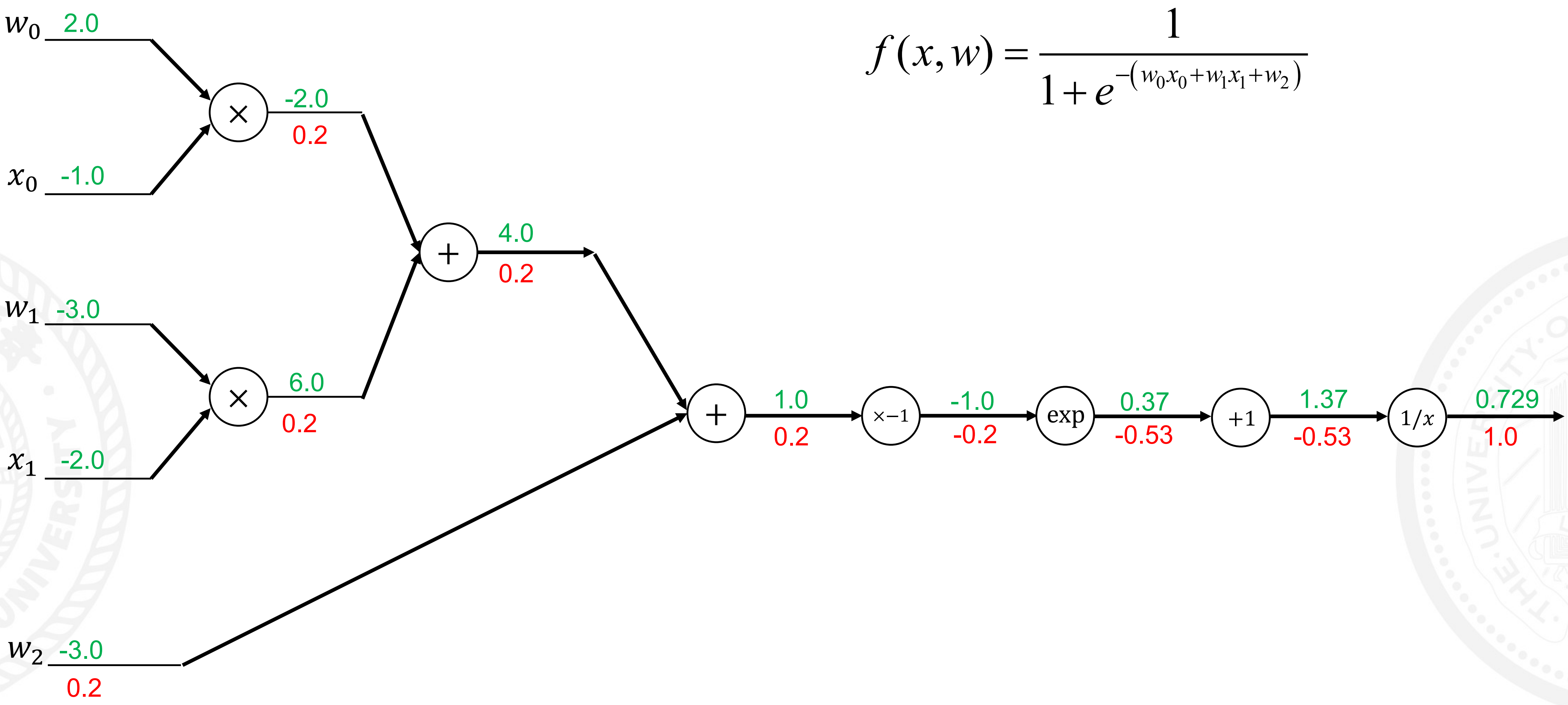
Backpropagation

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



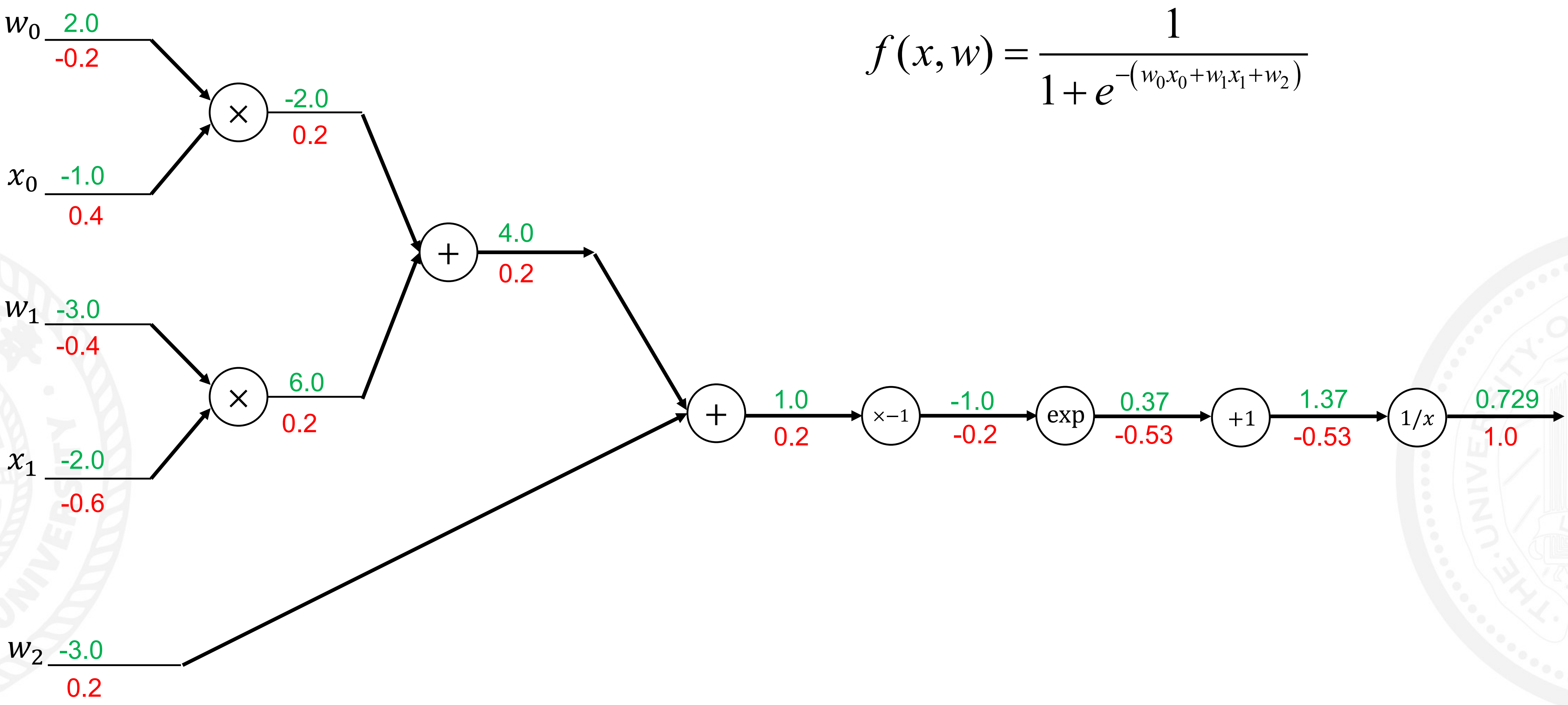
Backpropagation

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



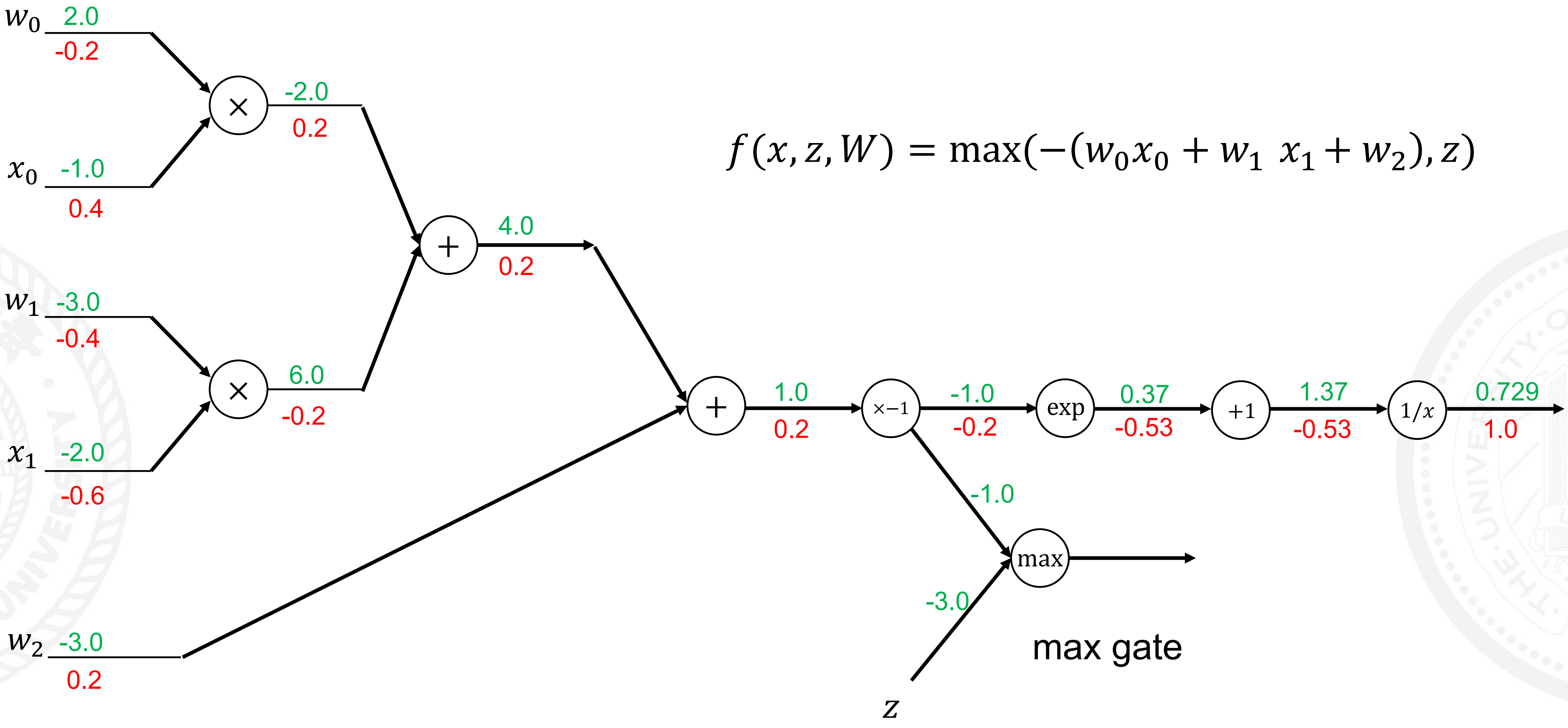
Backpropagation

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



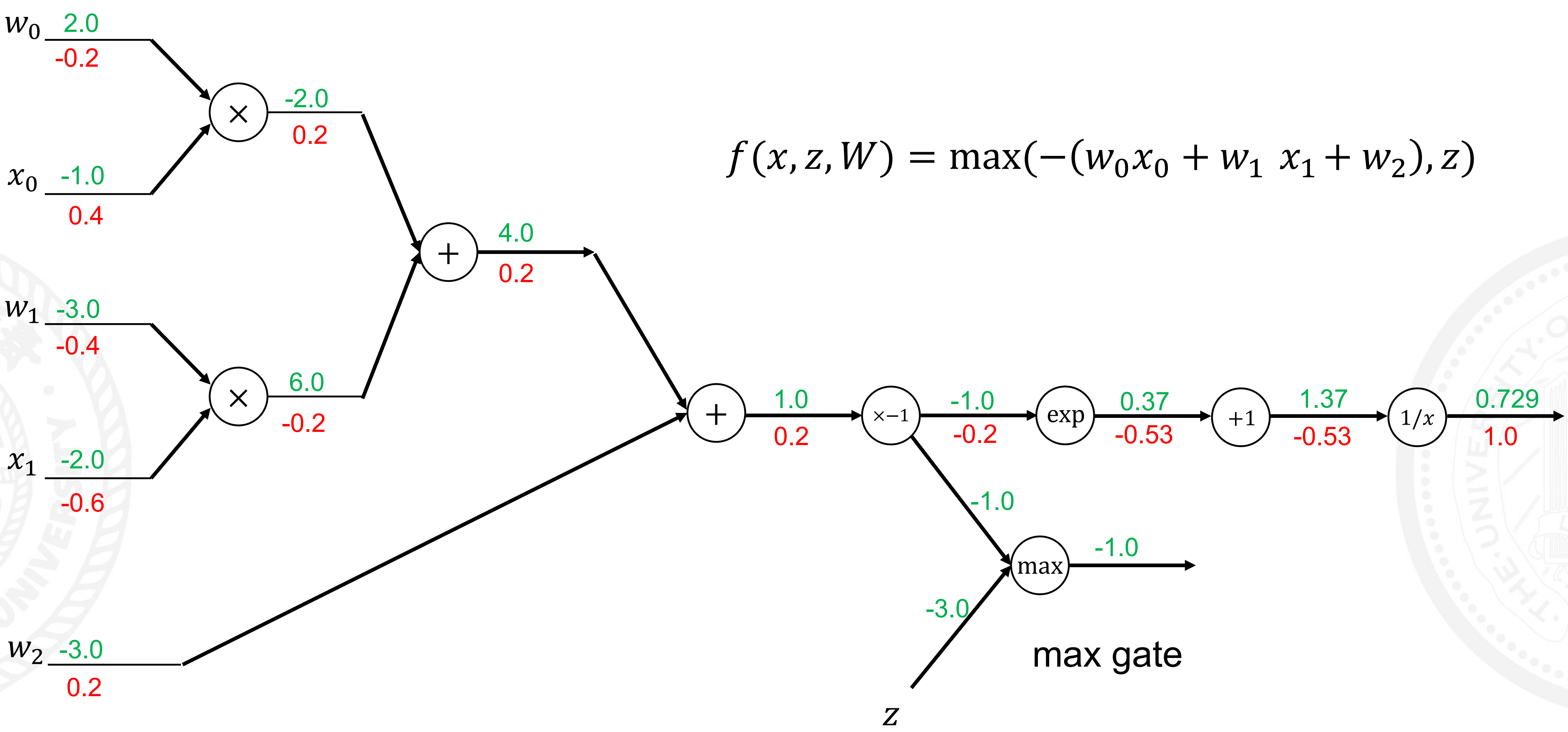
Backpropagation

$$f(x, z, W) = \max(-(w_0 x_0 + w_1\ x_1 + w_2), z)$$



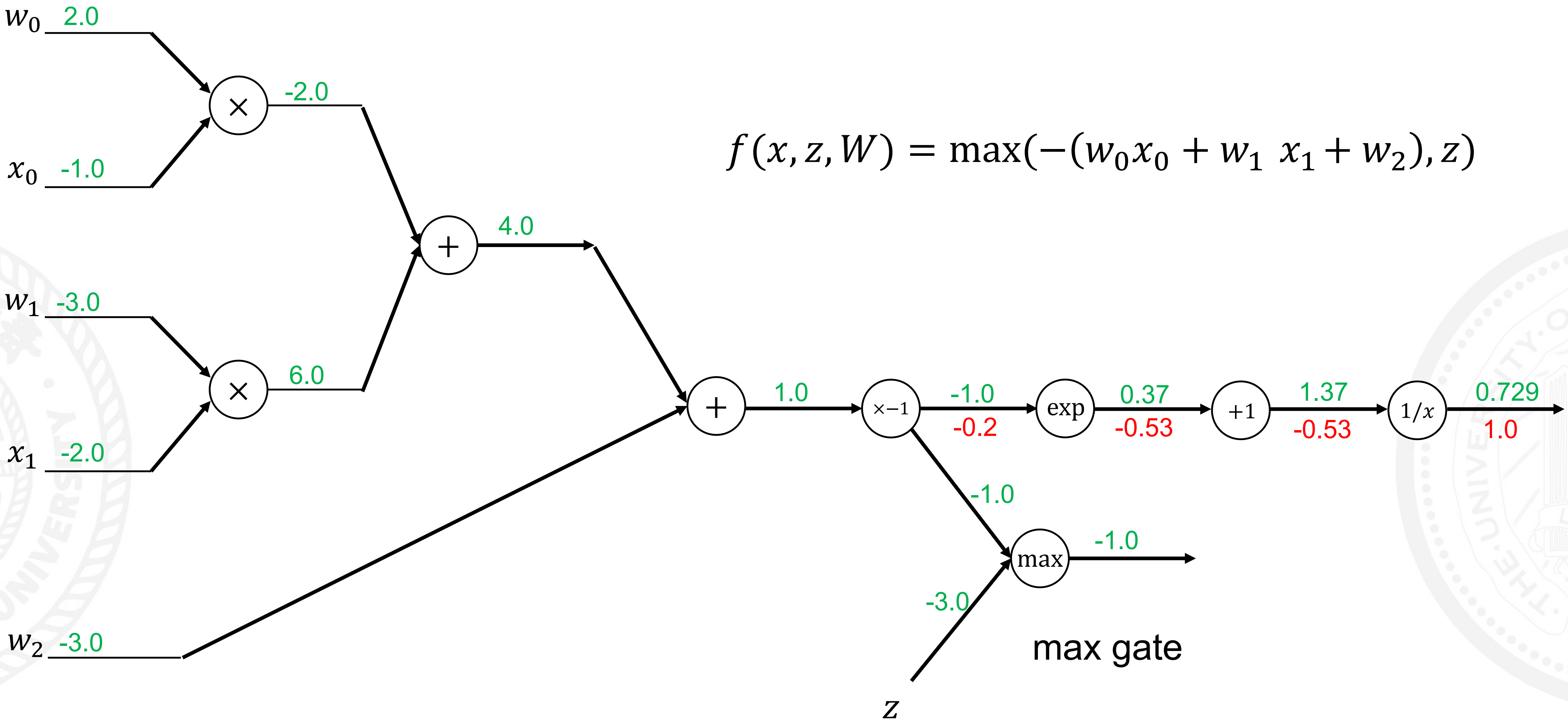
Backpropagation

$$f(x, z, W) = \max(-(w_0 x_0 + w_1\ x_1 + w_2), z)$$

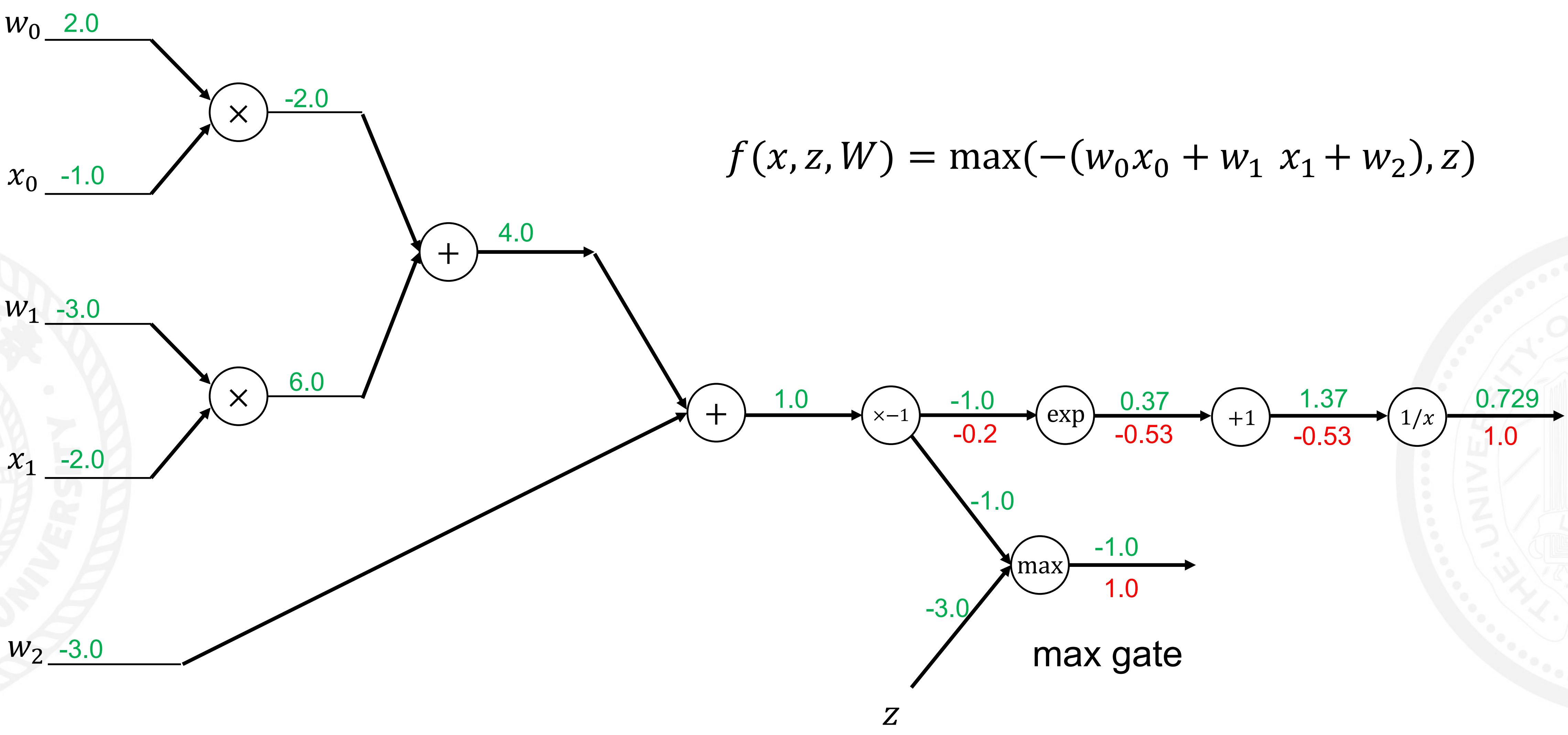


Backpropagation

$$f(x,z,W) = \max(-(w_0 x_0 + w_1\ x_1 + w_2), z)$$

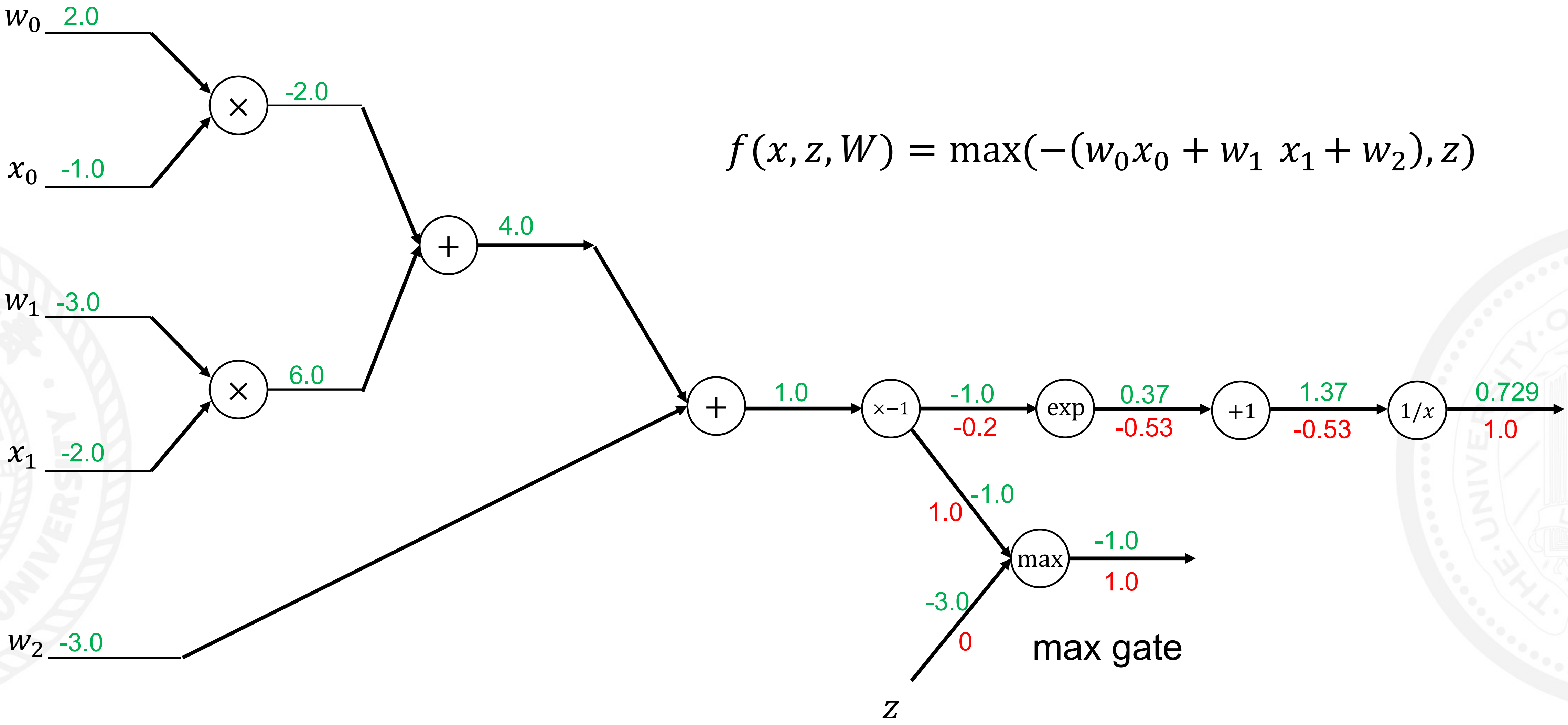


Backpropagation



$$f(x, z, W) = \max(-(w_0 x_0 + w_1 \ x_1 + w_2), z)$$

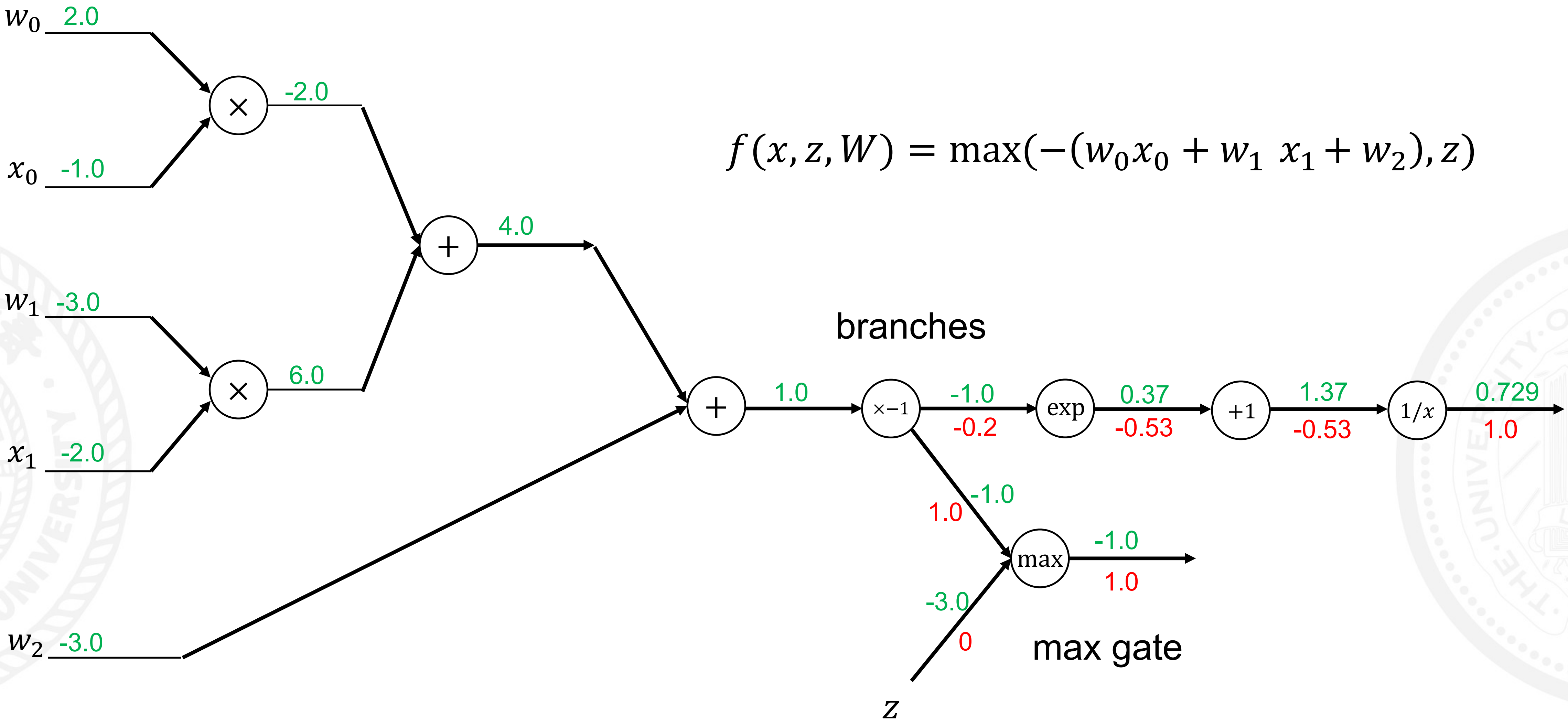
Backpropagation



$$f(x, z, W) = \max(-(w_0 x_0 + w_1\ x_1 + w_2), z)$$

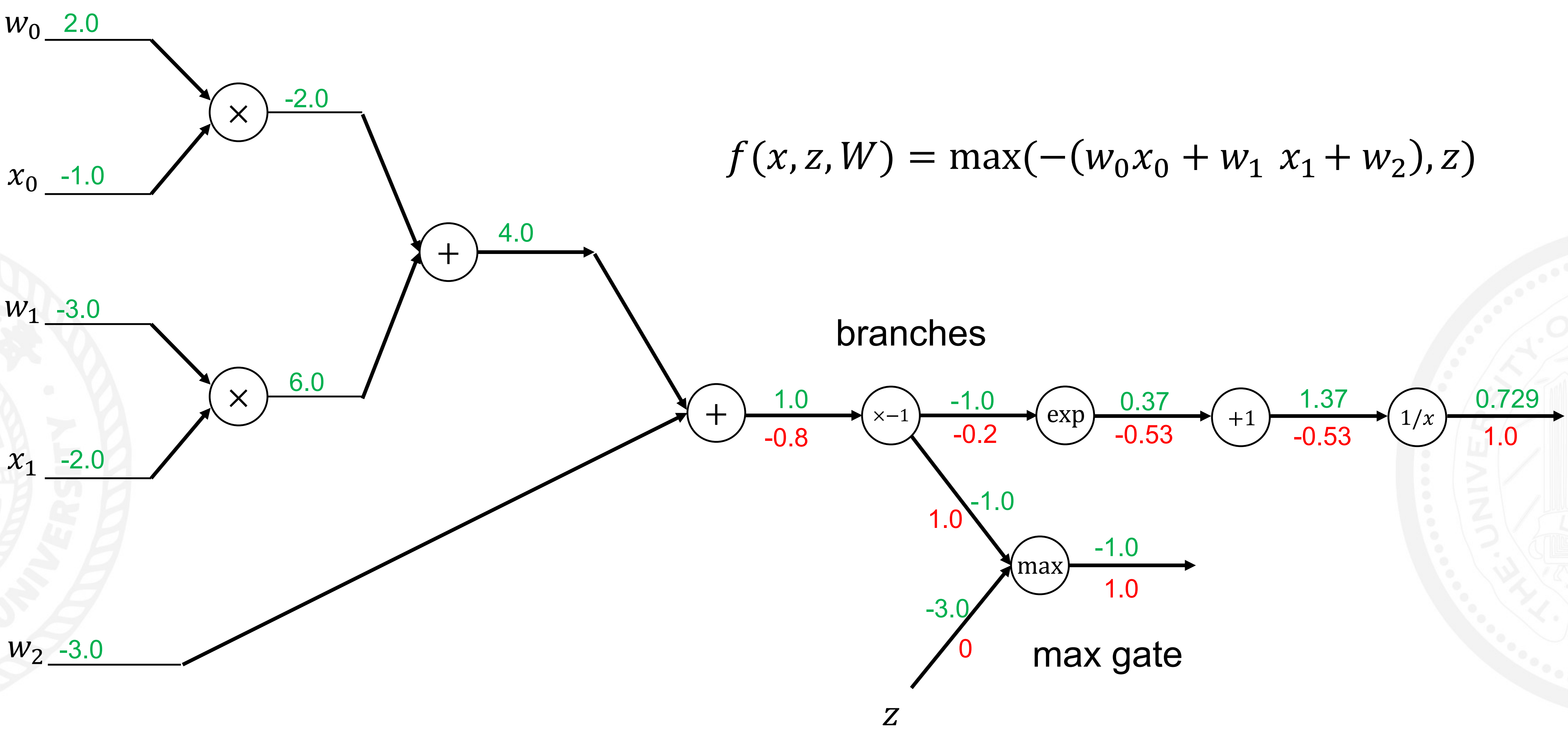
Backpropagation

$$f(x,z,W) = \max(-(w_0x_0 + w_1\ x_1 + w_2), z)$$



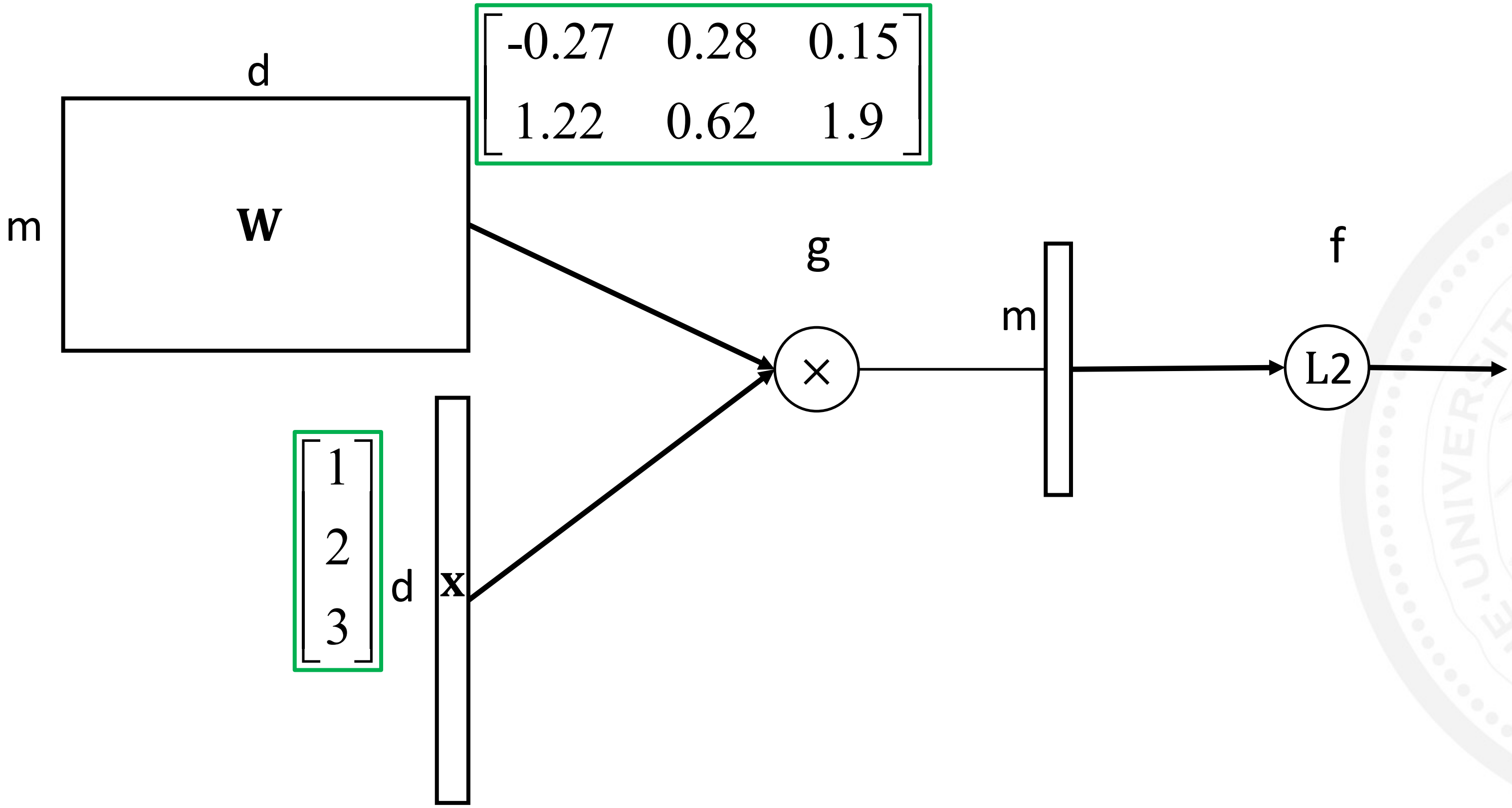
Backpropagation

$$f(x, z, W) = \max(-(w_0 x_0 + w_1\ x_1 + w_2), z)$$



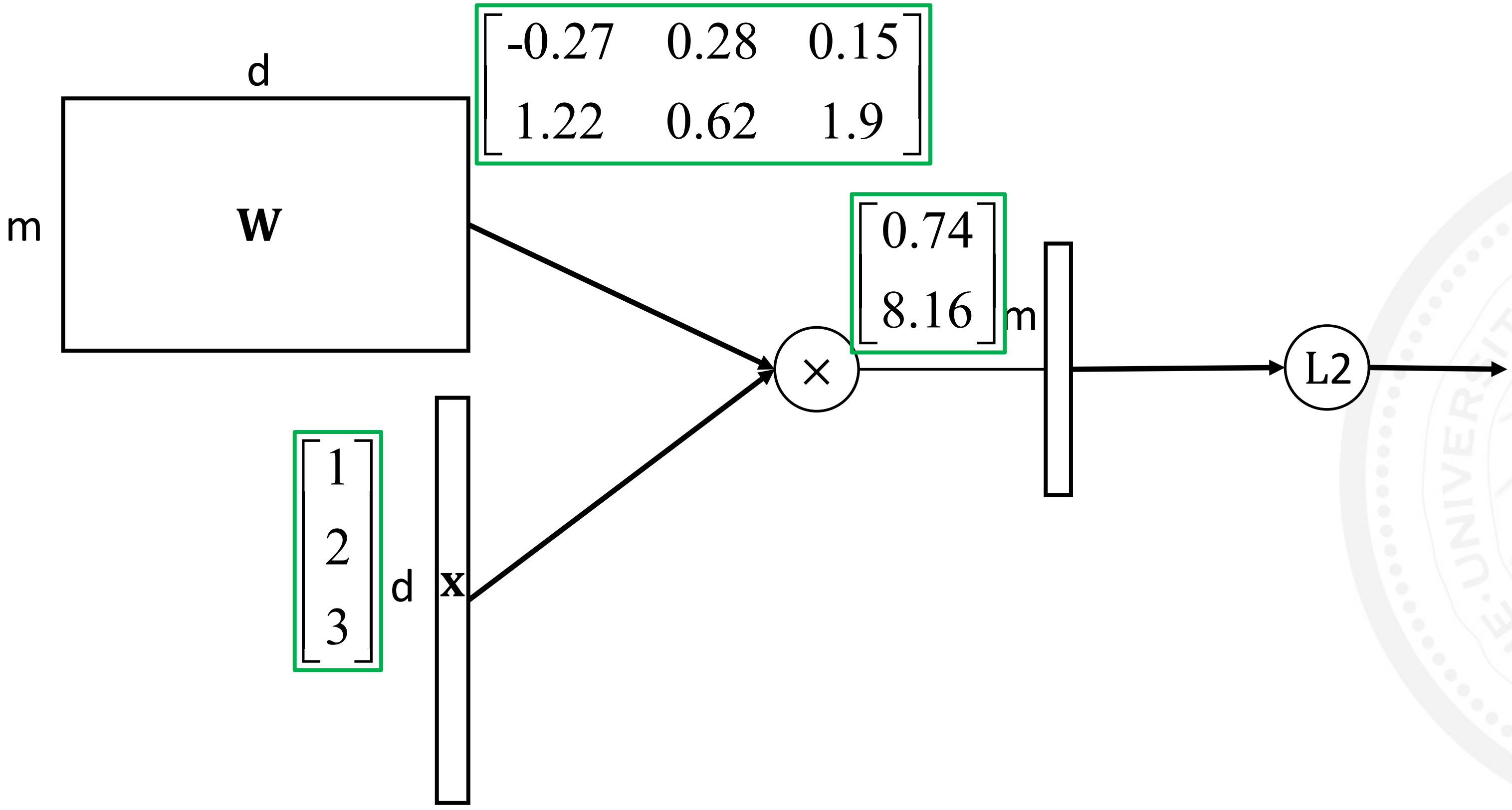
Vectorized example

$$f(\mathbf{x}, \mathbf{W}) = \|\mathbf{W} \cdot \mathbf{x}\|^2$$



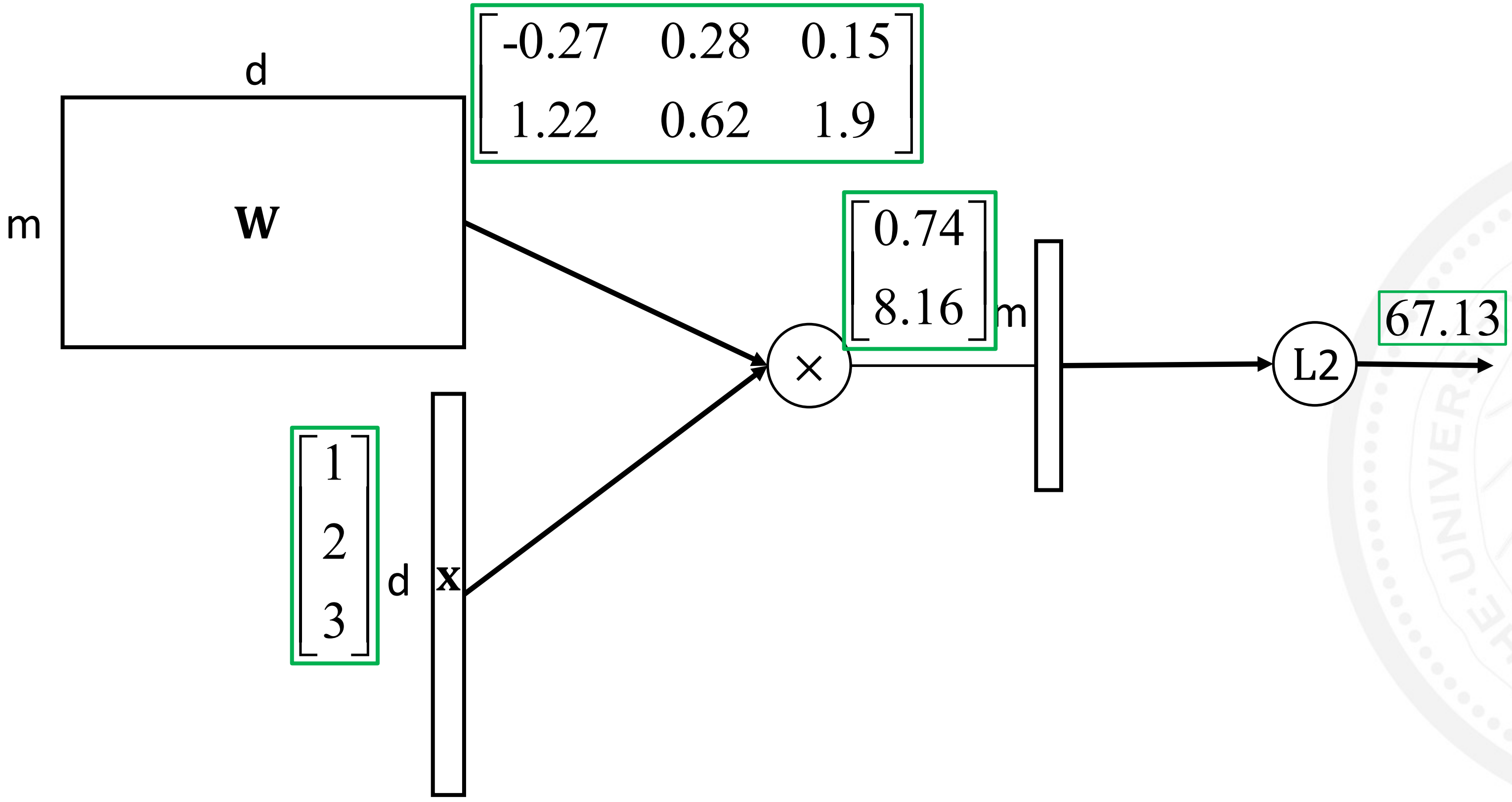
Vectorized example

$$f(\mathbf{x}, \mathbf{W}) = \|\mathbf{W} \cdot \mathbf{x}\|^2$$



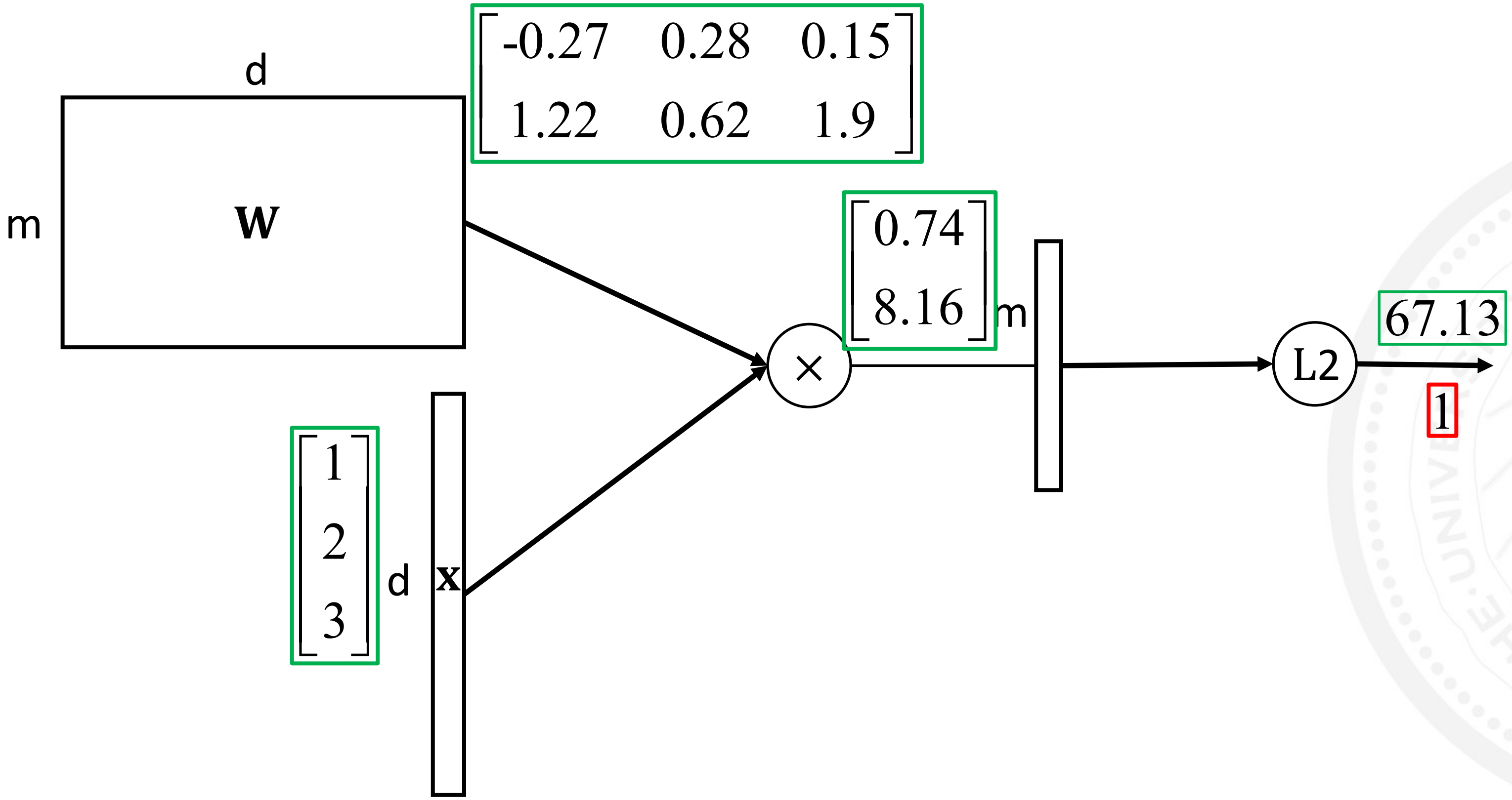
Vectorized example

$$f(\mathbf{x}, \mathbf{W}) = \|\mathbf{W} \cdot \mathbf{x}\|^2$$



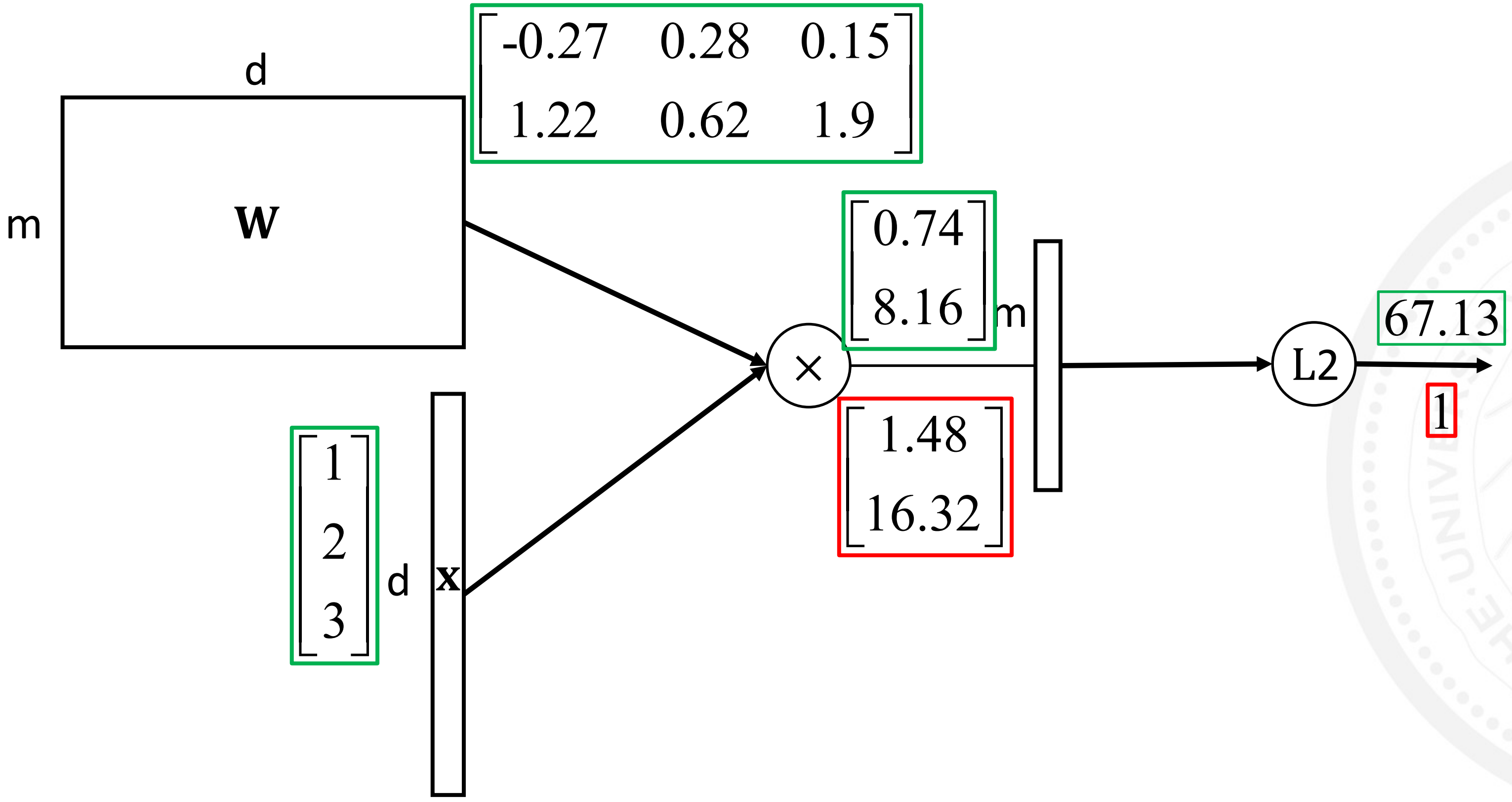
Vectorized example

$$f(\mathbf{x}, \mathbf{W}) = \|\mathbf{W} \cdot \mathbf{x}\|^2$$



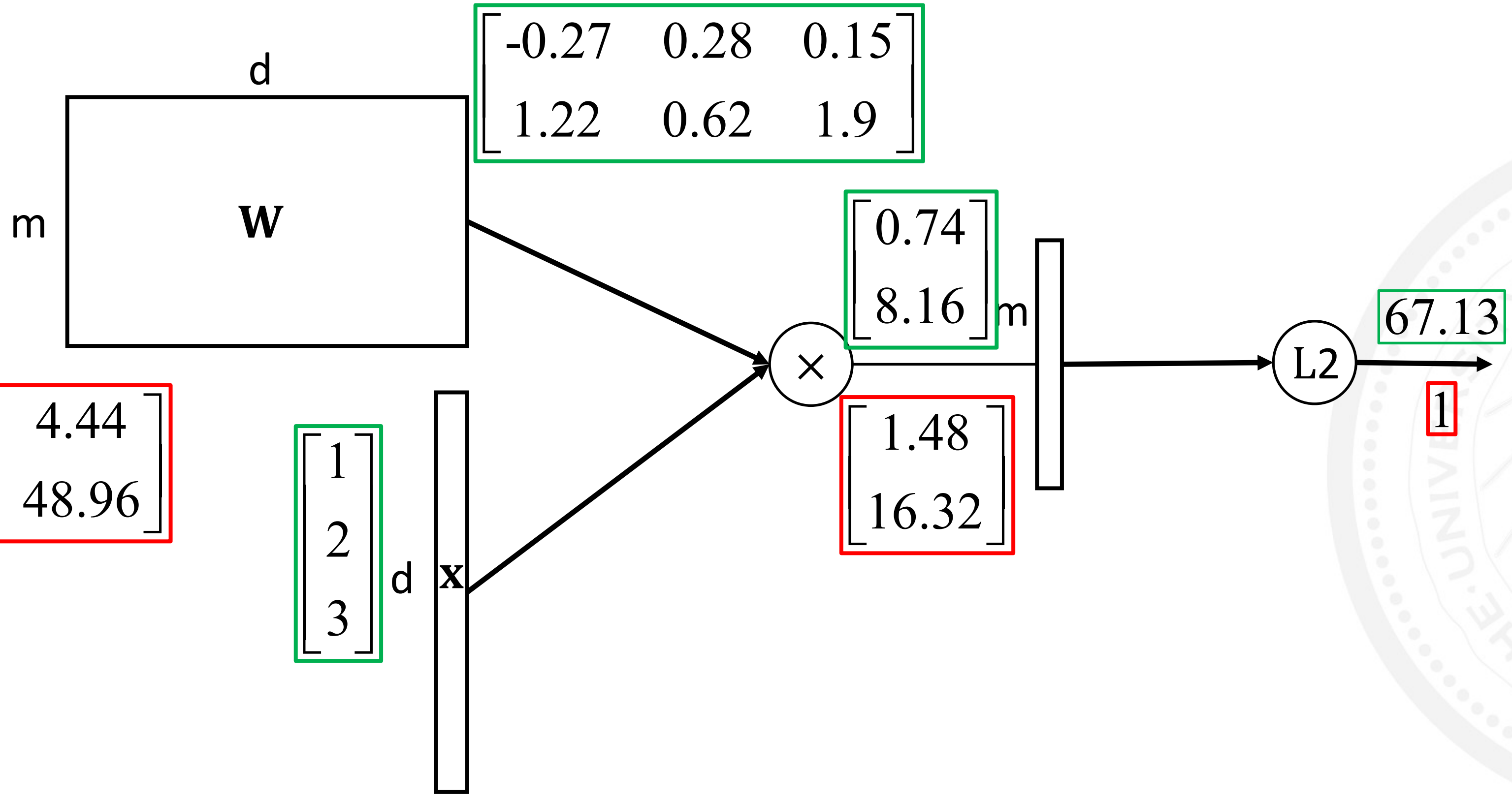
Vectorized example

$$f(\mathbf{x}, \mathbf{W}) = \|\mathbf{W} \cdot \mathbf{x}\|^2$$



Vectorized example

$$f(\mathbf{x},\mathbf{W}) = \|\mathbf{W}\cdot\mathbf{x}\|^2$$



$$\begin{bmatrix}1.48\\16.32\end{bmatrix}\times\begin{bmatrix}1 & 2 & 3\end{bmatrix}=\begin{bmatrix}1.48 & 2.96 & 4.44\\16.32 & 32.64 & 48.96\end{bmatrix}$$