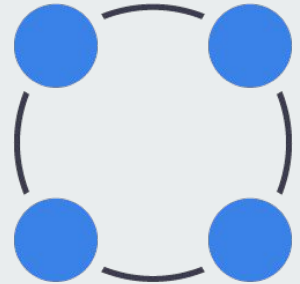




Машинное обучение

Лекция 7. Нейронные сети. Начало

(18.03.2024)

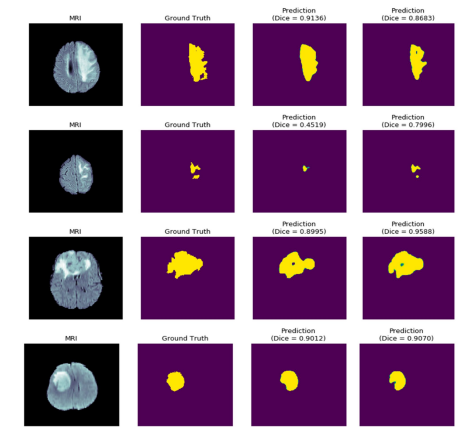
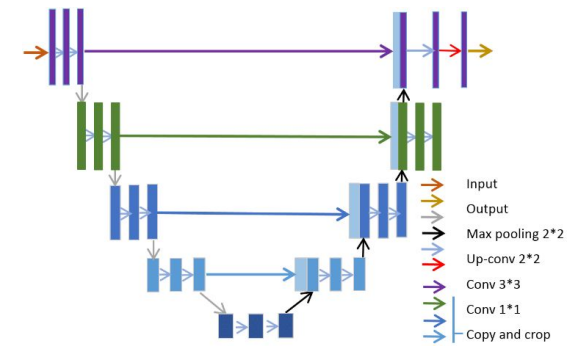
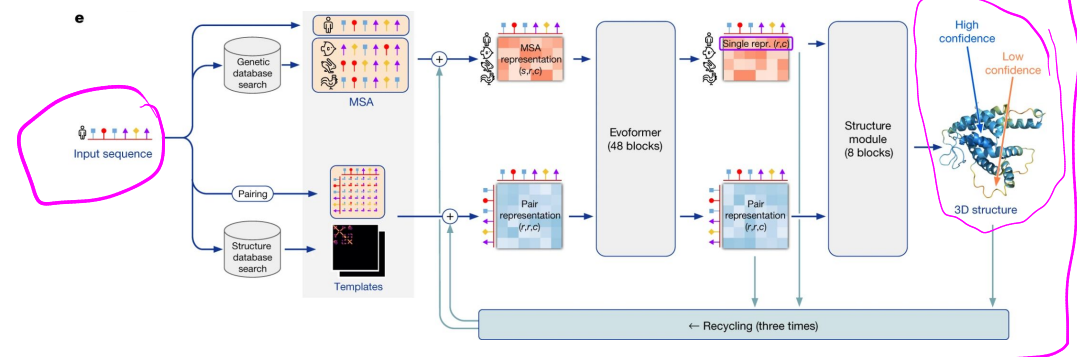
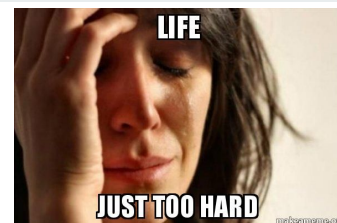




Начало.

UNET

AF2



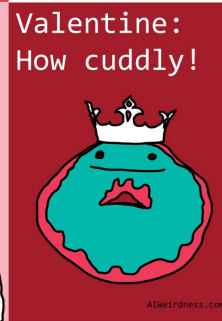
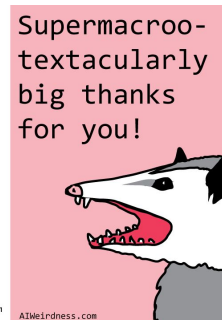
Inventing magic spells



I trained a neural network twice on [Dungeons and Dragons spells](#), and once on [spells from Harry Potter](#). See if you can figure out which list is which.

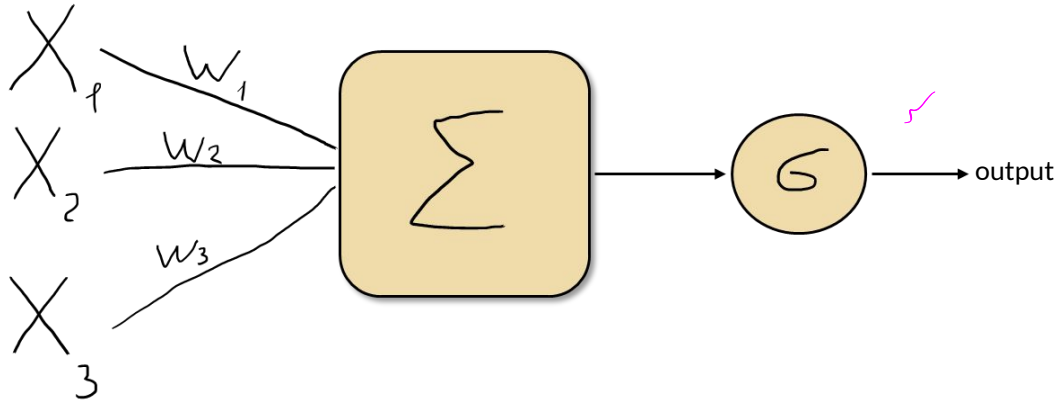
Chorus of the dave
Song of the doom goom
Barking Sphere
Gland Growth
Hold Mouse

Hurder-gerping Charm
Regrowing hair to curse of the Bogies
Brechaim hedbivicus Doobers Spell
Fubbledory Charm
Squggy-wing fart



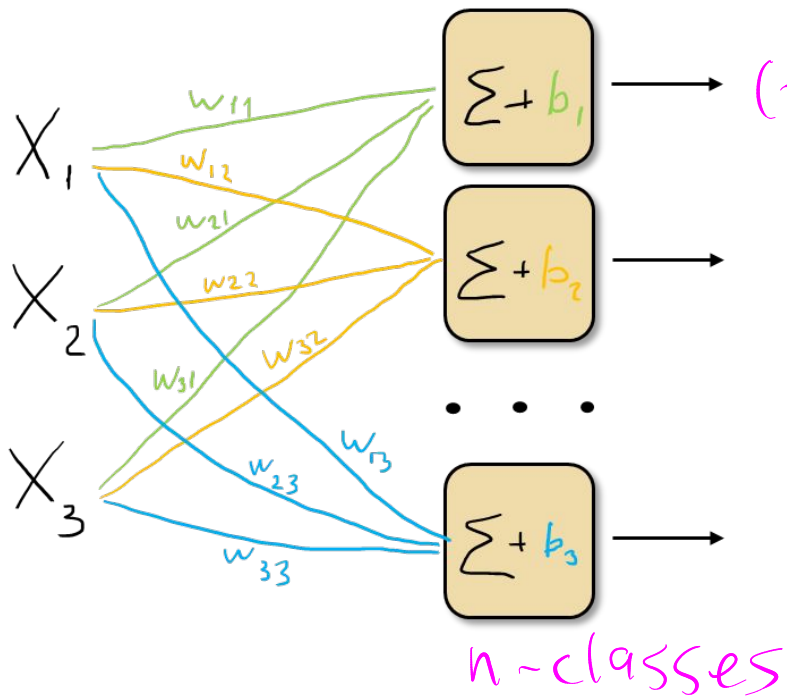
Вспомним логистическую регрессию

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



$w_0 + x_1 \cdot w_1 + x_2 \cdot w_2 + x_3 \cdot w_3$ - logits
 w_0 - bias
 w_i - weights

А если классов больше, чем 2...



logits

$(-\infty, +\infty)$ 10

0.5

⋮

-1

$\arg \max(\text{logits})$



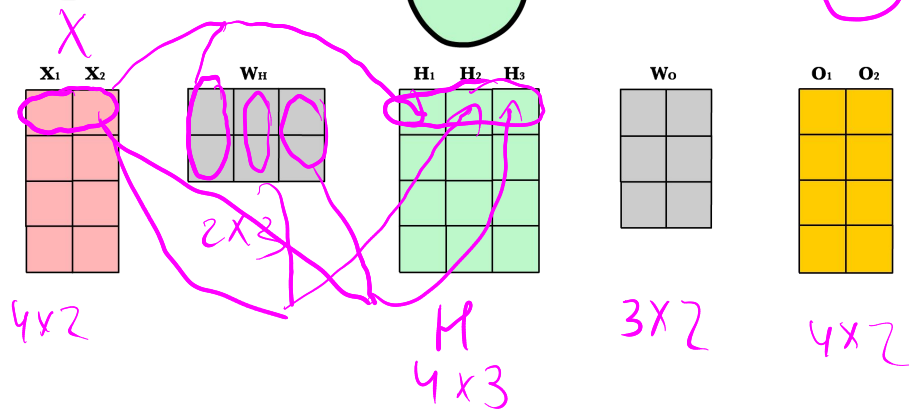
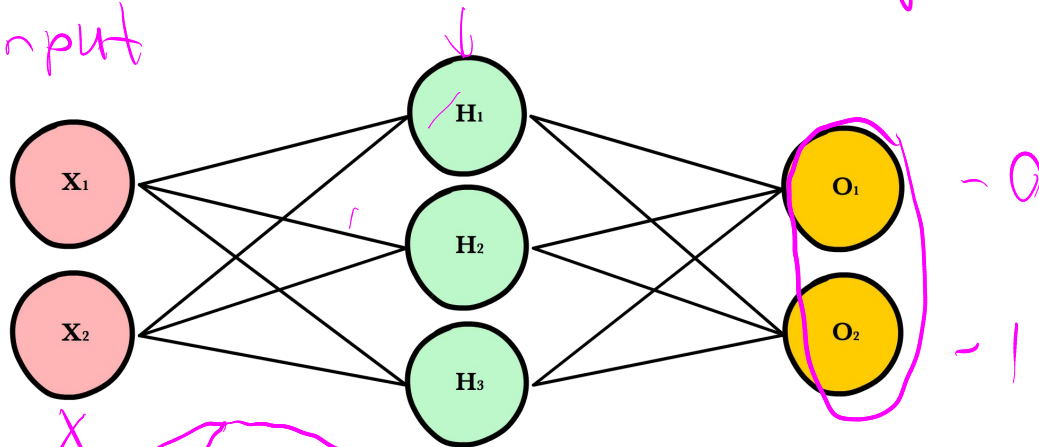
0



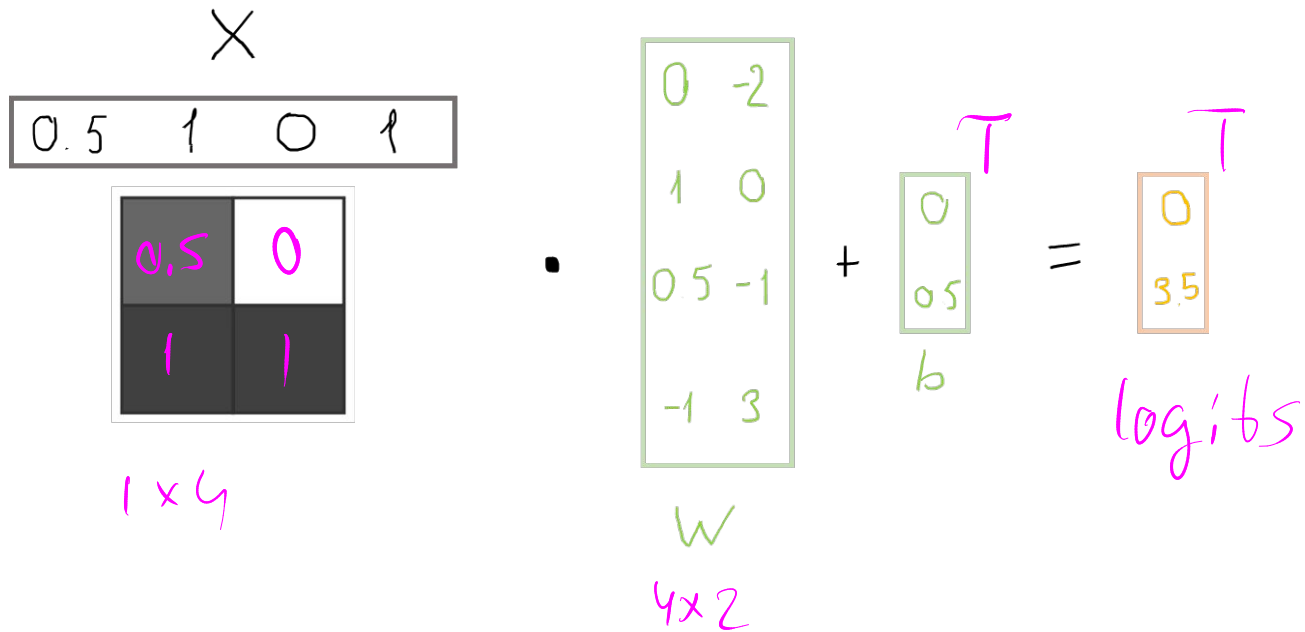
Ещё разок

linear $\hat{y} = \sigma(Xw)$
hidden output

input



Бинарная классификация



The diagram illustrates the calculation of logits for binary classification using matrix multiplication.

Input Vector X (1x4):

0.5	1	0	1
-----	---	---	---

Weight Matrix W (4x2):

0	-2
1	0
0.5	-1
-1	3

Bias Vector b (1x2):

0
0.5

Logits (1x2):

0
3.5

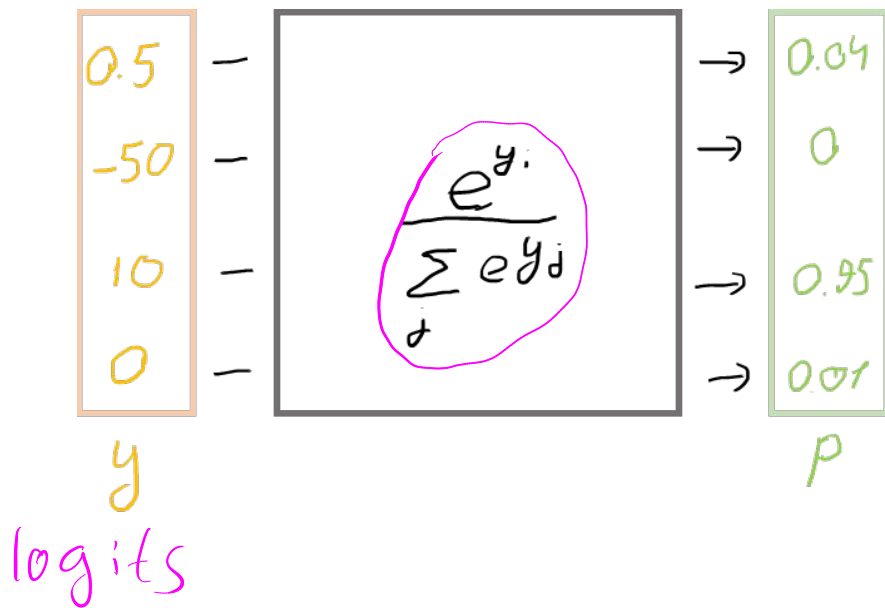
The calculation is represented as:

$$X \cdot W + b = \text{logits}$$

Handwritten annotations include:

- 1×4 for the input vector X .
- 4×2 for the weight matrix W .
- 0 and 0.5 for the bias vector b .
- 0 and 3.5 for the resulting logits.

Softmax



$$BCE = -\frac{1}{N} \sum_i [y_i \ln p_i + (1-y_i) \ln(1-p_i)]$$

Принцип максимального правдоподобия. Maximum likelihood

likelihood


$$0.8 + 0.9 + 1$$

$$\prod_s p(c = gt_s | x_s) \rightarrow \max$$

$$\sum_s \ln p(c = gt_s | x_s) \rightarrow \max$$

$$-\sum_s \ln p(c = gt_s | x_s) \rightarrow \min$$

cross entropy

samples	features	labels	predictions
1 	x_1, x_2, \dots, x_n	0	0.9 0.09 0.01 <u>0.1 0.9 0</u>
2 	...	1	0.1 0.9 0
3 	...	2	0.0 0.1

А как учиться?



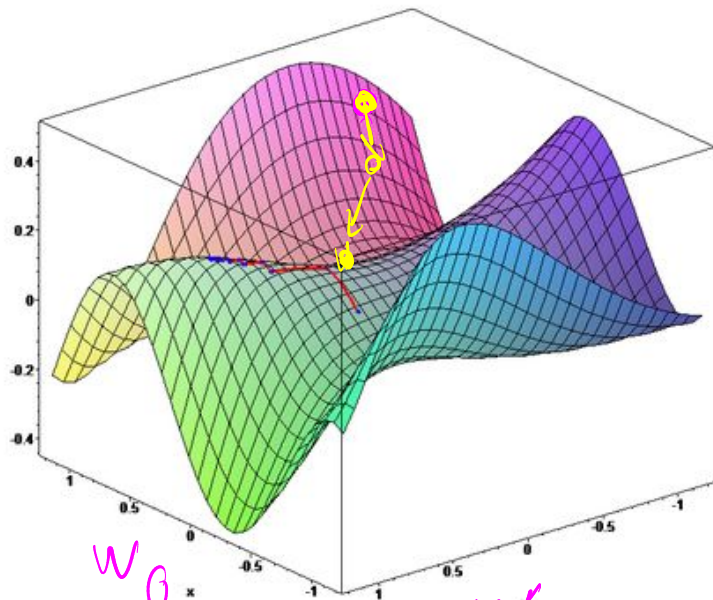
Лекции в универе похожи на просмотр Даши путешественницы: препод задаёт вопросы и несколько секунд пялится на аудиторию, а потом сам же отвечает.



А как учиться? Градиентный спуск



BCE



w_0

w_1

μ

RAM \sim 8 GB

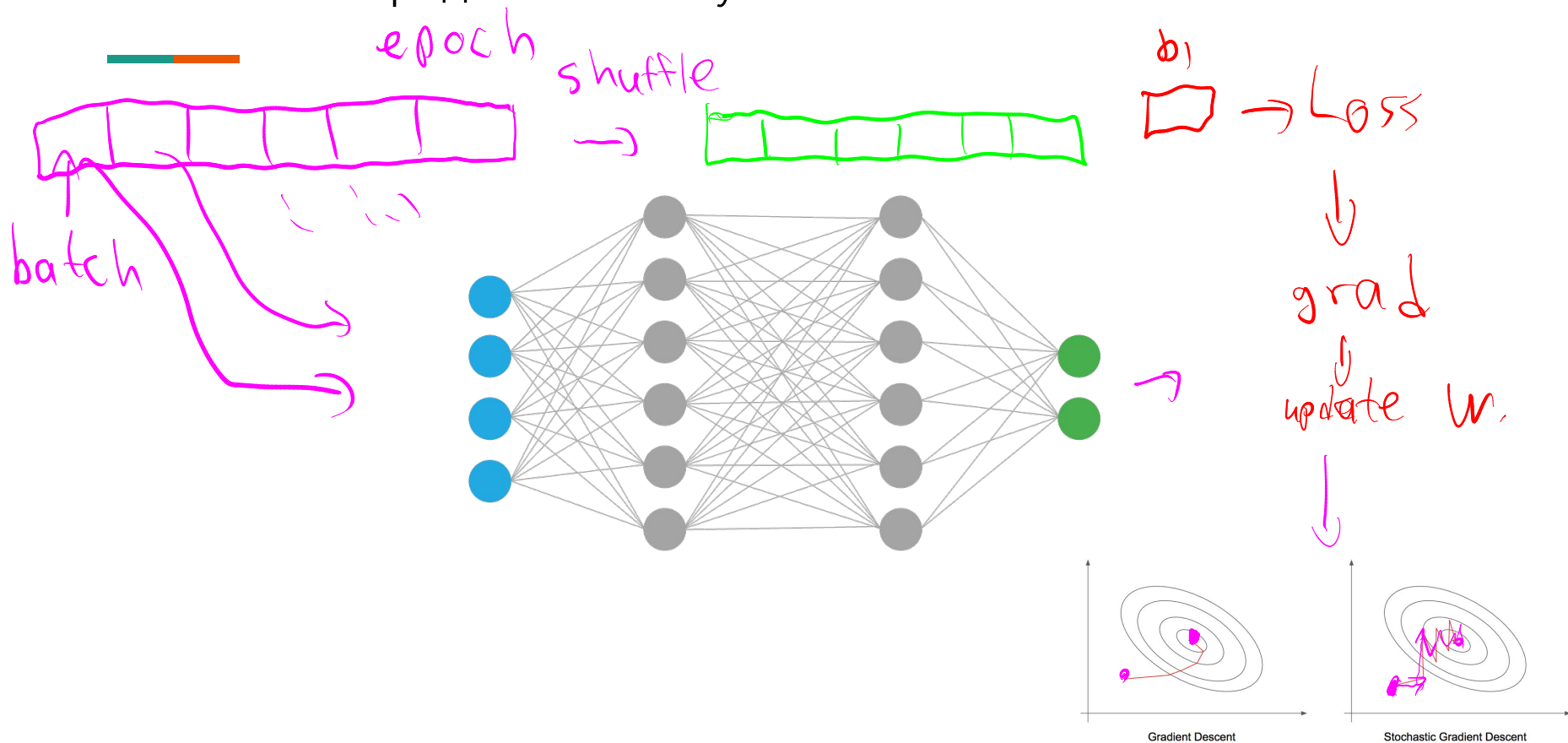
VRAM \sim 4 GB

float32 \sim 4 байт

1000000 \sim 4 MB

10^2 \sim 4 GB

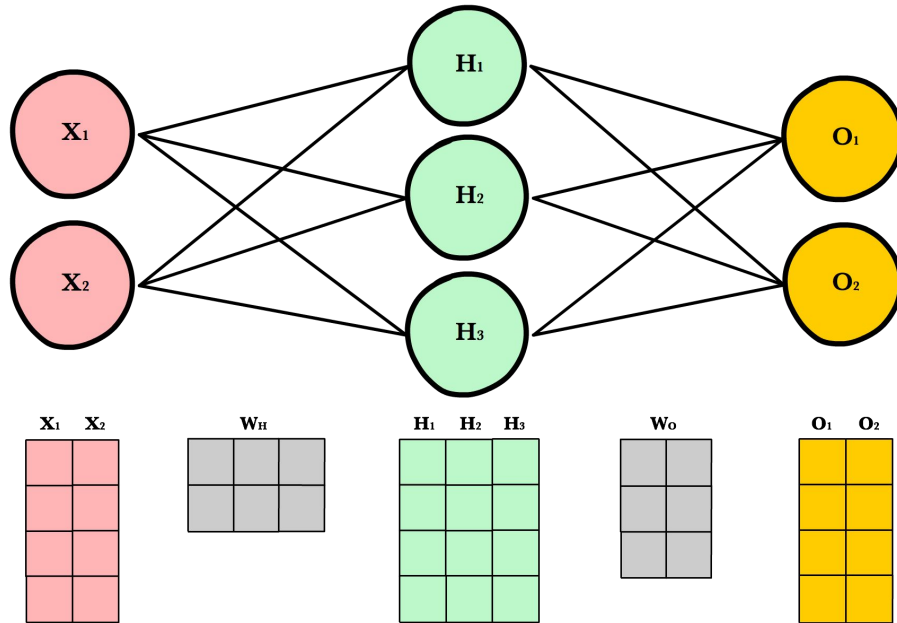
Стохастический градиентный спуск. Stochastic Gradient Descent (SGD)



ОТДЫХ

$$(2 \cdot 2) \cdot 3 = 2 \cdot (2 \cdot 3)$$

Будет ли работать просто так?



$$X \cdot W_{H \rightarrow O}$$

4x2 || 2x2

$$(X \cdot W_H) \cdot W_O \Rightarrow X \cdot (W_H \cdot W_O)$$

2x3 || 3x2

Функции активации

Хотим, чтобы активация была:

Нелинейная:

Функция активации необходима для введения нелинейности в нейронные сети. Если функция активации не применяется, выходной сигнал становится простой линейной функцией. Неактивированная нейронная сеть будет действовать как линейная регрессия с ограниченной способностью к обучению:

✓

$$y = NN(X, W_1, \dots, W_n) = X \cdot W_1 \cdot \dots \cdot W_n = X \cdot W$$

Только нелинейные функции активации позволяют нейронным сетям решать задачи аппроксимации нелинейных функций:

$$y = NN(X, W_1, \dots, W_n) = \sigma(\dots \sigma(X \cdot W_1) \dots W_n) \neq X \cdot W$$

Дифференцируемая:

Функции активации должны быть дифференцируемые, то есть от них можно взять производную

✓

Функции активации. Сигмоида

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

$\sigma(x)$

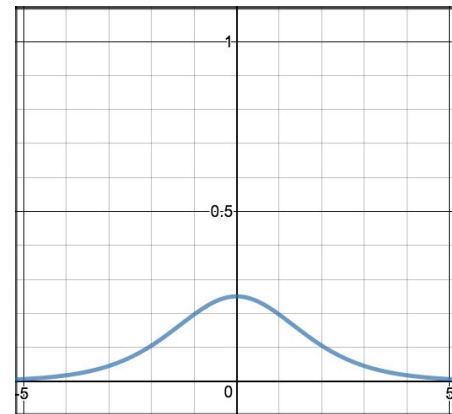
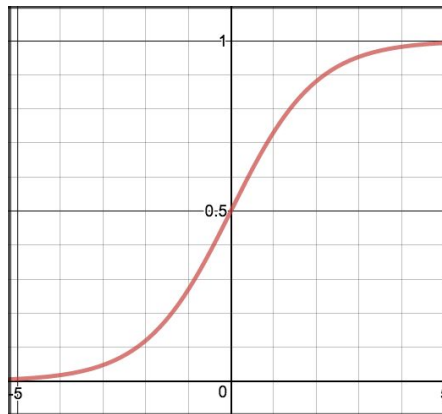
$\sigma'(x)$

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

$$\sigma'(x) = \frac{e^x \cdot (e^x+1) - e^{2x}}{(e^x+1)^2} =$$

$$= \frac{e^{2x} + e^x - e^{2x}}{(e^x+1)^2} = \frac{e^x}{(e^x+1)^2}$$


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

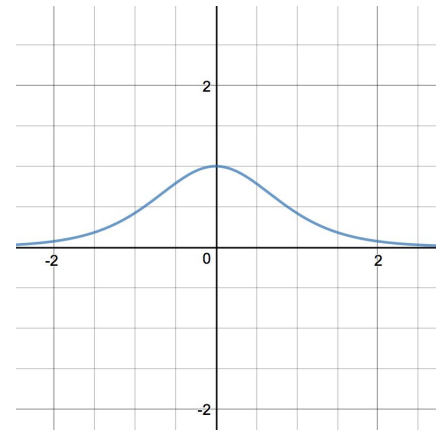
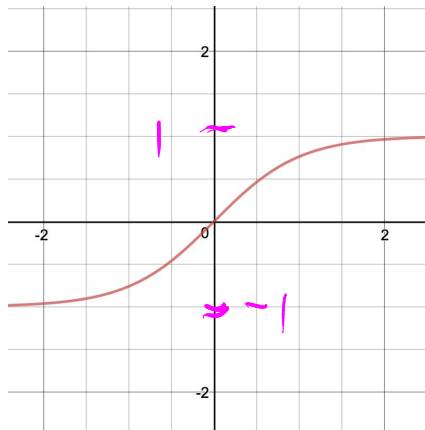


Недостатки:

1. Насыщение сигмоиды приводит к затуханию градиентов
2. Выход сигмоиды не центрирован относительно нуля

Функции активации. Гиперболический тангенс


$$\tanh(x) = \frac{2}{1+e^{-2x}} - 1$$



Недостатки:

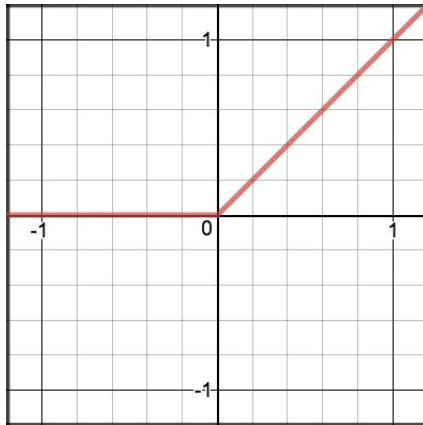
1. Снова затухание градиентов

Функции активации. ReLU (rectified linear unit)

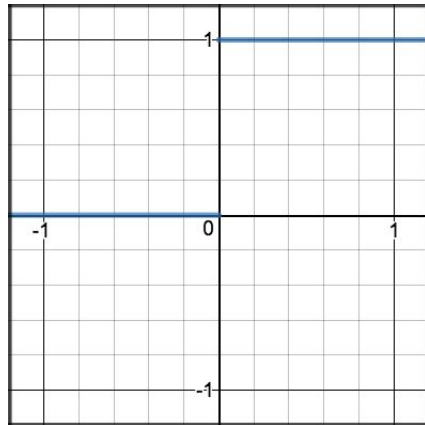


$$\text{relu}(x) = \max(0, x)$$

ReLU



ReLU'



Недостатки:

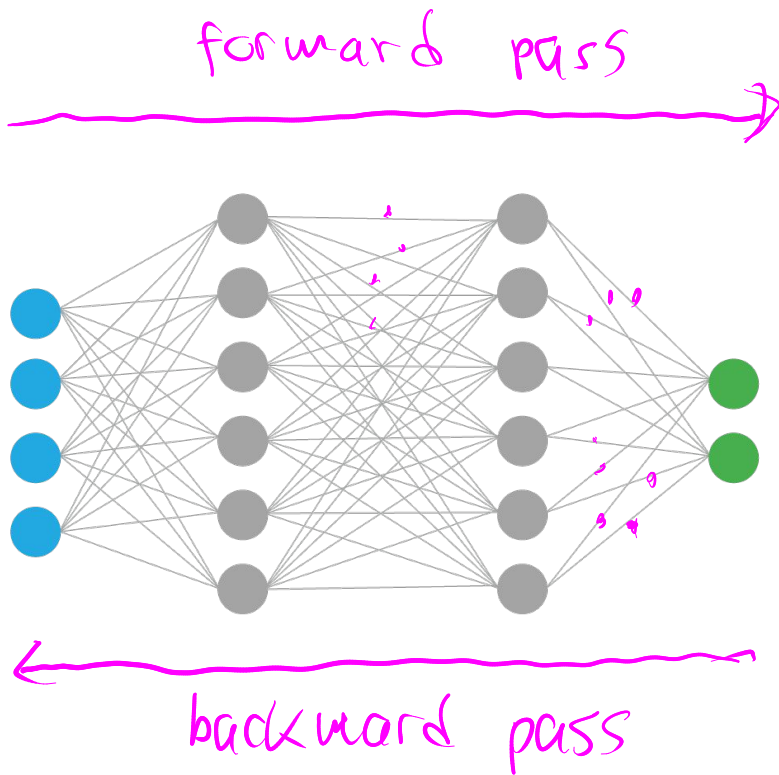
1. Может “умереть” в области слева от 0

Функции активации. Другие функции активации



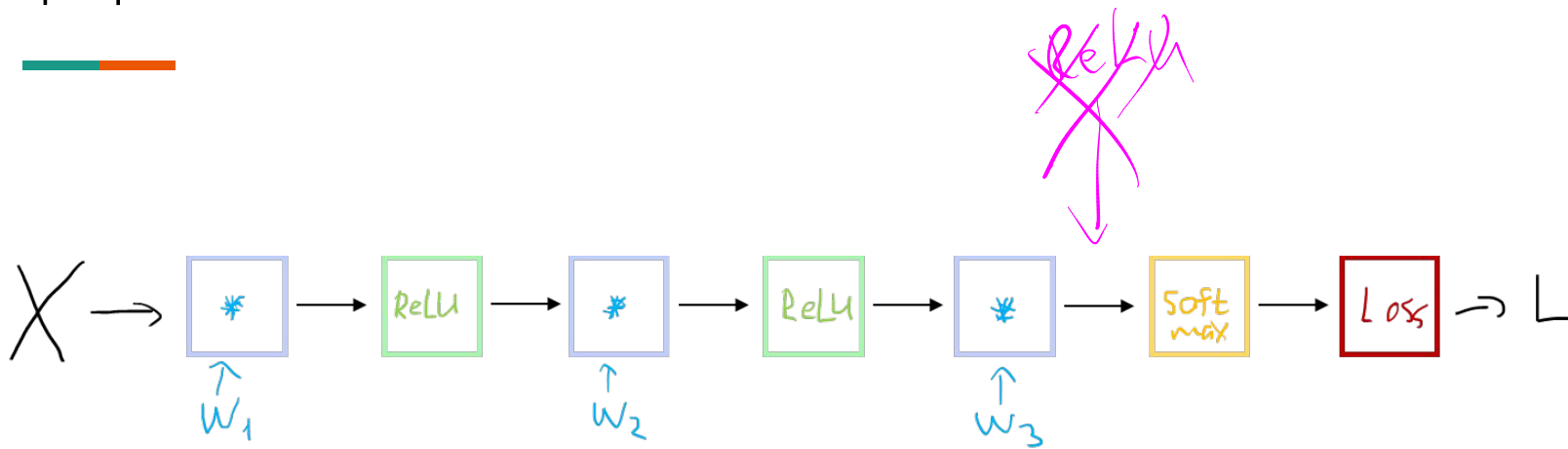
Identity	Sigmoid	TanH	ArcTan
ReLU	Leaky ReLU	Randomized ReLU	Parametric ReLU
Binary	Exponential Linear Unit	Soft Sign	Inverse Square Root Unit (ISRU)
Inverse Square Root Linear	Square Non-Linearity	Bipolar ReLU	Soft Plus

Тренировка



preds \rightarrow Loss
 $\nabla_{\mathbf{w}}$ Loss

Граф вычислений



Алгоритм обратного распространения ошибки. Backpropagation

Алгоритм обратного распространения ошибки позволяет находить градиенты для любого графа вычислений, если функция которую он описывает дифференцируема (каждый из узлов дифференцируемый). В его основе лежит правило взятия производной сложной функции (chain rule).

$$f(g(x))$$

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

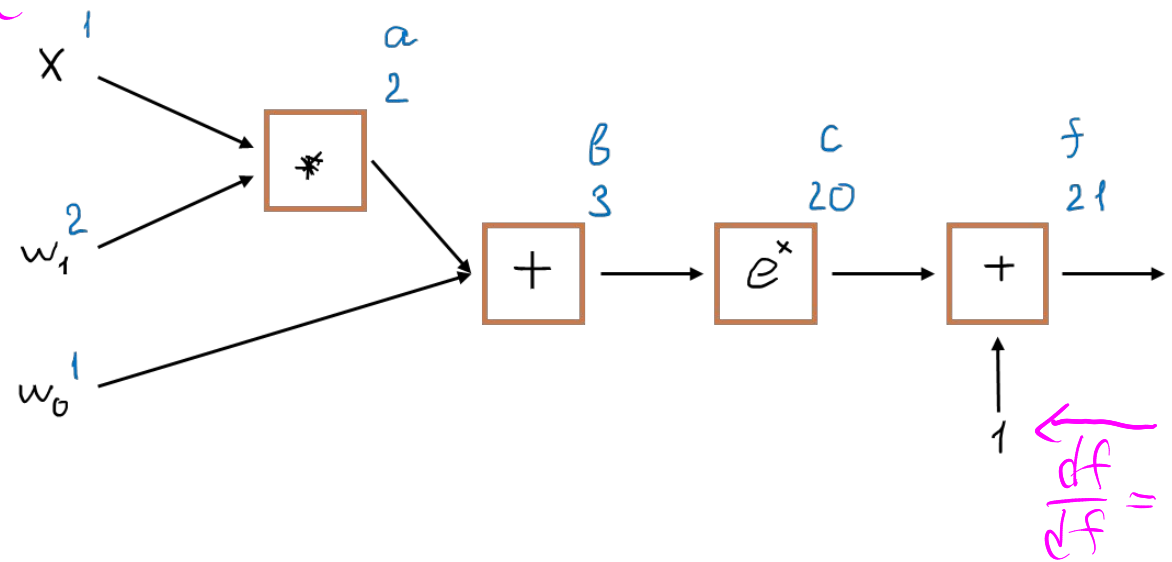
Тренируемся

$\frac{df}{dc} \approx \frac{df}{df} \cdot \frac{df}{dc}$
 $f = c + 1$

$f(g(x))$

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

$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$



$\frac{\partial f}{\partial w_0} - ?$
 $\frac{\partial f}{\partial w_1} - ?$
 $\frac{\partial f}{\partial x} - ?$

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

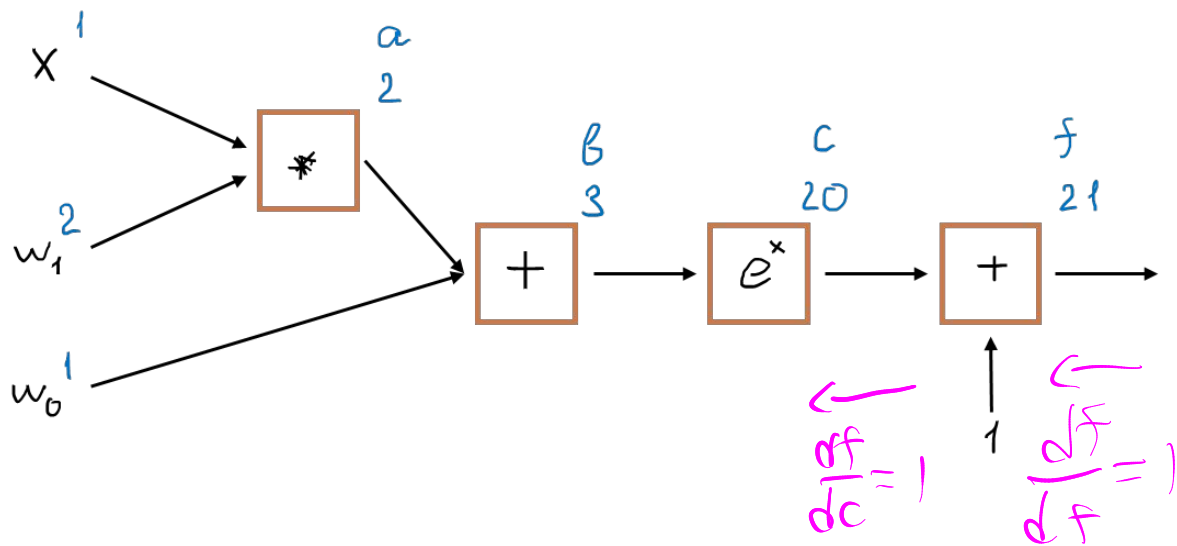
Тренируемся

$\frac{df}{db} = \frac{df}{dc} \cdot \frac{dc}{db}$
 $c = e^b$
 $\frac{dc}{db} = 20$

$$f(g(x))$$

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

$$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$$



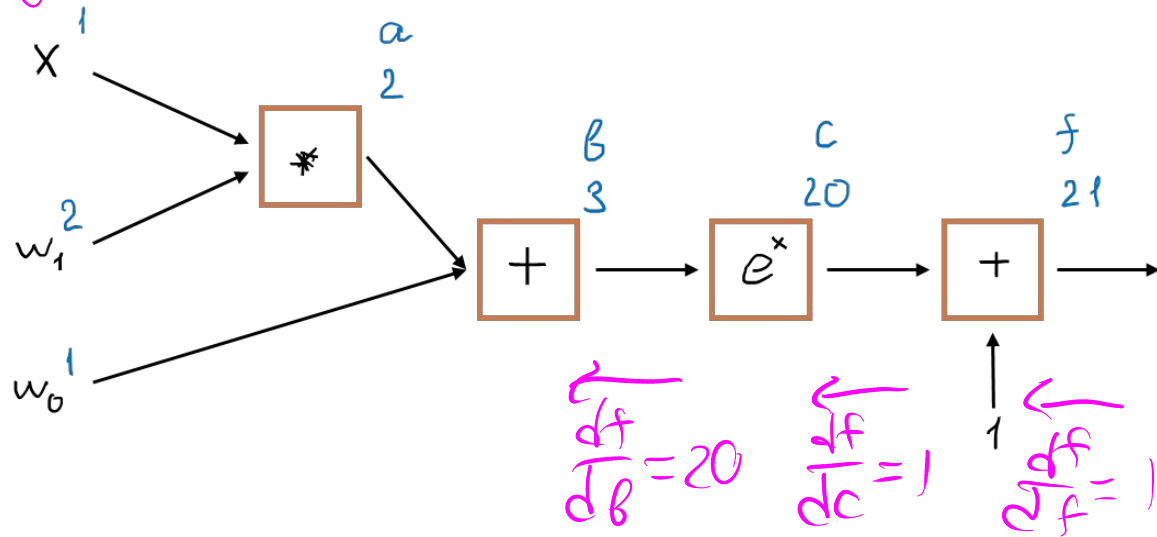
Тренируемся

$\frac{df}{dw_0} = \frac{df}{db} \cdot \frac{db}{dw_0} = 20$
 $b = w_0 + a$

$$f(g(x))$$

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

$$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$$



Тренируемся

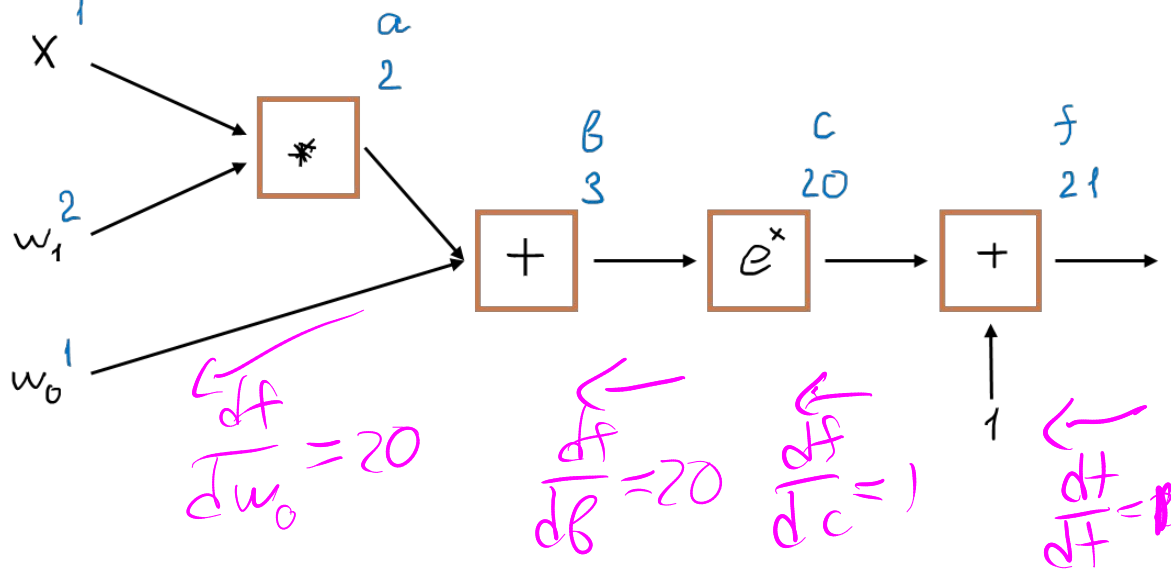
$$\frac{df}{da} = \frac{df}{db} \cdot \frac{db}{da}$$

$b = a + w_0$

$$f(g(x))$$

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

$$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$$

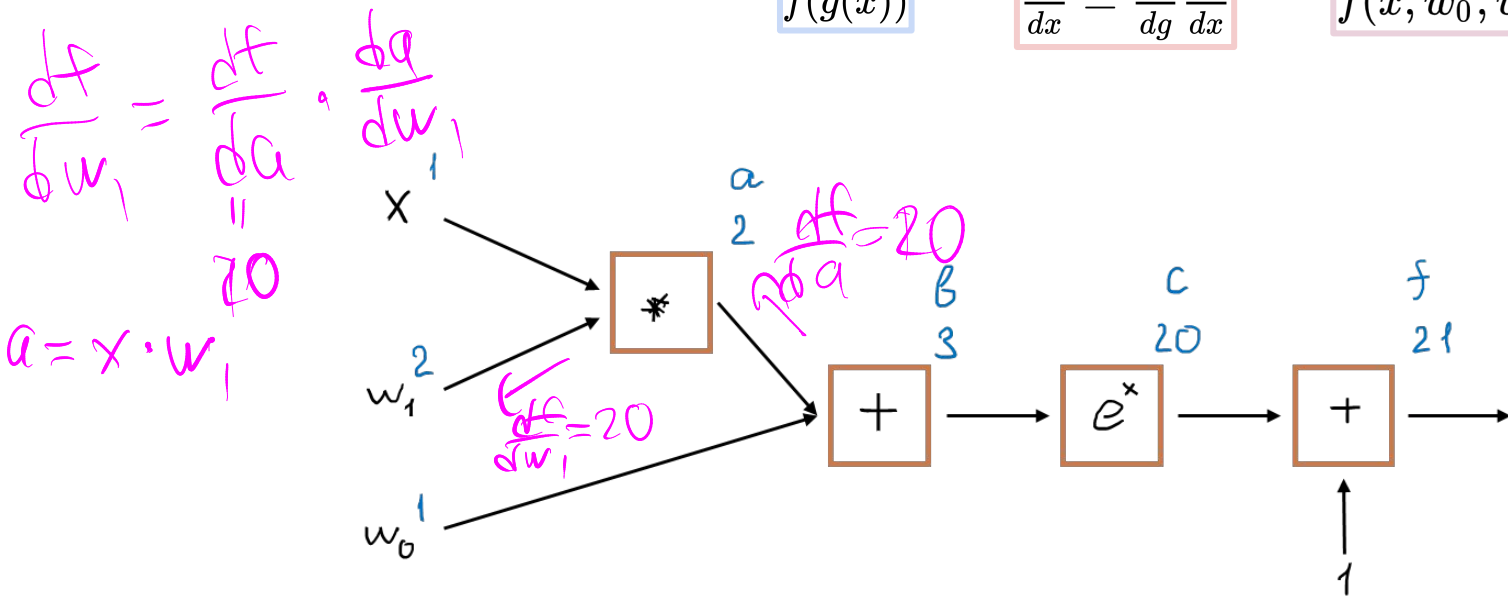


Тренируемся

$$f(g(x))$$

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

$$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$$

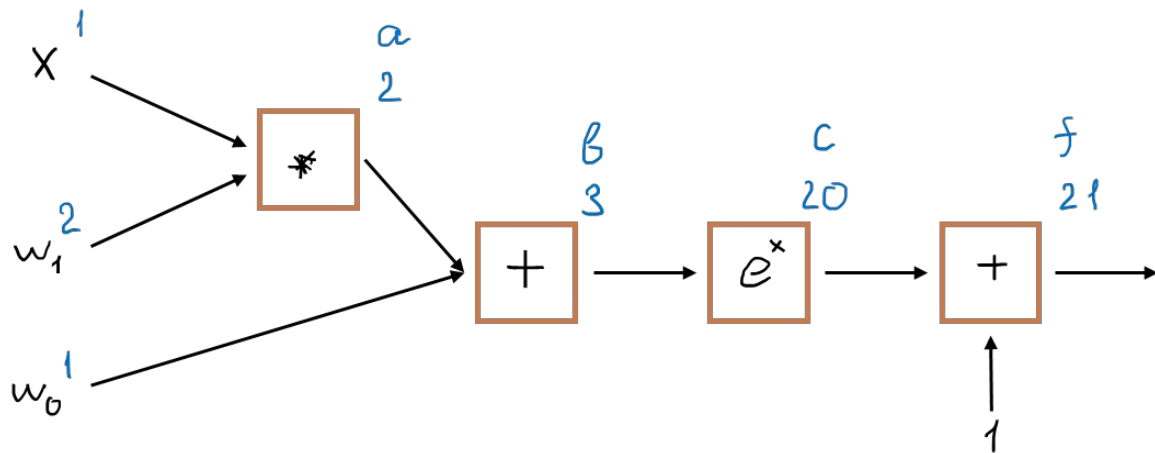


Тренируемся

$$\frac{df}{dx} = \frac{df}{da} \cdot \frac{da}{dx} = 40$$

11
20

$$a = x \cdot w_1$$



$$f(g(x))$$

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

$$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$$

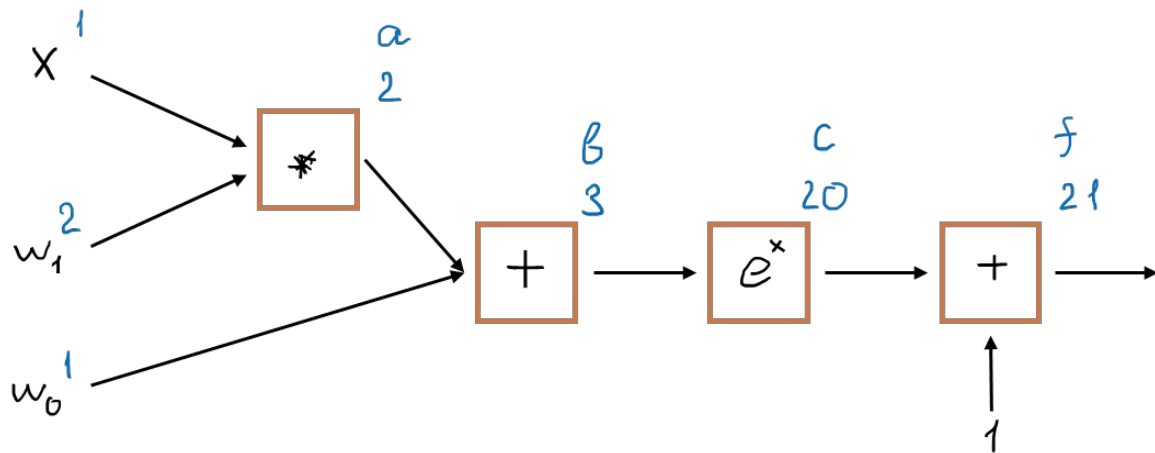
Тренируемся



$$f(g(x))$$

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

$$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$$



Тренируемся



$$f(g(x))$$

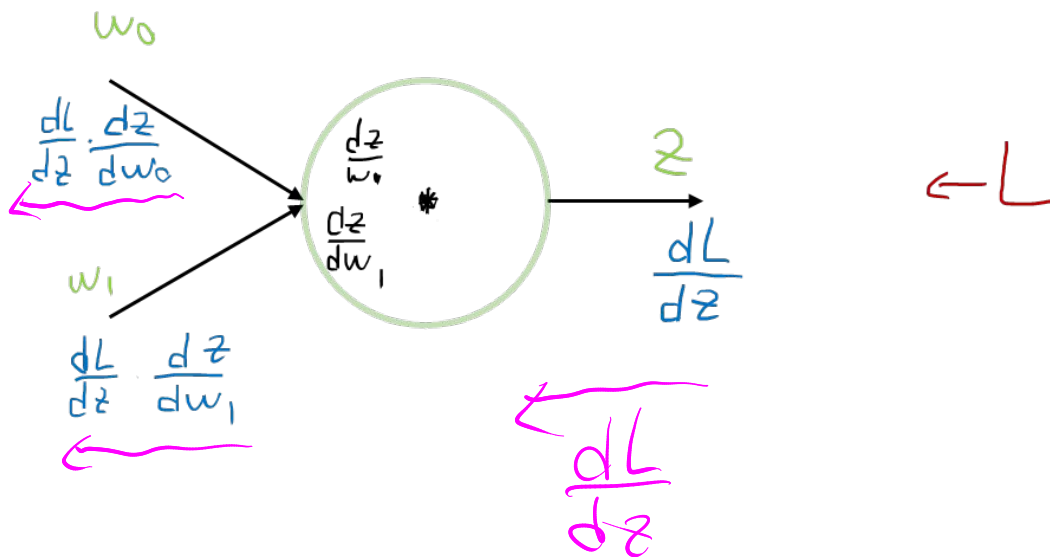
$$\frac{df}{dx} = \frac{df}{dg} \frac{dg}{dx}$$

$$f(x, w_0, w_1) = 1 + e^{w_0 + w_1 x}$$

А зачем мы все это считали?

$$\begin{aligned} w_0 &= w_0 - \frac{\partial f}{\partial w_0} \\ w_1 &= w_1 - \frac{\partial f}{\partial w_1} \end{aligned}$$

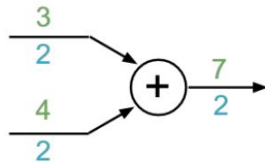
Общая схема вычисления градиента



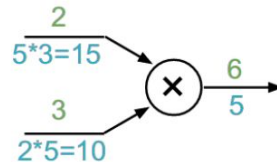
Уточнения



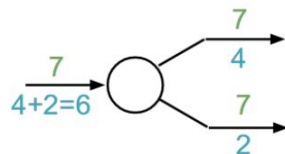
add gate: gradient distributor



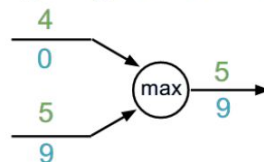
mul gate: "swap multiplier"



copy gate: gradient adder



max gate: gradient router



знаешь почему ты
так сильно устаёшь к концу дня?
потому что ты весь день был
замечательным котёночком,
а это тяжелый труд

