

# Лекция 4:

# Нейронные сети и обратное распространение ошибки

# Напомним...

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

Оценки линейного классификатора

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

SVM loss (или softmax)

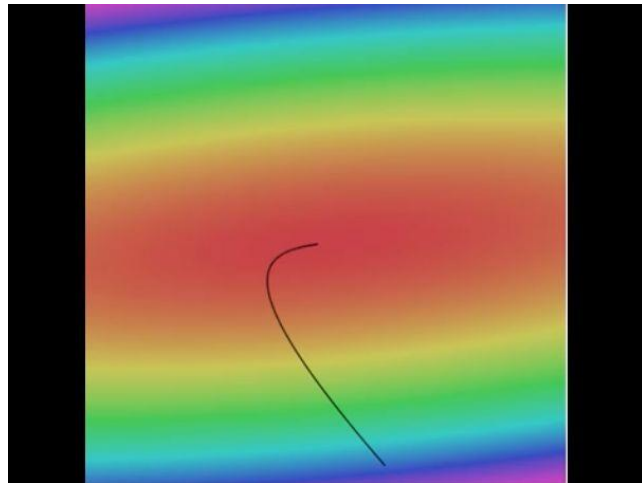
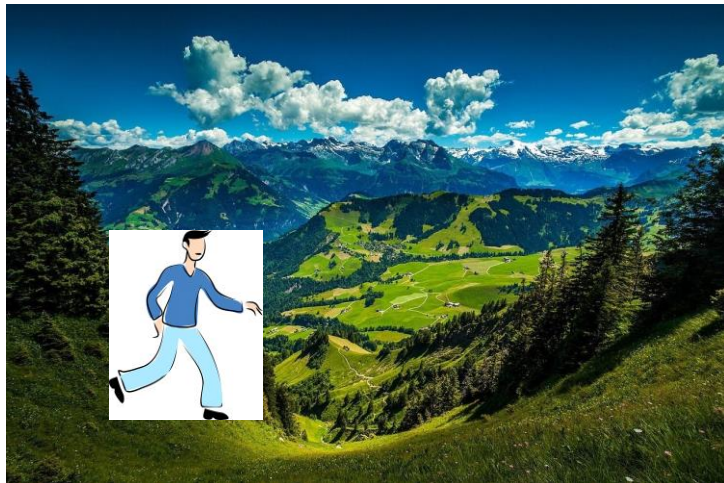
$$L = \frac{1}{N} \sum_{i=1}^N L_i + \lambda \sum_k W_k^2$$

data loss + regularization

Как найти  $W$ ?

$$\nabla_W L$$

# Поиск $W$ : градиентный спуск



```
# 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

# Градиентный спуск

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

- Численный градиент медленный, приближенный, но простой в реализации
- Аналитический градиент: точный, быстрый, можно ошибиться!

=>

На практике: Используем аналитический градиент но сверяем с численным - **gradient check**.

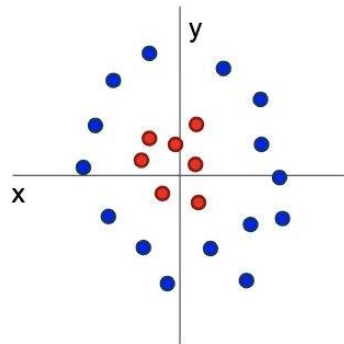
# Проблема: Линейная классификация не очень хороша...

## Visual Viewpoint



Линейный классификатор  
делает по одному шаблону на  
класс

## Geometric Viewpoint



Линейный классификатор  
требуется линейной  
разделимости классов

# Сопоставление с шаблоном – плохо

## Признаками выступают пикселы...

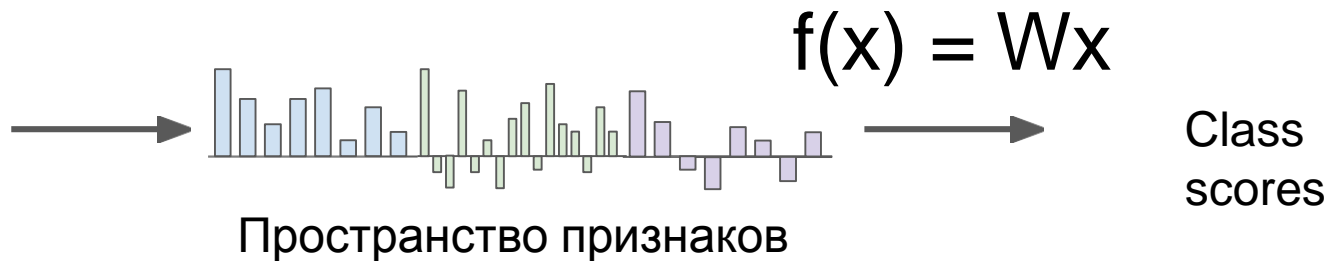


$$f(x) = Wx$$

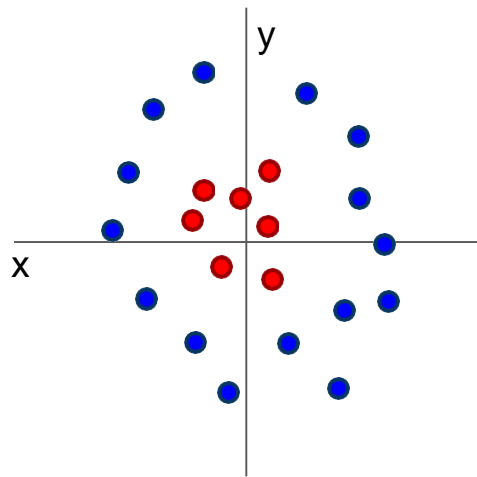
Оценки  
для  
классов



# Выделение признаков из изображения – лучше!

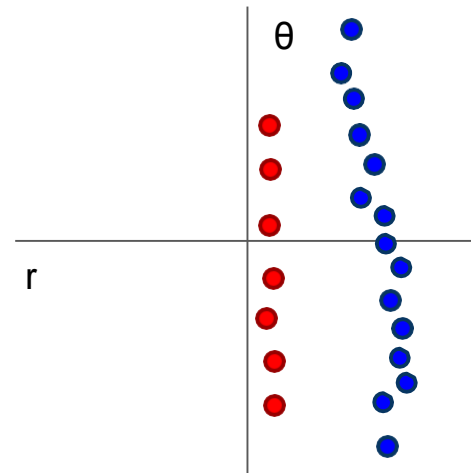


# Признаковое представление



Линейно  
неразделимые  
классы

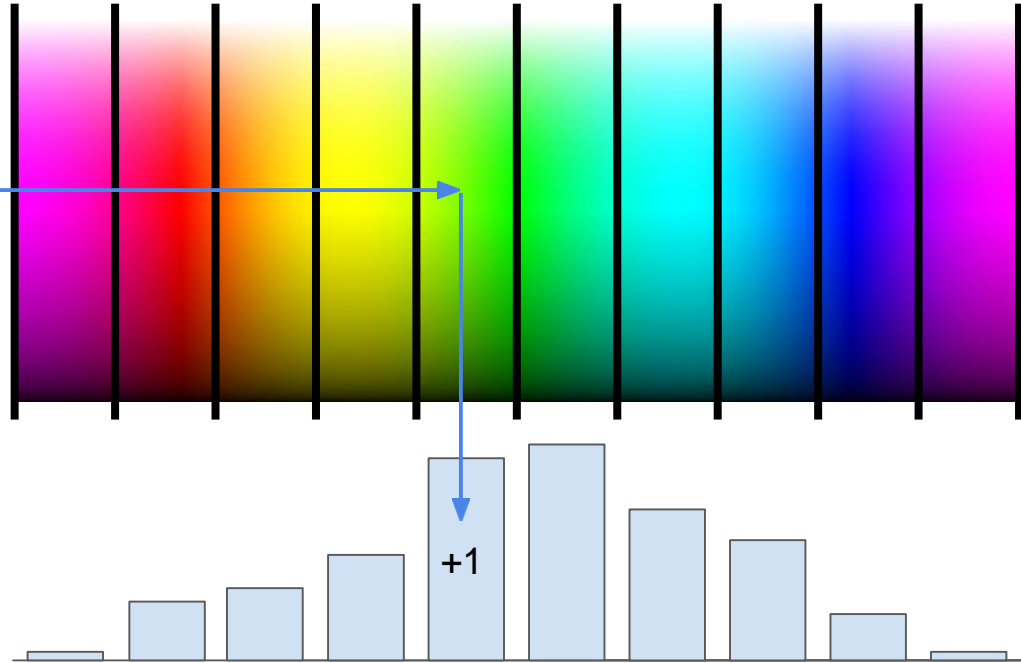
$$f(x, y) = (r(x, y), \theta(x, y))$$



Становятся  
разделимыми после  
выделения  
признаков



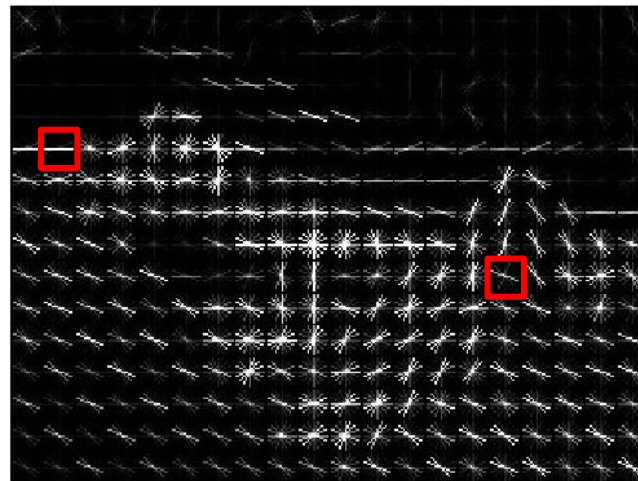
# Пример выделения признаков: цветовые гистограммы



# Пример: Histogram of Oriented Gradients (HoG)



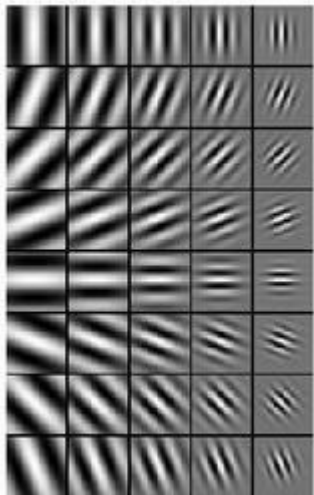
Разобьем на области 8x8 пикселей  
В каждом регионе подсчитаем 9  
направлений градиента



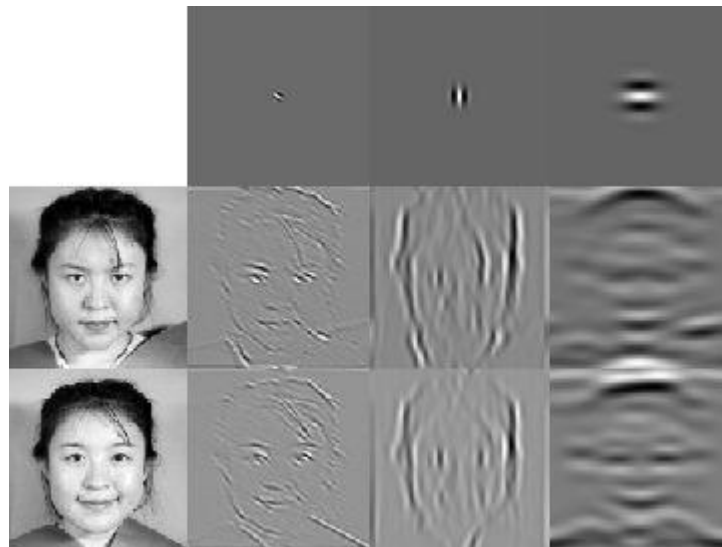
Пример: 320x240 пикселей  
поделили на 40x30 областей; в  
каждой области 9 значений  
градиента размер вектора  
признаков  $30 \times 40 \times 9 = 10\,800$  вместо  
 $320 \times 240 \times 3 = 230\,400$

Lowe, "Object recognition from local scale-invariant features", ICCV 1999  
Dalal and Triggs, "Histograms of oriented gradients for human detection," CVPR 2005

# Пример: Фильтры Габора



Примеры фильтров Габора  
разных размеров и ориентаций



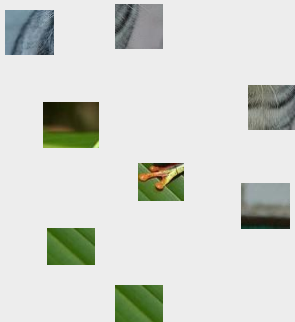
Применение фильтров Габора

# Пример: визуальное кодирование (bag of words)

## 1: Формируем словарь



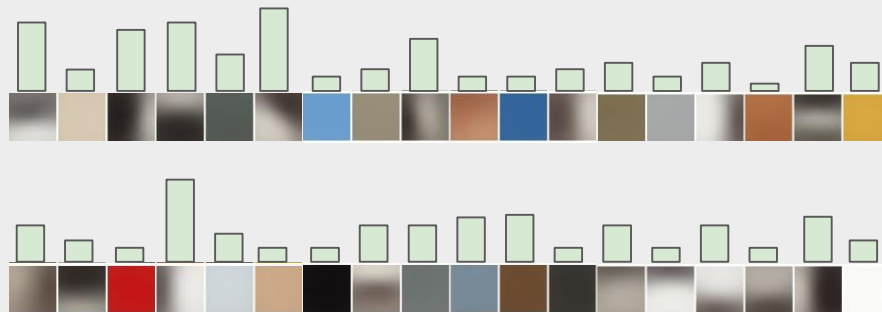
Выделяем случайные  
фрагменты (патчи)



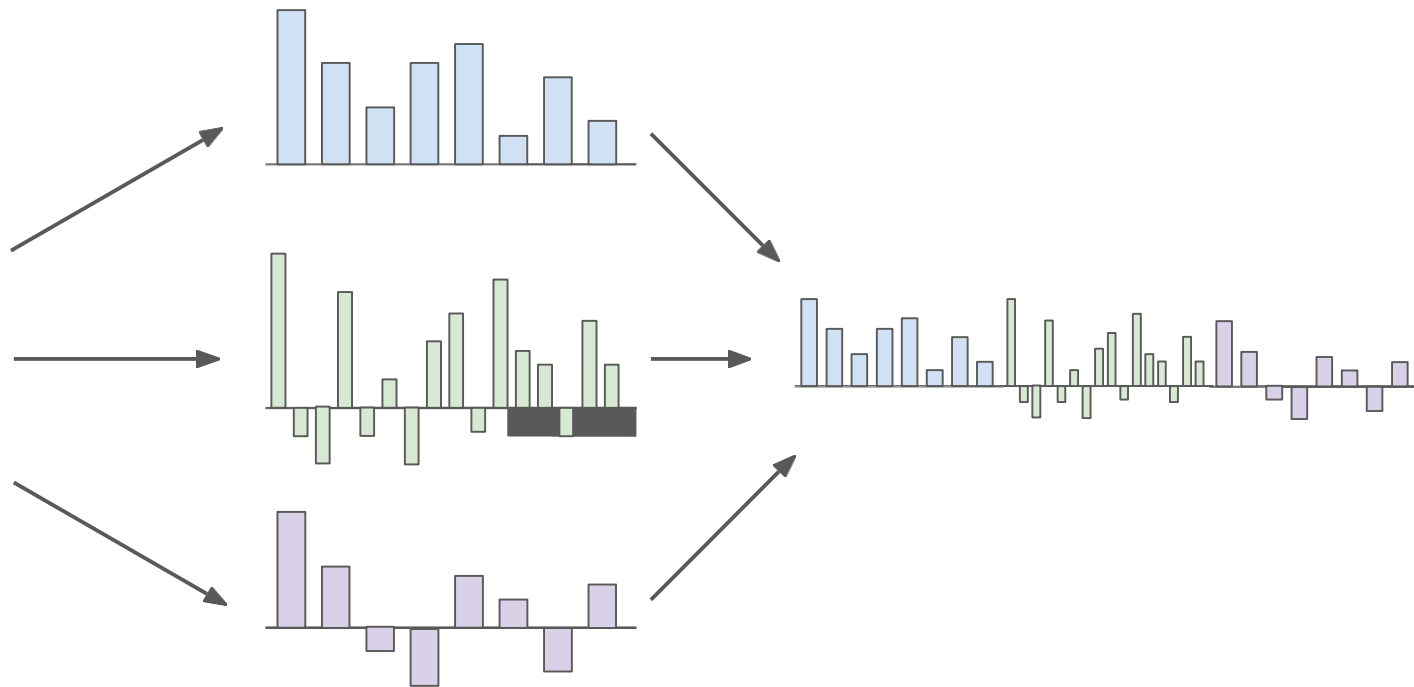
Кластеризуем  
патчи чтобы  
построить  
«словарь»



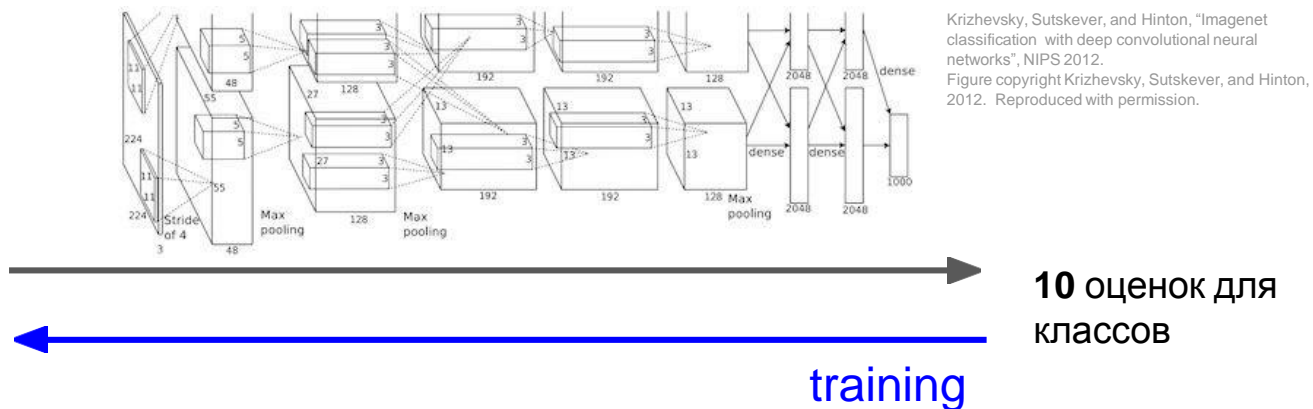
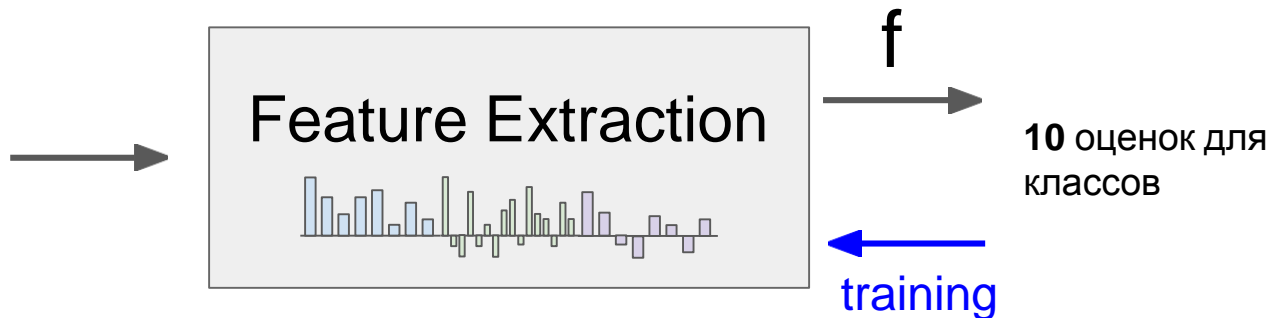
## 2: Кодлируем картинку по словарю



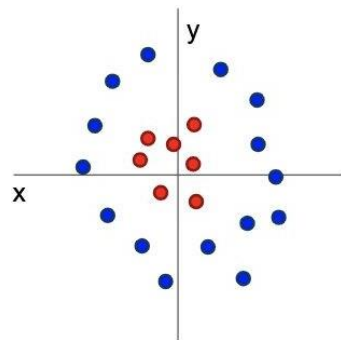
# Image Features



# Image features vs ConvNets

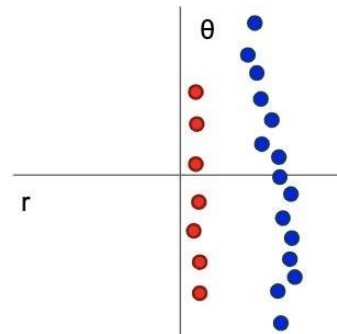


# One Solution: Feature Transformation



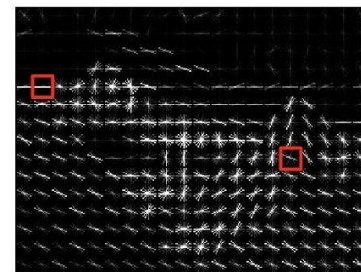
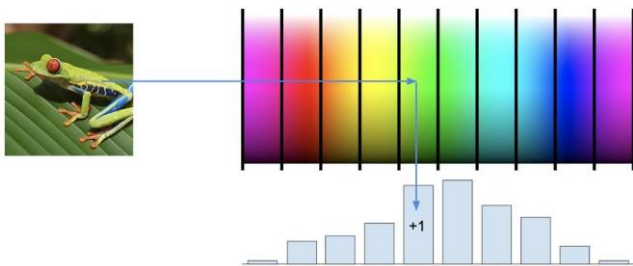
$$f(x, y) = (r(x, y), \theta(x, y))$$

Выделяем признаки, потом  
выполняем классификацию



Color Histogram

Histogram of Oriented Gradients (HoG)



# Нейронные сети



# Нейронные сети

(**Было**) линейная оценка:

$$f = Wx$$

(**Стало**) 2-слоя НС

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

$$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$$

“Нейронные сети” общий термин; эти сети называют полносвязными  
“fully-connected networks” или “multi-layer perceptrons” (MLP)

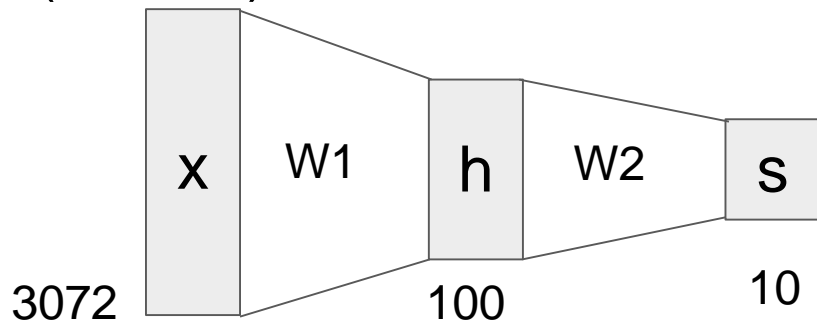
# Нейронные сети

(Было) линейная оценка:

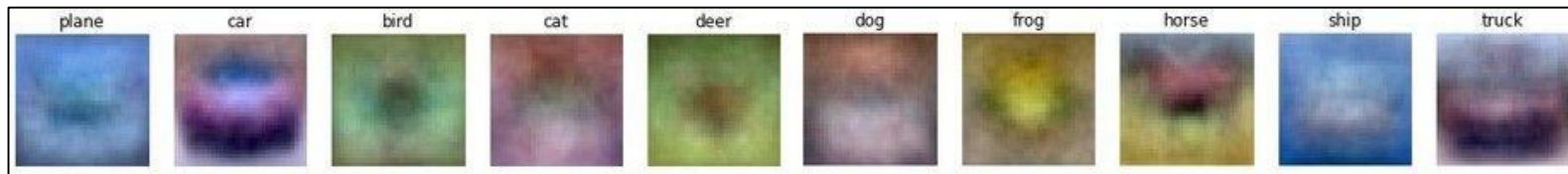
$$f = Wx$$

(Стало) 2-слоя НС

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



$$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$$



100 шаблонов вместо 10

Общие шаблоны для классов

# Нейронные сети

(**Было**) линейная оценка:

$$f = Wx$$

(**Стало**) 2-слоя НС

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

Функция  $\max(0, z)$  называется **функция активации**.

**Q:** что будет если ее убрать?

# Нейронные сети

(**Было**) линейная оценка:

$$f = Wx$$

(**Стало**) 2-слоя НС

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

Функция  $\max(0, z)$  называется **функция активации**.

**Q:** что будет если ее убрать?

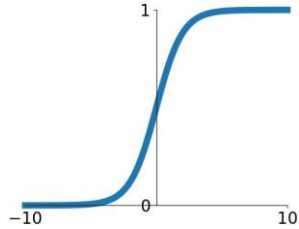
$$f = W_2 W_1 x \quad W_3 = W_2 W_1 \in \mathbb{R}^{C \times H}, \quad \underline{f = W_3 x}$$

**A:** опять получим линейный классификатор!

# Функции активации

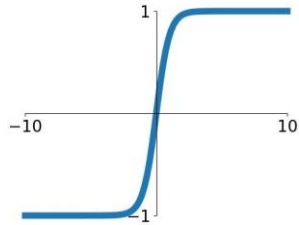
## Sigmoid

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



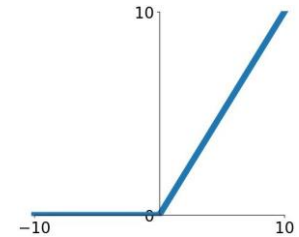
## tanh

$$\tanh(x)$$



## ReLU

$$\max(0, x)$$

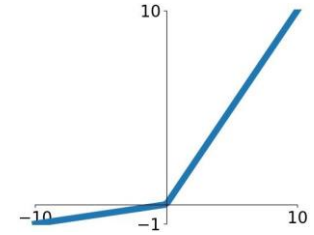


Rectified Linear Unit

ReLU выбор по умолчанию

## Leaky ReLU

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

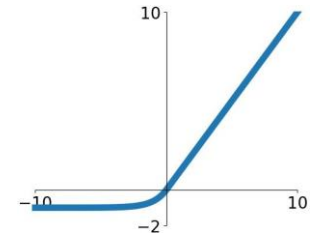


## Maxout

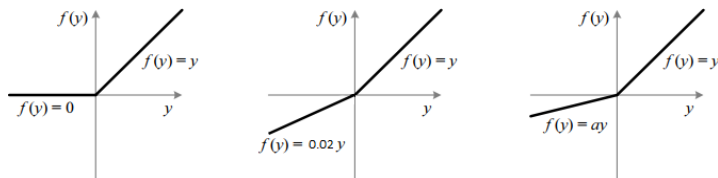
$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

## ELU

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



# PReLU - Parametric ReLU



[www.cv-foundation.org](http://www.cv-foundation.org) > He\_... ▾ PDF [Перевести эту страницу](#)

## Delving Deep into Rectifiers: Surpassing Human-Level ...

**Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet**

**Classification.** Kaiming He. Xiangyu Zhang. Shaoqing Ren. Jian Sun.

автор: К Хе - 2015 - [Цитируется: 9211](#) - [Похожие статьи](#)

## Сравните с цитируемостью работ Колмогорова и Цибенко

[link.springer.com](http://link.springer.com) > article - [Перевести эту страницу](#)

### Approximation by superpositions of a sigmoidal function ...

Jones, Constructive **approximations** for neural networks by **sigmoidal functions**, Technical Report Series, No. 7, Department of Mathematics, University of Lowell, ...

автор: G Cybenko - 1989 - [Цитируется: 13151](#) - [Похожие статьи](#)

[www.mathnet.ru](http://www.mathnet.ru) > dan22050 - [Перевести эту страницу](#)

### A. N. Kolmogorov, "On the representation of continuous ...

**On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition** A. N. Kolmogorov Full text: ...

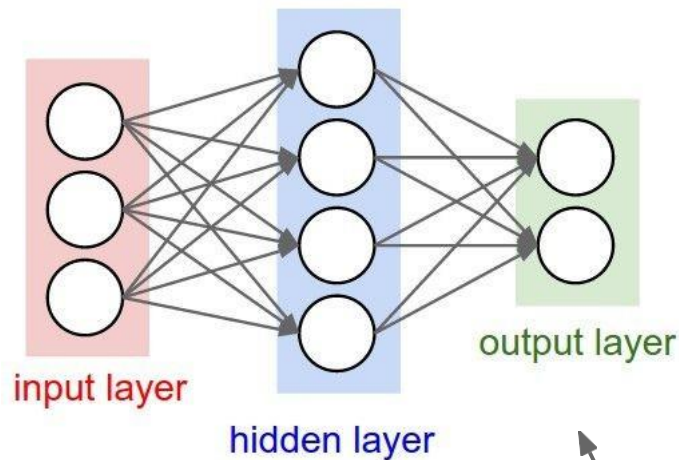
автор: AN Kolmogorov - 1957 - [Цитируется: 1194](#) - [Похожие статьи](#)

$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \leq 0 \end{cases}$$

$$f(x_1, \dots, x_n) = \sum_{i=1}^{2n+1} g_i \left( \sum_{j=1}^n \phi_{ji}(x_j) \right)$$

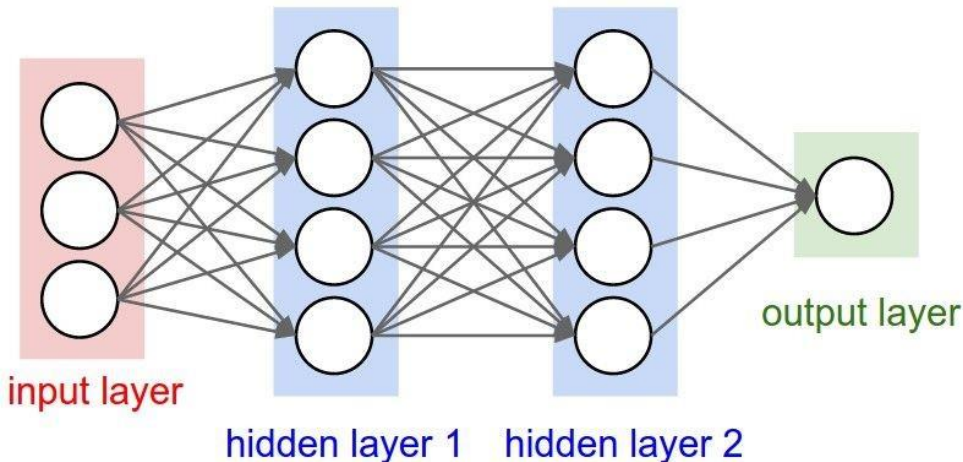
Kolmogorov's Theorem (1957)

# Нейронные сети: архитектура



“2-layer Neural Net”, or  
“1-hidden-layer Neural Net”

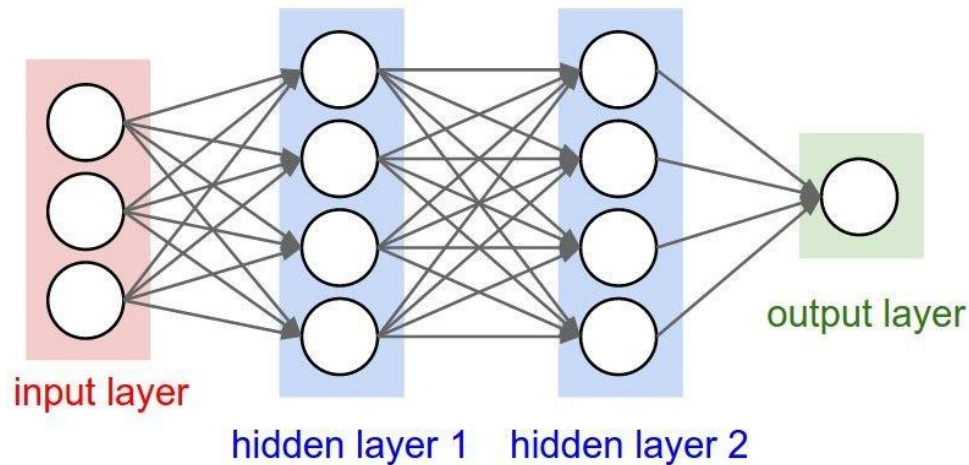
Двухслойная сеть,  
Сеть с одним скрытым  
слоем



“3-layer Neural Net”, or  
“2-hidden-layer Neural Net”

**“Fully-connected” layers – полносвязные слои**

# Пример: прямой проход НС (forward pass)



```
# 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)
```



# Реализация обучения двухслойной сети ~20 строчек:

```
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
```

Прямой проход

Расчет градиента  
(обратный проход)

Обновление весов

# Реализация обучения двухслойной сети ~20 строчек:

```
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
```

Определим сеть.

x, y – вход и выход

w1, w2 – матрицы весов

# Реализация обучения двухслойной сети ~20 строчек:

```
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
```

Определение сети

Прямой проход (forward pass)

Сигмоидальная активация  
sigmoid activation function

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

# Реализация обучения двухслойной сети ~20 строчек:

```
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
```

Определение сети

Forward pass

Аналитический расчет градиента

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

# Реализация обучения двухслойной сети ~20 строчек:

```
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
```

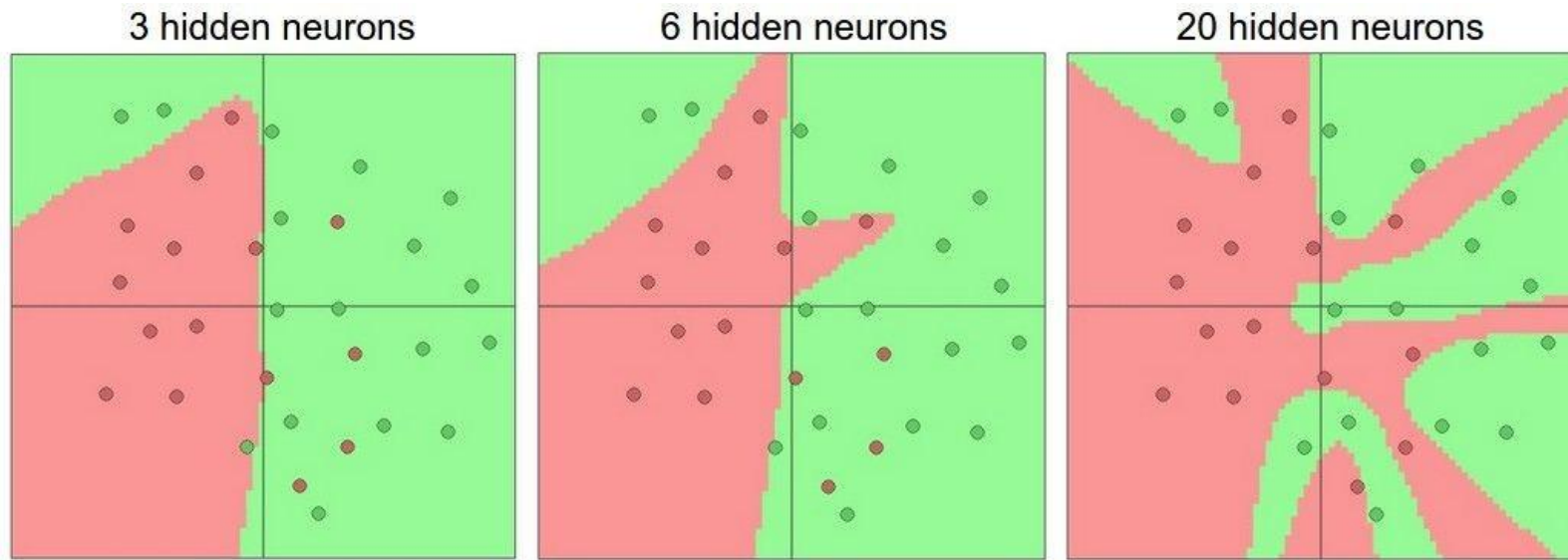
Определение сети

Forward pass

Аналитический расчет градиента

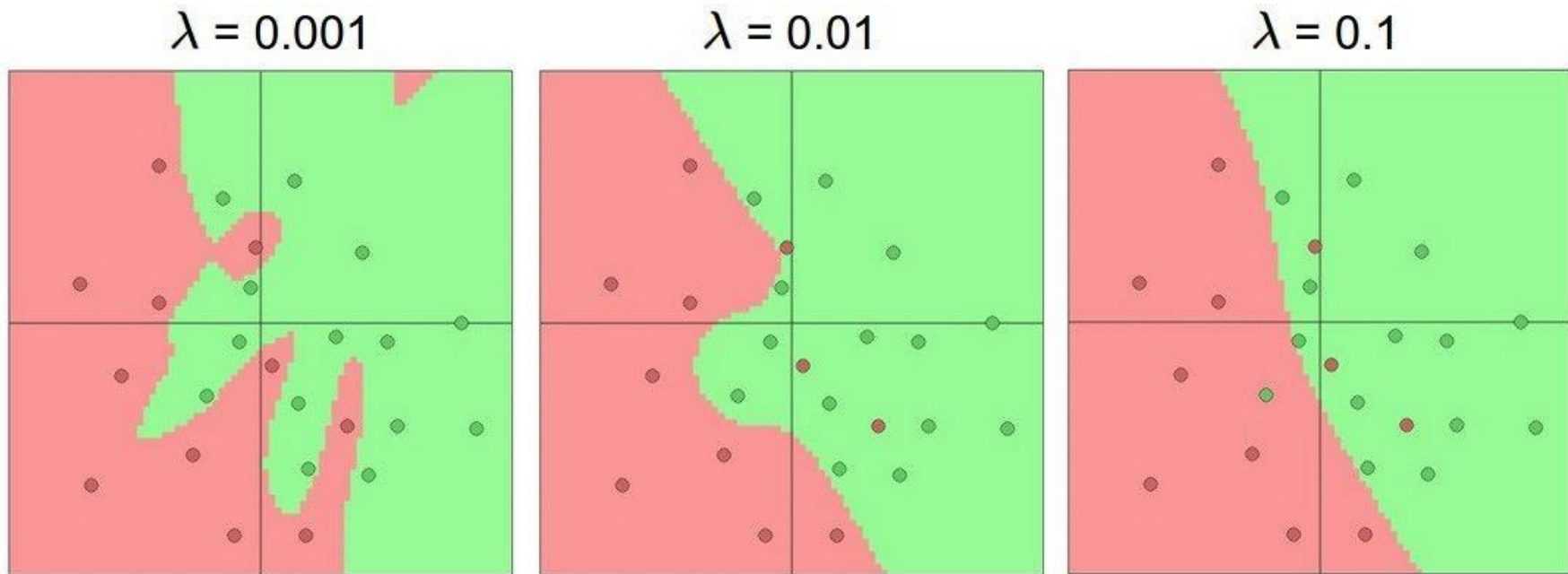
Градиентный спуск  
Gradient descent

# Размер и число слоев



↑  
Больше нейронов точнее

## Регуляризация:



(Демо ConvNetJS:

<http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html>)

$$L(W) = \frac{1}{N} \sum_{i=1}^N L_i(f(x_i, W), y_i) + \lambda R(W)$$

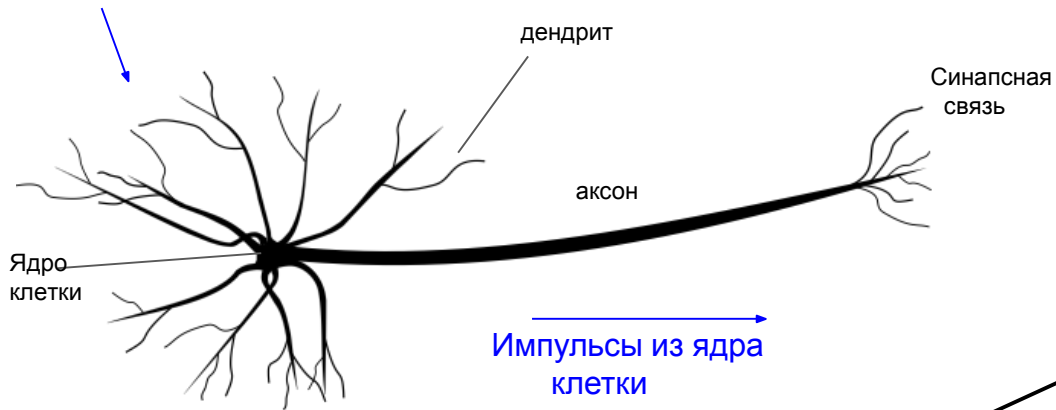
# Аналогия с мозгом



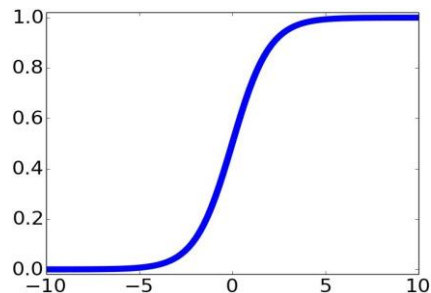
[This image](#) by [Fotis Bobolas](#) is  
licensed under [CC-BY 2.0](#)



## Импульсы в ядро клетки

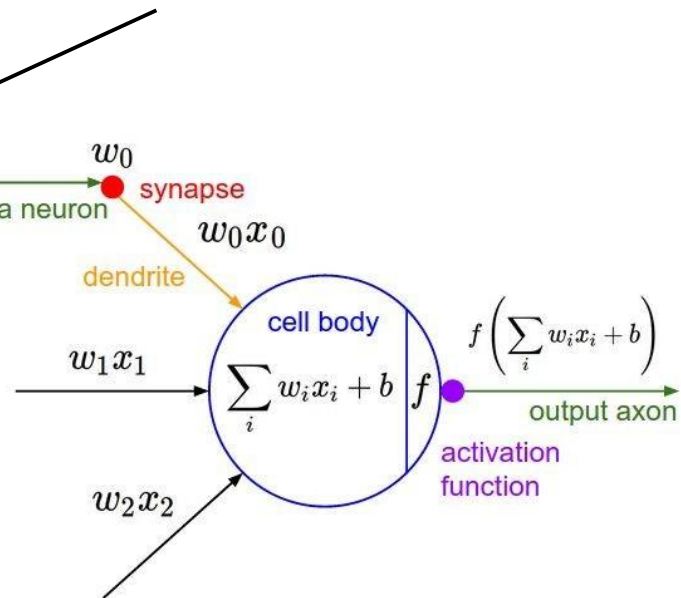


This image by Felipe Peruchio is licensed under [CC-BY 3.0](https://creativecommons.org/licenses/by/3.0/)

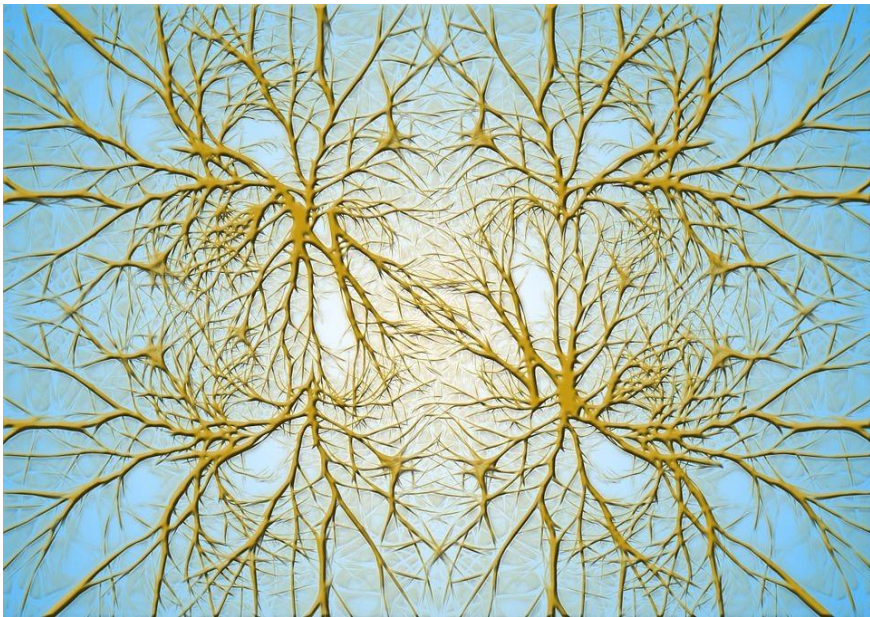


Сигмоидальная активация

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

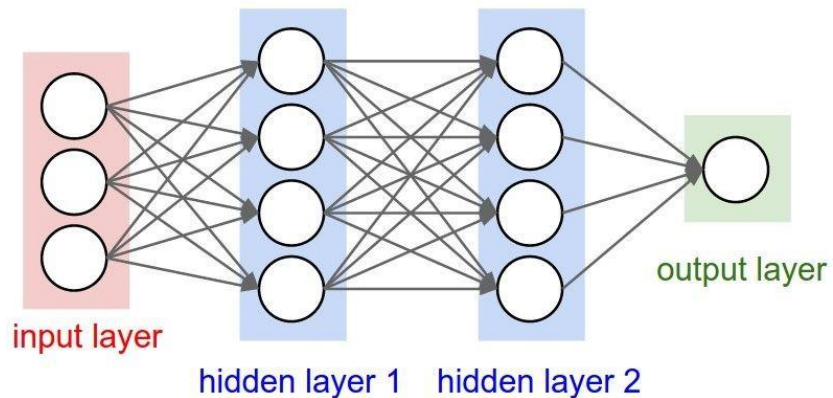


## Биологические нейроны: Сложная связность

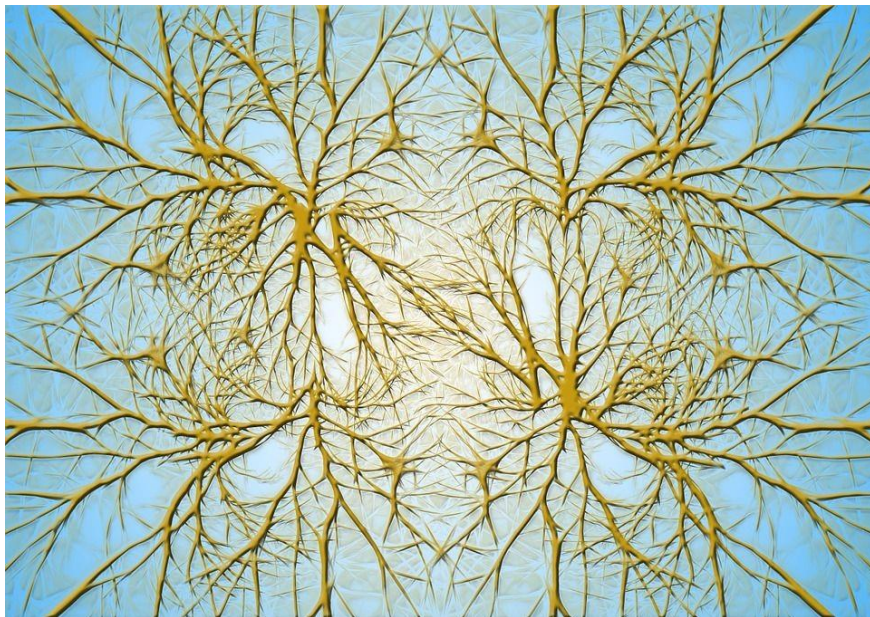


[This image is CC0 Public Domain](#)

## Искусственная НС: слоистая организация

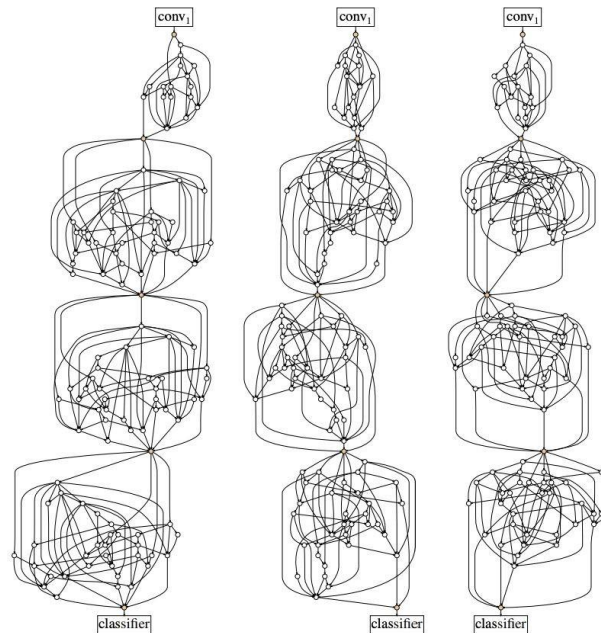


## Биологические нейроны: Сложная связность



[This image is CC0 Public Domain](#)

## Искусственная НС: Случайная связность тоже работает!



Xie et al, "Exploring Randomly Wired Neural Networks for Image Recognition", arXiv 2019



# Нейробиологические аналогии не работают!

## Биологические нейроны:

- Разные типы нейронов, импульсная активация!
- Возможна нелинейная связность в дендритах
- Синапсы не просто вес, а сложная нелинейность

[Dendritic Computation. London and Hausser]

**Michael Jordan:** Well, I want to be a little careful here. I think it's important to distinguish two areas where the word *neural* is currently being used.

One of them is in deep learning. And there, each “neuron” is really a cartoon.

<https://spectrum.ieee.org/artificial-intelligence/machine-learning/machinelearning-maestro-michael-jordan-on-the-delusions-of-big-data-and-other-huge-engineering-efforts>



# Главный вопрос: как считать градиенты?

$$s = f(x; W_1, W_2) = W_2 \max(0, W_1 x) \quad \text{Нелинейная оценка}$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \quad \text{Функция потерь: SVM Loss}$$

$$R(W) = \sum_k W_k^2 \quad \text{Регуляризация}$$

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \lambda R(W_1) + \lambda R(W_2) \quad \text{Итого: функция потерь+регуляризация}$$

Если можем посчитать  $\frac{\partial L}{\partial W_1}, \frac{\partial L}{\partial W_2}$  то можем найти  $W_1$  и  $W_2$

# (Плохое) решение: вывести на бумаге $\nabla_W L$

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

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

$$= \sum_{j \neq y_i} \max(0, W_{j,:} \cdot x + W_{y_i,:} \cdot x + 1)$$

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

$$= \frac{1}{N} \sum_{i=1}^N \sum_{j \neq y_i} \max(0, W_{j,:} \cdot x + W_{y_i,:} \cdot x + 1) + \lambda \sum_k W_k^2$$

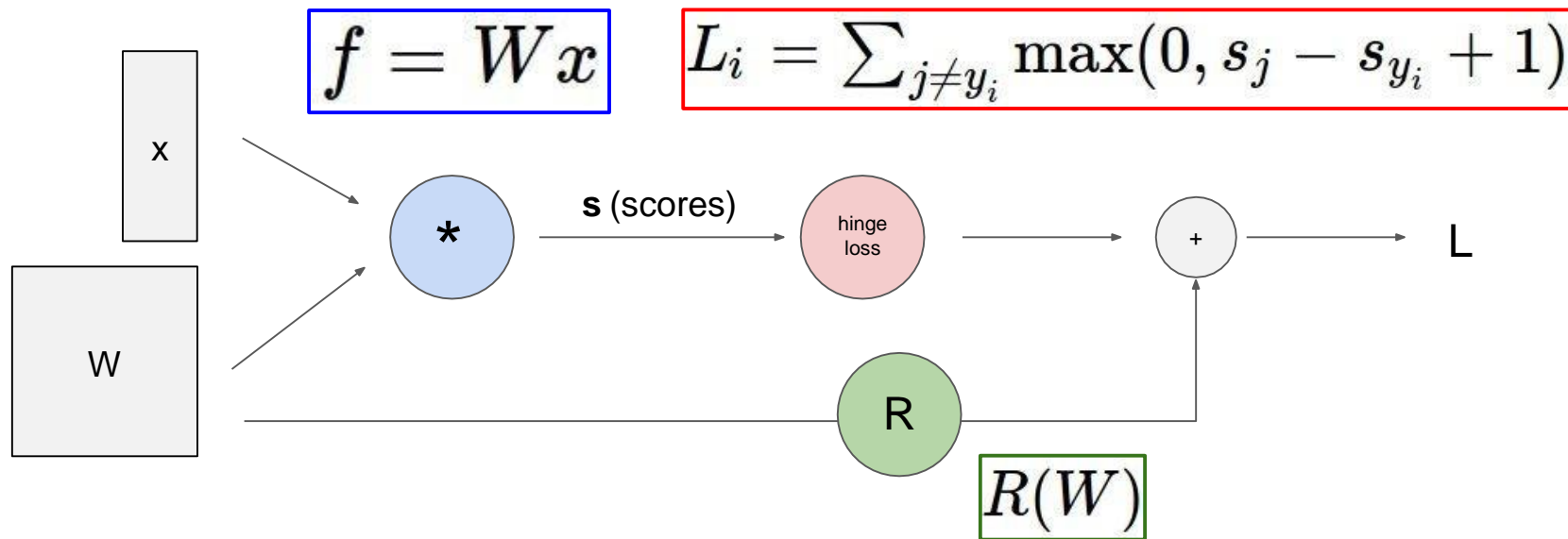
$$\nabla_W L = \nabla_W \left( \frac{1}{N} \sum_{i=1}^N \sum_{j \neq y_i} \max(0, W_{j,:} \cdot x + W_{y_i,:} \cdot x + 1) + \lambda \sum_k W_k^2 \right)$$

**Проблема:** Можно ошибиться

**Проблема:** а что если что то поменяется? Например мы захотели софтмакс? Все пересчитывать? А если еще один слой?

**Проблема:** не реализуемо для больших моделей с динамическим определением!

# Решение: Вычислительные графы + Backpropagation



# Convolutional network (AlexNet)

input image

weights

loss

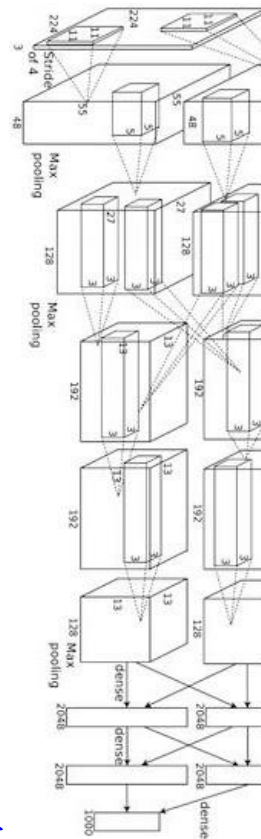


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.



# Neural Turing Machine

input image

loss

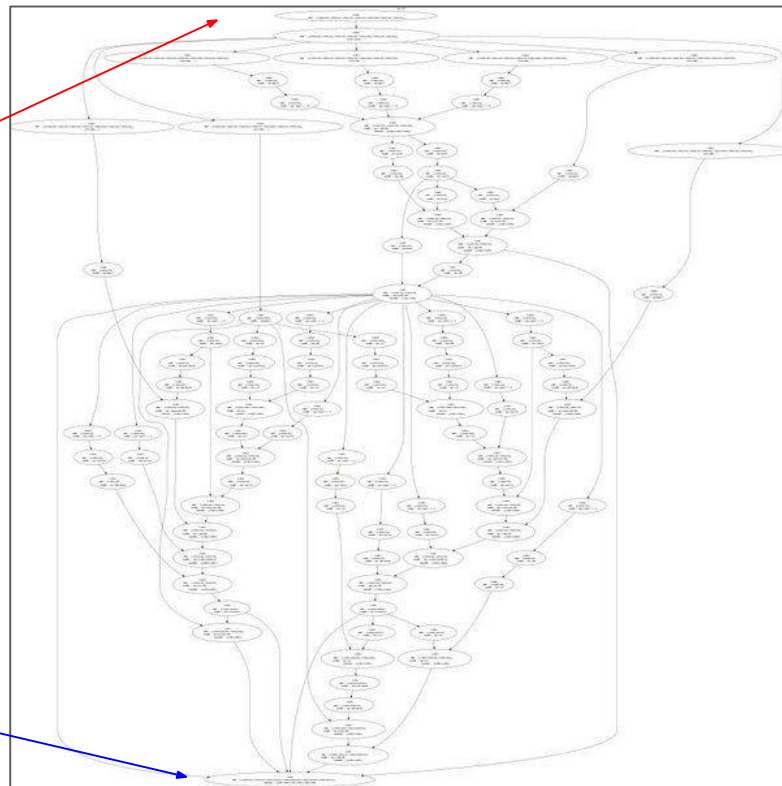


Figure reproduced with permission from a [Twitter post](#) by Andrej Karpathy.

Решение: Backpropagation

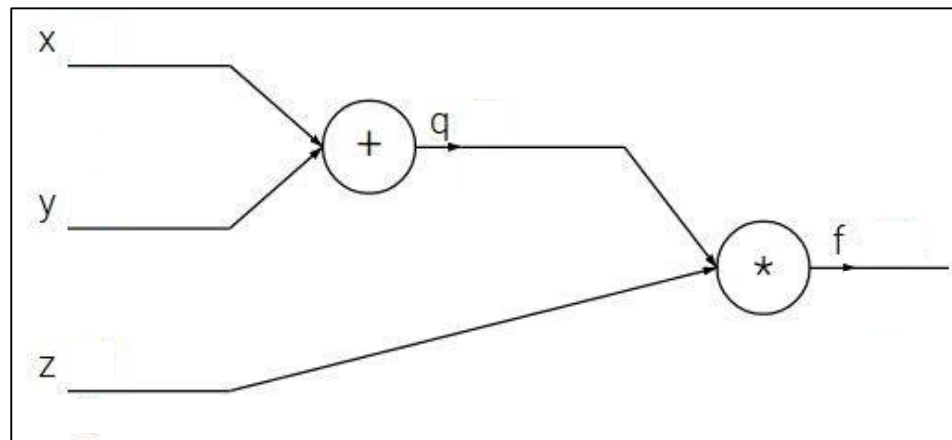
Обратное распространение ошибки

## Backpropagation: пример

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

## Backpropagation: пример

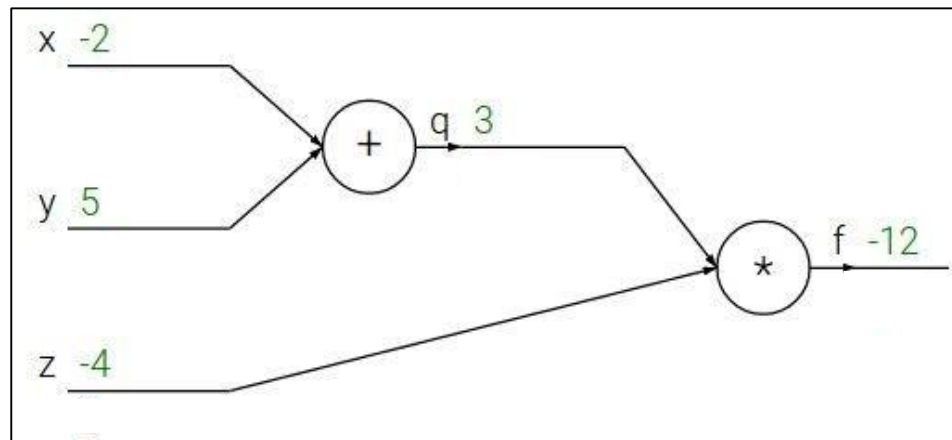
$$f(x, y, z) = (x + y)z$$



## Backpropagation: пример

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

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

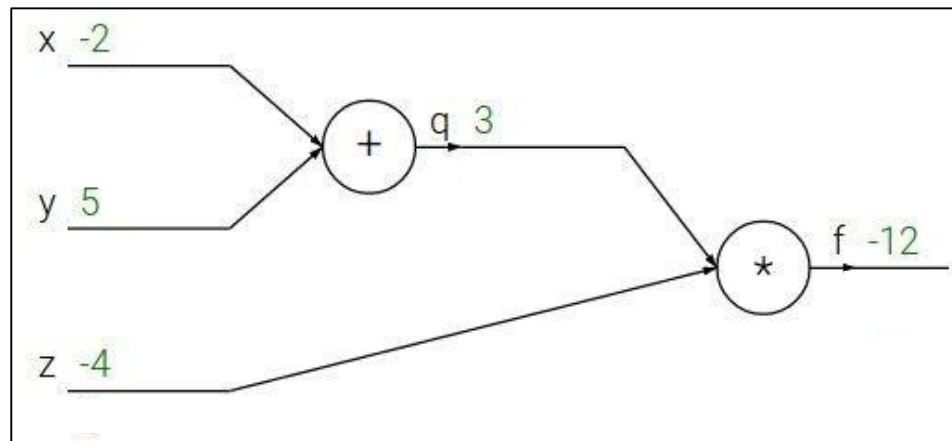


## Backpropagation: пример

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

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

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



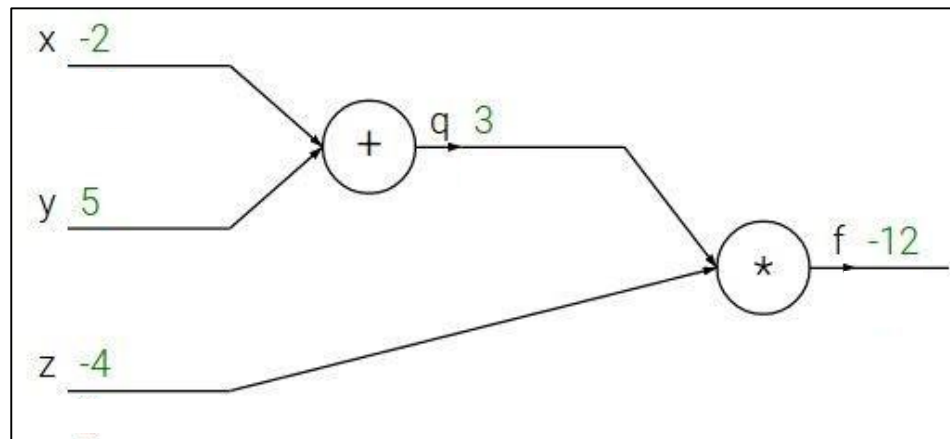
## Backpropagation: пример

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

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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



## Backpropagation: пример

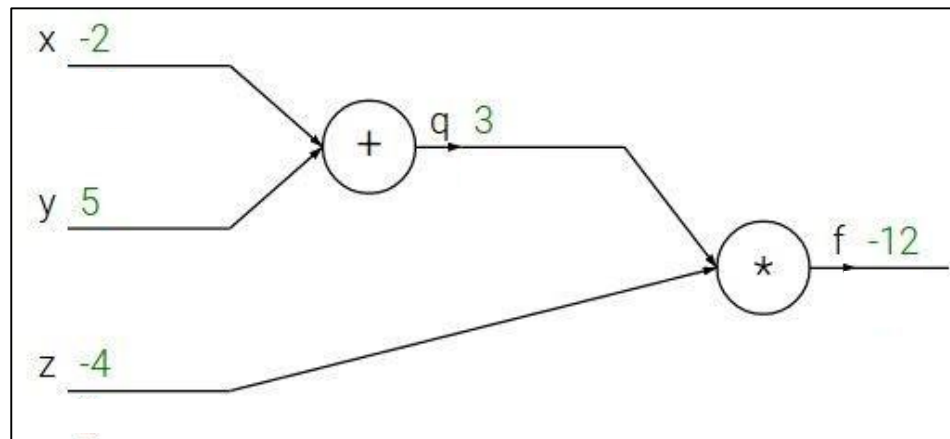
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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





## Backpropagation: пример

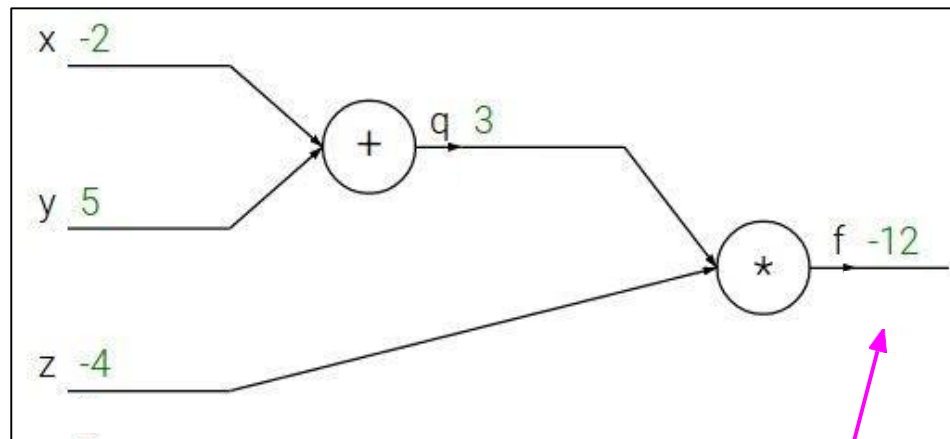
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



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

## Backpropagation: пример

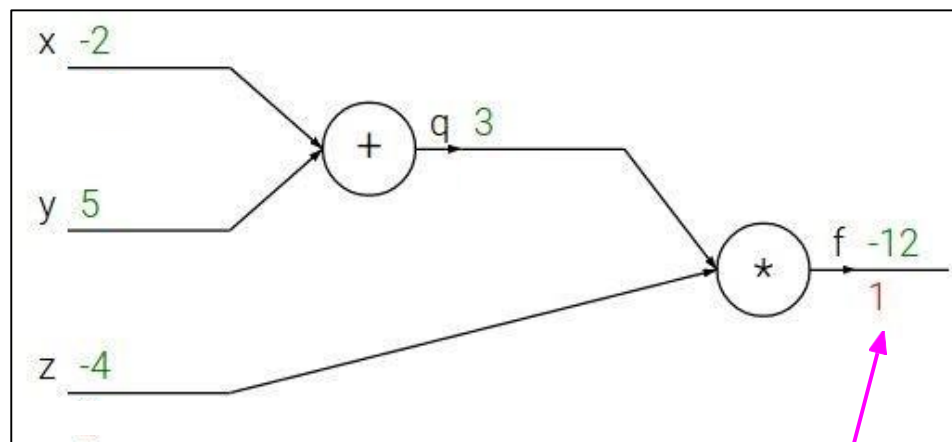
$$f(x, y, z) = (x + y)z$$

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



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

## Backpropagation: пример

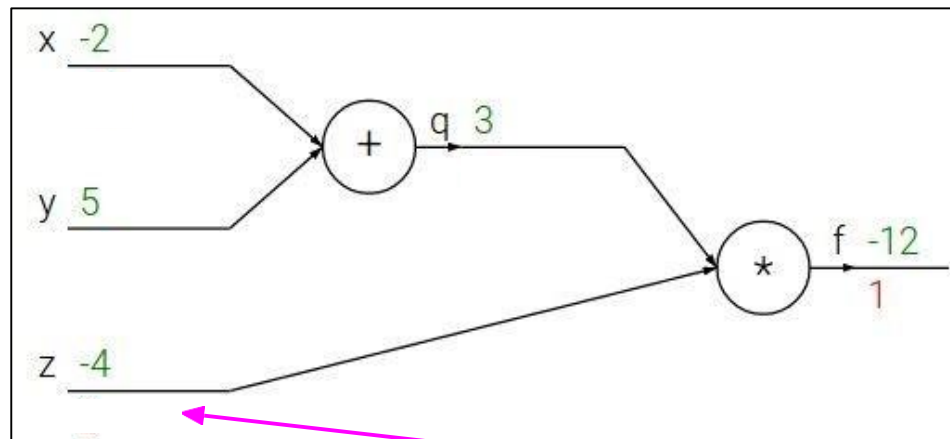
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



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

## Backpropagation: пример

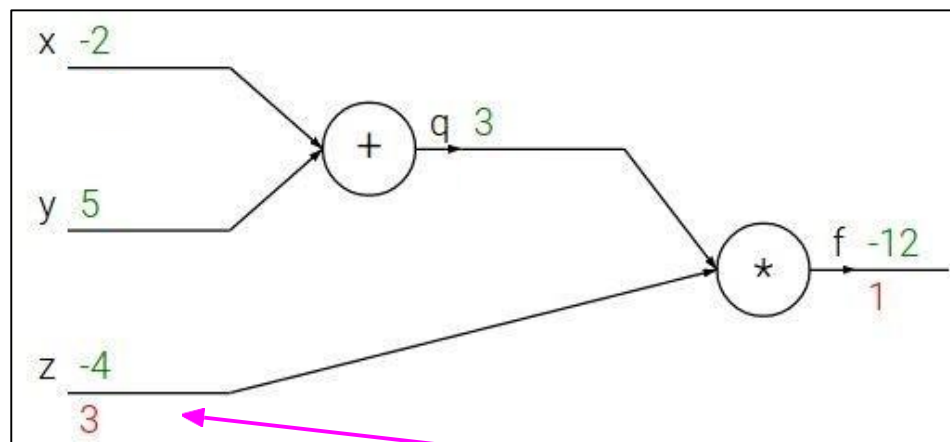
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



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

## Backpropagation: пример

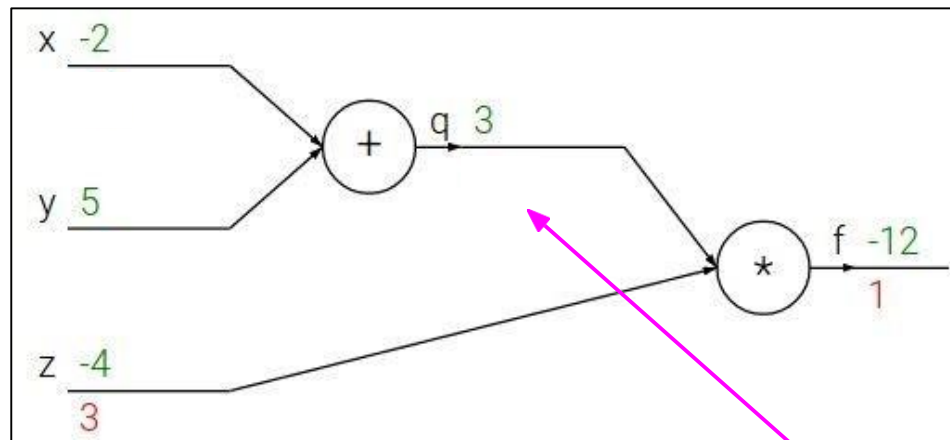
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



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

## Backpropagation: пример

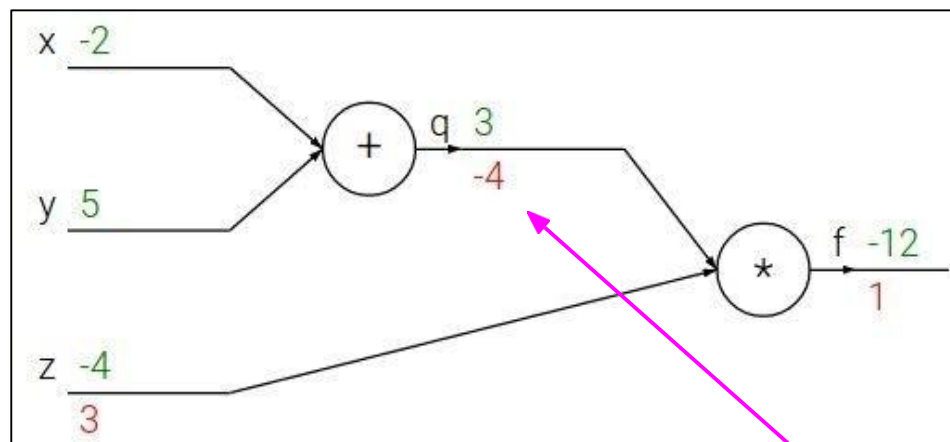
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



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

## Backpropagation: пример

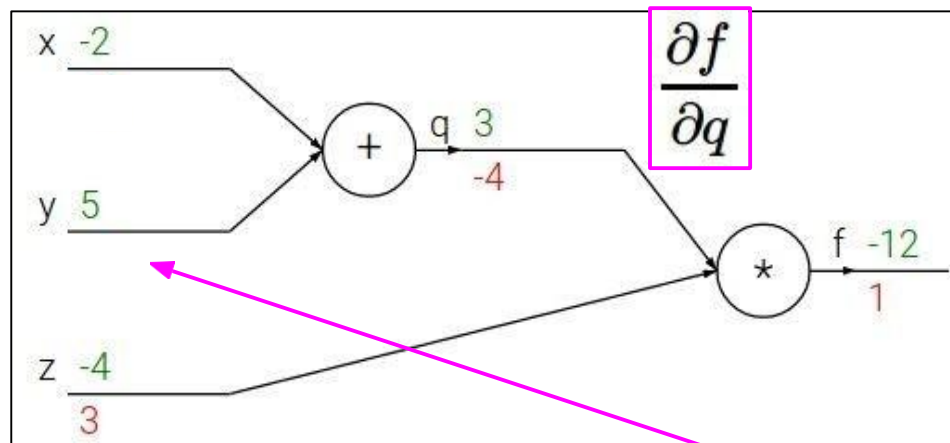
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



Chain rule:

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

Upstream  
gradient

Local  
gradient

Производная сложной функции  
Локальные и восходящие градиенты

## Backpropagation: пример

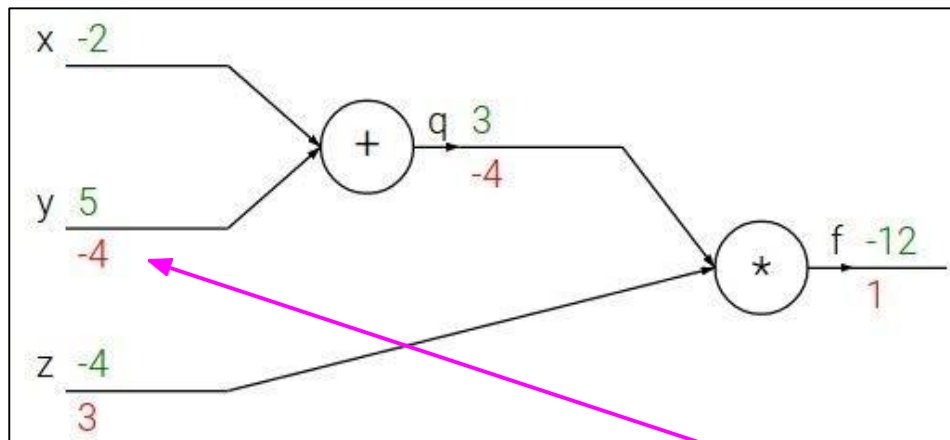
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



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

Chain rule:

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

Upstream  
gradient

Local  
gradient

Производная сложной функции  
Локальные и восходящие градиенты



## Backpropagation: пример

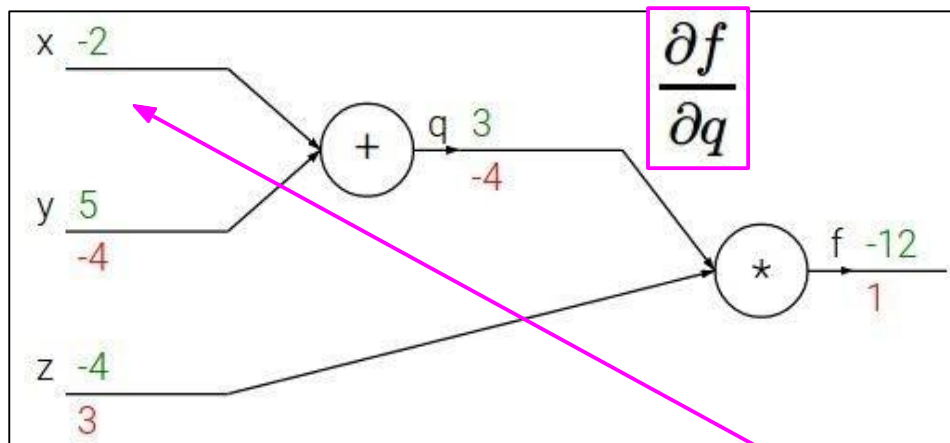
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



Chain rule:

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

Upstream  
gradient

Local  
gradient

Производная сложной функции  
Локальные и восходящие градиенты

## Backpropagation: пример

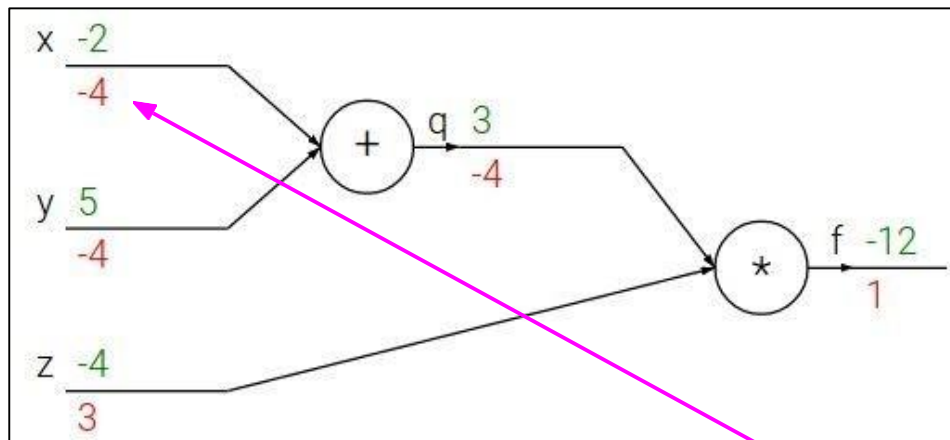
$$f(x, y, z) = (x + y)z$$

Пусть  $x = -2$ ,  $y = 5$ ,  $z = -4$

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

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

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



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

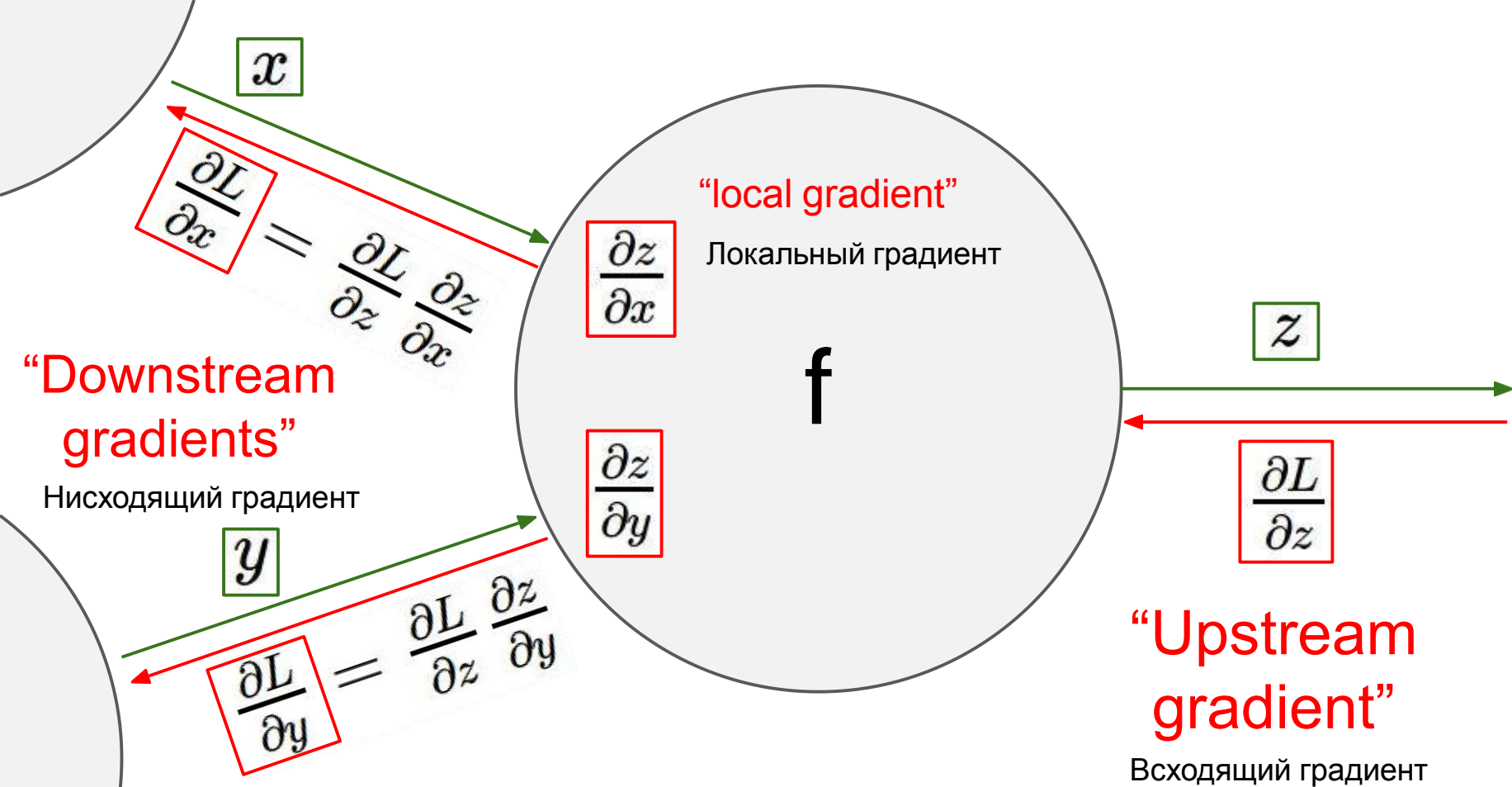
Chain rule:

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

Upstream  
gradient

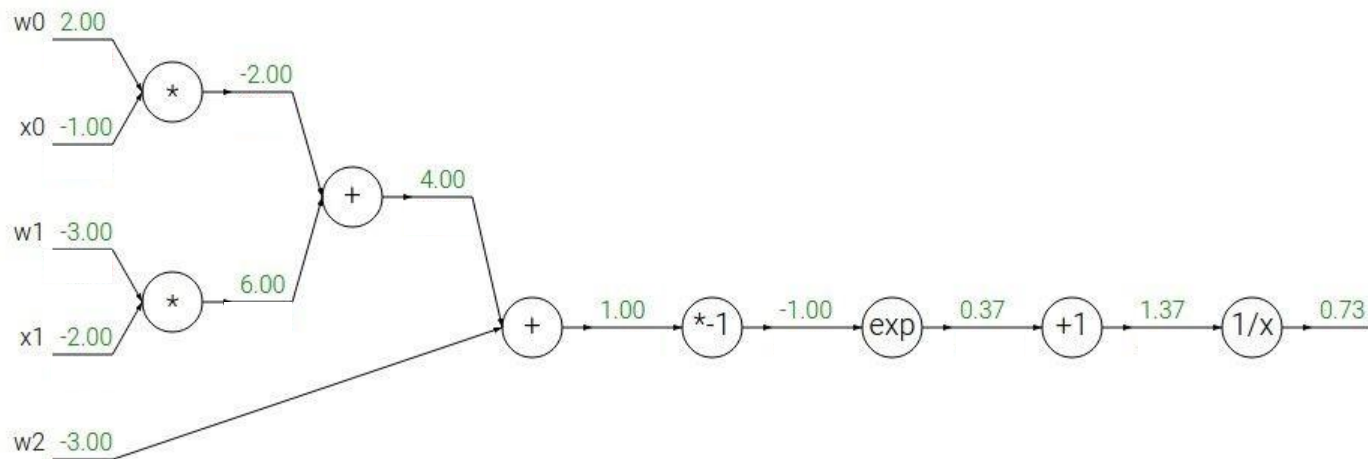
Local  
gradient

Производная сложной функции  
Локальные и восходящие градиенты



Another example:

$$f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

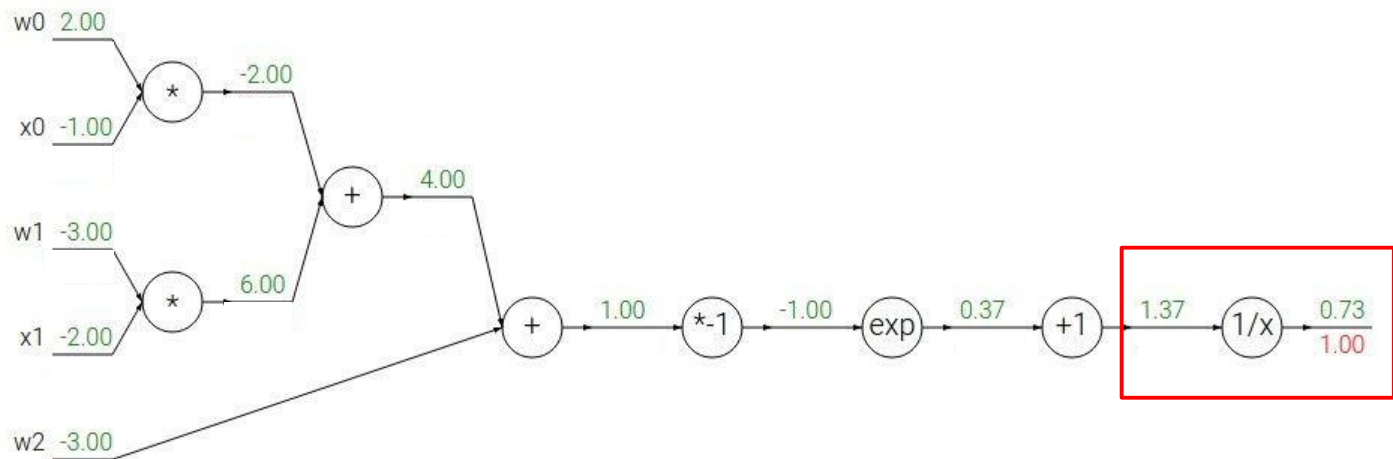
$$f_c(x) = c + x$$

→

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

Another example:

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



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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

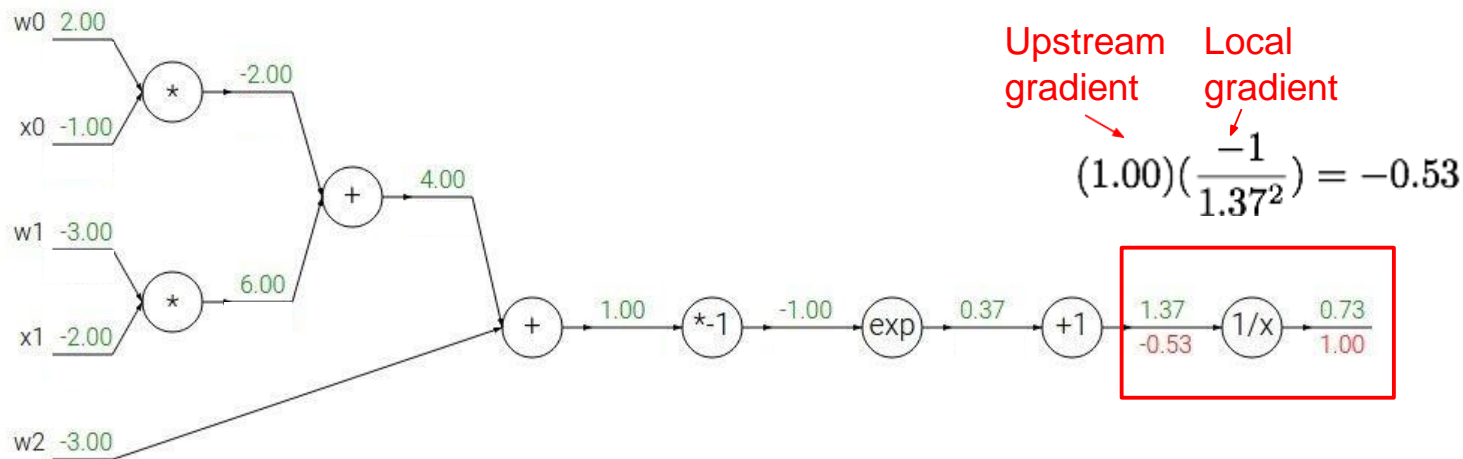
$$f_c(x) = c + x$$

→

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

Another example:

$$f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

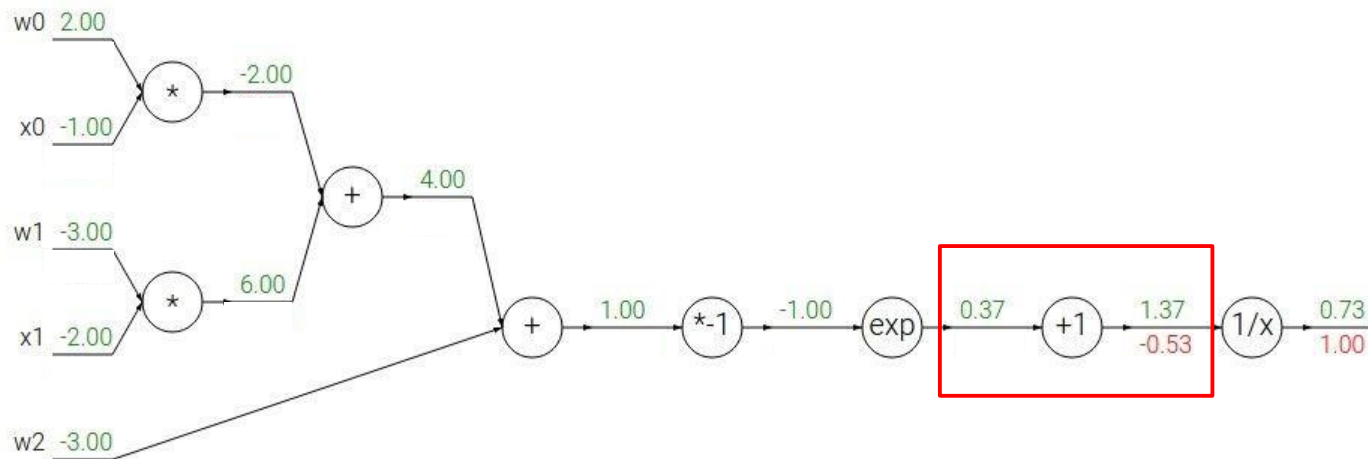
$$f_c(x) = c + x$$

→

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

Another example:

$$f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

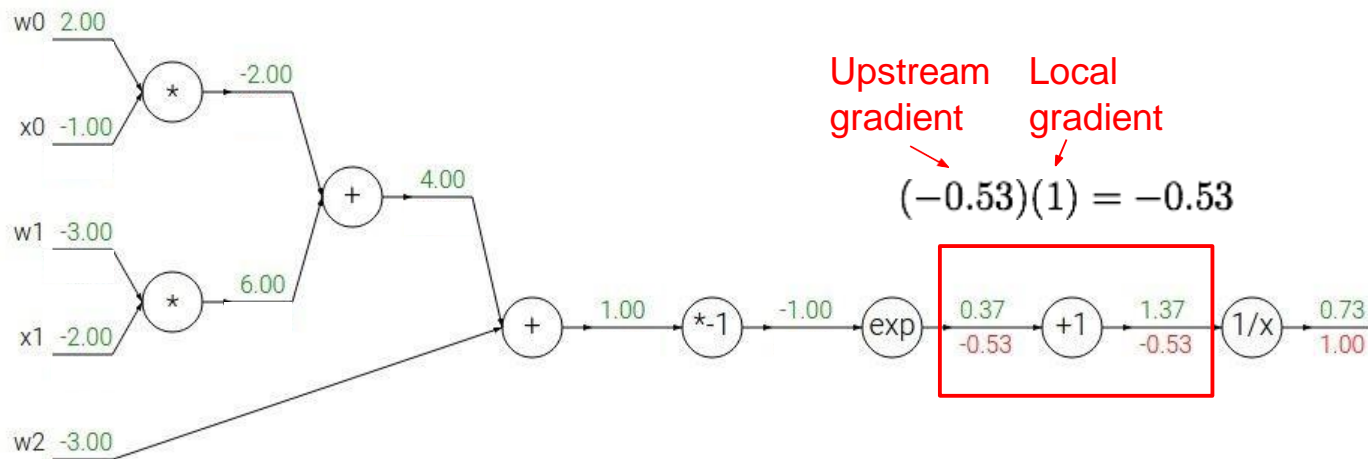
$$f_c(x) = c + x$$

→

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

Another example:

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



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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

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

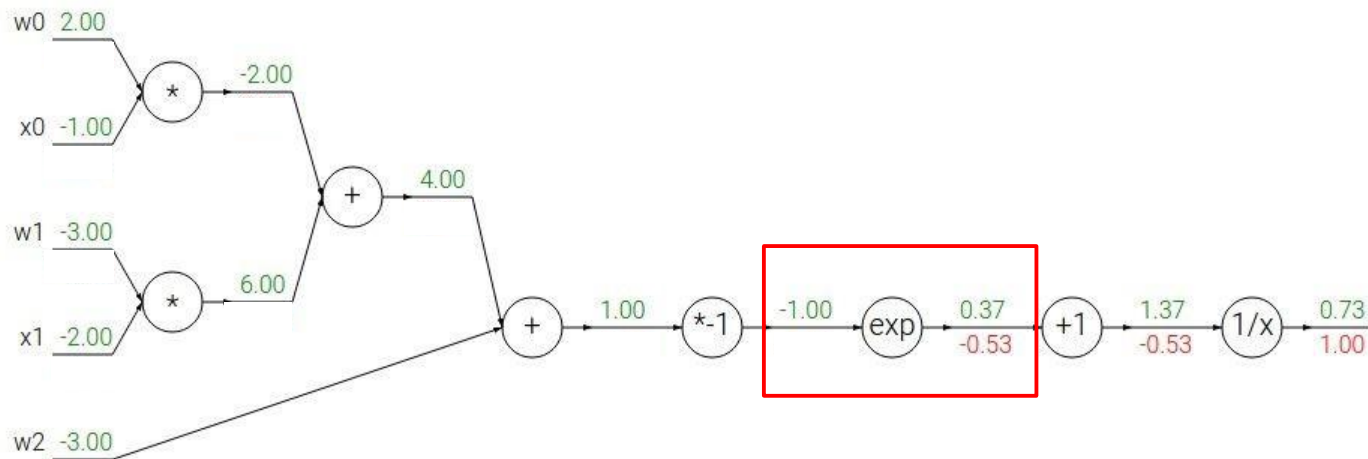
→

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



Another example:

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



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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

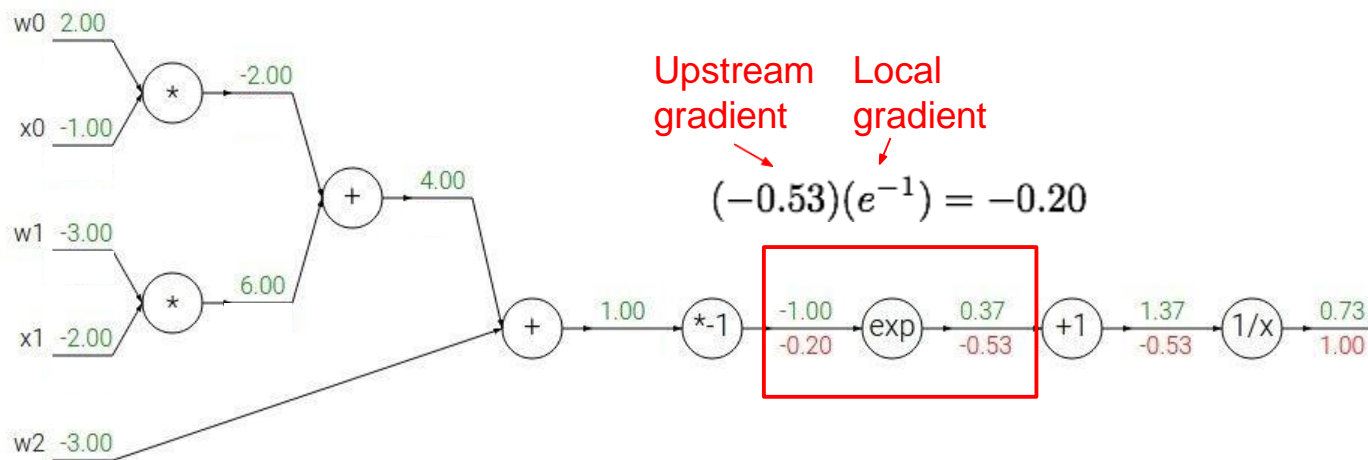
$$f_c(x) = c + x$$

→

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

Another example:

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



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

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

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

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

→

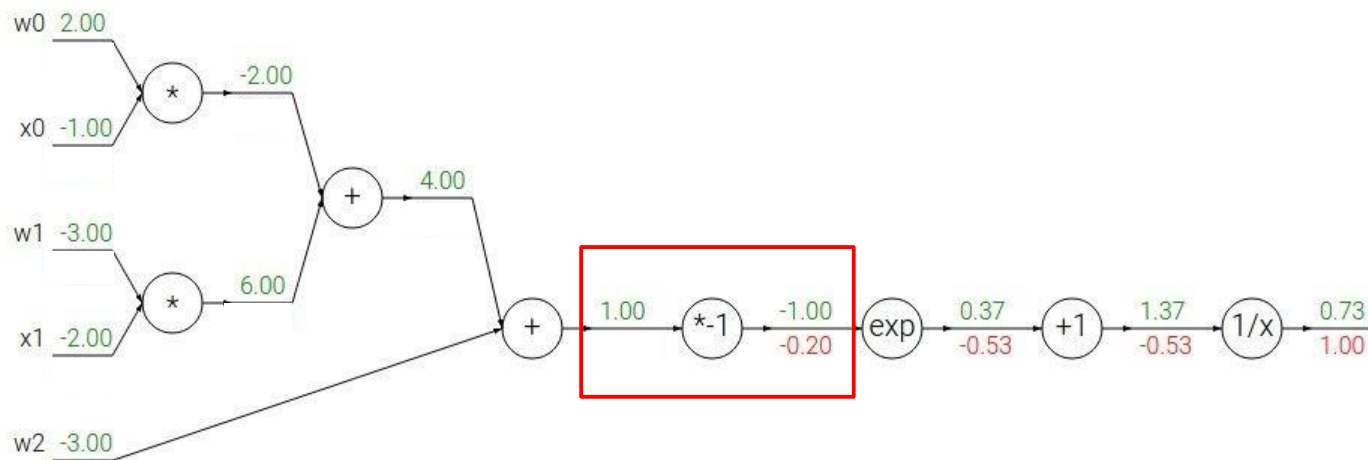
$$\frac{df}{dx} = -1/x^2$$

→

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

Another example:

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



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

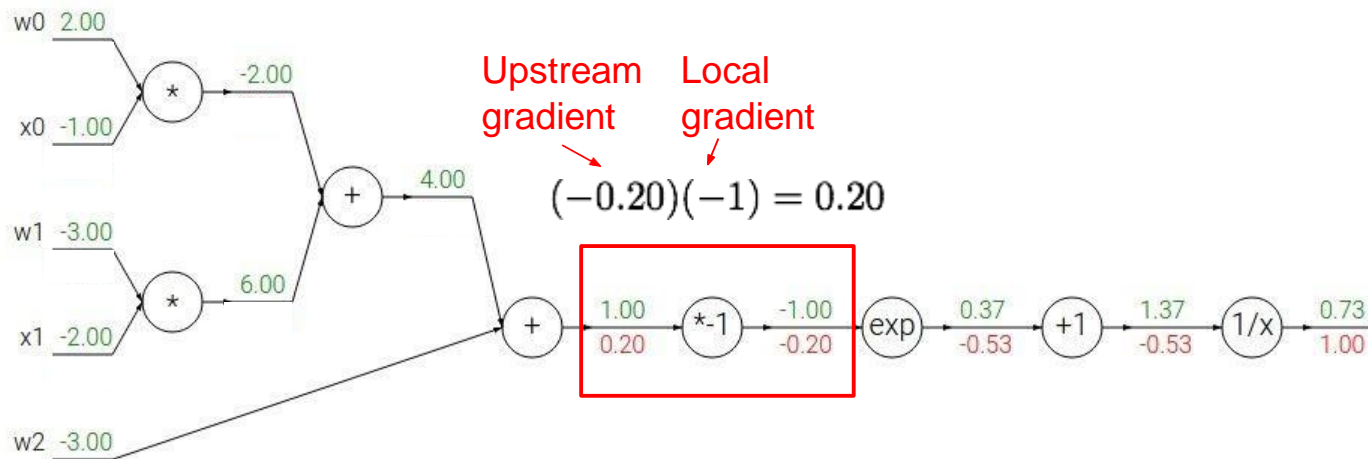
$$f_a(x) = ax \rightarrow \frac{df}{dx} = a$$

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

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

Another example:

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



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

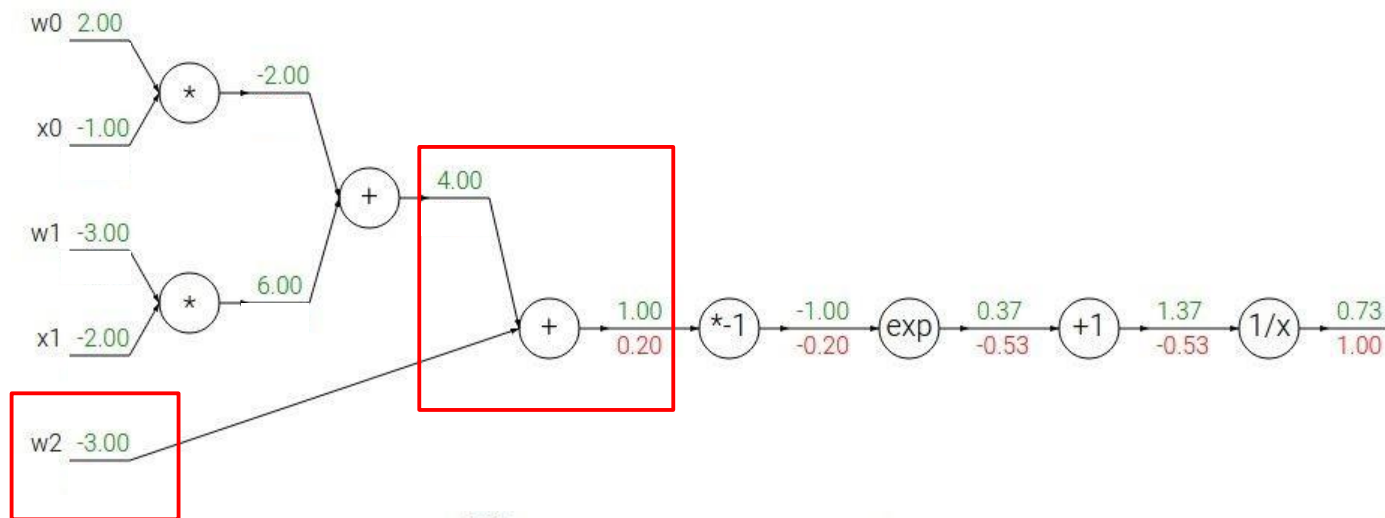
$$f_a(x) = ax \rightarrow \frac{df}{dx} = a$$

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

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

Another example:

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



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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

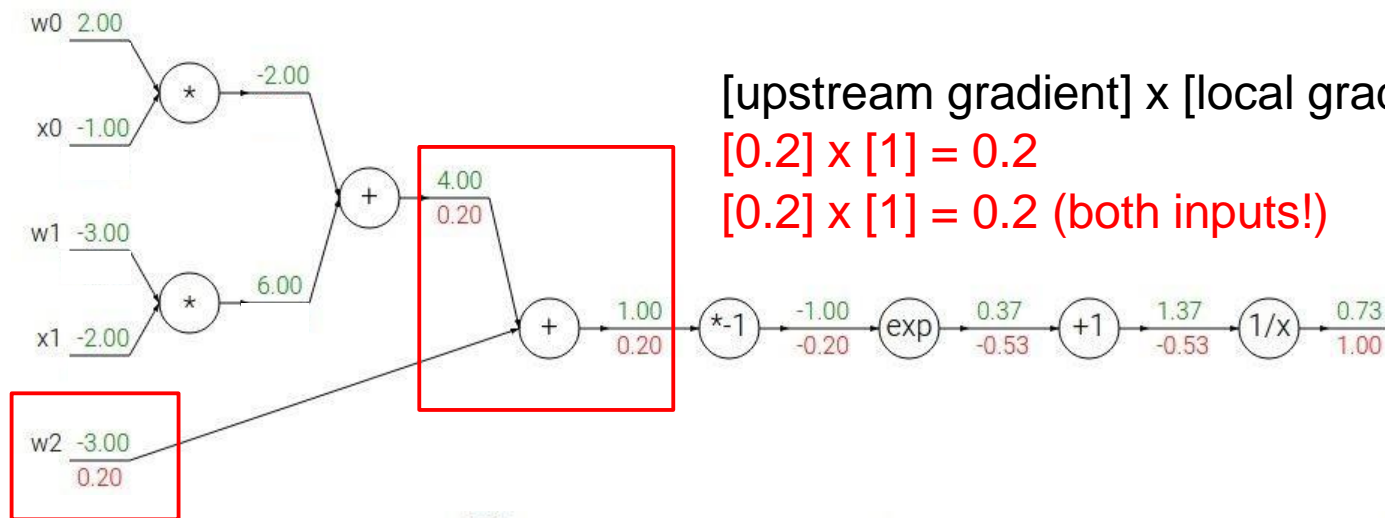
$$f_c(x) = c + x$$

→

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

Another example:

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



[upstream gradient] x [local gradient]

$$[0.2] \times [1] = 0.2$$

$$[0.2] \times [1] = 0.2 \text{ (both inputs!)}$$

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

→

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

$$f_a(x) = ax$$

→

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

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

→

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

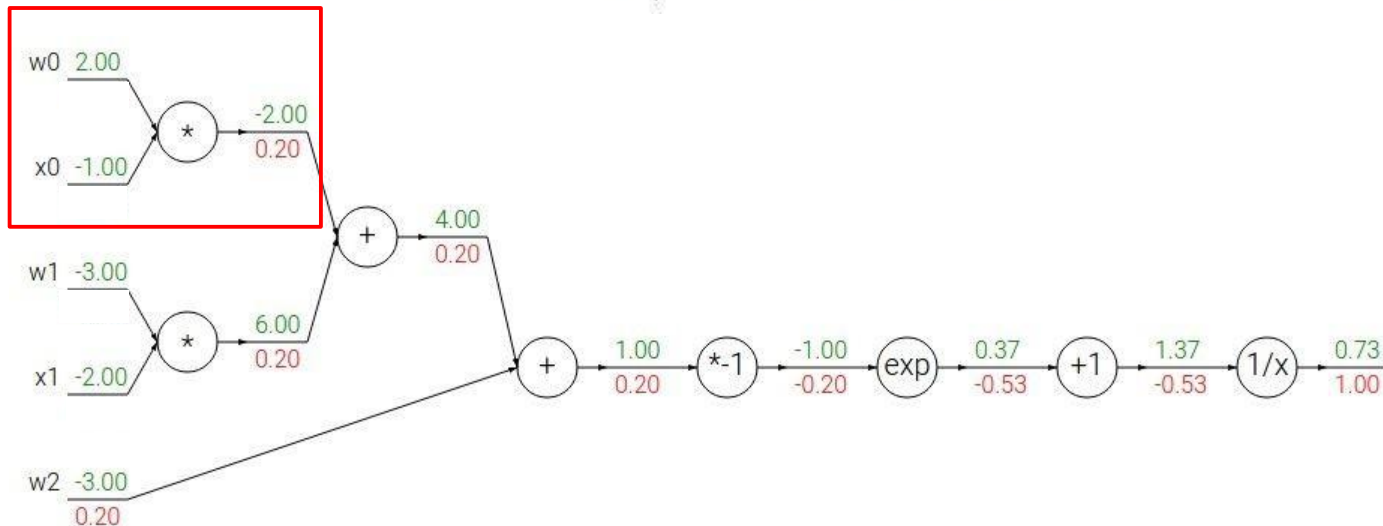
$$f_c(x) = c + x$$

→

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

Another example:

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



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

→

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

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

→

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

$$f_a(x) = ax$$

→

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

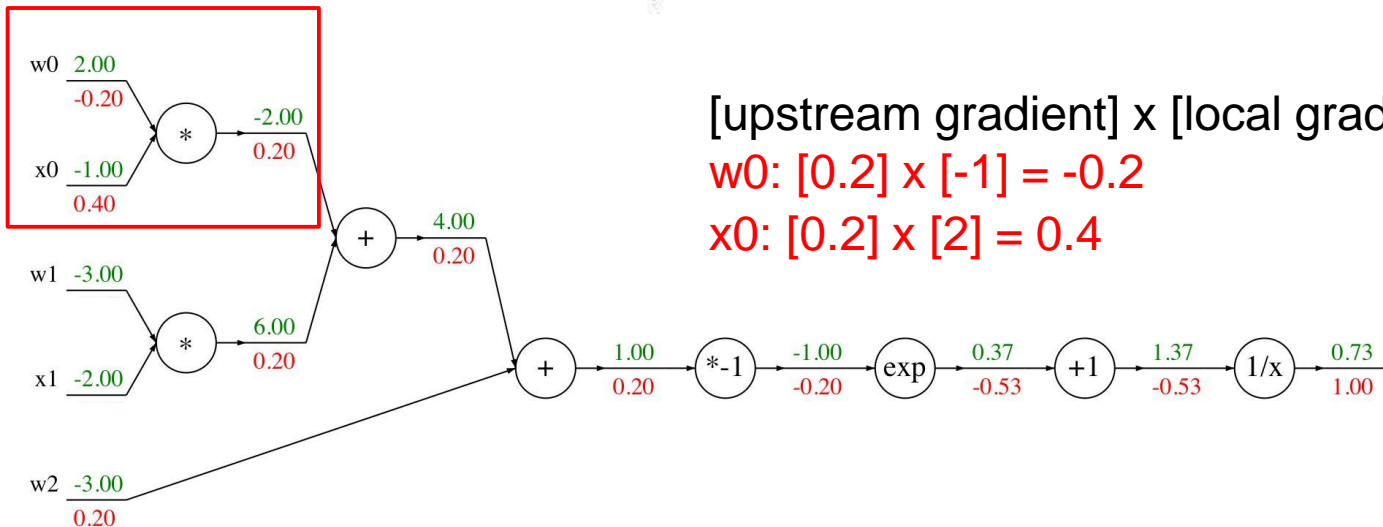
$$f_c(x) = c + x$$

→

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

Another example:

$$f(w, x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2x_2)}}$$



[upstream gradient] x [local gradient]

$$w_0: [0.2] \times [-1] = -0.2$$

$$x_0: [0.2] \times [2] = 0.4$$

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

→

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

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

→

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

$$f_a(x) = ax$$

→

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

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

→

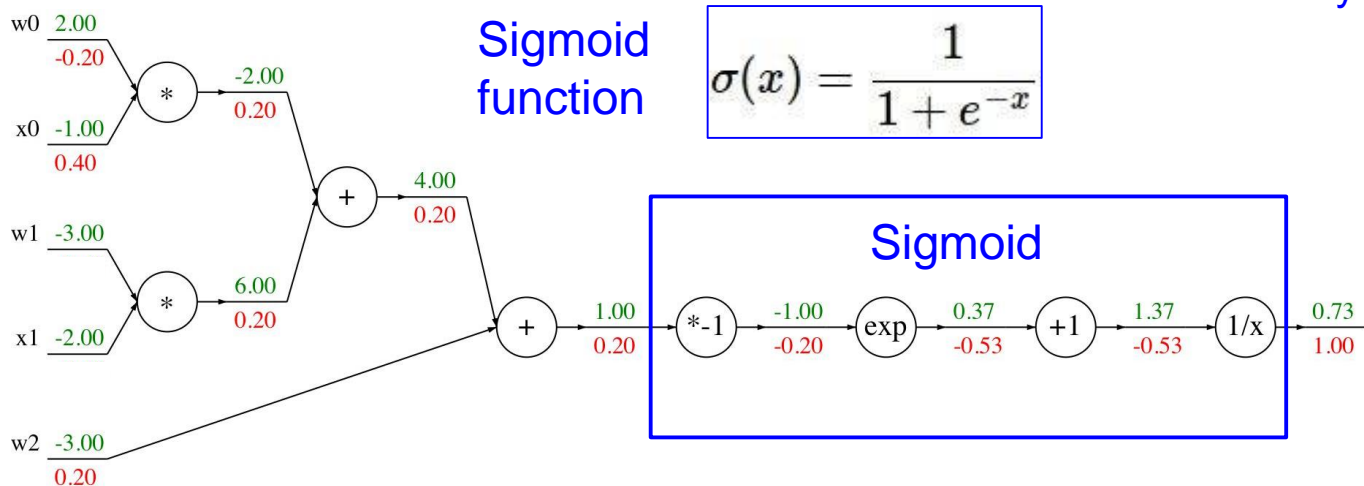
$$\frac{df}{dx} = 1$$



Another example:

$$f(w, x) = \frac{1}{1 + e^{-(w_0x_0 + w_1x_1 + w_2x_2)}}$$

Вычислительный граф не единственный, и может быть упрощен!



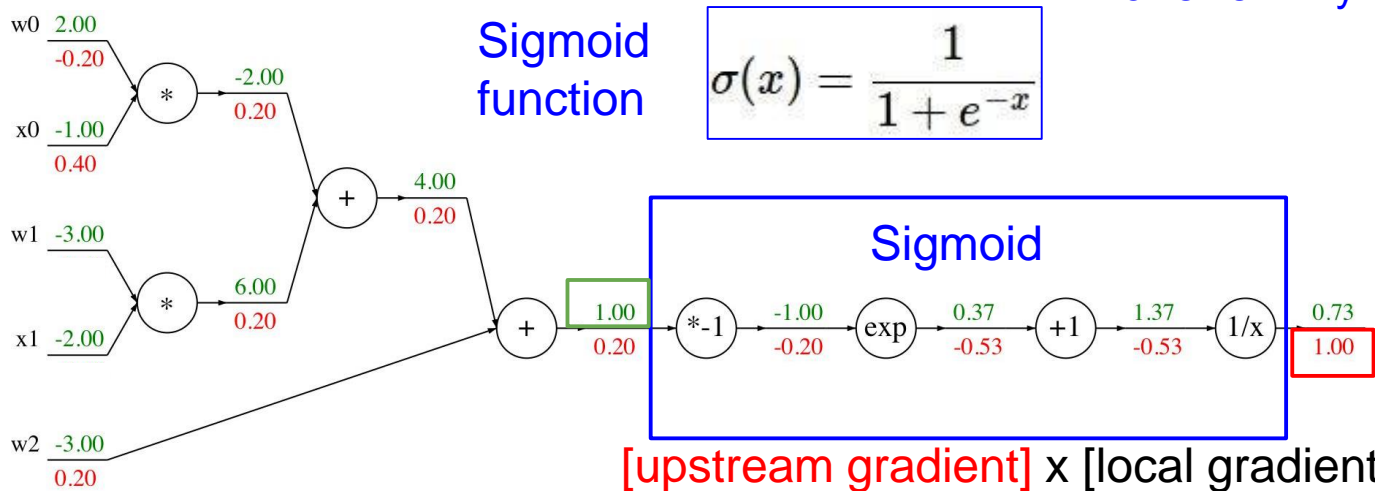
Sigmoid local gradient:

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

Another example:

$$f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$

Вычислительный граф не единственный, и может быть упрощен!



[upstream gradient] x [local gradient]  
[1.00] x [(1 - 1/(1+e<sup>1</sup>)) (1/(1+e<sup>1</sup>))] = 0.2

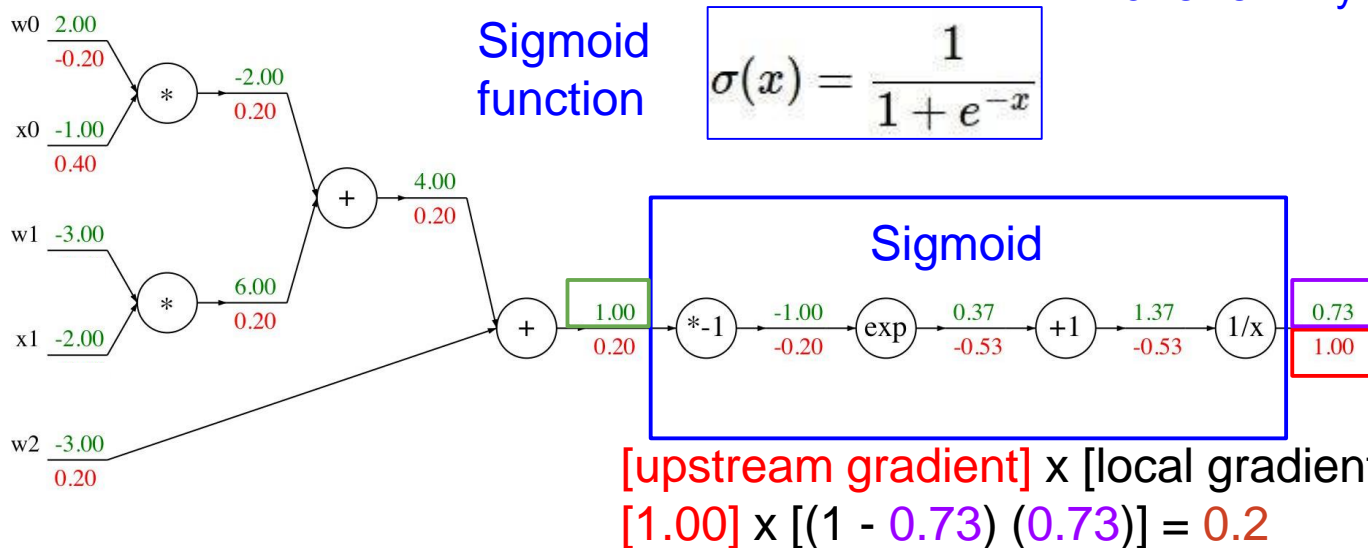
Sigmoid local gradient:

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

Another example:

$$f(w, x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$

Вычислительный граф не единственный, и может быть упрощен!

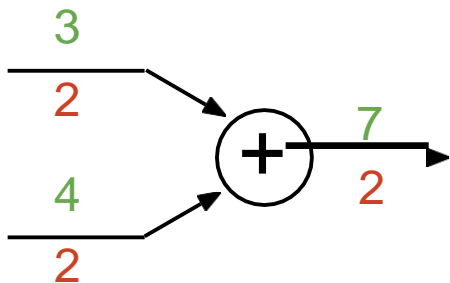


Sigmoid local gradient:

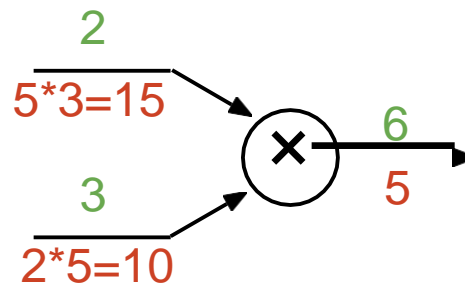
$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2} = \left( \frac{1 + e^{-x} - 1}{1 + e^{-x}} \right) \left( \frac{1}{1 + e^{-x}} \right) = (1 - \sigma(x)) \sigma(x)$$

# Правила вычисления градиентов

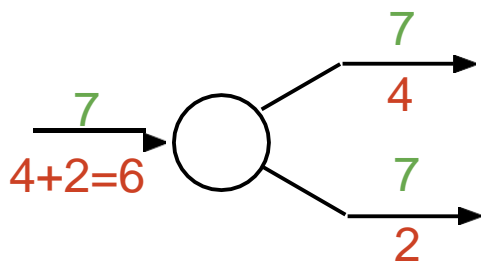
**add** gate: gradient distributor



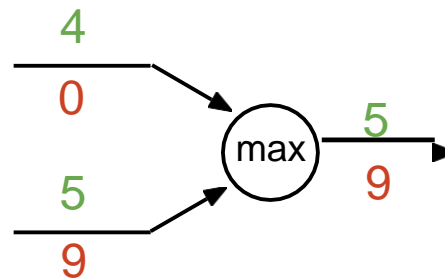
**mul** gate: “swap multiplier”



**copy** gate: gradient adder

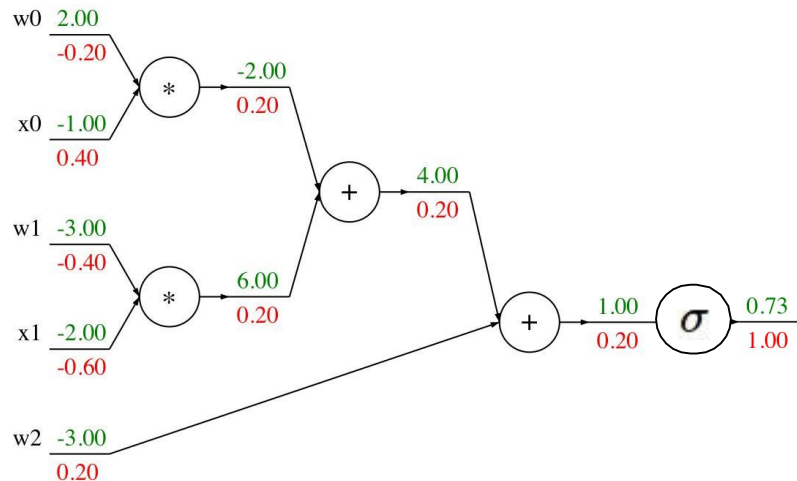


**max** gate: gradient router



# Реализация backprop: “в лоб”

Прямой проход:  
Считаем выход



Обратный  
проход:  
Считаем  
градиенты

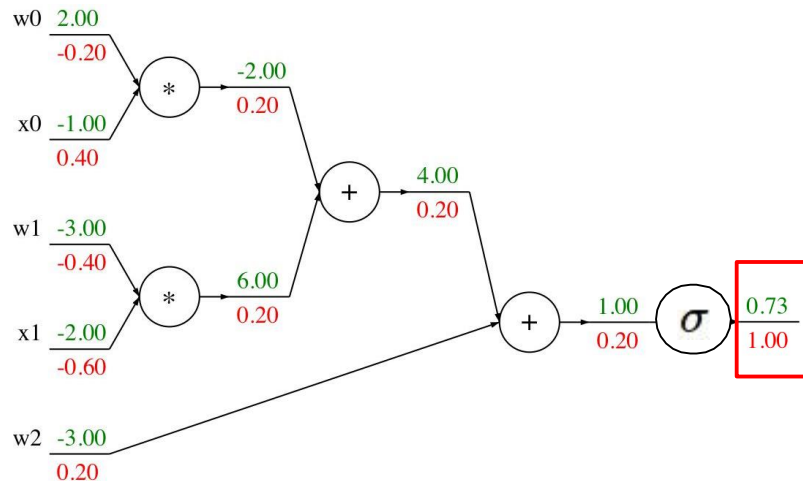
```
def f(w0, x0, w1, x1, w2):
```

```
    s0 = w0 * x0
    s1 = w1 * x1
    s2 = s0 + s1
    s3 = s2 + w2
    L = sigmoid(s3)
```

```
    grad_L = 1.0
    grad_s3 = grad_L * (1 - L) * L
    grad_w2 = grad_s3
    grad_s2 = grad_s3
    grad_s0 = grad_s2
    grad_s1 = grad_s2
    grad_w1 = grad_s1 * x1
    grad_x1 = grad_s1 * w1
    grad_w0 = grad_s0 * x0
    grad_x0 = grad_s0 * w0
```

# Реализация backprop: “в лоб”

Прямой проход:  
Считаем выход



Инициализация

```
def f(w0, x0, w1, x1, w2):
```

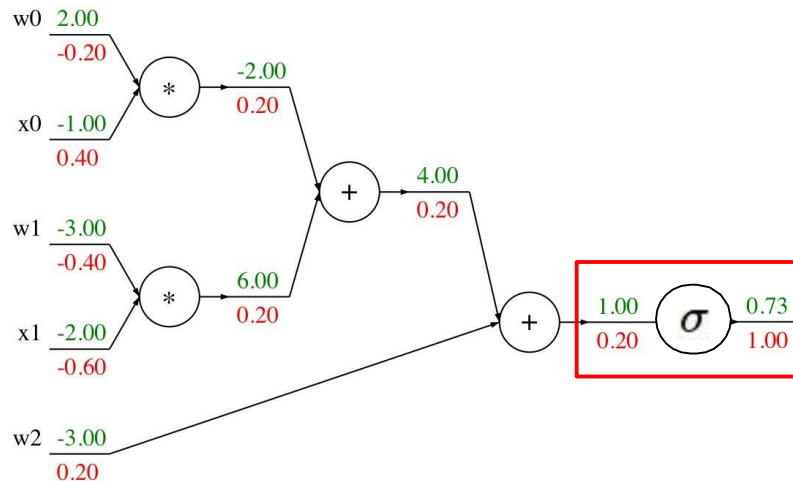
```
    s0 = w0 * x0
    s1 = w1 * x1
    s2 = s0 + s1
    s3 = s2 + w2
    L = sigmoid(s3)
```

```
    grad_L = 1.0
```

```
    grad_s3 = grad_L * (1 - L) * L
    grad_w2 = grad_s3
    grad_s2 = grad_s3
    grad_s0 = grad_s2
    grad_s1 = grad_s2
    grad_w1 = grad_s1 * x1
    grad_x1 = grad_s1 * w1
    grad_w0 = grad_s0 * x0
    grad_x0 = grad_s0 * w0
```

# Реализация backprop: “в лоб”

Прямой проход:  
Считаем выход



Sigmoid

```
def f(w0, x0, w1, x1, w2):
```

```
    s0 = w0 * x0
```

```
    s1 = w1 * x1
```

```
    s2 = s0 + s1
```

```
    s3 = s2 + w2
```

```
    L = sigmoid(s3)
```

```
    grad_L = 1.0
```

```
    grad_s3 = grad_L * (1 - L) * L
```

```
    grad_w2 = grad_s3
```

```
    grad_s2 = grad_s3
```

```
    grad_s0 = grad_s2
```

```
    grad_s1 = grad_s2
```

```
    grad_w1 = grad_s1 * x1
```

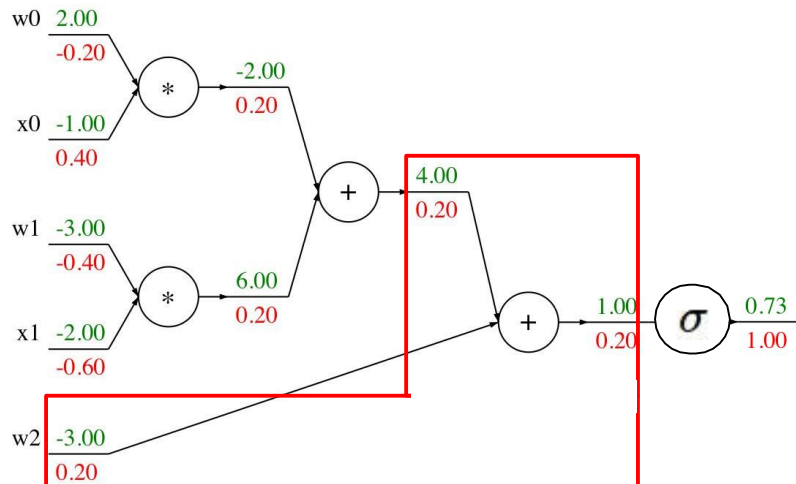
```
    grad_x1 = grad_s1 * w1
```

```
    grad_w0 = grad_s0 * x0
```

```
    grad_x0 = grad_s0 * w0
```

# Реализация backprop: “в лоб”

Прямой проход:  
Считаем выход



```
def f(w0, x0, w1, x1, w2):
```

```
    s0 = w0 * x0  
    s1 = w1 * x1  
    s2 = s0 + s1  
    s3 = s2 + w2  
    L = sigmoid(s3)
```

```
    grad_L = 1.0
```

```
    grad_s3 = grad_L * (1 - L) * L
```

```
    grad_w2 = grad_s3
```

```
    grad_s2 = grad_s3
```

```
    grad_s0 = grad_s2
```

```
    grad_s1 = grad_s2
```

```
    grad_w1 = grad_s1 * x1
```

```
    grad_x1 = grad_s1 * w1
```

```
    grad_w0 = grad_s0 * x0
```

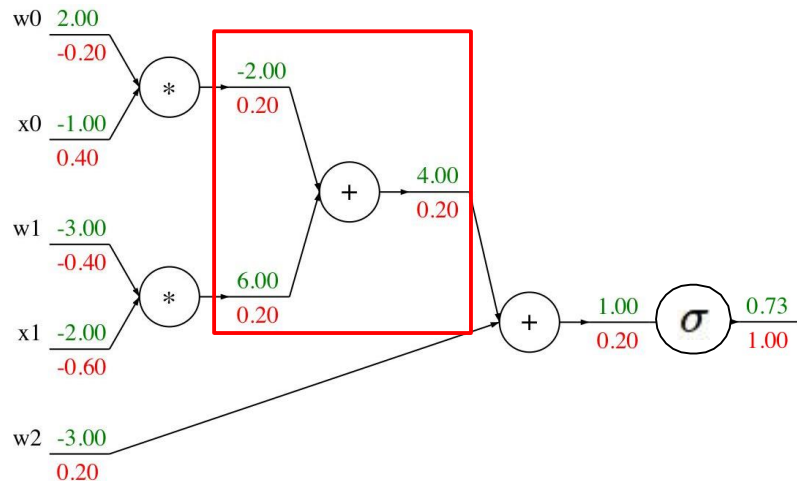
```
    grad_x0 = grad_s0 * w0
```

Add gate



# Реализация backprop: “в лоб”

Прямой проход:  
Считаем выход



```
def f(w0, x0, w1, x1, w2):
```

```
    s0 = w0 * x0
```

```
    s1 = w1 * x1
```

```
    s2 = s0 + s1
```

```
    s3 = s2 + w2
```

```
    L = sigmoid(s3)
```

```
grad_L = 1.0
```

```
grad_s3 = grad_L * (1 - L) * L
```

```
grad_w2 = grad_s3
```

```
grad_s2 = grad_s3
```

```
grad_s0 = grad_s2
```

```
grad_s1 = grad_s2
```

```
grad_w1 = grad_s1 * x1
```

```
grad_x1 = grad_s1 * w1
```

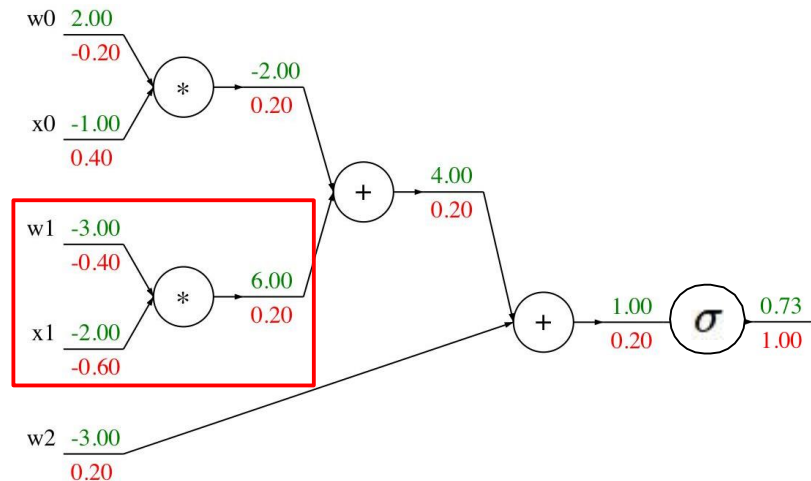
```
grad_w0 = grad_s0 * x0
```

```
grad_x0 = grad_s0 * w0
```

Add gate

# Реализация backprop: “в лоб”

Прямой проход:  
Считаем выход



```
def f(w0, x0, w1, x1, w2):
```

```
    s0 = w0 * x0
```

```
    s1 = w1 * x1
```

```
    s2 = s0 + s1
```

```
    s3 = s2 + w2
```

```
    L = sigmoid(s3)
```

```
grad_L = 1.0
```

```
grad_s3 = grad_L * (1 - L) * L
```

```
grad_w2 = grad_s3
```

```
grad_s2 = grad_s3
```

```
grad_s0 = grad_s2
```

```
grad_s1 = grad_s2
```

```
grad_w1 = grad_s1 * x1
```

```
grad_x1 = grad_s1 * w1
```

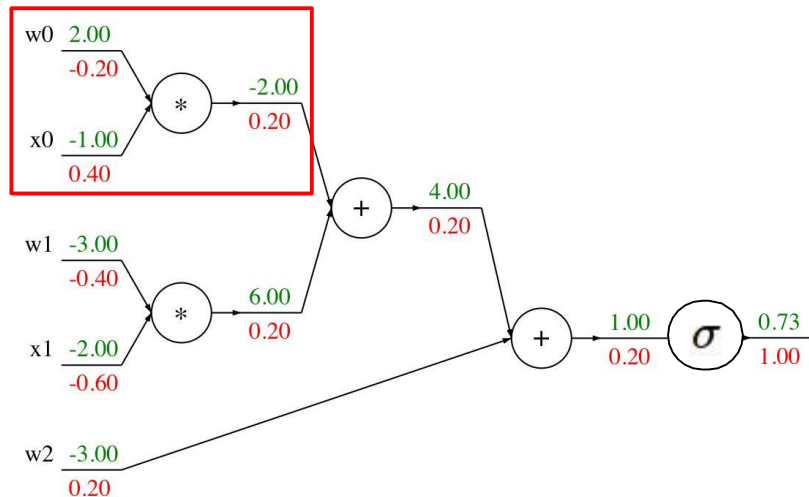
```
grad_w0 = grad_s0 * x0
```

```
grad_x0 = grad_s0 * w0
```

Multiply gate

# Реализация backprop: “в лоб”

Прямой проход:  
Считаем выход



```
def f(w0, x0, w1, x1, w2):
```

```
    s0 = w0 * x0
```

```
    s1 = w1 * x1
```

```
    s2 = s0 + s1
```

```
    s3 = s2 + w2
```

```
    L = sigmoid(s3)
```

```
grad_L = 1.0
```

```
grad_s3 = grad_L * (1 - L) * L
```

```
grad_w2 = grad_s3
```

```
grad_s2 = grad_s3
```

```
grad_s0 = grad_s2
```

```
grad_s1 = grad_s2
```

```
grad_w1 = grad_s1 * x1
```

```
grad_x1 = grad_s1 * w1
```

```
grad_w0 = grad_s0 * x0
```

```
grad_x0 = grad_s0 * w0
```

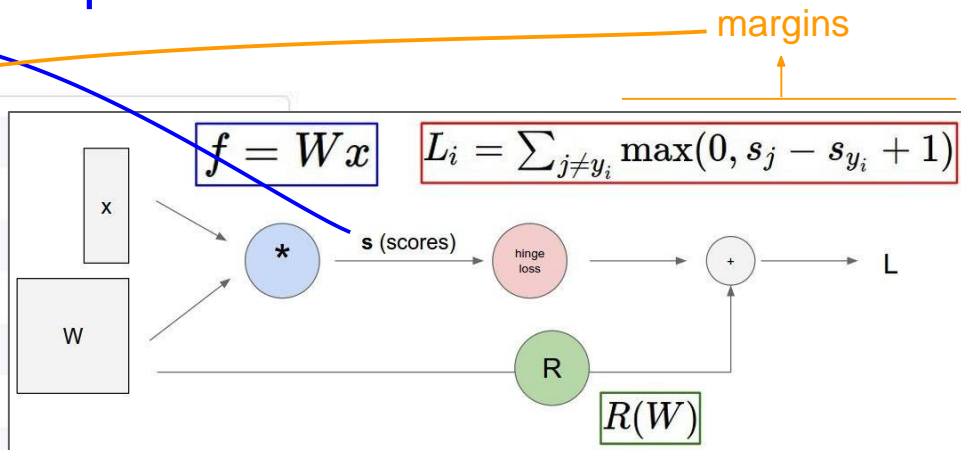
Multiply gate

# “Плоский” Backprop

Можно подставить ваши лабораторки!

Например для SVM:

```
# receive W (weights), X (data)
# forward pass (we have 8 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 = #...
```



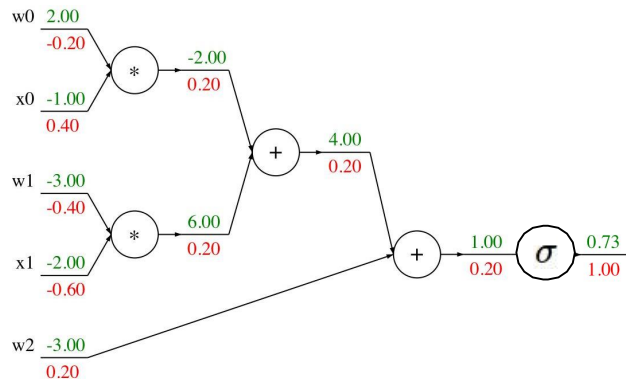
# “Плоский” Backprop

Двуслойная сеть:

```
# 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 = #...
```

# Модульное API для Backprop

Graph (Net) object (псевдокод)



```
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
```

Далее: векторный backprop

# Вспомним векторные производные

Скаляр на скаляр    Вектор на скаляр

$$x \in \mathbb{R}, y \in \mathbb{R}$$

Производная:

$$\frac{\partial y}{\partial x} \in \mathbb{R}$$

Как изменится  $y$   
при малом  
изменении  $x$ ?

Вектор на скаляр

$$x \in \mathbb{R}^N, y \in \mathbb{R}$$

Производная стала  
**градиентом:**

$$\frac{\partial y}{\partial x} \in \mathbb{R}^N \quad \left( \frac{\partial y}{\partial x} \right)_n = \frac{\partial y}{\partial x_n}$$

Как изменится  $y$  при  
малом изменении  
каждого элемента  $x$ ?

Вектор на вектор

$$x \in \mathbb{R}^N, y \in \mathbb{R}^M$$

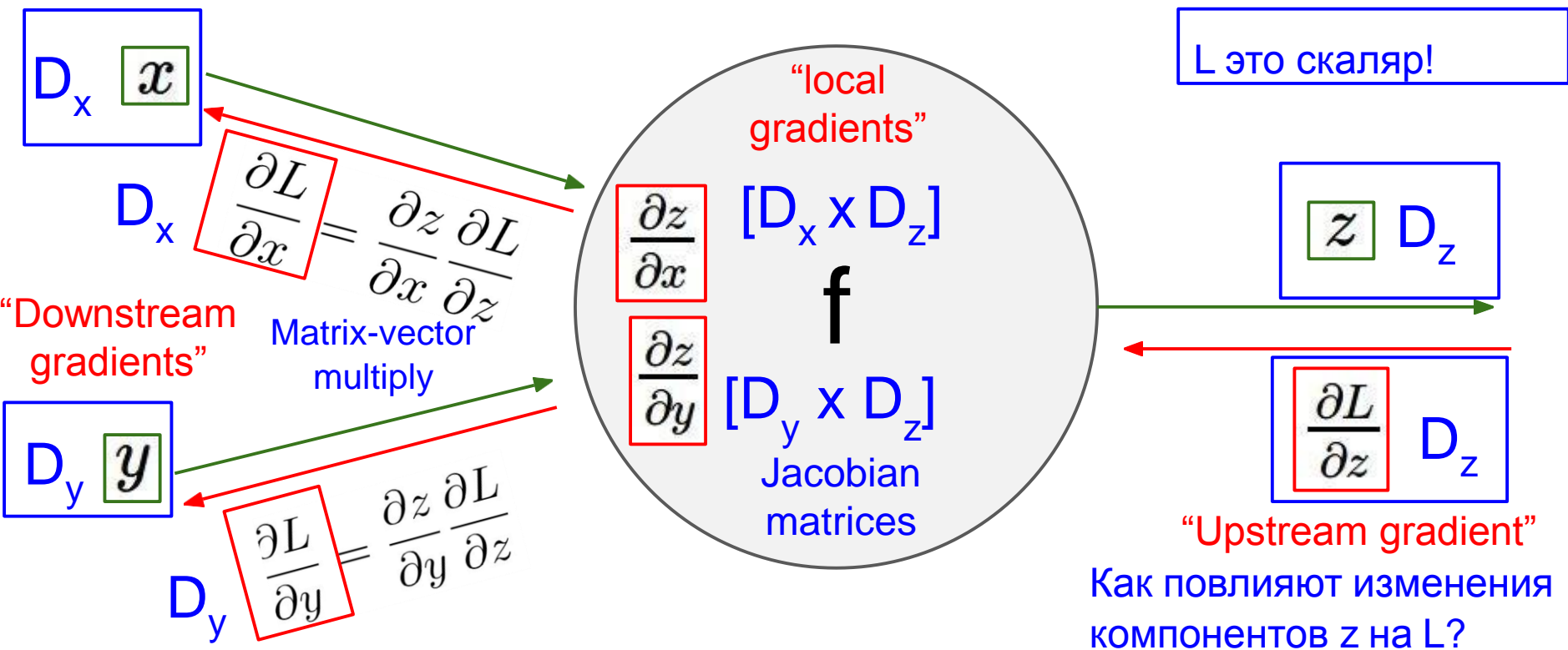
Производная стала  
**якобианом:**

$$\frac{\partial y}{\partial x} \in \mathbb{R}^{N \times M} \quad \left( \frac{\partial y}{\partial x} \right)_{n,m} = \frac{\partial y_m}{\partial x_n}$$

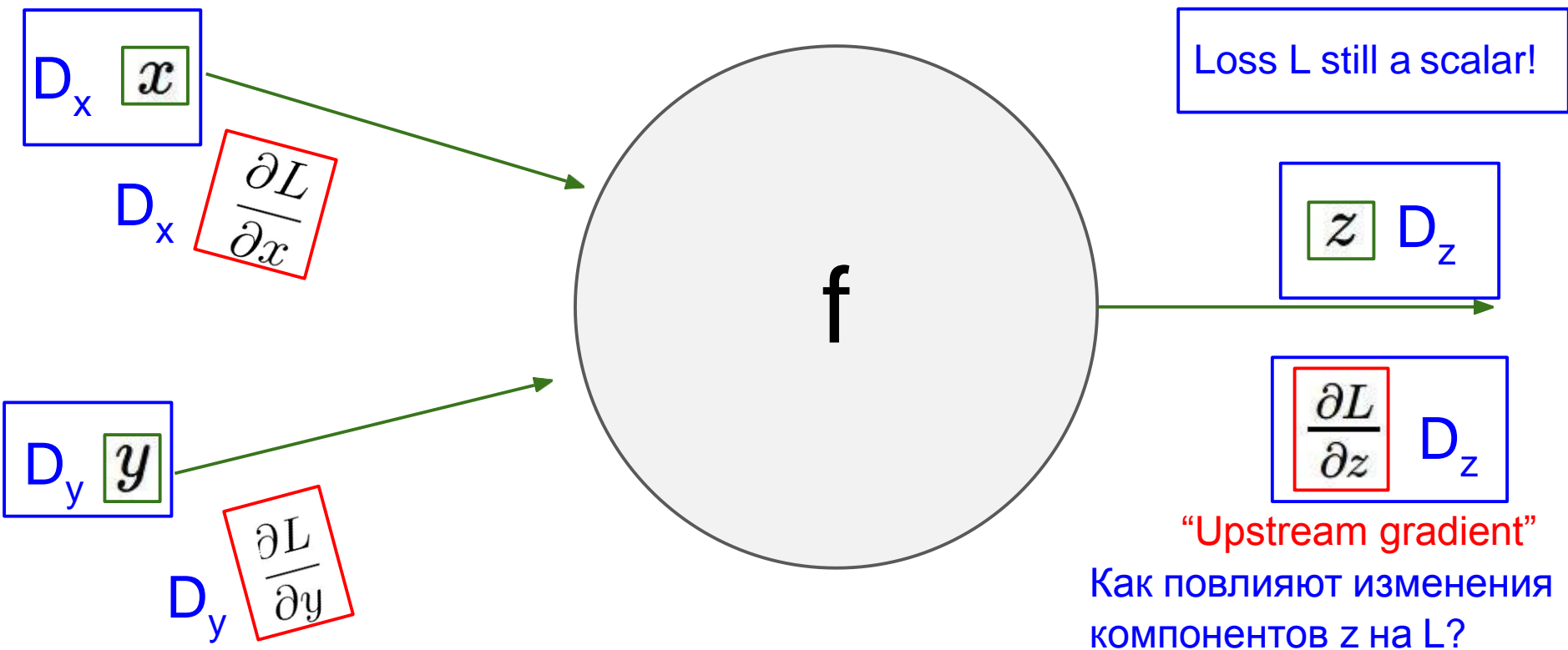
Как изменится каждый  
элемент  $y$  при малом  
изменении каждого  
элемента  $x$ ?



# Векторный Backprop



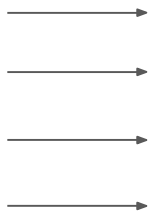
Градиенты переменных имеют ту же размерность что и сами переменные



# Векторный Backprop

4D input x:

$\begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$

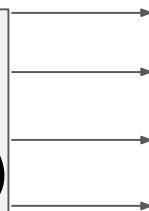


$$f(x) = \max(0, x)$$

(поэлементно)

4D output z:

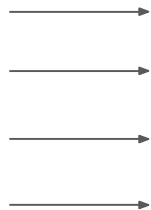
$\begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \end{bmatrix}$



# Векторный Backprop

4D input x:

$\begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$

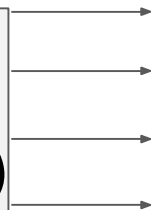


$$f(x) = \max(0, x)$$

(поэлементно)

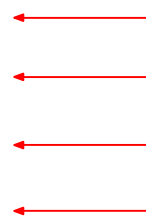
4D output z:

$\begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \end{bmatrix}$



4D dL/dz:

$\begin{bmatrix} 4 \\ -1 \\ 5 \\ 9 \end{bmatrix}$

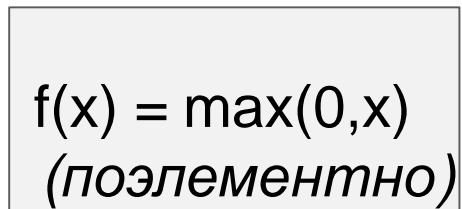
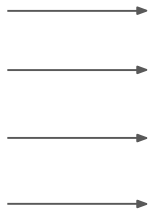


Upstream  
gradient

# Векторный Backprop

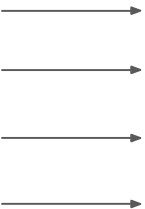
4D input x:

$\begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$



4D output z:

$\begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \end{bmatrix}$



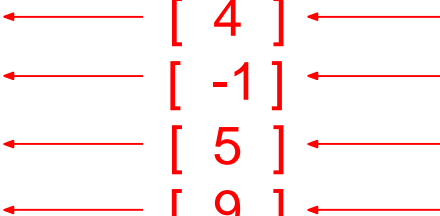
Jacobian  $dz/dx$

$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

4D  $dL/dz$ :

$\begin{bmatrix} 4 \\ -1 \\ 5 \\ 9 \end{bmatrix}$

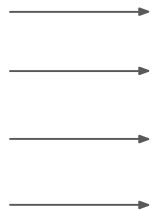
Upstream  
gradient



# Векторный Backprop

4D input x:

$\begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$

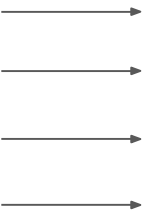


$$f(x) = \max(0, x)$$

(поэлементно)

4D output z:

$\begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \end{bmatrix}$



$[dz/dx] \quad [dL/dz]$

$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 4 \end{bmatrix}$

$\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 \end{bmatrix}$

$\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 5 \end{bmatrix}$

$\begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 9 \end{bmatrix}$

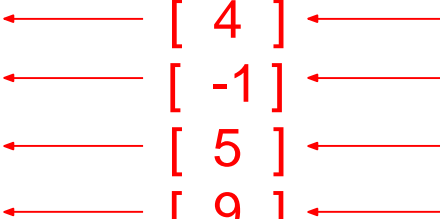
4D dL/dz:

$\begin{bmatrix} 4 \end{bmatrix}$

$\begin{bmatrix} -1 \end{bmatrix}$

$\begin{bmatrix} 5 \end{bmatrix}$

$\begin{bmatrix} 9 \end{bmatrix}$

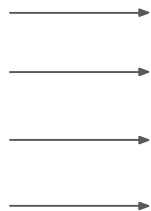


Upstream  
gradient

# Backprop with Vectors

4D input x:

$\begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$

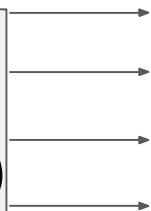


$$f(x) = \max(0, x)$$

(поэлементно)

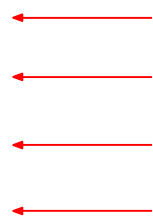
4D output z:

$\begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \end{bmatrix}$



4D dL/dx:

$\begin{bmatrix} 4 \\ 0 \\ 5 \\ 0 \end{bmatrix}$

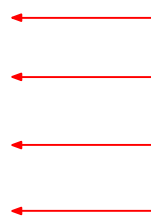


$\begin{bmatrix} dz/dx & dL/dz \end{bmatrix}$

$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 4 \\ -1 \\ 5 \\ 9 \end{bmatrix}$

4D dL/dz:

$\begin{bmatrix} 4 \\ -1 \\ 5 \\ 9 \end{bmatrix}$

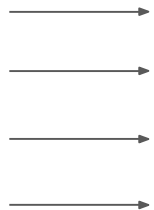


Upstream  
gradient

# Векторный Backprop

4D input x:

$\begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$



$$f(x) = \max(0, x)$$

(поэлементно)

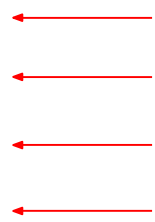
4D output z:

$\begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \end{bmatrix}$



4D dL/dx:

$\begin{bmatrix} 4 \\ 0 \\ 5 \\ 0 \end{bmatrix}$

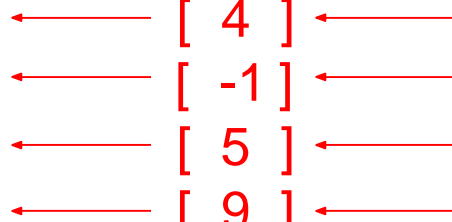


$\begin{bmatrix} dz/dx & dL/dz \end{bmatrix}$

$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 4 \\ -1 \\ 5 \\ 9 \end{bmatrix}$

4D dL/dz:

$\begin{bmatrix} 4 \\ -1 \\ 5 \\ 9 \end{bmatrix}$



Upstream  
gradient

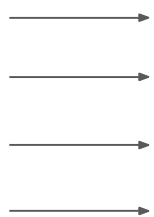
Якобиан всегда **разряженный**: внедиагональные элементы нулевые! Матрицу якобиана никогда не формируется **явно** – всегда используется **неявное умножение**



# Векторный Backprop

4D input x:

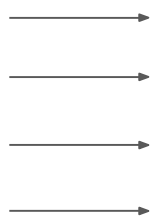
$\begin{bmatrix} 1 \\ -2 \\ 3 \\ -1 \end{bmatrix}$



$$f(x) = \max(0, x) \\ (\textit{elementwise})$$

4D output z:

$\begin{bmatrix} 1 \\ 0 \\ 3 \\ 0 \end{bmatrix}$



4D dL/dx:

$\begin{bmatrix} 4 \\ 0 \\ 5 \\ 0 \end{bmatrix}$

$[dz/dx] [dL/dz]$

$$\left( \frac{\partial L}{\partial x} \right)_i = \begin{cases} \left( \frac{\partial L}{\partial z} \right)_i & \text{if } x_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

4D dL/dz:

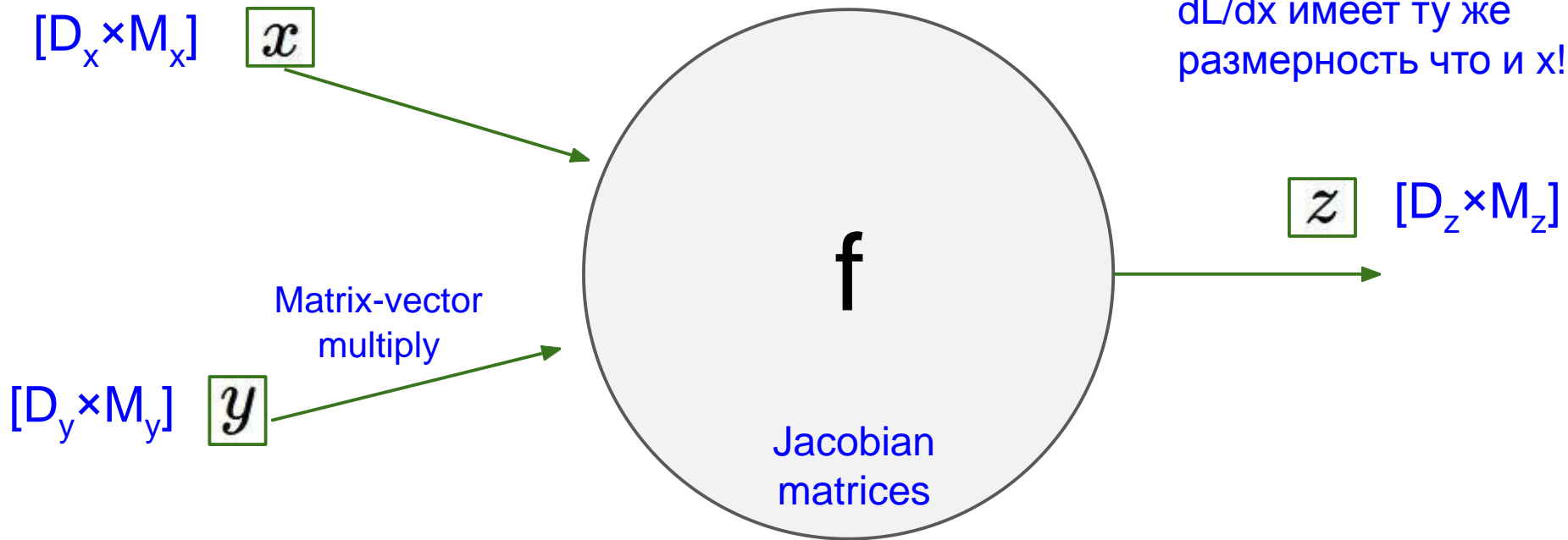
$\begin{bmatrix} 4 \\ -1 \\ 5 \\ 9 \end{bmatrix}$

Upstream  
gradient

Якобиан всегда **разряженный**: внедиагональные элементы нулевые! Матрицу якобиана никогда не формируется **явно** – всегда используется **неявное умножение**

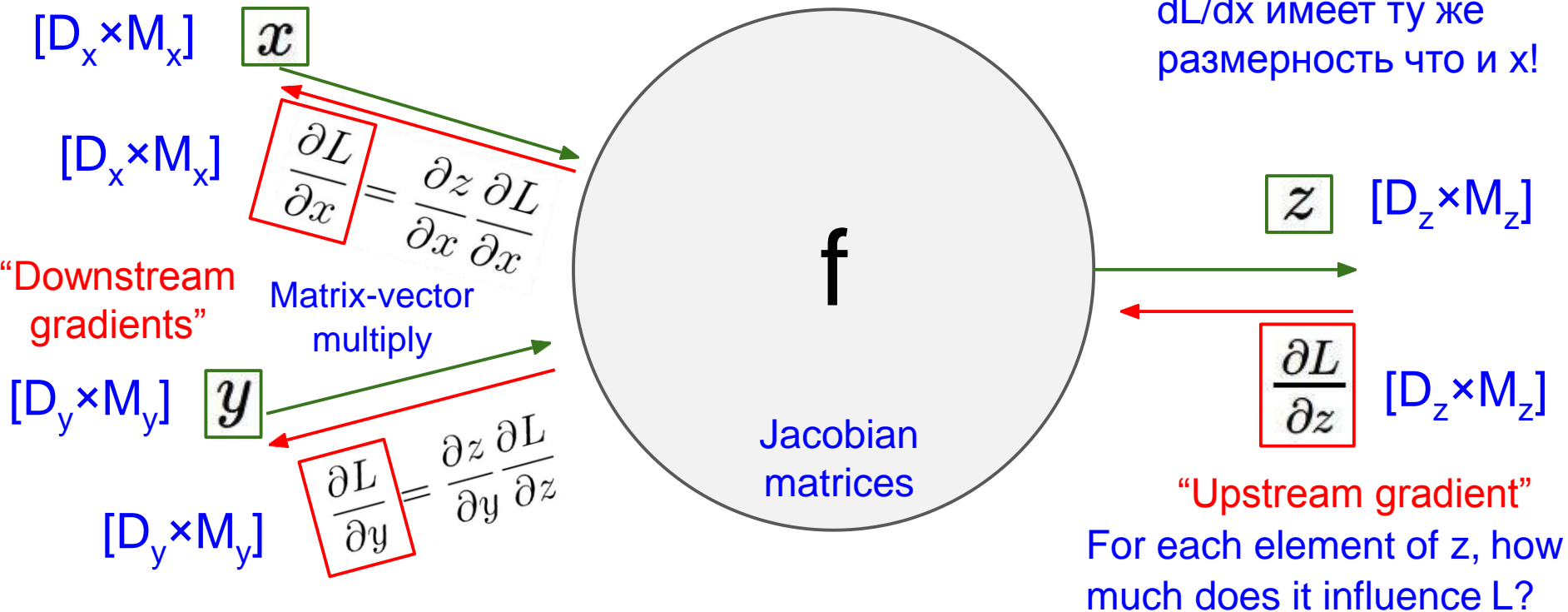
# Матричный (тензорный) Backprop

L скаляр!



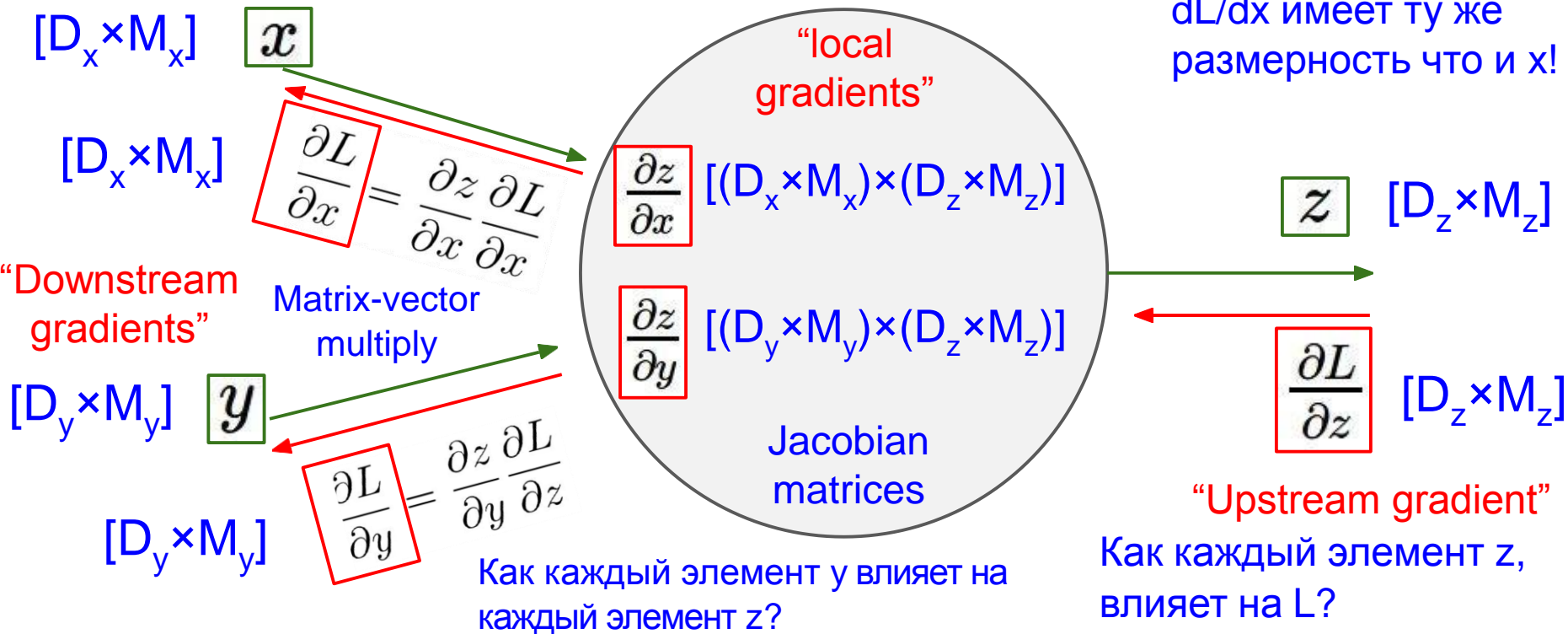
# Матричный (тензорный) Backprop

L скаляр!

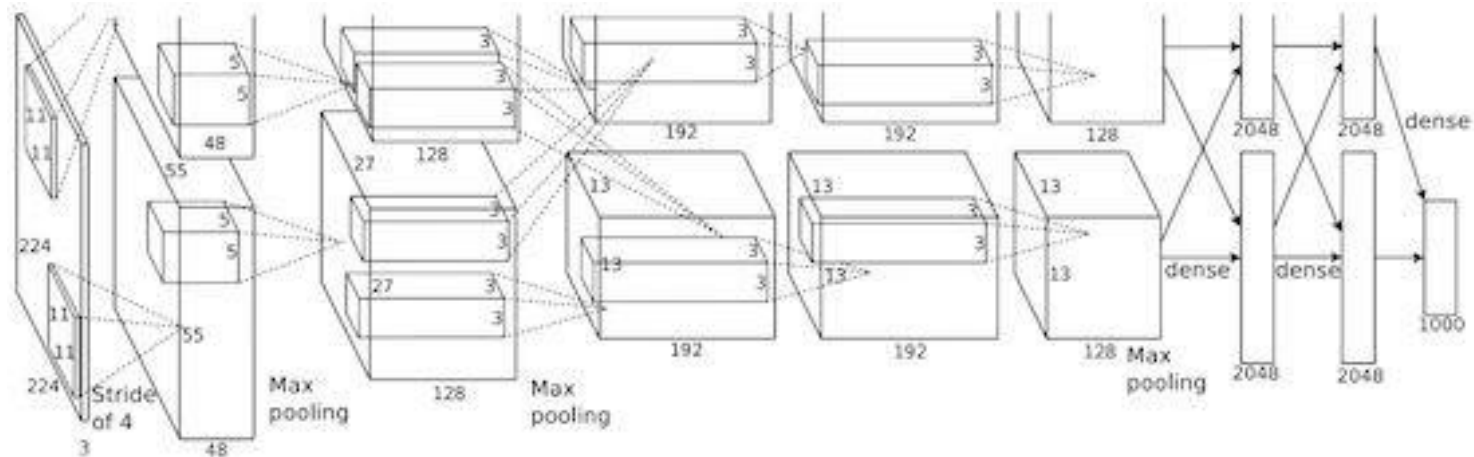


# Матричный (тензорный) Backprop

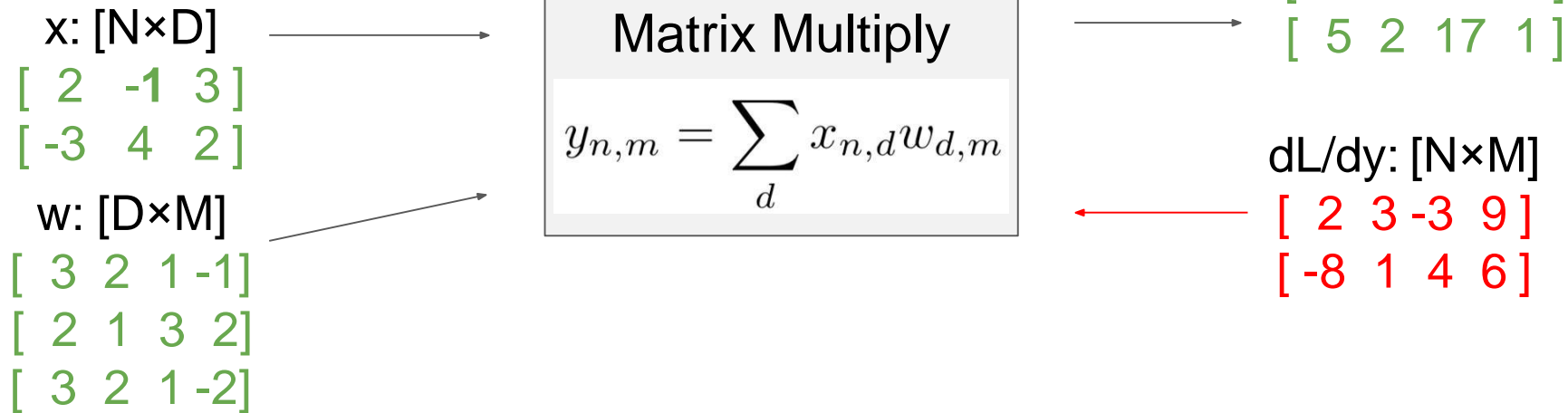
L скаляр!



# Далее сверточные сетки!



# Backprop with Matrices



Полезно посмотреть:

<http://cs231n.stanford.edu/handouts/linear-backprop.pdf>

# Backprop with Matrices

x: [N×D]

[ 2 -1 3 ]

[ -3 4 2 ]

w: [D×M]

[ 3 2 1 -1 ]

[ 2 1 3 2 ]

[ 3 2 1 -2 ]

Matrix Multiply

$$y_{n,m} = \sum_d x_{n,d} w_{d,m}$$

**Jacobians:** dy/dx:

[(N×D)×(N×M)]

dy/dw: [(D×M)×(N×M)]

y: [N×M]

[13 9 2 -10]

[ 5 2 17 1 ]

dL/dy: [N×M]

[ 2 3 -3 9 ]

[ -8 1 4 6 ]

Для нейронной сети может быть

N=64, D=M=4096

Каждый якобиан требует 256 GB!

Только неявное умножение!

# Backprop with Matrices

$x: [N \times D]$   
 $\begin{bmatrix} 2 & -1 & 3 \\ -3 & 4 & 2 \end{bmatrix}$

$w: [D \times M]$   
 $\begin{bmatrix} 3 & 2 & 1 & -1 \\ 2 & 1 & 3 & 2 \\ 3 & 2 & 1 & -2 \end{bmatrix}$



Matrix Multiply

$$y_{n,m} = \sum_d x_{n,d} w_{d,m}$$



$y: [N \times M]$   
 $\begin{bmatrix} 13 & 9 & 2 & -10 \\ 5 & 2 & 17 & 1 \end{bmatrix}$



$dL/dy: [N \times M]$   
 $\begin{bmatrix} 2 & 3 & -3 & 9 \\ -8 & 1 & 4 & 6 \end{bmatrix}$

**Q:** на какую часть  
у влияет один  
элемент x?



# Backprop with Matrices

$x: [N \times D]$   
 $\begin{bmatrix} 2 & -1 & 3 \\ -3 & 4 & 2 \end{bmatrix}$

$w: [D \times M]$   
 $\begin{bmatrix} 3 & 2 & 1 & -1 \\ 2 & 1 & 3 & 2 \\ 3 & 2 & 1 & -2 \end{bmatrix}$

Matrix Multiply

$$y_{n,m} = \sum_d x_{n,d} w_{d,m}$$

$y: [N \times M]$   
 $\begin{bmatrix} 13 & 9 & 2 & -10 \\ 5 & 2 & 17 & 1 \end{bmatrix}$

$dL/dy: [N \times M]$   
 $\begin{bmatrix} 2 & 3 & -3 & 9 \\ -8 & 1 & 4 & 6 \end{bmatrix}$

**Q:** на какую часть  
у влияет один  
элемент x?

**A:**  $x_{n,d}$  влияет на  
всю строку  $y_{n,\cdot}$

$$\frac{\partial L}{\partial x_{n,d}} = \sum_m \frac{\partial L}{\partial y_{n,m}} \frac{\partial y_{n,m}}{\partial x_{n,d}}$$

# Backprop with Matrices

$$x: [N \times D]$$

$$\begin{bmatrix} 2 & \boxed{1} & -3 \\ -3 & 4 & 2 \end{bmatrix}$$

$$w: [D \times M]$$

$$\begin{bmatrix} 3 & 2 & 1 & -1 \\ 2 & 1 & 3 & 2 \\ 3 & 2 & 1 & -2 \end{bmatrix}$$

Matrix Multiply

$$y_{n,m} = \sum_d x_{n,d} w_{d,m}$$

$$y: [N \times M]$$

$$\begin{bmatrix} \boxed{13} & 9 & \boxed{-2} & -6 \\ 5 & 2 & 17 & 1 \end{bmatrix}$$

$$dL/dy: [N \times M]$$

$$\begin{bmatrix} \boxed{2} & 3 & -3 & 9 \\ -8 & 1 & 4 & 6 \end{bmatrix}$$

**Q:** на какую часть  $y$  влияет один элемент  $x$ ?

**A:**  $x_{n,d}$  влияет на всю строку  $y_{n,\cdot}$ .

**Q:** Как влияет  $x_{n,d}$  на  $y_{n,m}$ ?

$$\frac{\partial L}{\partial x_{n,d}} = \sum_m \frac{\partial L}{\partial y_{n,m}} \frac{\partial y_{n,m}}{\partial x_{n,d}}$$

# Backprop with Matrices

$x: [N \times D]$

$\begin{bmatrix} 2 & -1 & 3 \\ -3 & 4 & 2 \end{bmatrix}$

$w: [D \times M]$

$\begin{bmatrix} 3 & 2 & 1 & -1 \\ 2 & 1 & 3 & 2 \\ 3 & 2 & 1 & -2 \end{bmatrix}$

Matrix Multiply

$$y_{n,m} = \sum_d x_{n,d} w_{d,m}$$

$y: [N \times M]$

$\begin{bmatrix} 13 & 9 & 2 & -10 \\ 5 & 2 & 17 & 1 \end{bmatrix}$

$dL/dy: [N \times M]$

$\begin{bmatrix} 2 & 3 & -3 & 9 \\ -8 & 1 & 4 & 6 \end{bmatrix}$

**Q:** на какую часть  $y$  влияет один элемент  $x$ ?

**A:**  $x_{n,d}$  влияет на всю строку  $y_{n,\cdot}$ .

**Q:** Как влияет  $x_{n,d}$  на  $y_{n,m}$ ?

**A:**  $w_{d,m}$

$$\frac{\partial L}{\partial x} = \left( \frac{\partial L}{\partial y} \right) w^T$$

$$\frac{\partial L}{\partial x_{n,d}} = \sum_m \frac{\partial L}{\partial y_{n,m}} \frac{\partial y_{n,m}}{\partial x_{n,d}} = \sum_m \frac{\partial L}{\partial y_{n,m}} w_{d,m}$$

# Backprop with Matrices

$$x: [N \times D]$$

$$\begin{bmatrix} 2 & -1 & 3 \\ -3 & 4 & 2 \end{bmatrix}$$

$$w: [D \times M]$$

$$\begin{bmatrix} 3 & 2 & 1 & -1 \\ 2 & 1 & 3 & 2 \\ 3 & 2 & 1 & -2 \end{bmatrix}$$

Matrix Multiply

$$y_{n,m} = \sum_d x_{n,d} w_{d,m}$$

Аналогично:

$$y: [N \times M]$$

$$\begin{bmatrix} 13 & 9 & 2 & -10 \\ 5 & 2 & 17 & 1 \end{bmatrix}$$

$$dL/dy: [N \times M]$$

$$\begin{bmatrix} 2 & 3 & -3 & 9 \\ -8 & 1 & 4 & 6 \end{bmatrix}$$

$$[N \times D] \quad [N \times M] \quad [M \times D]$$

$$\frac{\partial L}{\partial x} = \left( \frac{\partial L}{\partial y} \right) w^T$$

$$[D \times M] \quad [D \times N] \quad [N \times M]$$

$$\frac{\partial L}{\partial w} = x^T \left( \frac{\partial L}{\partial y} \right)$$

Тут все просто –  
главное чтобы  
совпадали  
размерности!

# Итоги:

- **Полносвязные (fully-connected, dense)** это последовательность линейных классификаторов с нелинейными активационными функциями; они обеспечивают лучшие результаты чем линейные классификаторы
- **backpropagation** = рекурсивное применение цепного правила к вычислительному графу при обратном распространении позволяет получить значение градиента функции потерь по параметрам
- Граф может быть реализован универсальным API с функциями **forward()** / **backward()**
- **forward**: вычисляем результат прямого распространения с сохранением всех промежуточных значений, необходимых для обратного распространения
- **backward**: применяя цепное правило вычисляем значения градиента функции потерь по параметрам

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

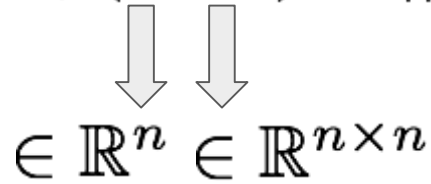
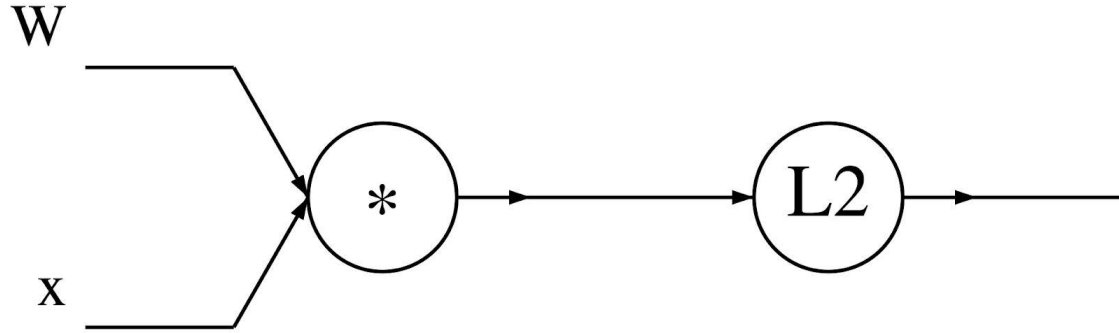


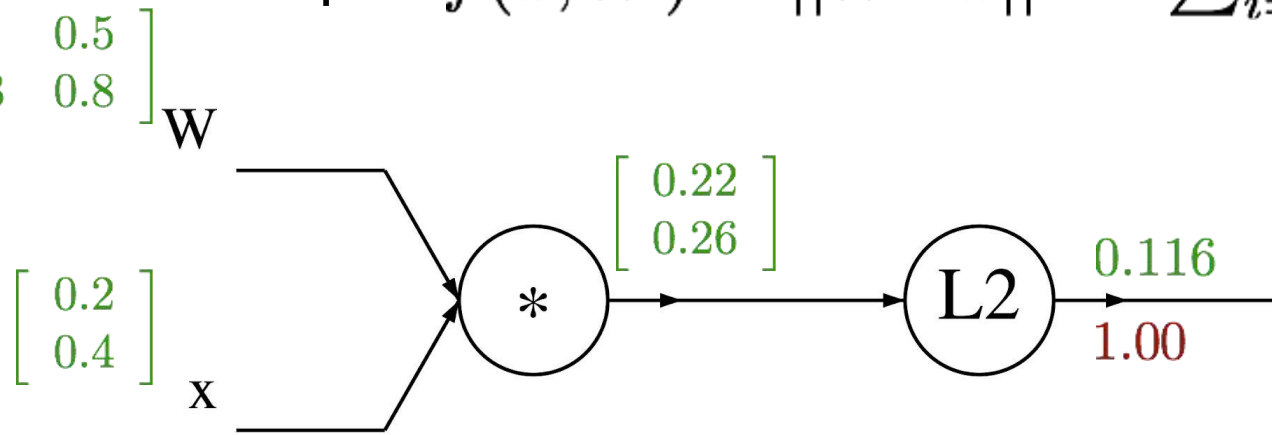
Diagram illustrating the domains of the variables  $x$  and  $W$  in the vectorized example:

$$\begin{array}{ccc} & \downarrow & \downarrow \\ & \in \mathbb{R}^n & \in \mathbb{R}^{n \times n} \end{array}$$

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



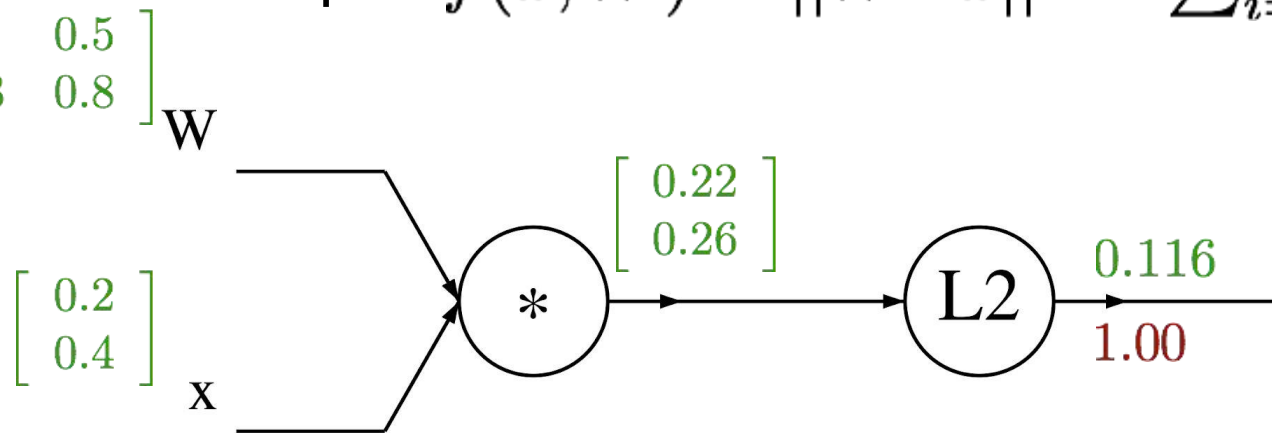
A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$



$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$
$$f(q) = ||q||^2 = q_1^2 + \cdots + q_n^2$$



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



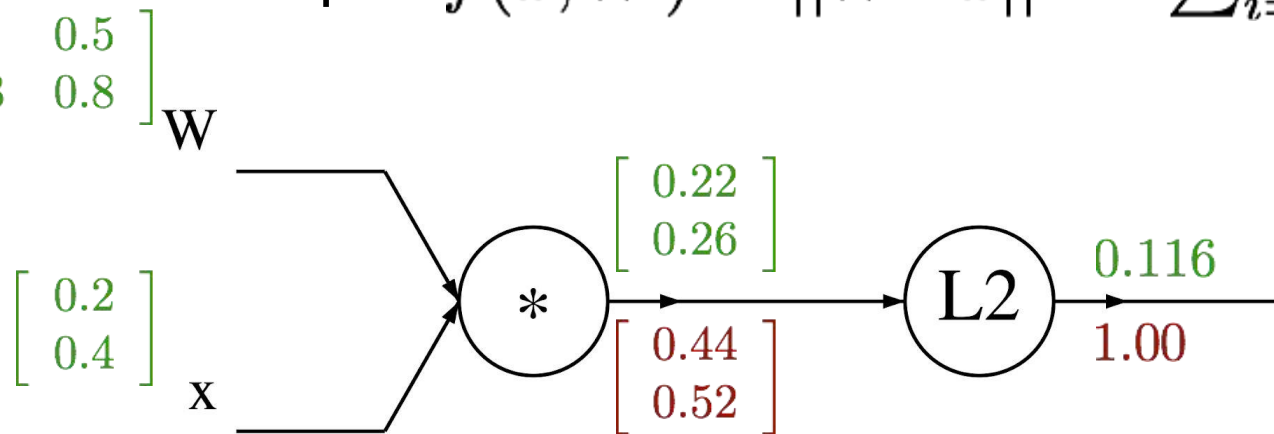
$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

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

$$\nabla_q f = 2q$$

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



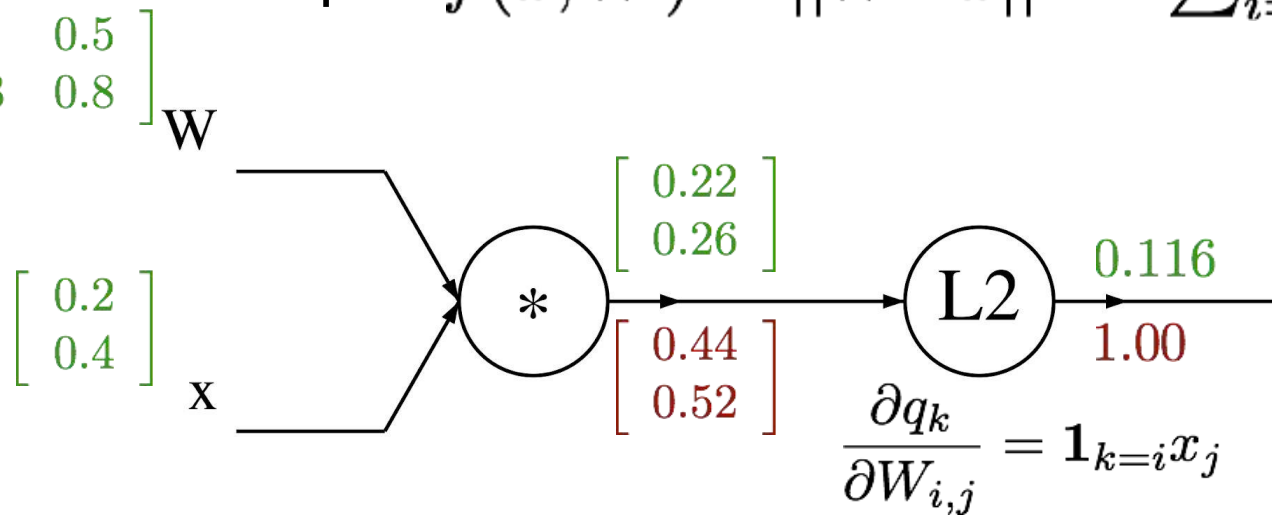
$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

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

$$\nabla_q f = 2q$$

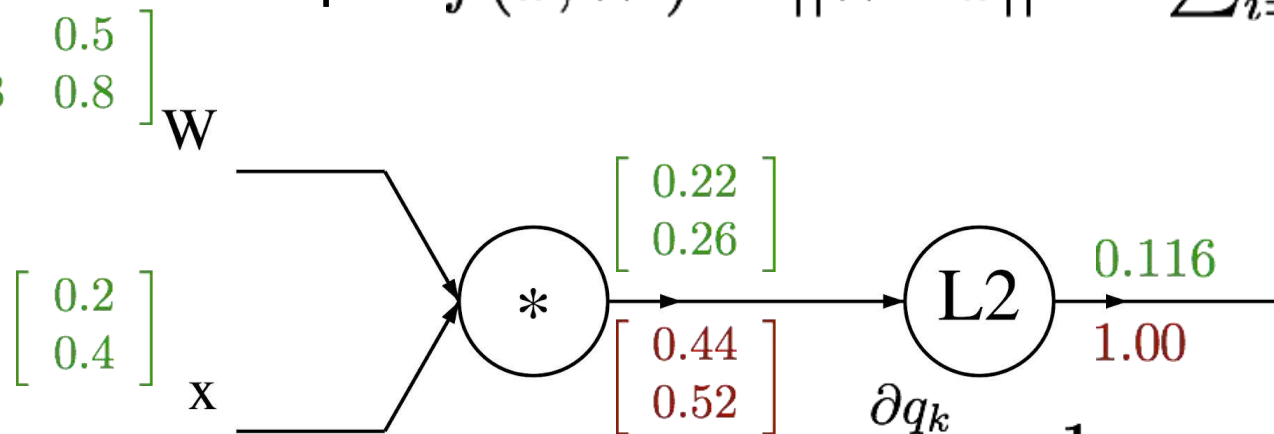
A vectorized example:  $f(x, W) = ||W \cdot x||^2 = \sum_{i=1}^n (W \cdot x)_i^2$



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

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

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

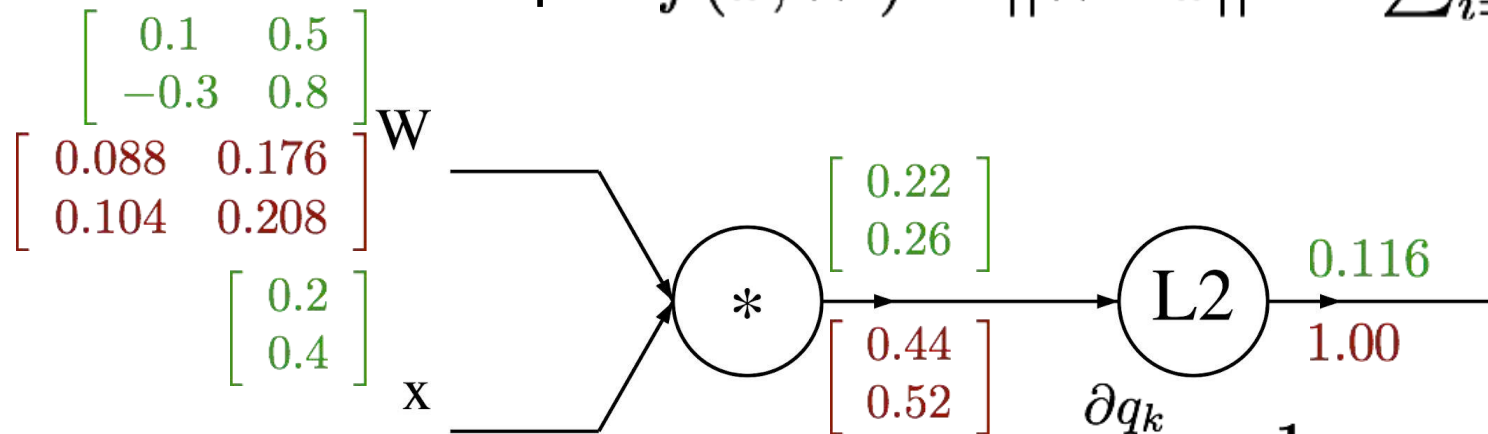


$$\begin{aligned} \frac{\partial q_k}{\partial W_{i,j}} &= \mathbf{1}_{k=i} x_j \\ \frac{\partial f}{\partial W_{i,j}} &= \sum_k \frac{\partial f}{\partial q_k} \frac{\partial q_k}{\partial W_{i,j}} \\ &= \sum_k (2q_k) (\mathbf{1}_{k=i} x_j) \\ &= 2q_i x_j \end{aligned}$$

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

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

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

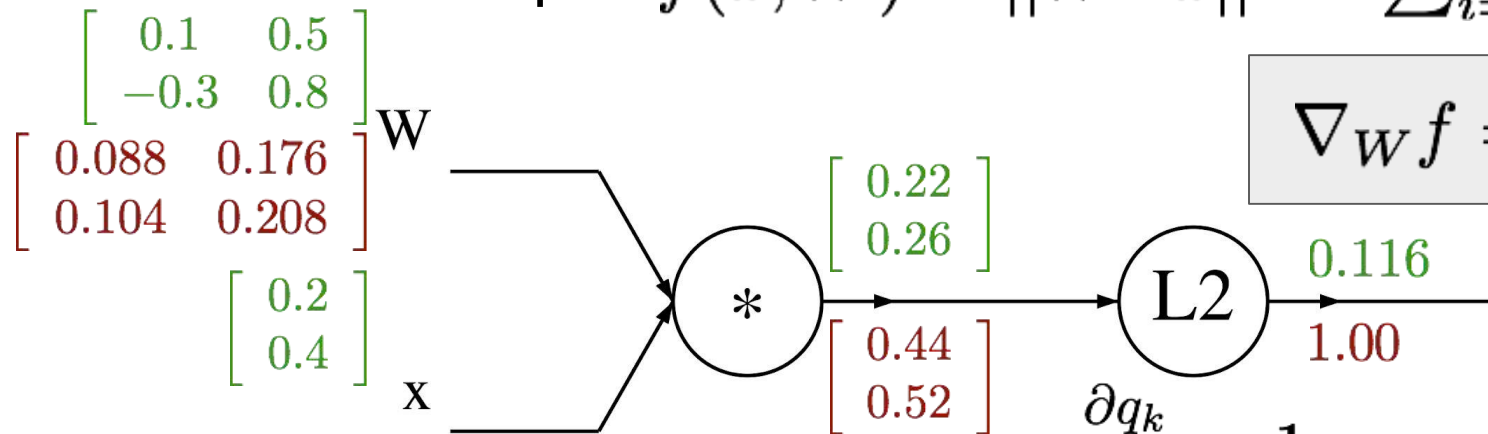


$$\begin{aligned} \frac{\partial q_k}{\partial W_{i,j}} &= \mathbf{1}_{k=i} x_j \\ \frac{\partial f}{\partial W_{i,j}} &= \sum_k \frac{\partial f}{\partial q_k} \frac{\partial q_k}{\partial W_{i,j}} \\ &= \sum_k (2q_k) (\mathbf{1}_{k=i} x_j) \\ &= 2q_i x_j \end{aligned}$$

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

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

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



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

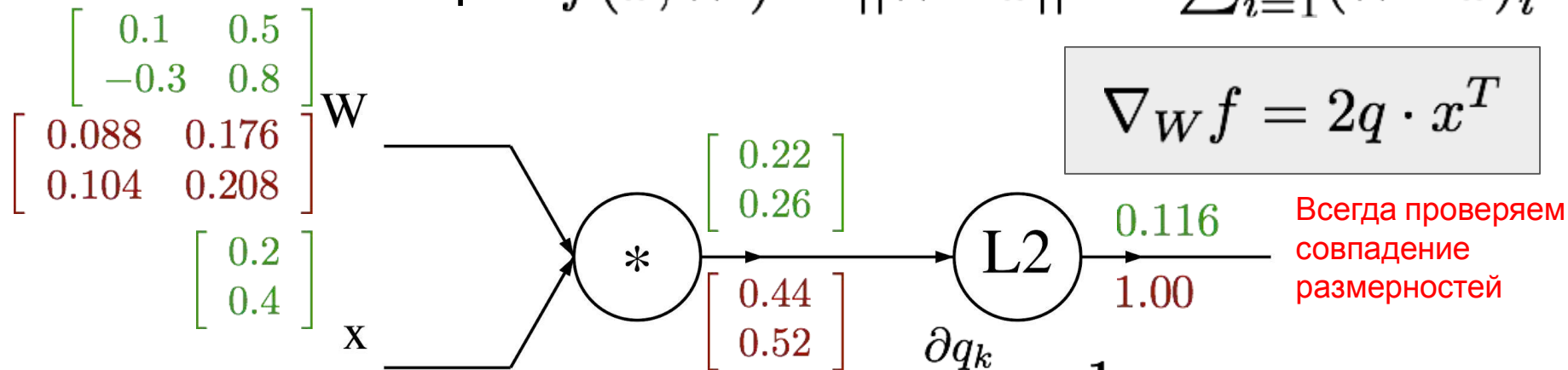
$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \dots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \dots + W_{n,n}x_n \end{pmatrix}$$

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

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

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

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

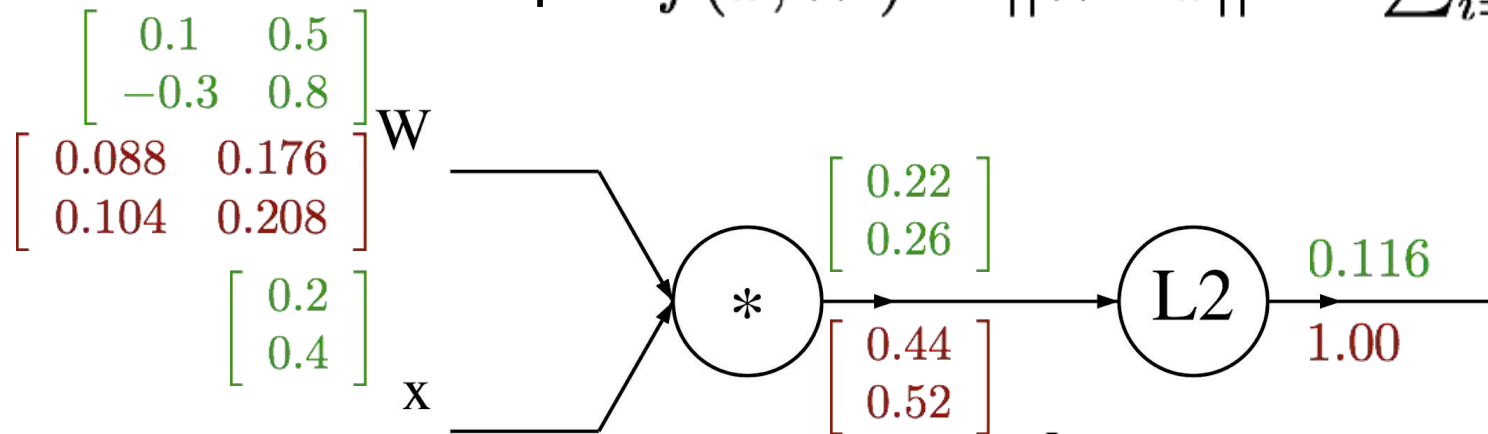


$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \dots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \dots + W_{n,n}x_n \end{pmatrix}$$

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

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

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



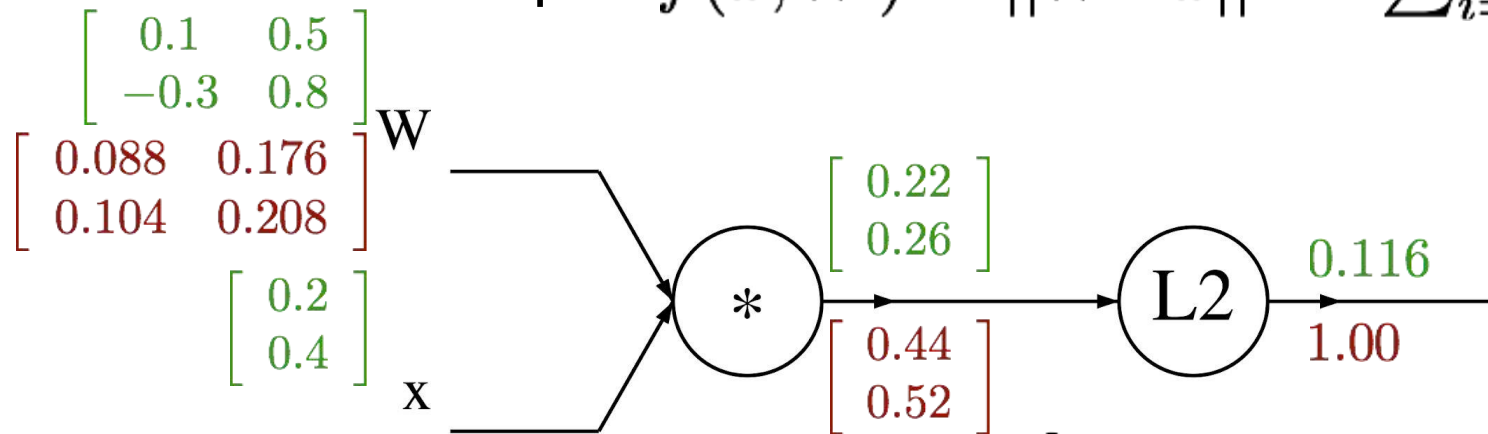
$$\frac{\partial q_k}{\partial x_i} = W_{k,i}$$

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

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



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

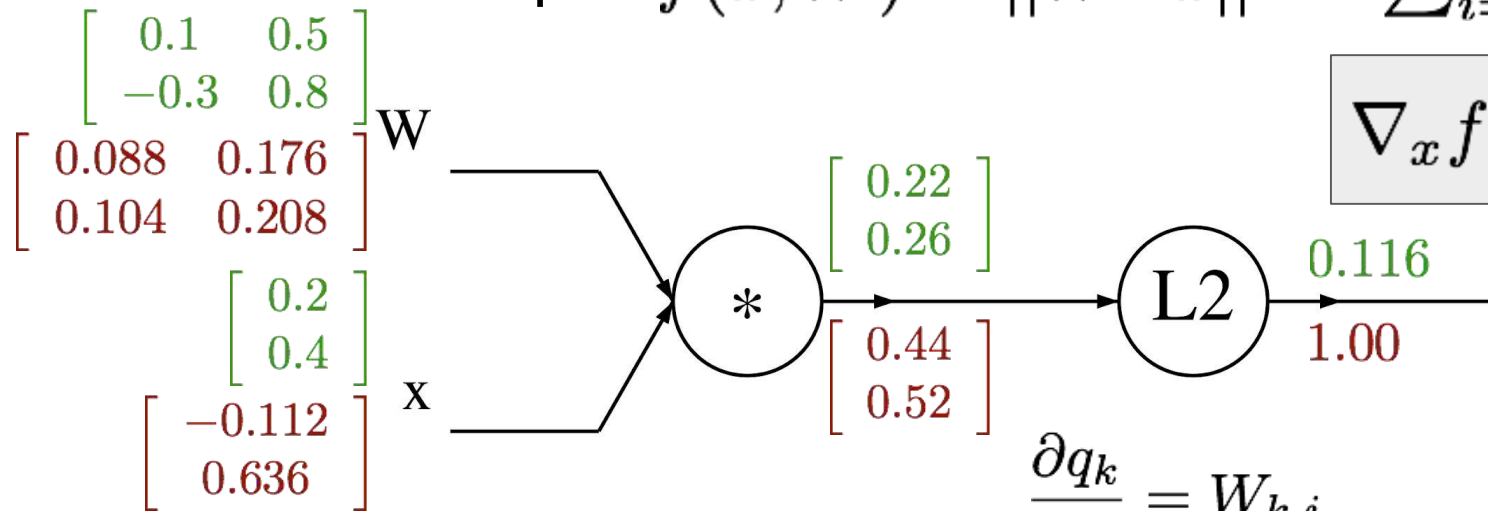


$$q = W \cdot x = \begin{pmatrix} W_{1,1}x_1 + \cdots + W_{1,n}x_n \\ \vdots \\ W_{n,1}x_1 + \cdots + W_{n,n}x_n \end{pmatrix}$$

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

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

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



$$\nabla_x f = 2W^T \cdot q$$

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

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

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