Cs231n Lecture 4
Introduction to Neural Networks

2170051 송여진

CONTENTS

1. Backpropagation

1. Backpropagation: a simple example

Backpropagation(오차역전파)

Chain Rule

Backward Pass

1. Backpropagation: a simple example

Chain Rule : 합성함수의 미분에서 사용하는 공식

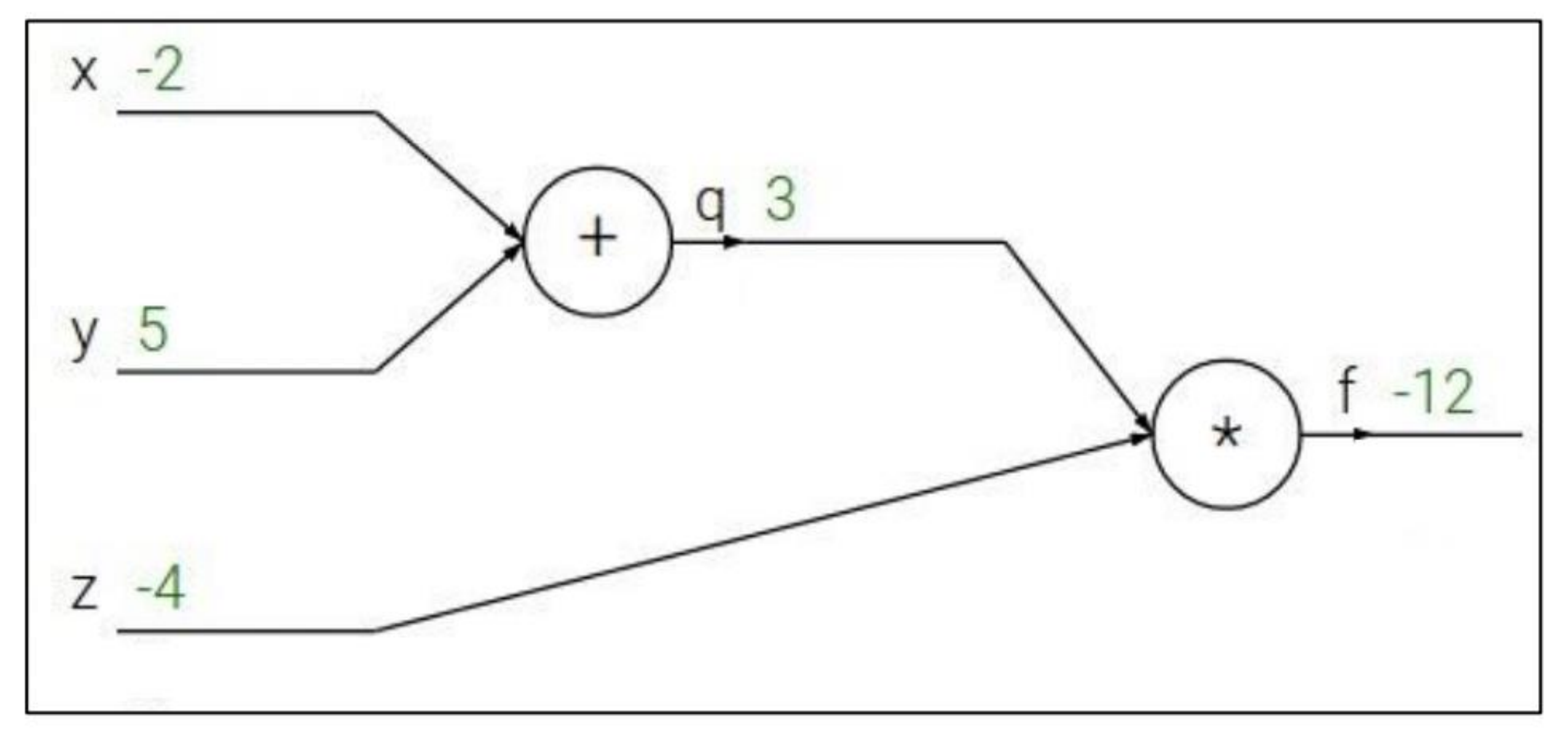
$$[f(g(x))]' = f'(g(x)) \times g'(x)$$

$$\frac{df}{dx} = \frac{df}{dg} \times \frac{dg}{dx}$$

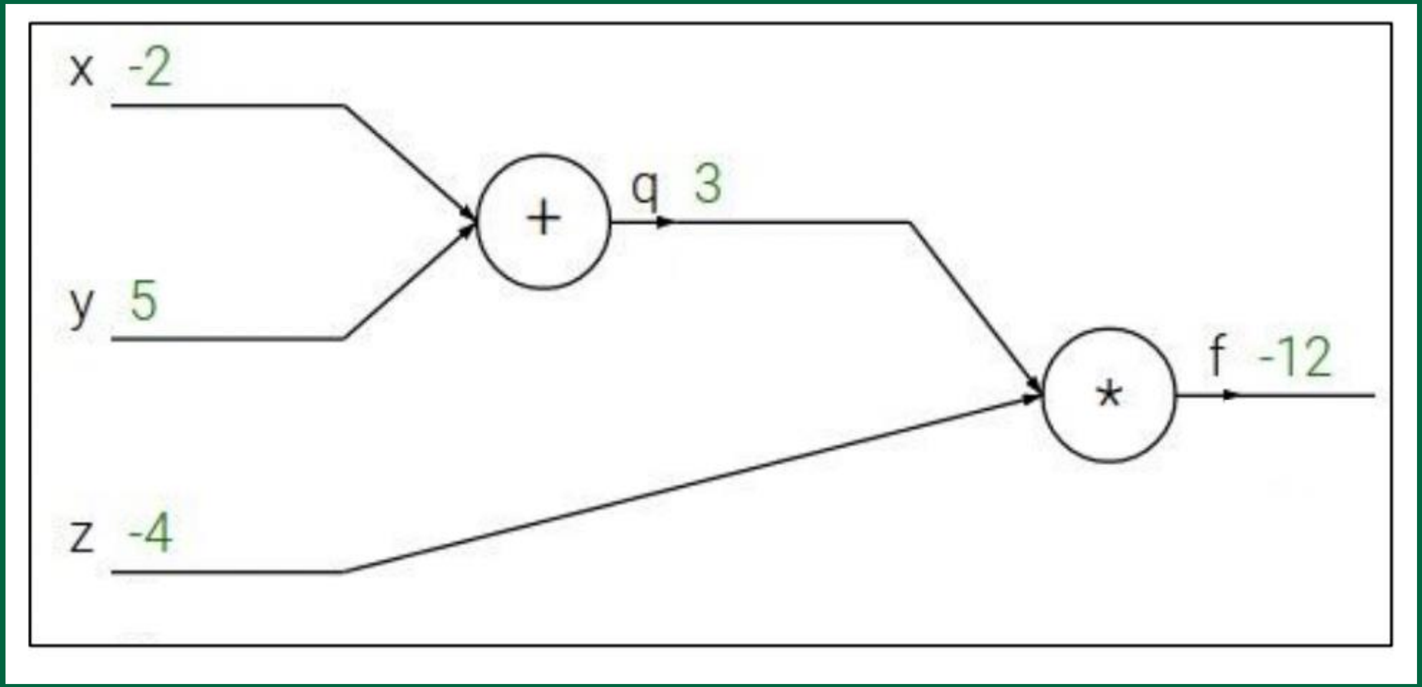
$$f(g(h(x)))' \rightarrow \frac{df}{dx} = \frac{df}{dg} \times \frac{dg}{dh} \times \frac{dh}{dx}$$

1. Backpropagation: a simple example

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



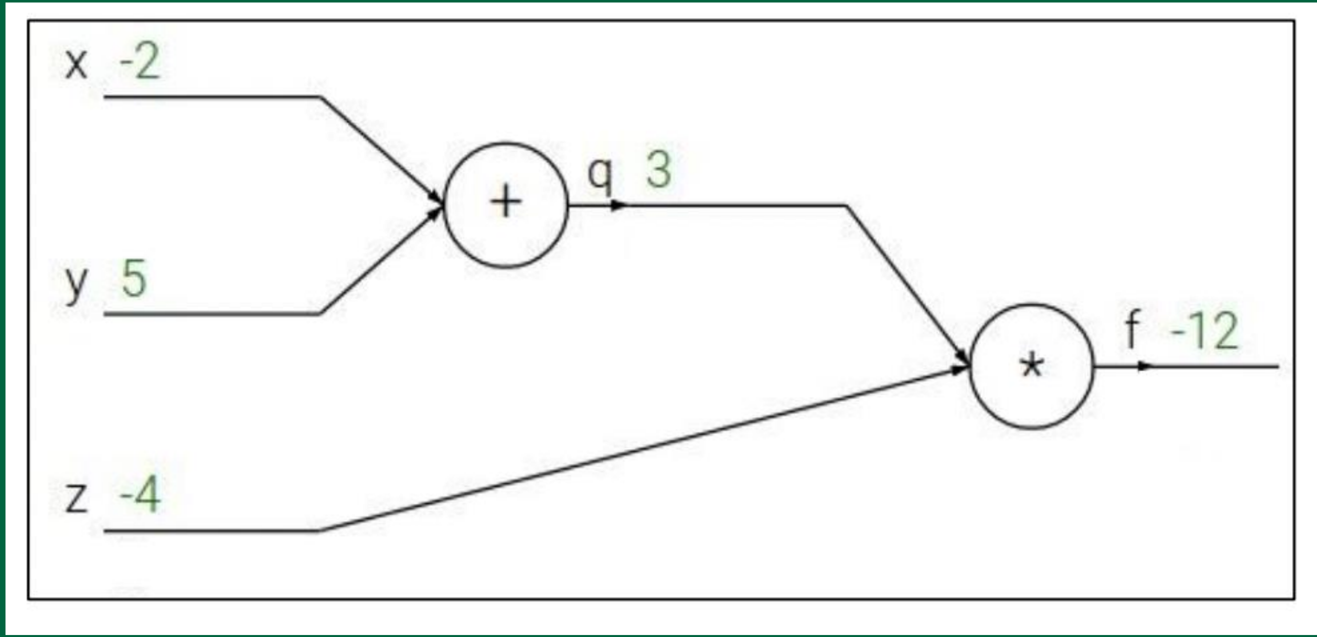
1. Backpropagation: a simple example



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

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

1. Backpropagation: a simple example



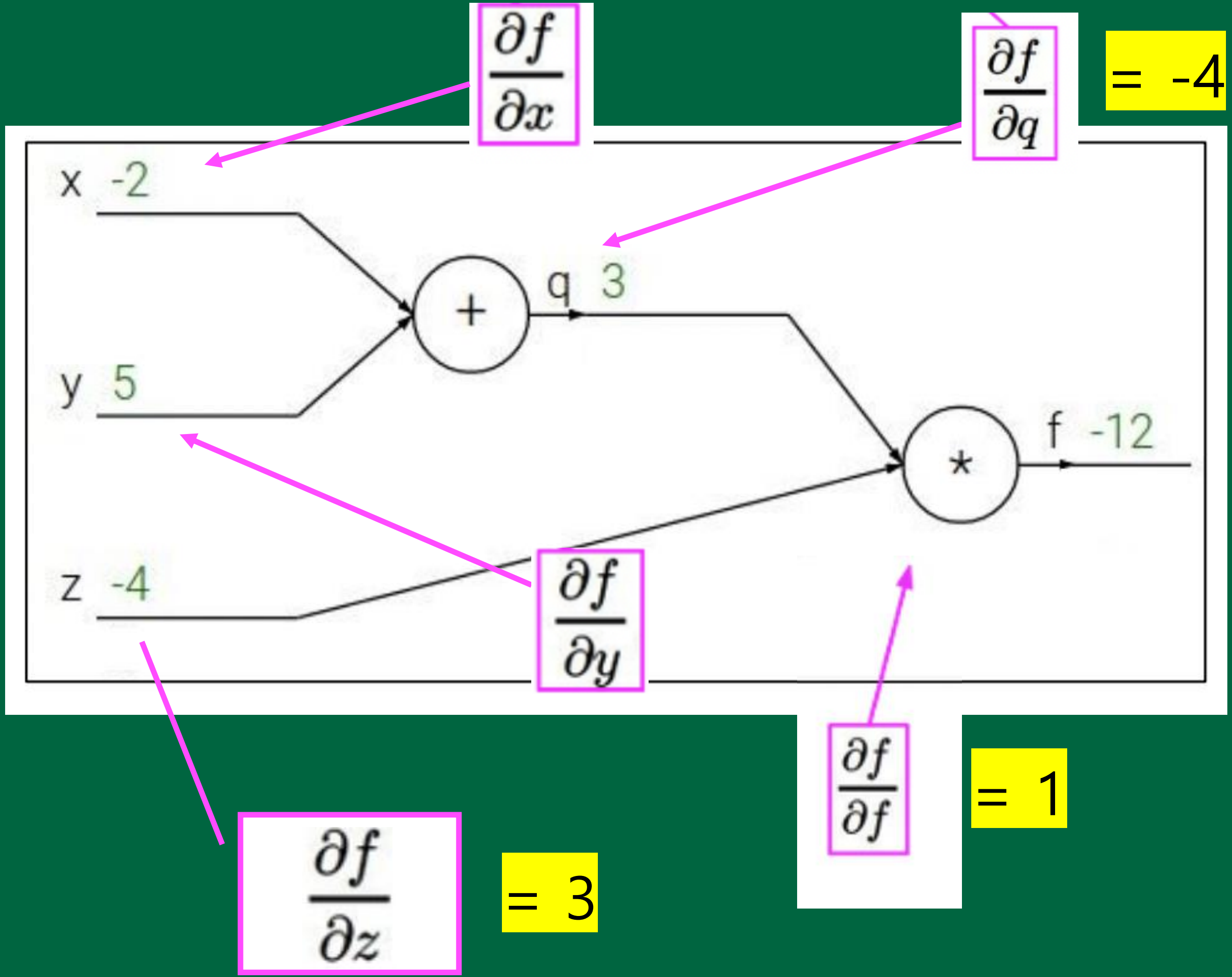
$$f(x,y,z) = (x+y)z$$

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

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

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

1. Backpropagation: a simple example



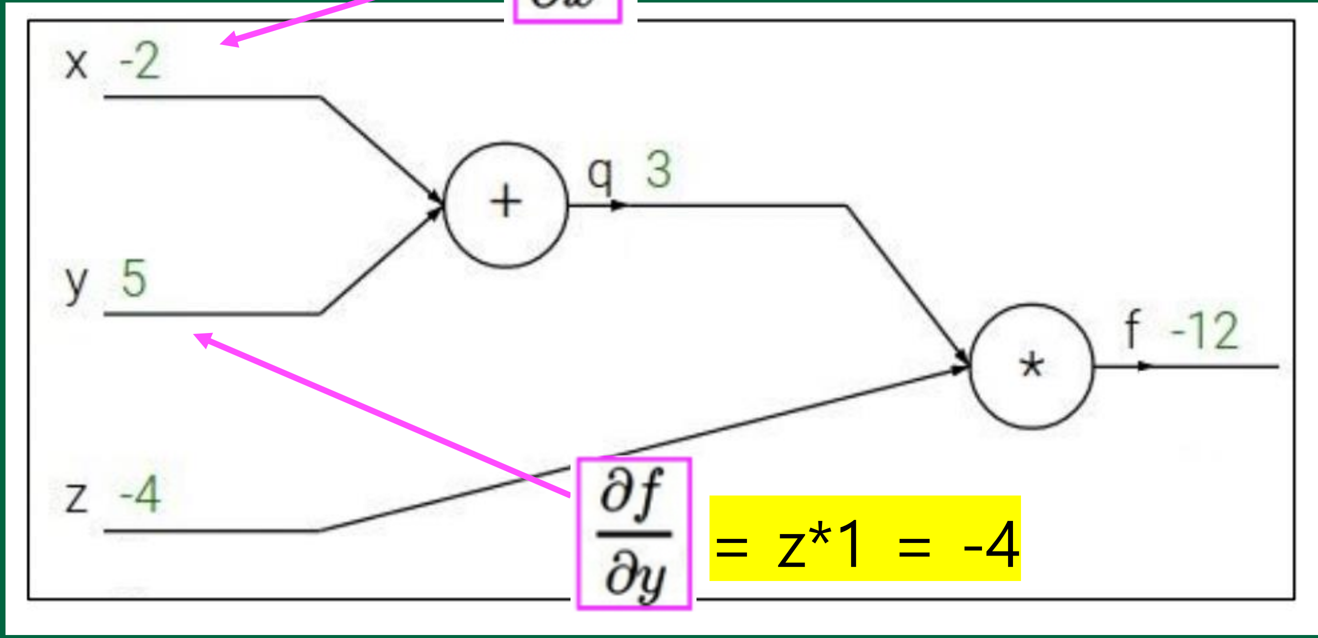
$$f(x, y, z) = (x + y)z$$

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

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

1. Backpropagation: a simple example

$$\frac{\partial f}{\partial x} = z*1 = -4$$



$$\frac{\partial f}{\partial y} = z*1 = -4$$

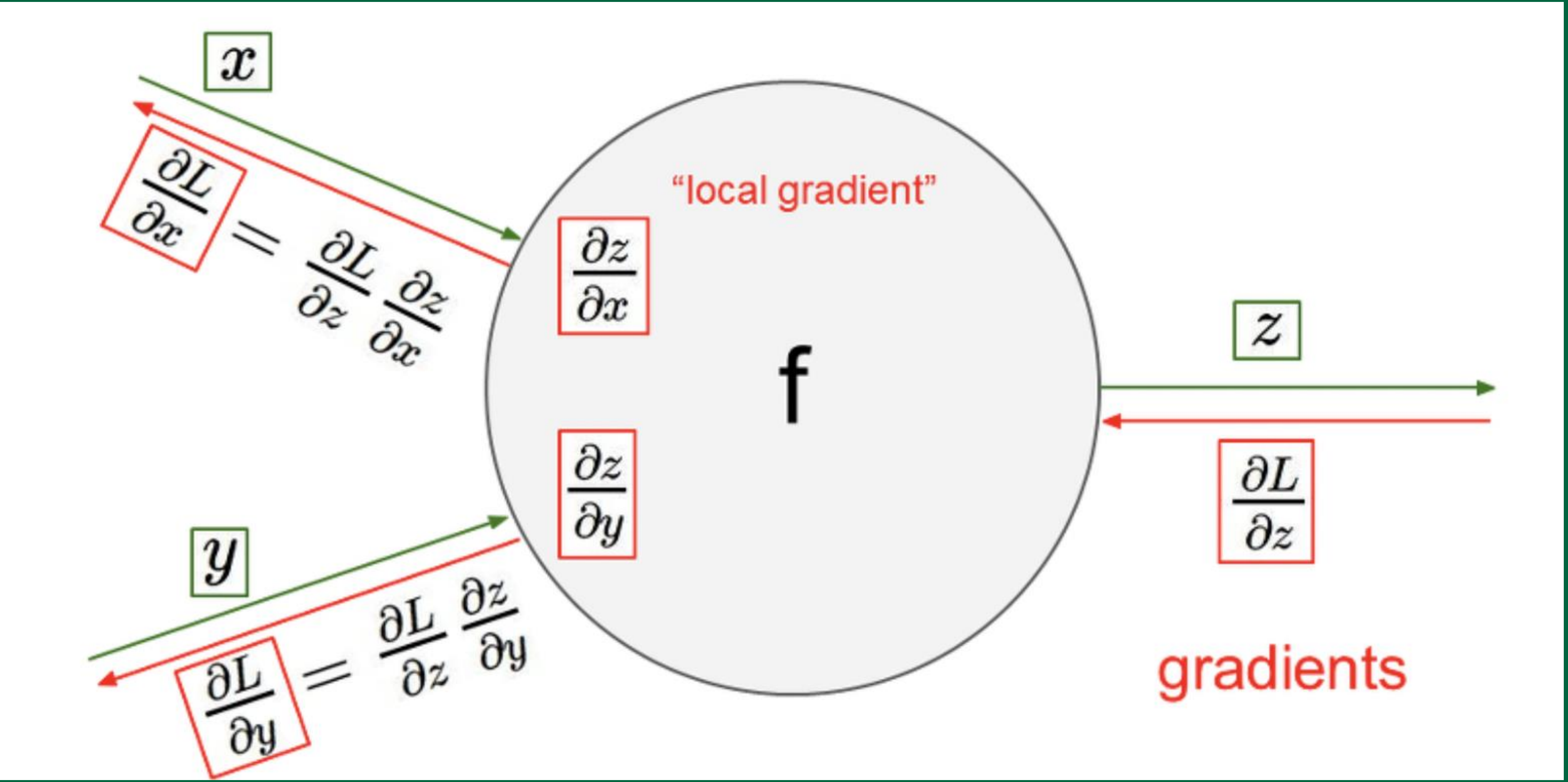
Chain rule:

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

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

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

1. Backpropagation: a simple example



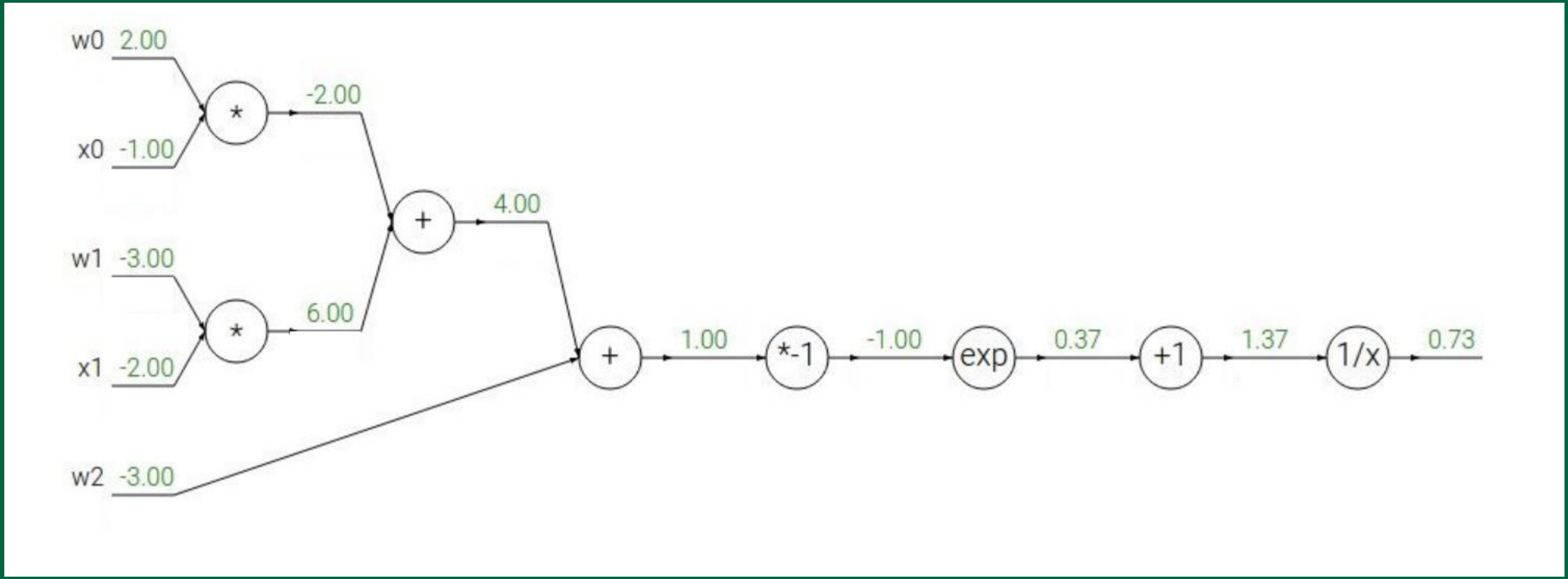
1. Backpropagation: a simple example

Remember:

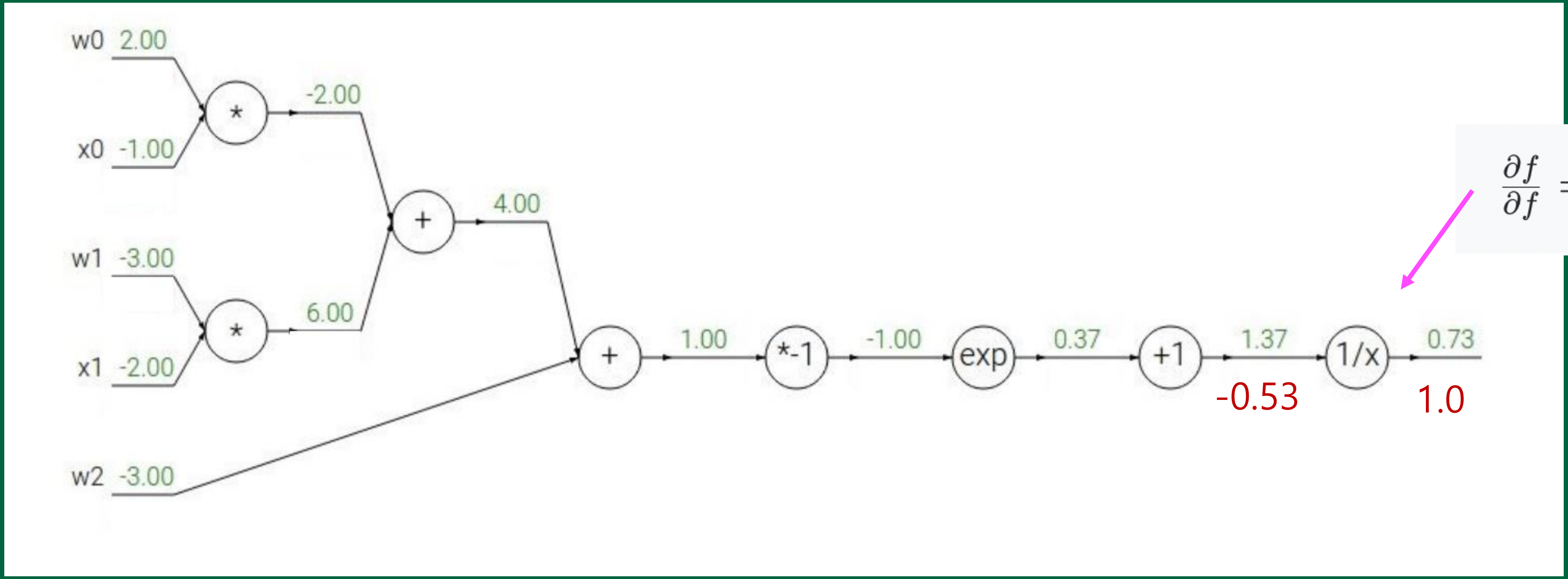
Gradient = Local Gradient * Global Gradient

1. Backpropagation: another example

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



1. Backpropagation: another example

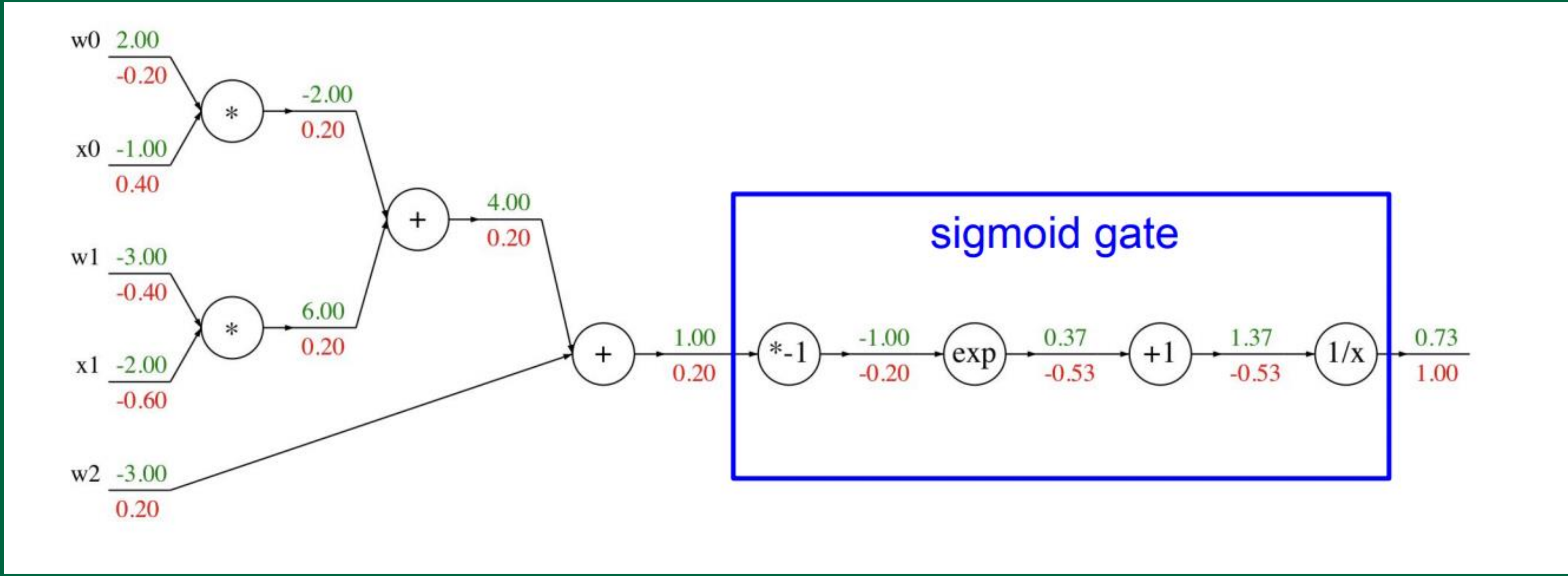


$$\frac{\partial f}{\partial f} = 1$$

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

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

1. Backpropagation: another example



Green color text : local gradient
Red color text: upstream gradient

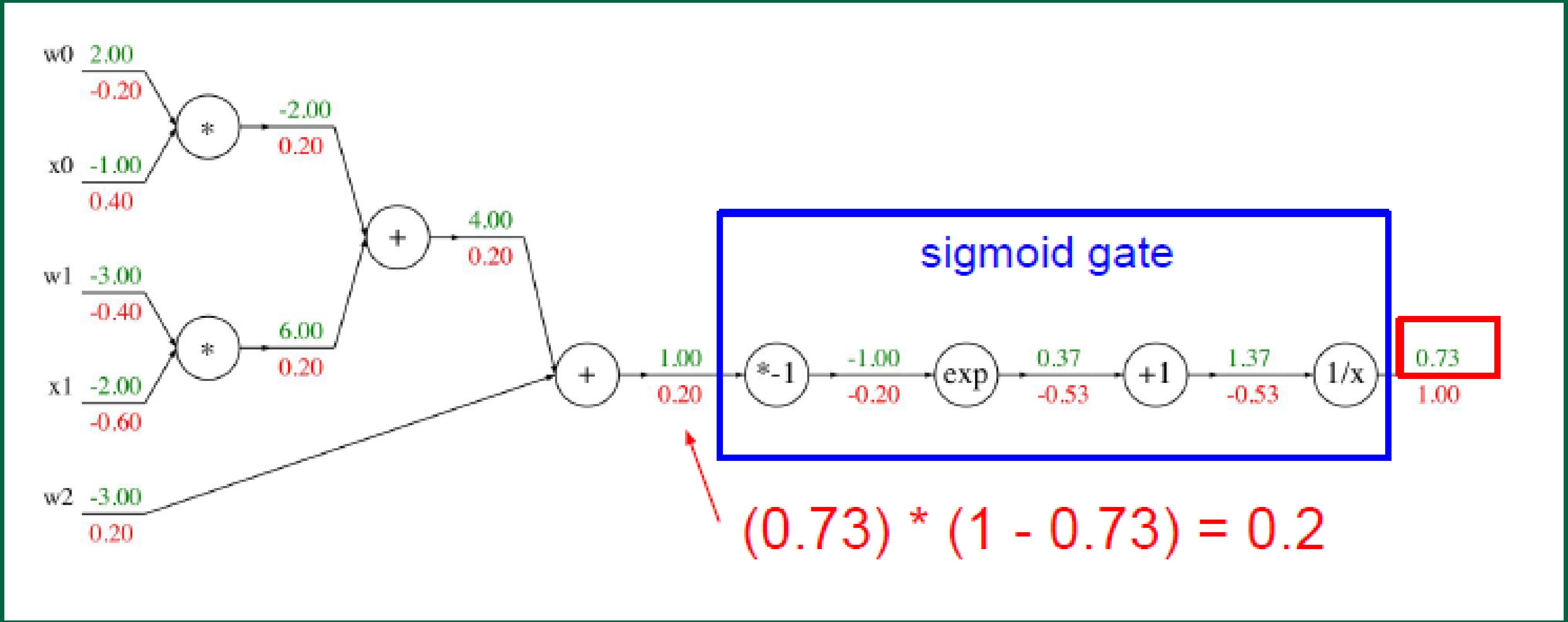
1. Backpropagation: another example

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

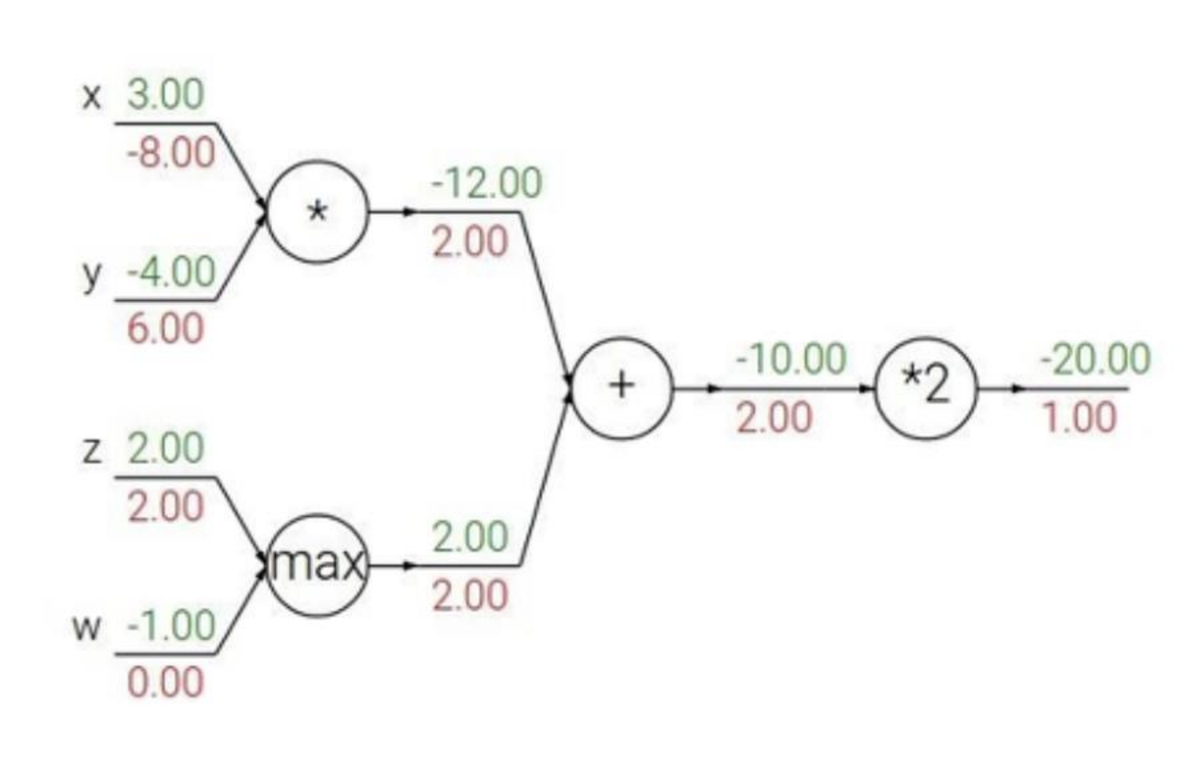
$$\boxed{\sigma(x) = \frac{1}{1+e^{-x}}}$$ sigmoid function

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

1. Backpropagation: another example



1. Backpropagation: Patterns in backward flow

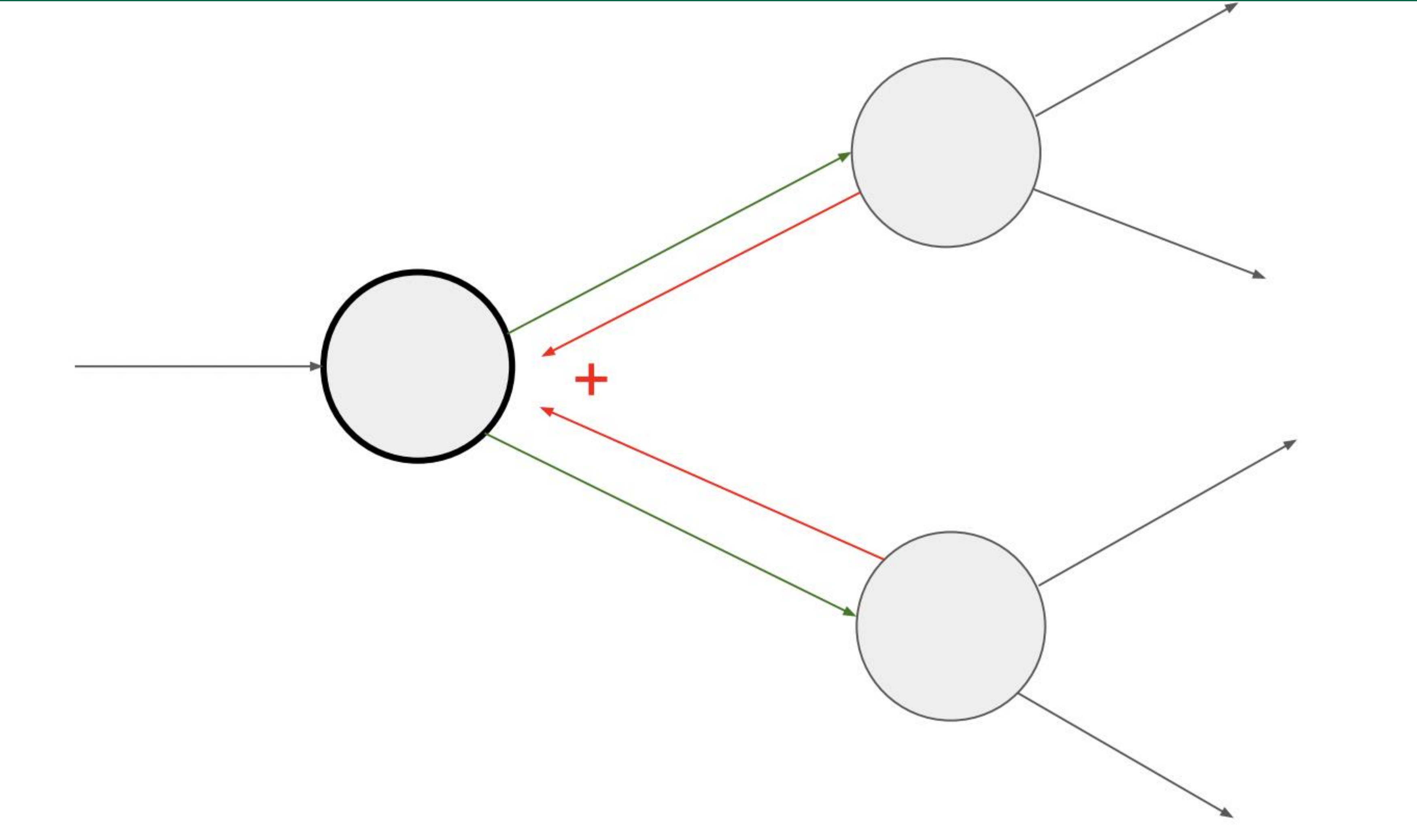


Add gate: gradient distributor

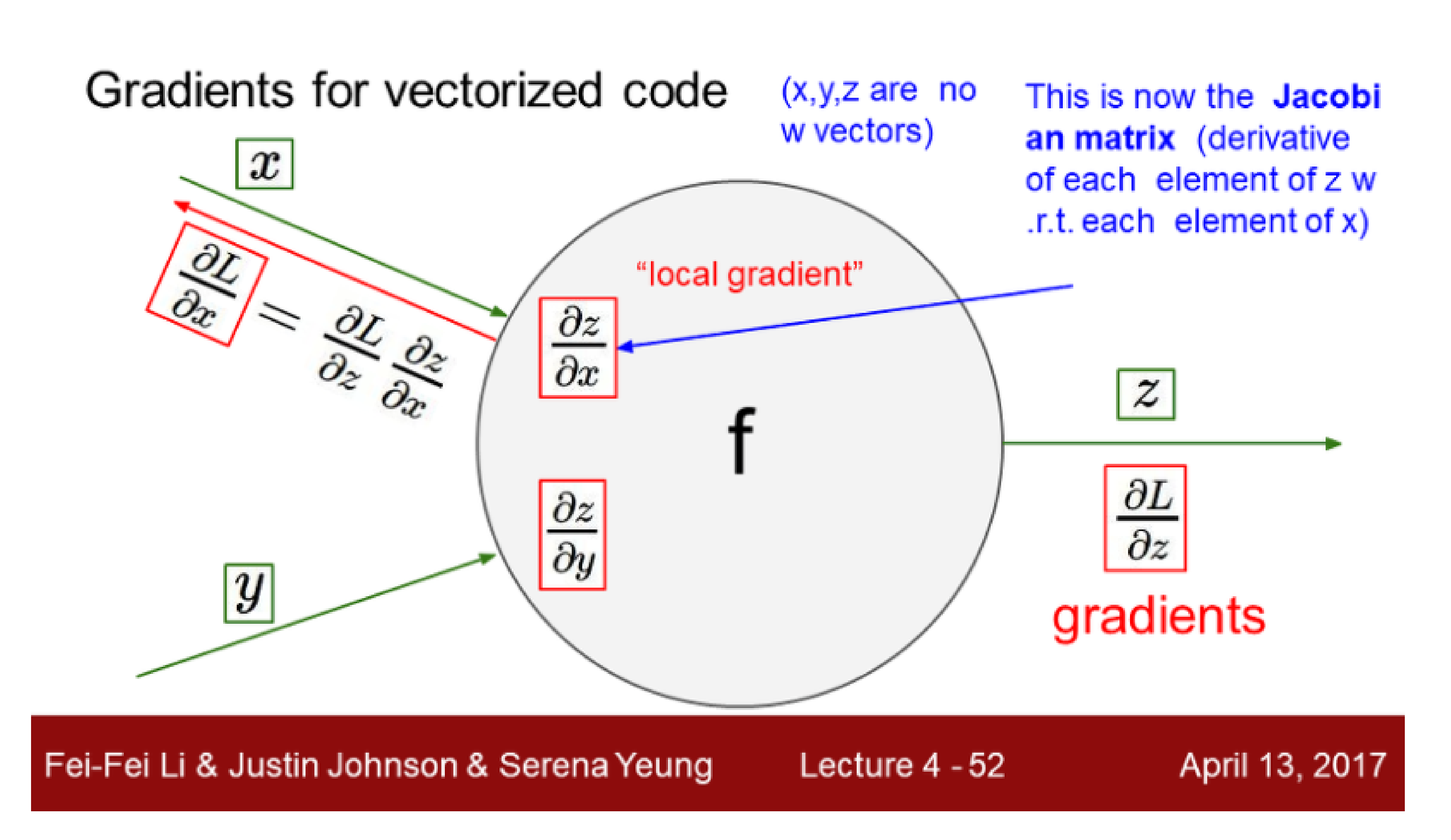
Max gate: gradient router

Mul gate: gradient switcher

1. Backpropagation: Gradients add at branches



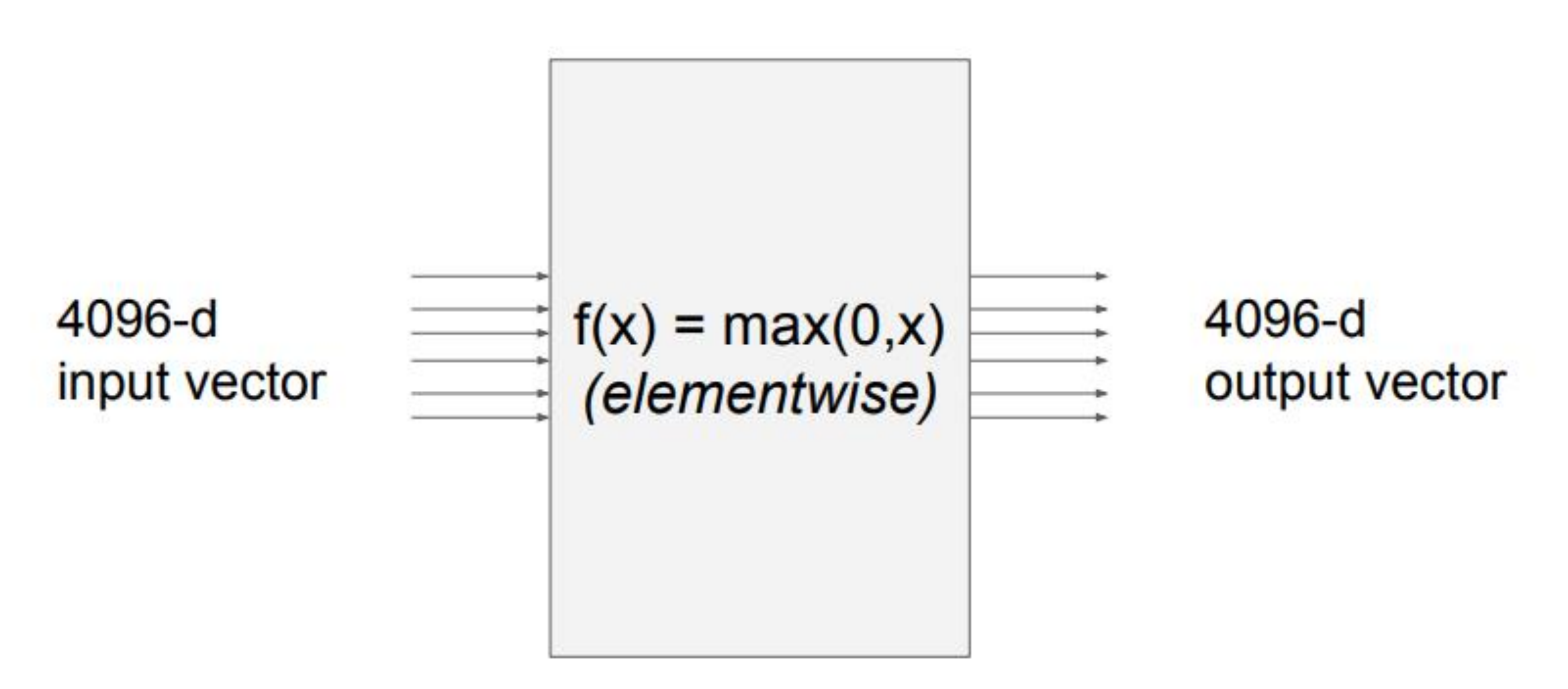
1. Backpropagation: Gradients for vectorized code



1. Backpropagation: Gradients for vectorized code

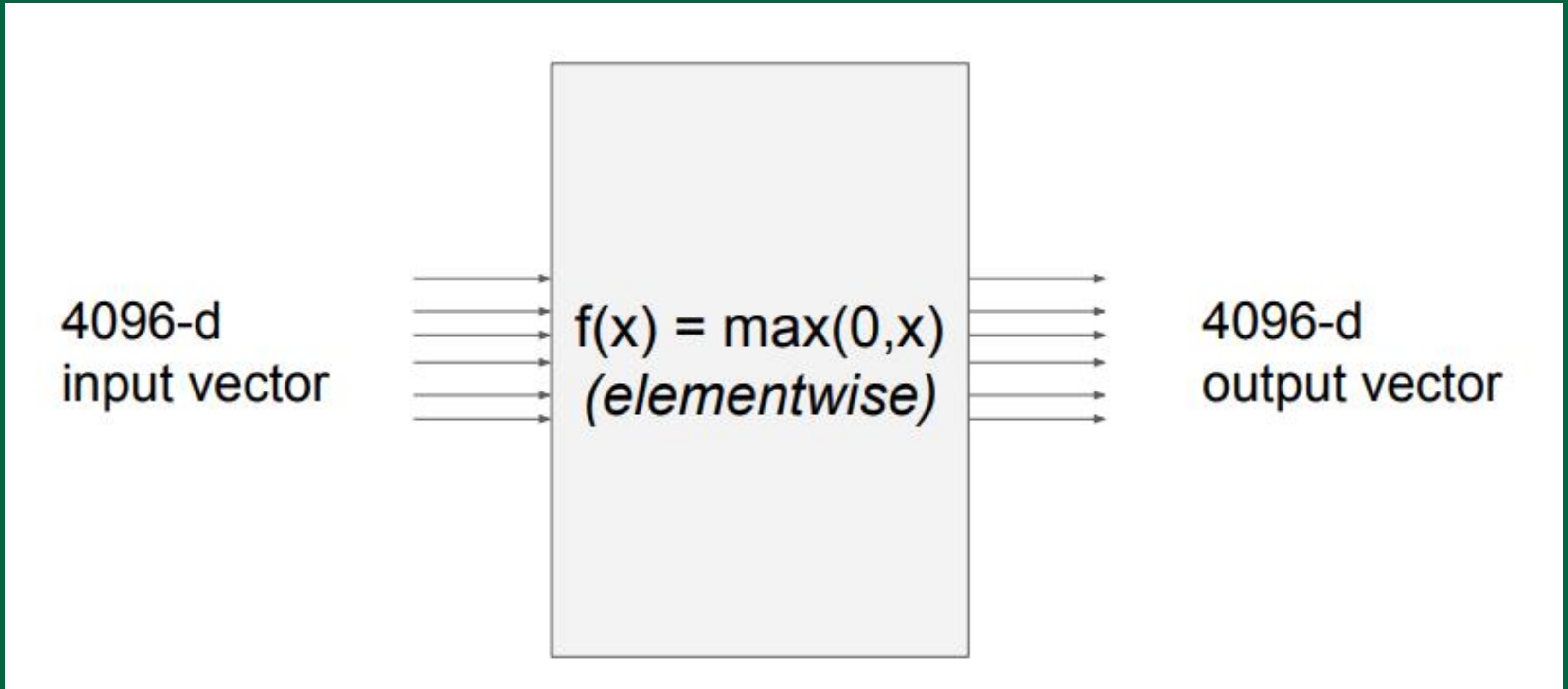
$$J = \frac{\partial(x, y)}{\partial(u, v)} = \begin{vmatrix} \dfrac{\partial x}{\partial u} & \dfrac{\partial x}{\partial v} \\ \dfrac{\partial y}{\partial u} & \dfrac{\partial y}{\partial v} \end{vmatrix} = \frac{\partial x}{\partial u}\frac{\partial y}{\partial v} - \frac{\partial x}{\partial v}\frac{\partial y}{\partial u}.$$

1. Backpropagation: Vectorized operations



Jacobian Matrix size : 4096 x 4096!

1. Backpropagation: Vectorized operations



Jacobian Matrix size(100 minibatch) : 409600 x 409600!

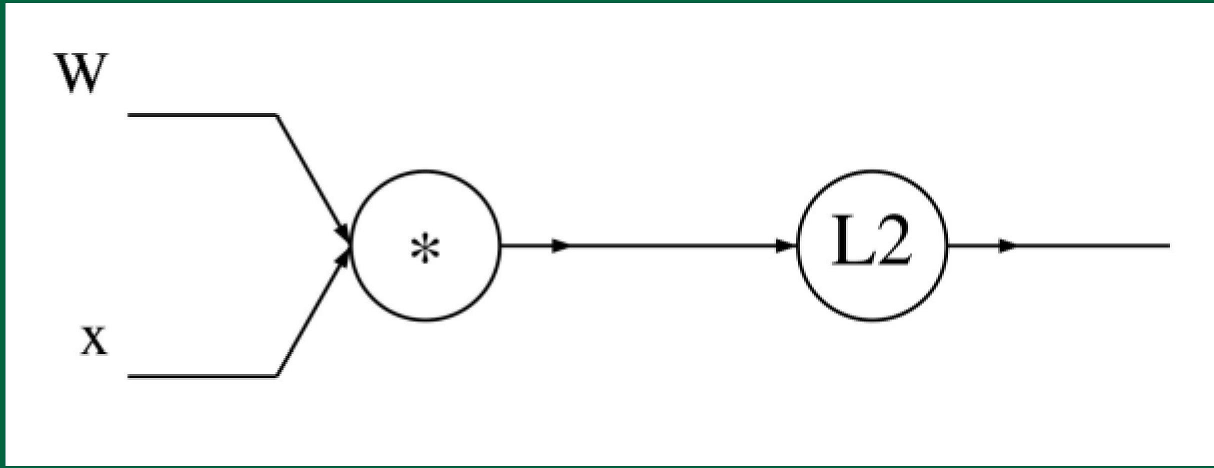
1. Backpropagation: Vectorized operations

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 5 \end{pmatrix}$$

Diagonal Matrix

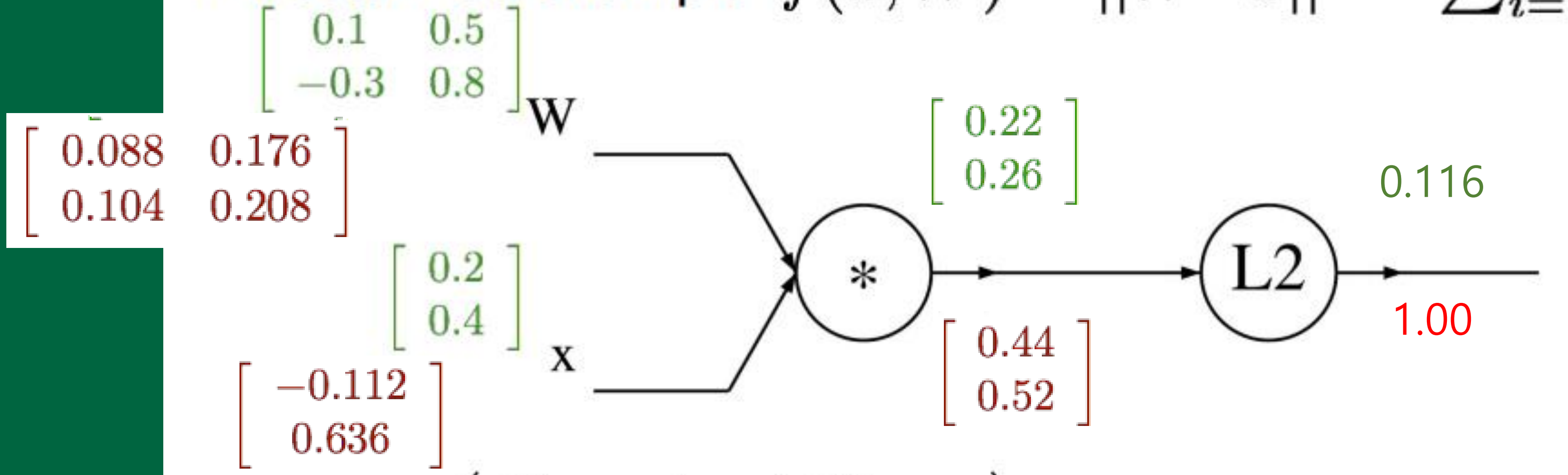
1. Backpropagation: A Vectorized example

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



1. Backpropagation: Vectorized operations

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



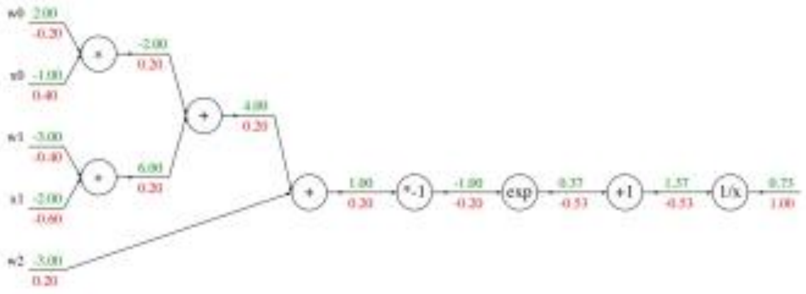
$$\frac{\partial f}{\partial q_i} = 2q_i$$

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

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

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

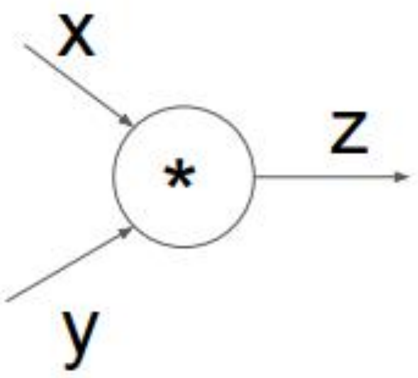
1. Backpropagation: code example



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

```
class ComputationalGraph(object):
    #...
    def forward(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():
        for gate in reversed(self.graph.nodes_topologically_sorted()):
            gate.backward() # little piece of backprop (chain rule applied)
        return inputs_gradients
```

1. Backpropagation: code example



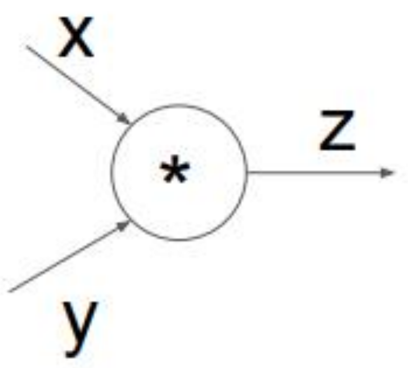
(x,y,z are scalars)

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

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

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

1. Backpropagation: code example



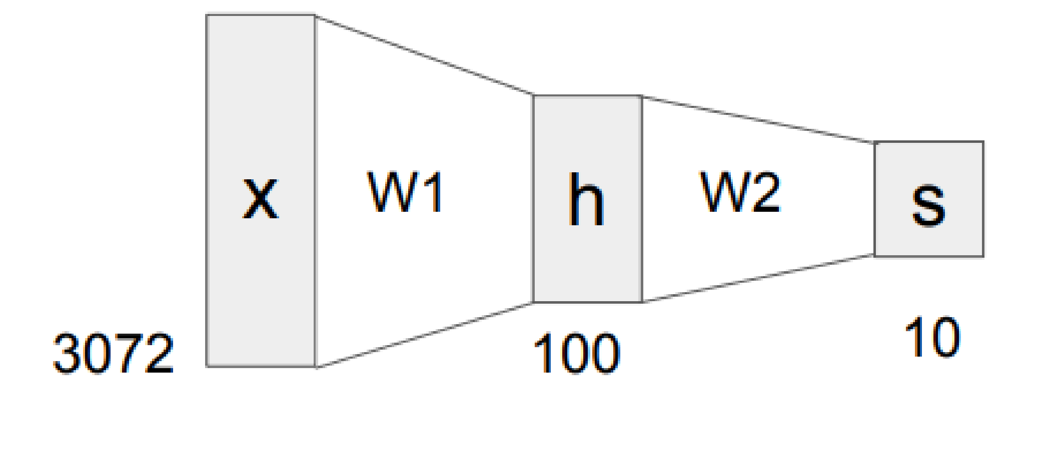
(x,y,z are scalars)

```
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]
```

2. Neural Networks

2. Neural Networks: without the brain stuff

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



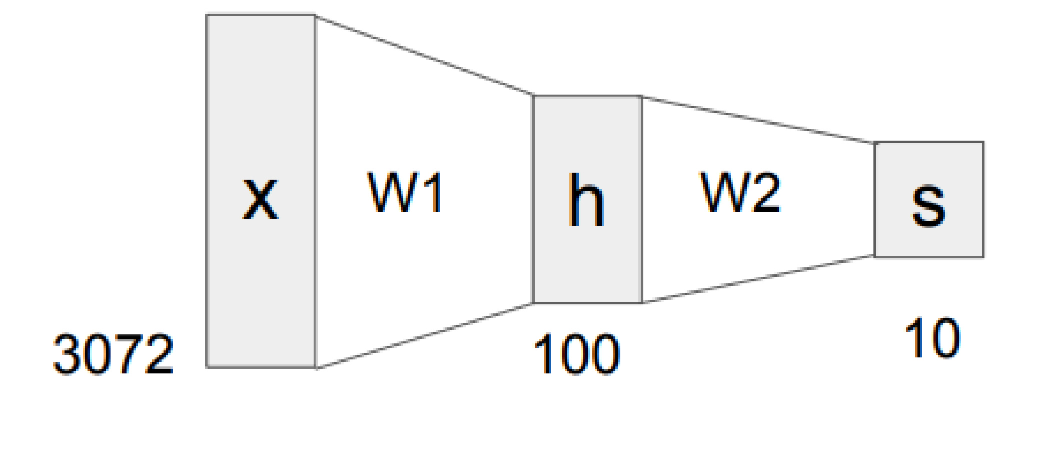
X = 3072 x 1

W = 10 x 3072

S(score) = 10 x 1

2. Neural Networks: without the brain stuff

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



W1 = 100 x 3072

W2 = 10 x 100

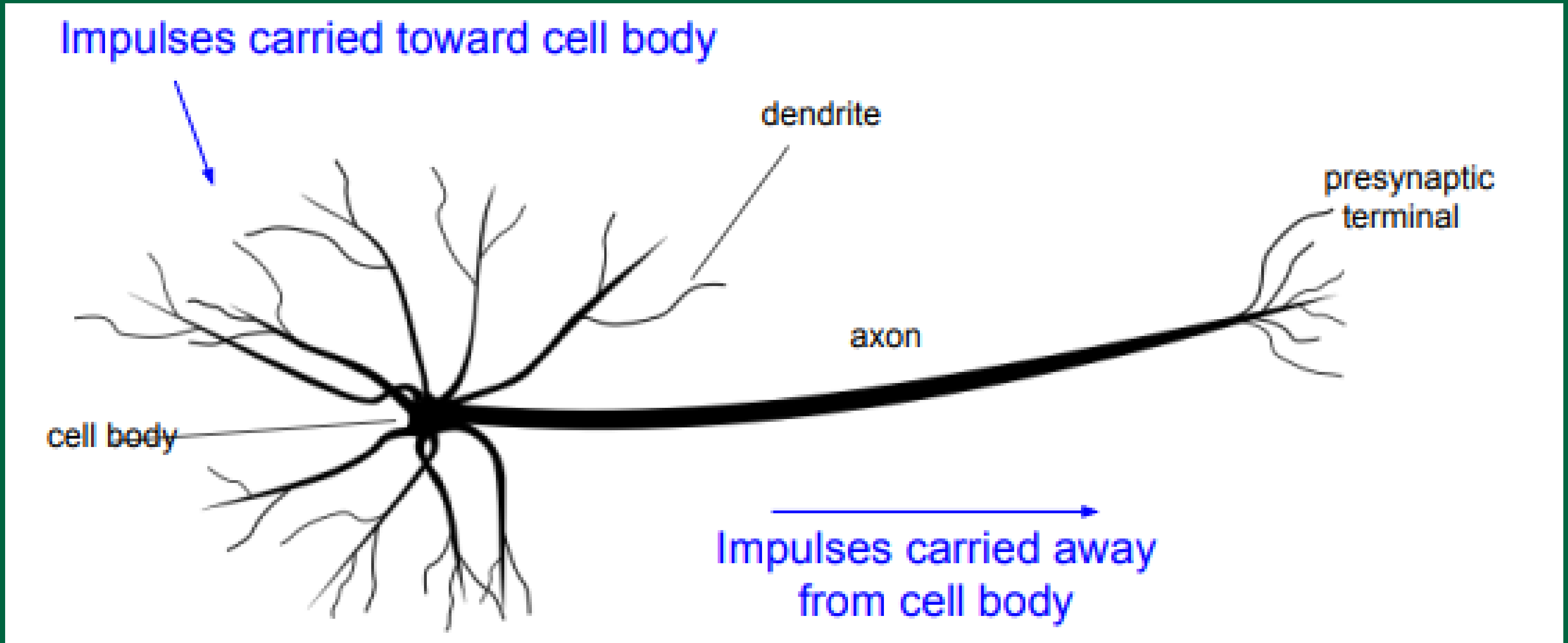
$$f = W_3 \max(0, W_2 \max(0, W_1 x))$$

2. Neural Networks: code example

```
import numpy as np
from numpy.random import randn

N, D_in, H, D_out = 64, 1000, 100, 10
x, y = randn(N, D_in), randn(N, D_out)
w1, w2 = randn(D_in, H), randn(H, D_out)

for t in range(2000):
  h = 1 / (1 + np.exp(-x.dot(w1)))
  y_pred = h.dot(w2)
  loss = np.square(y_pred - y).sum()
  print(t, loss)

  grad_y_pred = 2.0 * (y_pred - y)
  grad_w2 = h.T.dot(grad_y_pred)
  grad_h = grad_y_pred.dot(w2.T)
  grad_w1 = x.T.dot(grad_h * h * (1 - h))

  w1 -= 1e-4 * grad_w1
  w2 -= 1e-4 * grad_w2
```

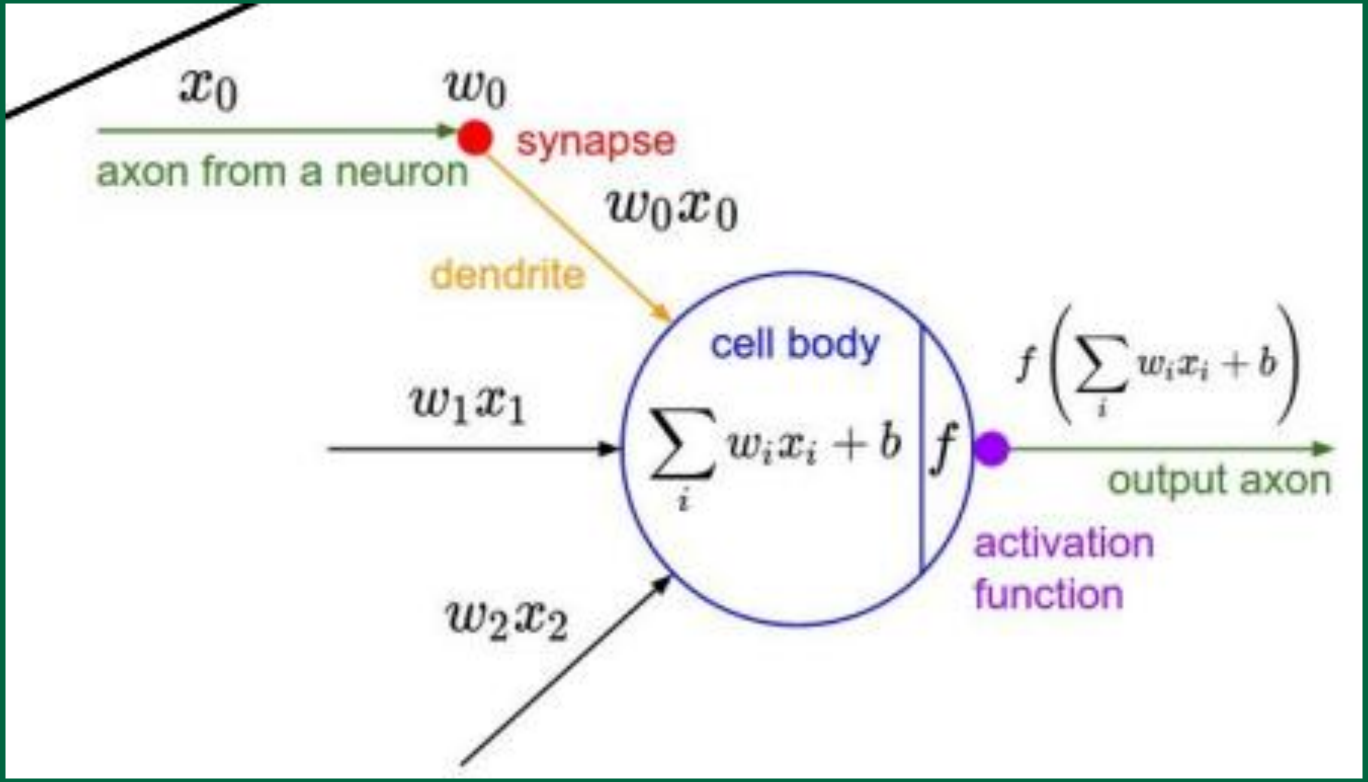
3. Artificial Neural Network

3. Neural Networks

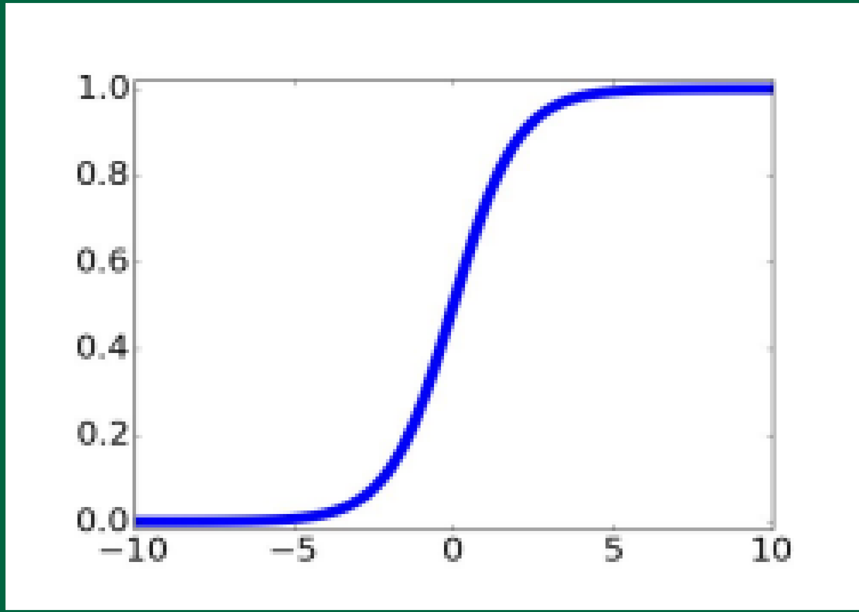


This image by Felipe Perucho is licensed under CC-BY 3.0

3. Neural Networks



3. Neural Networks: Activation functions

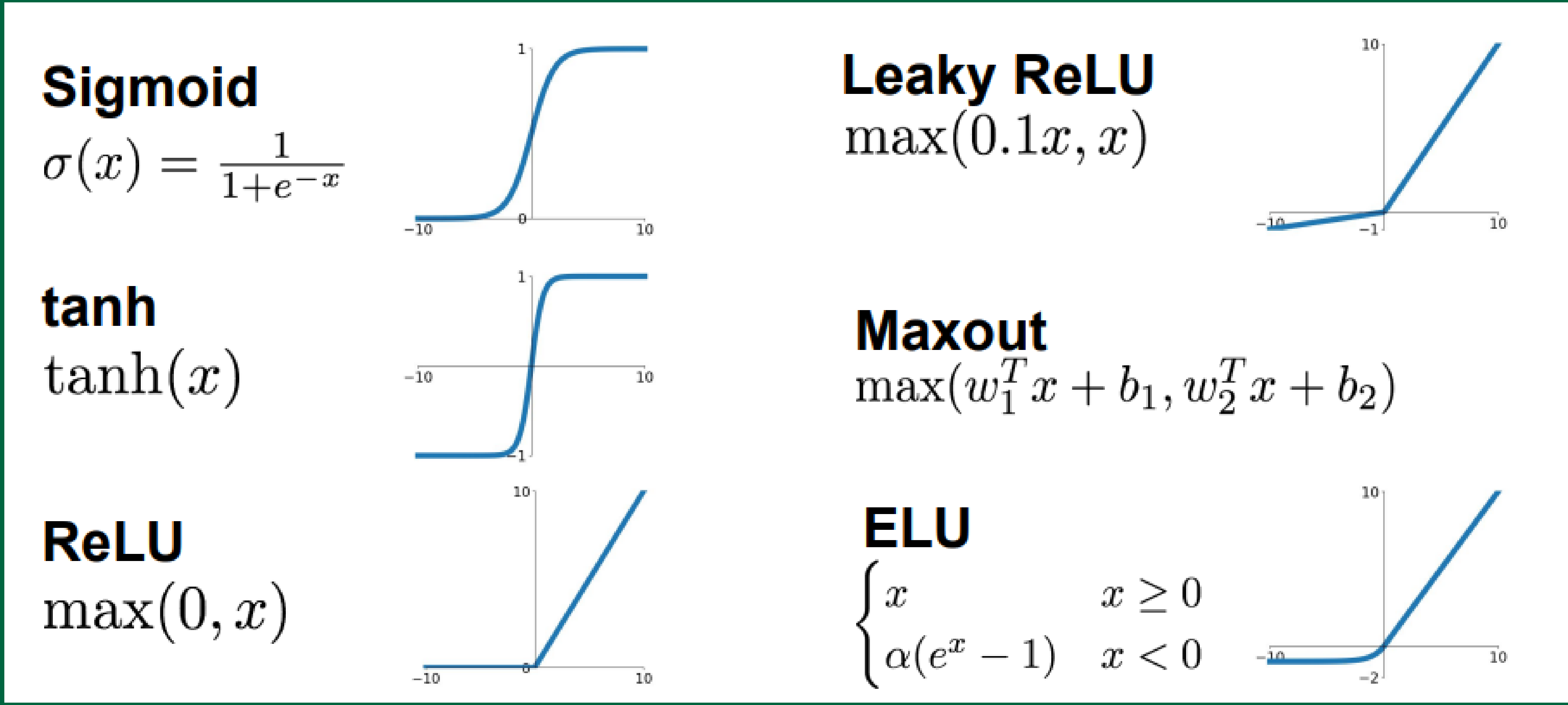


sigmoid activation function

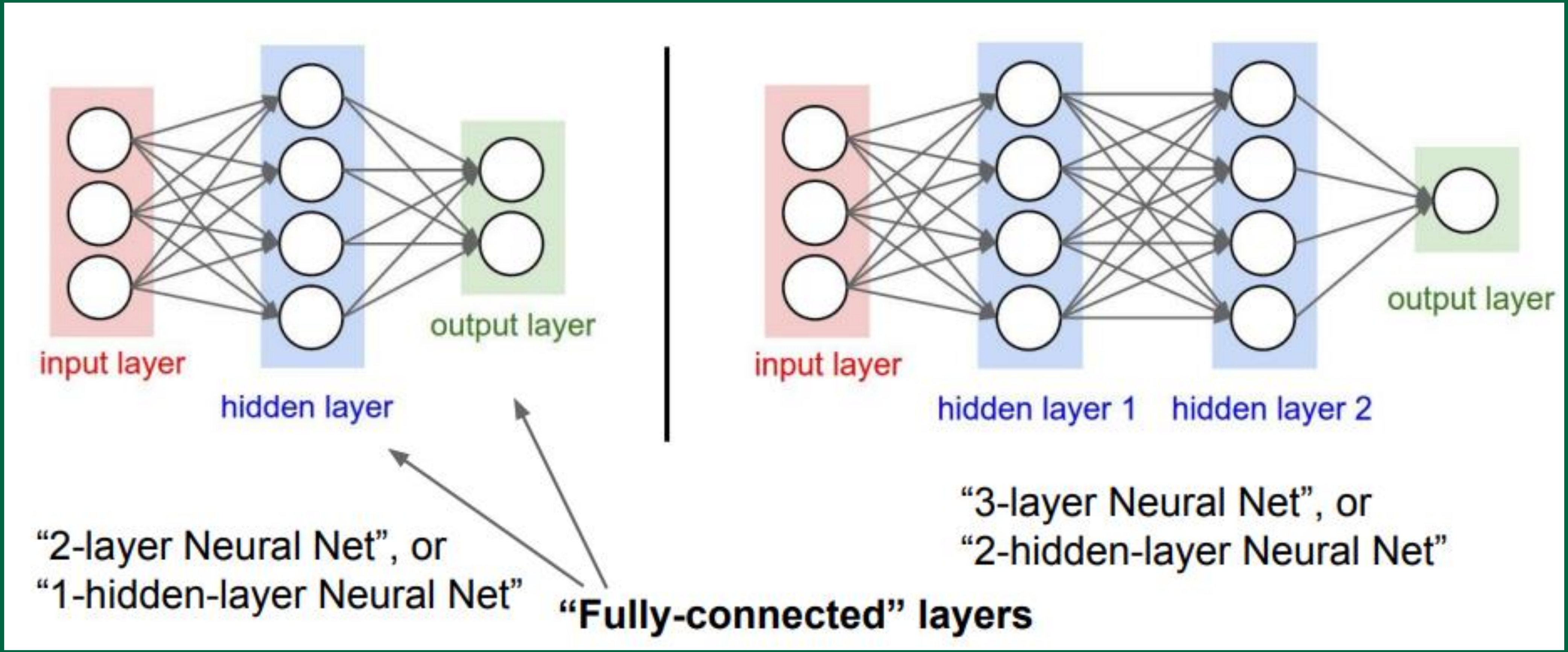
$$\frac{1}{1+e^{-x}}$$

```python
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 func
    return firing_rate
```

3. Neural Networks: Activation functions



3. Neural Networks: Architectures



3. Neural Networks: code example

```
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
```

THANK YOU