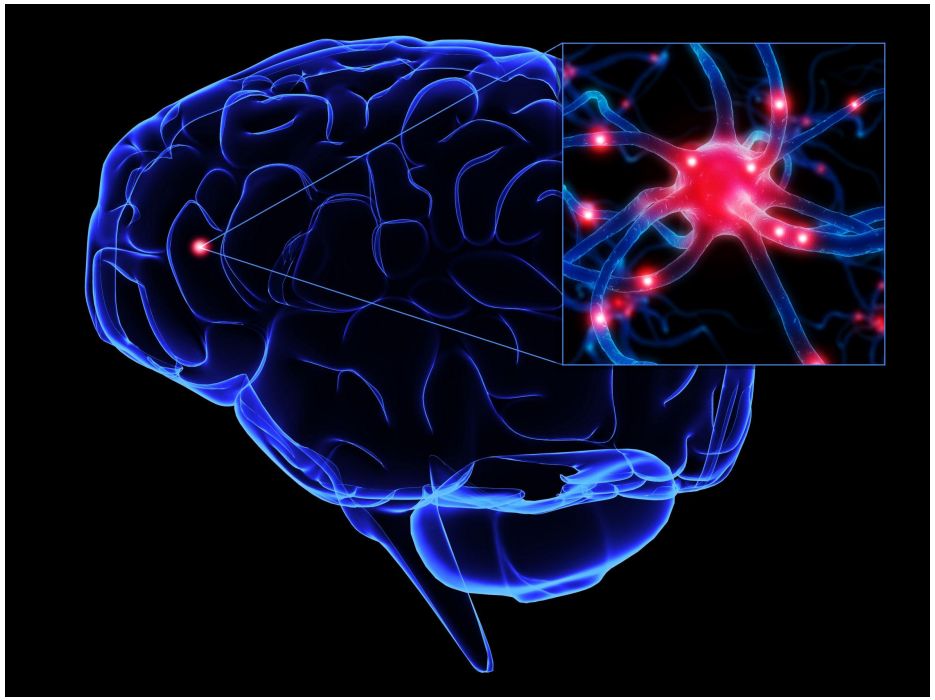
Введение в нейросети

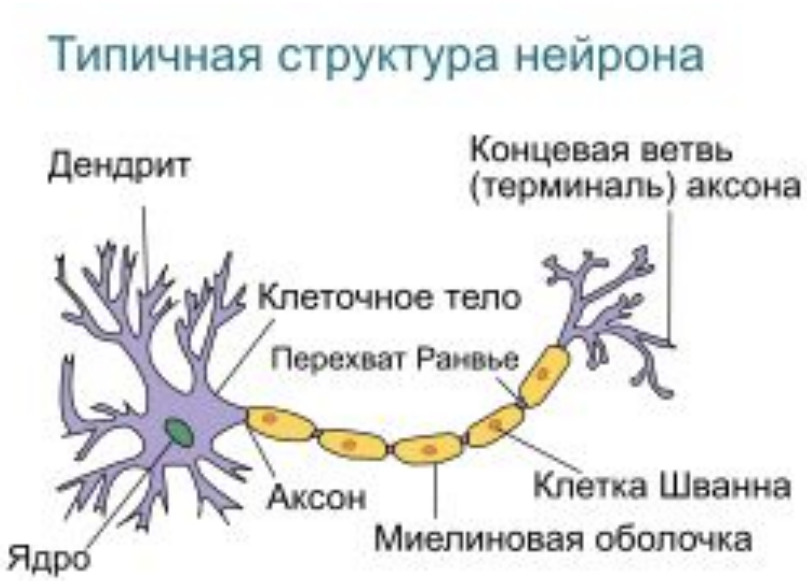


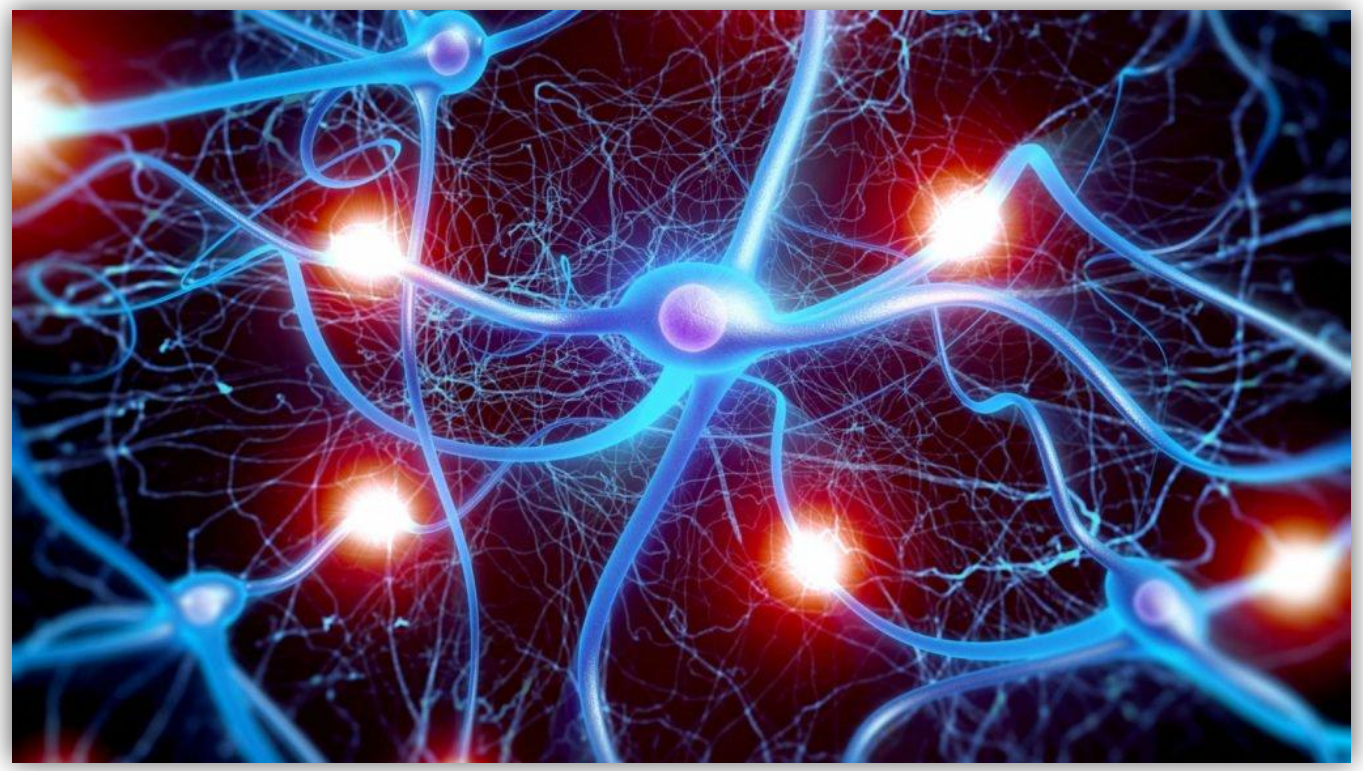
Исторический очерк

- Термин **«нейронная сеть»** появился в середине XX века. Первые работы, в которых были получены основные результаты в данном направлении, были проделаны **Мак-Каллоком** и **Питтсом**. В **1943** году ими была разработана **компьютерная модель нейронной сети** на основе математических алгоритмов и теории деятельности головного мозга.
- Они выдвинули предположение, что нейроны можно упрощённо рассматривать как устройства, оперирующие двоичными числами, и назвали эту модель «пороговой логикой». Подобно своему биологическому прототипу **нейроны** Мак-Каллока–Питтса были способны обучаться путём подстройки параметров, описывающих **синптическую** проводимость.
- Исследователи предложили конструкцию сети из электронных нейронов и показали, что подобная сеть может выполнять **практически любые вообразимые числовые или логические операции**. Мак-Каллок и Питтс предположили, что такая сеть в состоянии также **обучаться**, распознавать образы, обобщать, т. е. обладает всеми чертами интеллекта.

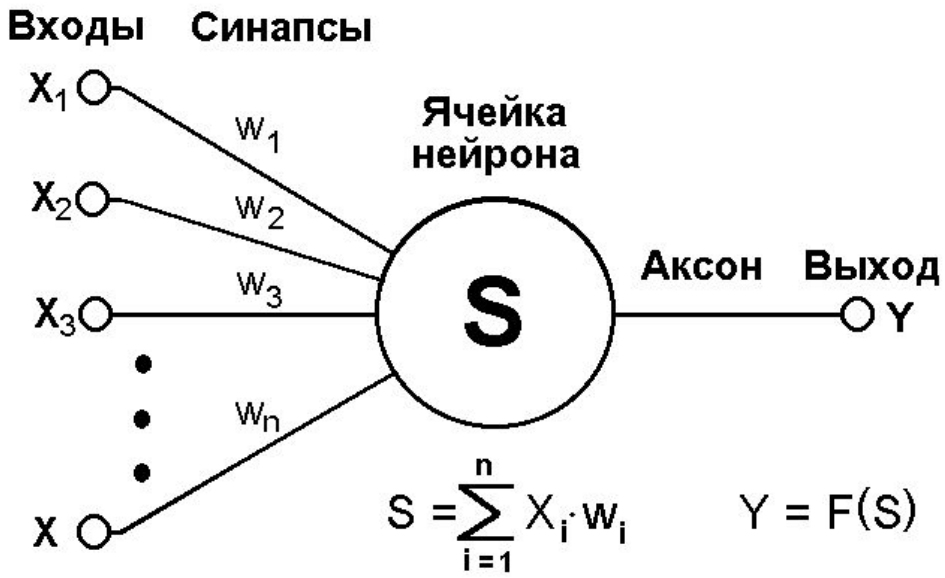
Источник: https://neuronus.com/history/5-istoriya-nejronnykh-setej.html

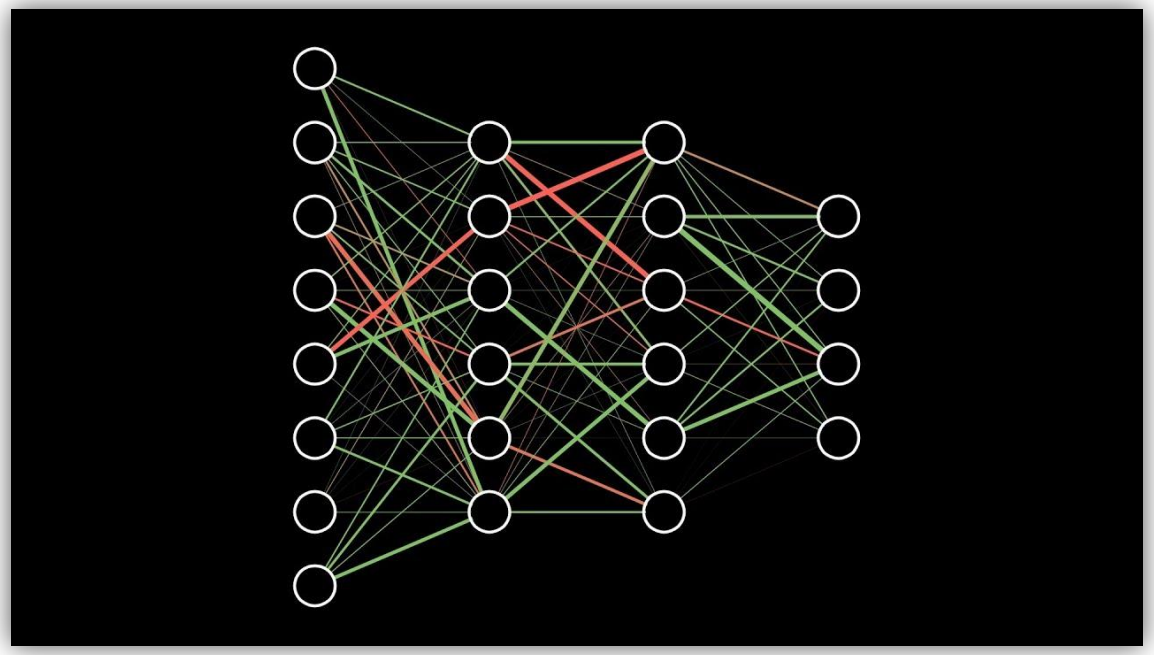
Человеческий мозг





Искусственная НС





На пальцах

Предположим, у нас есть три разных бинарных условия (да или нет) и одно бинарное решение на выходе (да или нет):

На вечеринке будет водка

На вечеринке будут друзья

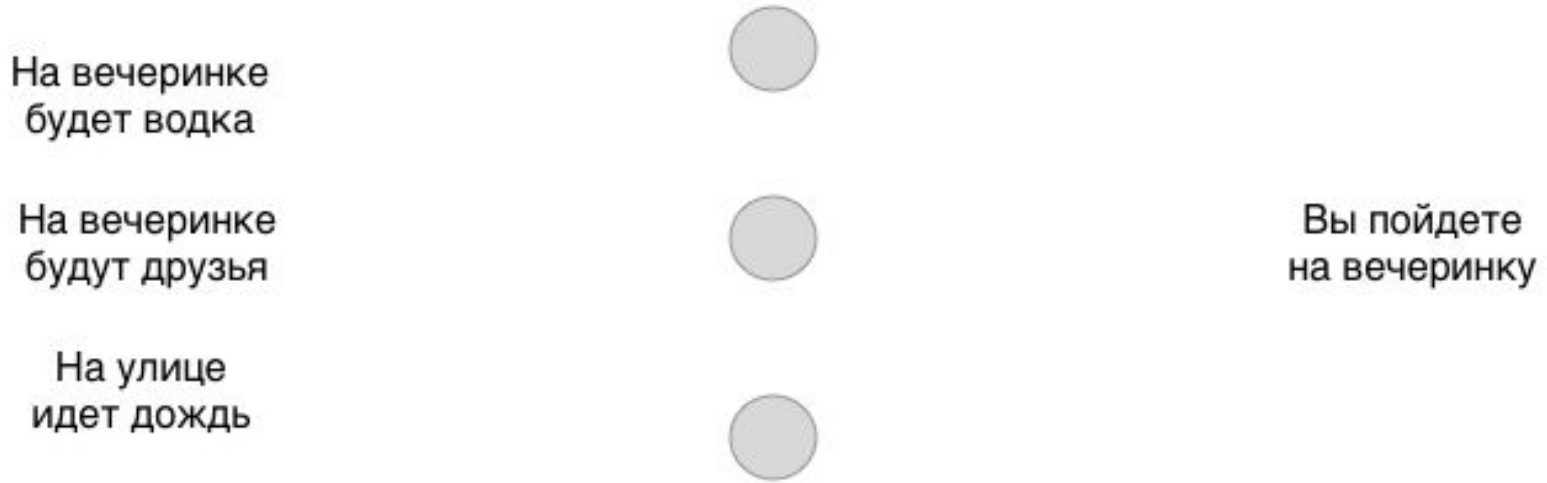
На улице идет дождь

Вы пойдете на вечеринку

Источник: https://habr.com/post/416211/

На пальцах

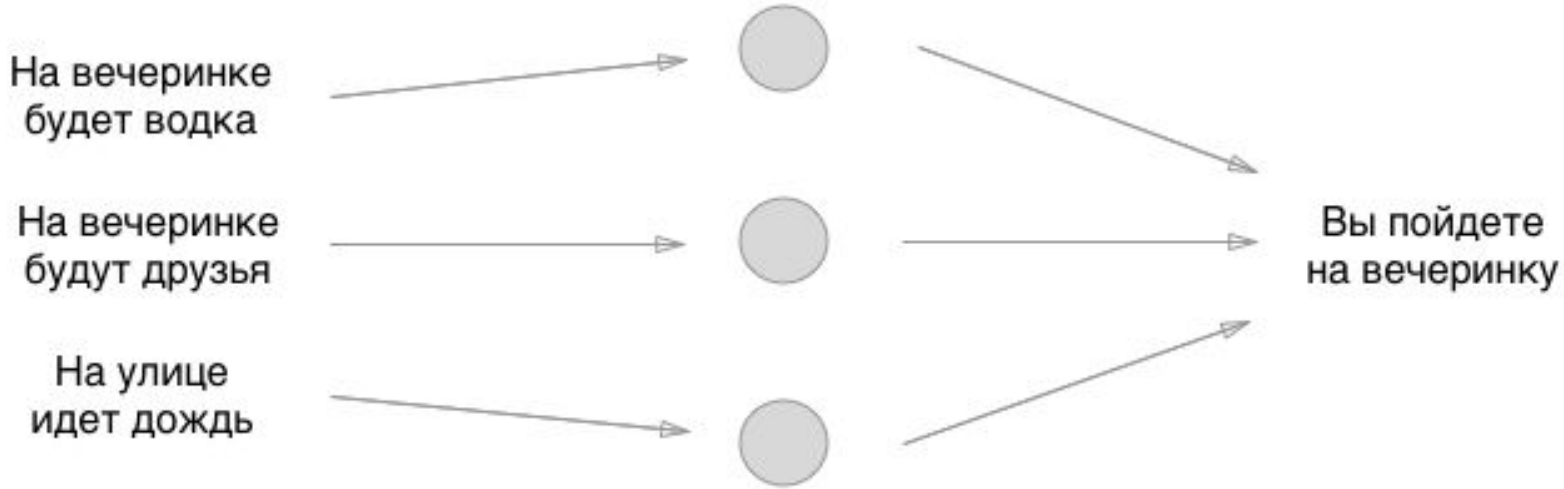
Простая модель с тремя вводными и одним выводом. Эта модель может абсолютно отлично работать для разных людей и выдавать им разные результаты, в зависимости от того, как они обучили нейронную сеть.



Источник: https://habr.com/post/416211/

На пальцах

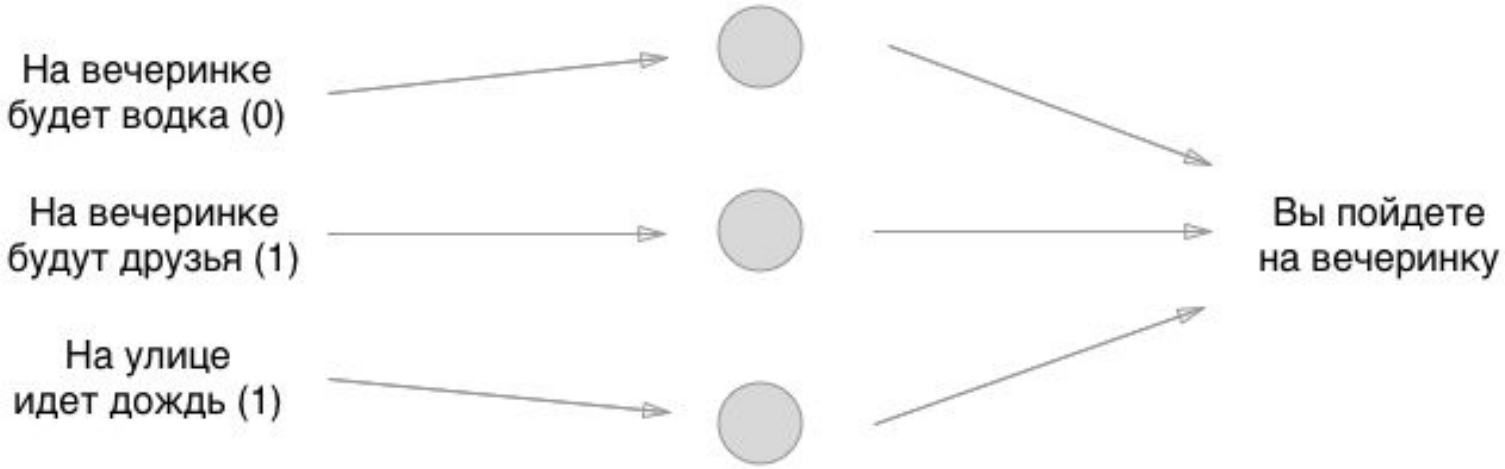
То, что вы видите между вводом и выводом — это нейроны. Пока что они ни с чем не связаны, но это и отражает их главную особенность, о которой все забывают сказать: они — это полностью абстрактны.



Источник: https://habr.com/post/416211/

На пальцах

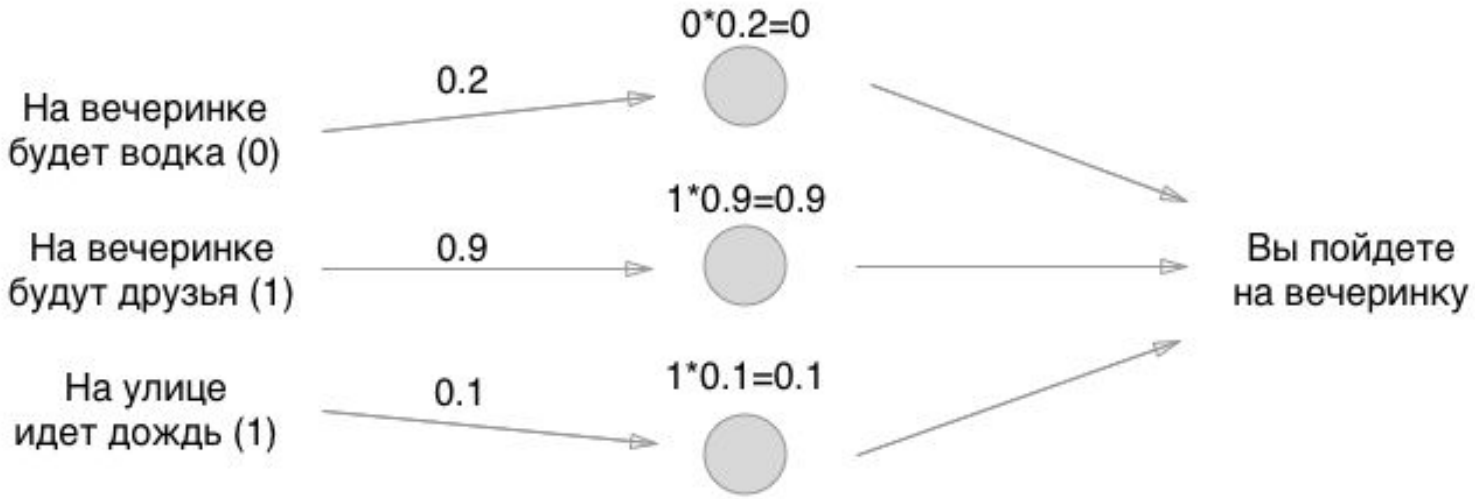
У каждого ввода слева есть значение: 0 или 1, да или нет. Давайте добавим эти значения вводу, предположим, что на вечеринке не будет водки, будут друзья да будет идти дождь:



Источник: https://habr.com/post/416211/

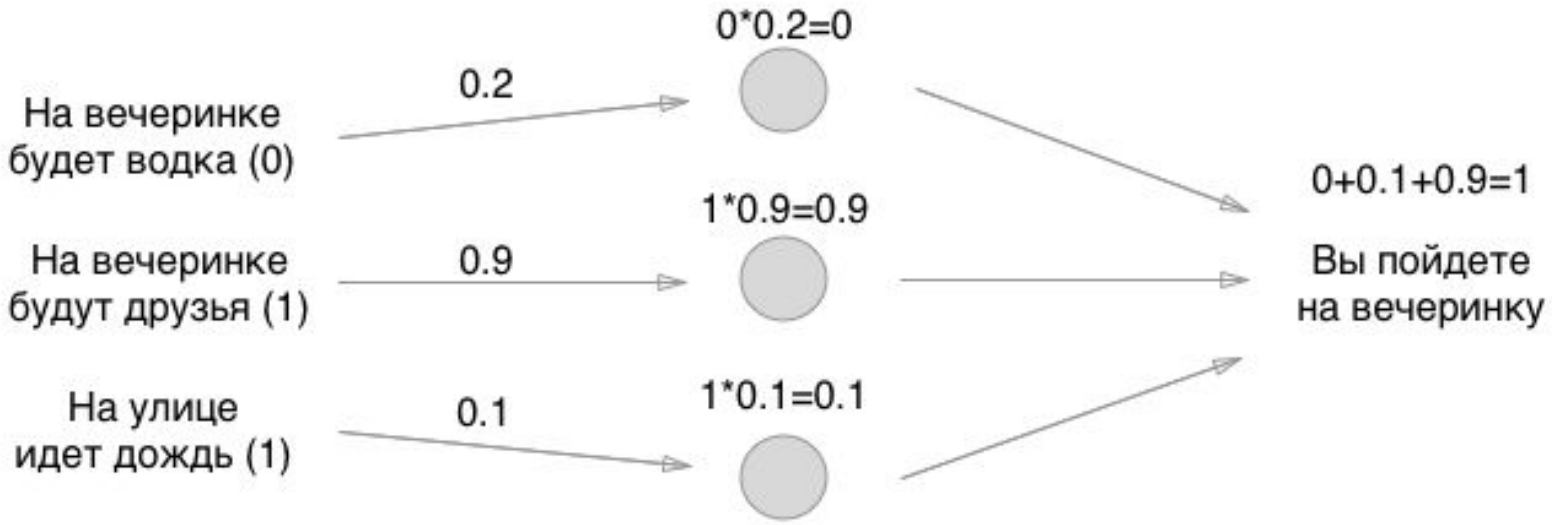
На пальцах

Цифры, которые мы расставили — это веса связей. Помните, что нейроны — это абстракция? Так вот, связи — это именно то, из чего и состоит нейронная сеть.



Источник: https://habr.com/post/416211/

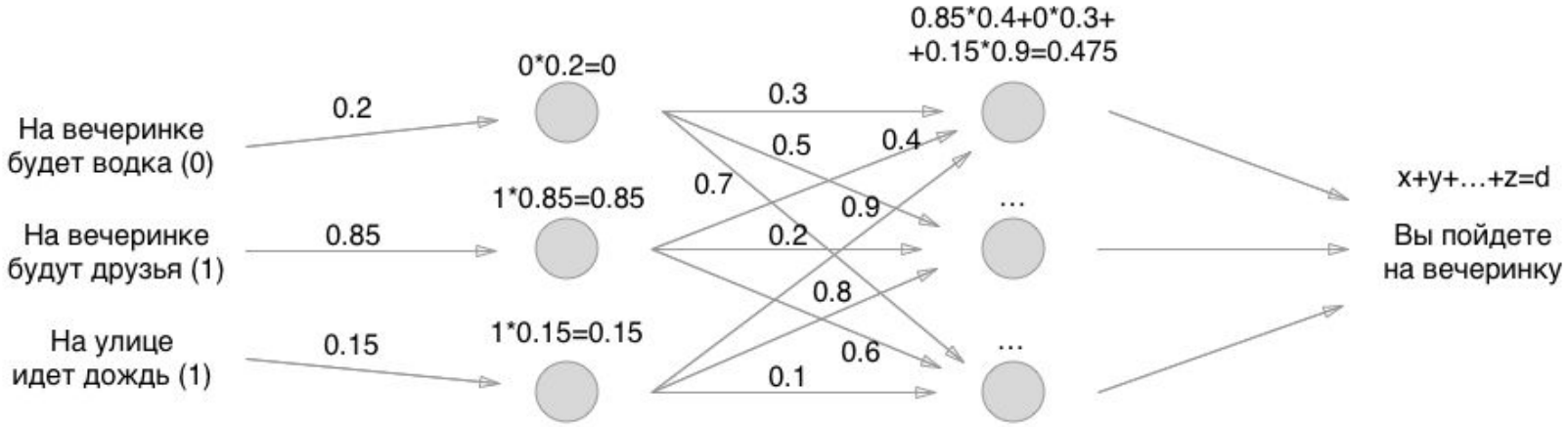
На пальцах



Ну, вот и все! Нейронка создана, а вы можете ее использовать для любых нужд. Если сумма получается больше 0.5 — идти на вечеринку нужно. Если меньше или равно — на вечеринку идти не нужно. Спасибо за внимание!

Источник: https://habr.com/post/416211/

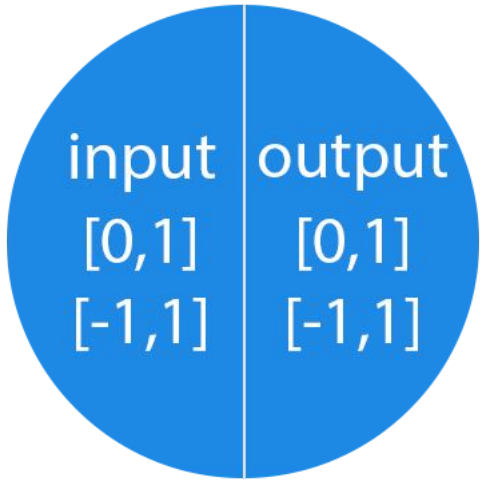
На пальцах

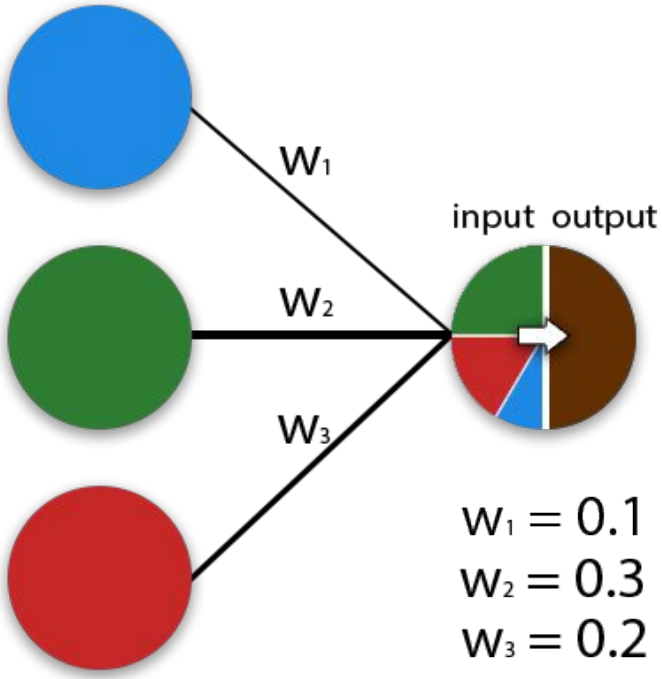


Дальше все просто: вместо одного слоя нейронов мы делаем два и снова все перебираем по точно тем же самым принципам, только уже все нейроны отдают значения другим нейронам. Если сначала у нас было только 3 связи, то теперь 3 + 9 связей с весами. А потом три слоя, четыре, рекурсивные слои, зацикленные на себе и тому подобная дичь:

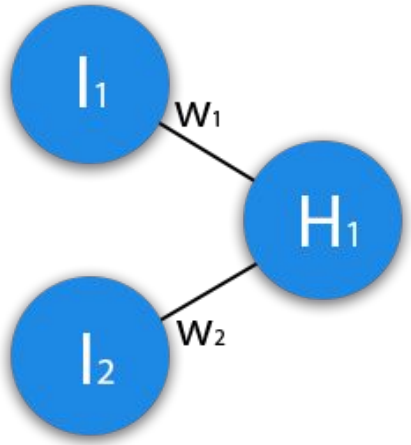
Источник: https://habr.com/post/416211/

Нейрон и вес





$w_1 = 0.1$
$w_2 = 0.3$
$w_3 = 0.2$



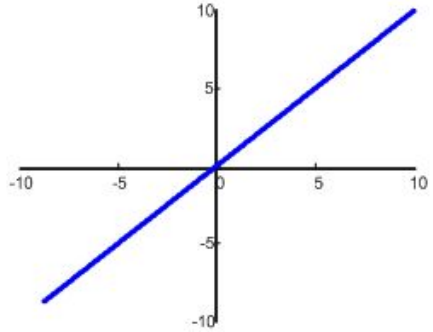
$$1) H1_{input} = (I_1 * w_1) + (I_2 * w_2)$$

$$2) H1_{output} = f_{activation}(H1_{input})$$

Источник: https://habr.com/post/312450/

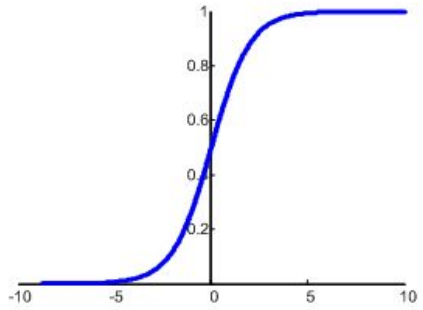
Функция активации (ФА)

$$f(x) = x$$



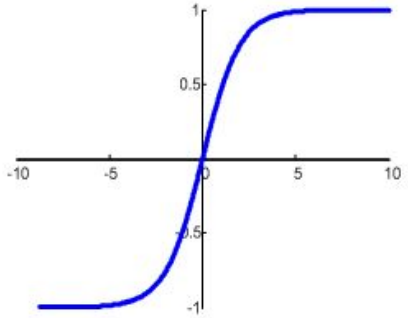
Линейная функция

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



Сигмоид

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



Гиперболический тангенс

Источник: https://habr.com/post/312450/

Эпохи и шаги

```
for (int i=0;i<maxEpoch;i++)
  for (int j=0;j<trainSet;j++)
```

```
for (int j=0;j<trainSet;j++)
  for (int i=0;i<maxEpoch;i++)
```

Итерация

Это своеобразный счетчик, который увеличивается каждый раз, когда нейронная сеть проходит один тренировочный сет. Другими словами, это общее количество тренировочных сетов пройденных нейронной сетью.

Эпоха

При инициализации нейронной сети эта величина устанавливается в 0 и имеет потолок, задаваемый вручную. Чем больше эпоха, тем лучше натренирована сеть и соответственно, ее результат. Эпоха увеличивается каждый раз, когда мы проходим весь набор тренировочных сетов, в нашем случае, 4 сетов или 4 итераций.

Источник: https://habr.com/post/416211/

Ошибка

Ошибка — это процентная величина, отражающая расхождение между ожидаемым и полученным ответами. Ошибка формируется каждую эпоху и должна идти на спад. Если этого не происходит, значит, вы что-то делаете не так. Ошибку можно вычислить разными путями, но мы рассмотрим лишь три основных способа: Mean Squared Error (далее MSE), Root MSE и Arctan.

MSE

$$\frac{(i_1-a_1)^2+(i_2-a_2)^2+\ldots+(i_n-a_n)^2}{n}$$

Root MSE

$$\sqrt{\frac{(i_1-a_1)^2+(i_2-a_2)^2+\ldots+(i_n-a_n)^2}{n}}$$

Arctan

$$\frac{\arctan^2(i_1-a_1)+\ldots+\arctan^2(i_n-a_n)}{n}$$

Источник: https://habr.com/post/416211/

Формулы

Расчет входа нейрона

$$\text{net} = \sum_{i=1}^{n} x_i \cdot w_i$$

Правило Видроу-Хоффа для пересчета весов

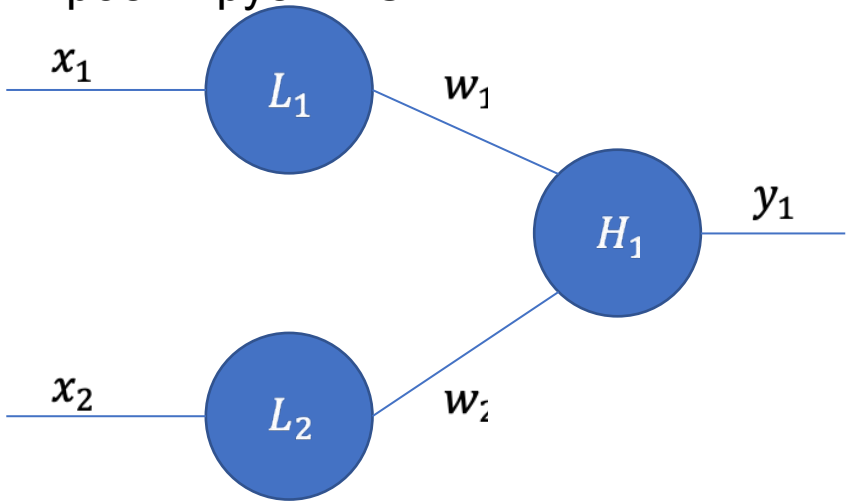
$$w_i^{(l+1)} = w_i^{(l)} + \Delta w_i^{(l)}$$

$$\Delta w_i^{(l)} = \eta \cdot \delta^{(l)} \cdot \frac{df(net)}{d\,net} \cdot w_i^{(l)}$$

i – количество нейронов, l – количество эпох, x – входы, w – веса, η – коэффициент обучения, δ – ошибка, net – сетевой вход

Пример. Инициализация

1. Выбираем задачу (Например логическое ИЛИ)
2. Проектируем НС



3. Выбираем случайные веса $w_i \in [0,1] \mid [-1,1]$
$w_1 = w_2 = 0$

4. Выбираем коэффициент обучения $\eta \in [0, 1]$
$\eta = 0.3$

5. Строим таблицу истинности

0	0	0
0	1	1
1	0	1
1	1	1

6. Выбираем ФА (например пороговую)

Сетевой выход:

$$out = y(net) = \begin{cases} 1, net \geq 0.5; \\ 0, net < 0.5 \end{cases}$$

Ошибка: $\delta = |t - y|$

Пример. Обучение

Сетевой выход:

$$out = y(net) = \begin{cases} 1, net \geq 0.5; \\ 0, net < 0.5 \end{cases}$$

Ошибка: δ = **|t – y|**

ЭПОХА I

Итерация 1

$x_1 = 0, x_2 = 0$, w = [0, 0]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0$

$y = 0$, t = 0, δ = 0 Ошибки НЕТ

Итерация 2

$x_1 = 0, x_2 = 1$, w = [0, 0]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0$

$y = 0$, t = 1, δ = 1 ОШИБКА

Пересчитываем веса

$w_1 = 0 + 0.3 * 0 = 0$

$w_2 = 0 + 0.3 * 1 = 0.3$

Итерация 3

$x_1 = 1, x_2 = 0$, w = [0, 0.3]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0$

$y = 0$, t = 1, δ = 1 ОШИБКА

Пересчитываем веса

$w_1 = 0 + 0.3 * 1 = 0.3$

$w_2 = 0.3 + 0.3 * 0 = 0.3$

Итерация 4

$x_1 = 1, x_2 = 1$, w = [0.3, 0.3]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0.6$

$y = 1$, t = 1, δ = 0 Ошибки НЕТ

0	0	0
0	1	1
1	0	1
1	1	1

		error
0	0	No
0	1	Yes
0	1	Yes
1	1	No

Пример. Обучение

Сетевой выход:

$$out = y(net) = \begin{cases} 1, net \geq 0.5; \\ 0, net < 0.5 \end{cases}$$

Ошибка: δ = **|t – y|**

ЭПОХА II

Итерация 1

$x_1 = 0, x_2 = 0$, w = [0.3, 0.3]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0$

$y = 0$, t = 0, δ = 0 Ошибки НЕТ

Итерация 2

$x_1 = 0, x_2 = 1$, w = [0.3, 0.3]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0.3$

$y = 0$, t = 1, δ = 1 ОШИБКА

Пересчитываем веса

$w_1 = 0.3 + 0.3 * 0 = 0.3$

$w_2 = 0.3 + 0.3 * 1 = 0.6$

Итерация 3

$x_1 = 1, x_2 = 0$, w = [0.3, 0.6]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0.3$

$y = 0$, t = 1, δ = 1 ОШИБКА

Пересчитываем веса

$w_1 = 0.3 + 0.3 * 1 = 0.6$

$w_2 = 0.6 + 0.3 * 0 = 0.6$

Итерация 4

$x_1 = 1, x_2 = 1$, w = [0.6, 0.6]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 1.2$

$y = 1$, t = 1, δ = 0 Ошибки НЕТ

0	0	0
0	1	1
1	0	1
1	1	1

		error
0	0	No
0	1	Yes
0	1	Yes
1	1	No

Пример. Обучение

Сетевой выход:

$$out = y(net) = \begin{cases} 1, net \geq 0.5; \\ 0, net < 0.5 \end{cases}$$

Ошибка: δ = **|t – y|**

ЭПОХА III

Итерация 1

$x_1 = 0, x_2 = 0$, w = [0.6, 0.6]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0$

$y = 0$, t = 0, δ = 0 Ошибки НЕТ

Итерация 2

$x_1 = 0, x_2 = 1$, w = [0.6, 0.6]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0.6$

$y = 1$, t = 1, δ = 0 Ошибки НЕТ

Итерация 3

$x_1 = 1, x_2 = 0$, w = [0.6, 0.6]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 0.6$

$y = 1$, t = 1, δ = 0 Ошибки НЕТ

Итерация 4

$x_1 = 1, x_2 = 1$, w = [0.6, 0.6]

$net = x_1 \cdot w_1 + x_2 \cdot w_2 = 1.2$

$y = 1$, t = 1, δ = 0 Ошибки НЕТ

Обучение завершено!

0	0	0
0	1	1
1	0	1
1	1	1

		error
0	0	No
0	0	No
0	0	No
1	1	No

Пример кода на Python

```
import numpy as np

def generate_vectors():
    from itertools import product as p
    return(p(range(2), repeat=2))

def f(x):
    return(x[0] or x[1])


x_vectors = generate_vectors()
t_vector = np.array([f(x) for x in x_vectors])

x_vectors = generate_vectors()

nu = 0.3
weight = np.array([0., 0.])
epoch = 1
error = 1
```

Пример кода на Python

```
while error != 0:
    x_vectors = generate_vectors()
    print(f'\n********** epoch {epoch} **********')
    print(f'weight = {weight}')
    print(f'_____________________________________________')

    error = 0
    for i, x_vec in enumerate(x_vectors):
        net = sum(x_vec[j]*weight[j] for j in range(2))

        if net >= 0.5: y = 1
        else: y = 0

        delta = t_vector[i] - y
        err = t_vector[i] - net
        print(f'| net = {net:.3f} | y = {y}| t = {t_vector[i]}  | err =
{abs(err):.3f} |')
        if delta != 0:
            error += 1
            weight = [weight[j] + nu * abs(err) * x_vec[j] for j in range(2)]
    epoch += 1

    print(f'---------------------------------------------')
    print(f'{error} errors\n')
```

Пример кода на Python

********** epoch 1 **********
weight = [0. 0.]

---

net = 0.000	y = 0	t = 0	err = 0.000
net = 0.000	y = 0	t = 1	err = 1.000
net = 0.000	y = 0	t = 1	err = 1.000
net = 0.600	y = 1	t = 1	err = 0.400

---

2 errors

********** epoch 2 **********
weight = [0.3, 0.3]

---

net = 0.000	y = 0	t = 0	err = 0.000
net = 0.300	y = 0	t = 1	err = 0.700
net = 0.300	y = 0	t = 1	err = 0.700
net = 1.020	y = 1	t = 1	err = 0.020

---

2 errors

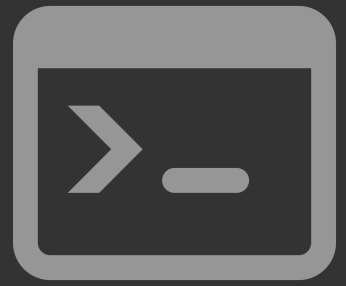
********** epoch 3 **********
weight = [0.51, 0.51]

---

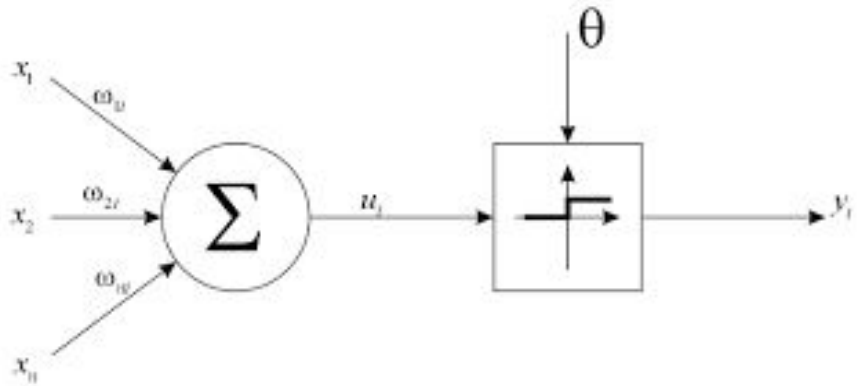
net = 0.000	y = 0	t = 0	err = 0.000
net = 0.510	y = 1	t = 1	err = 0.490
net = 0.510	y = 1	t = 1	err = 0.490
net = 1.020	y = 1	t = 1	err = 0.020

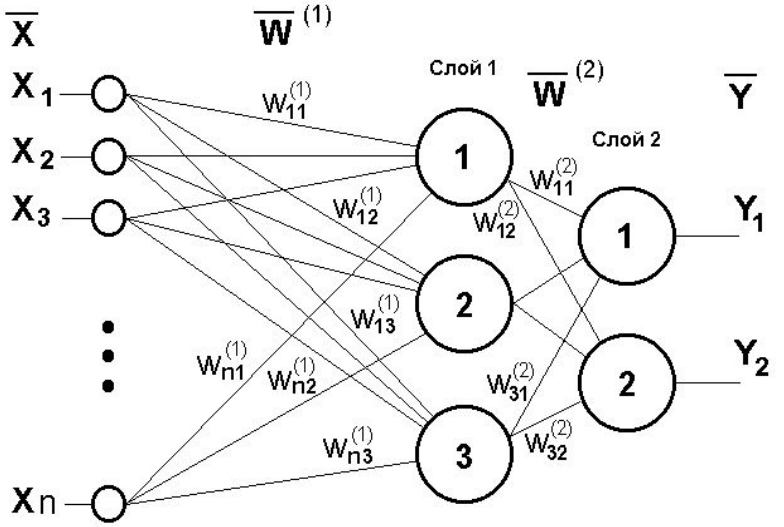
---

0 errors



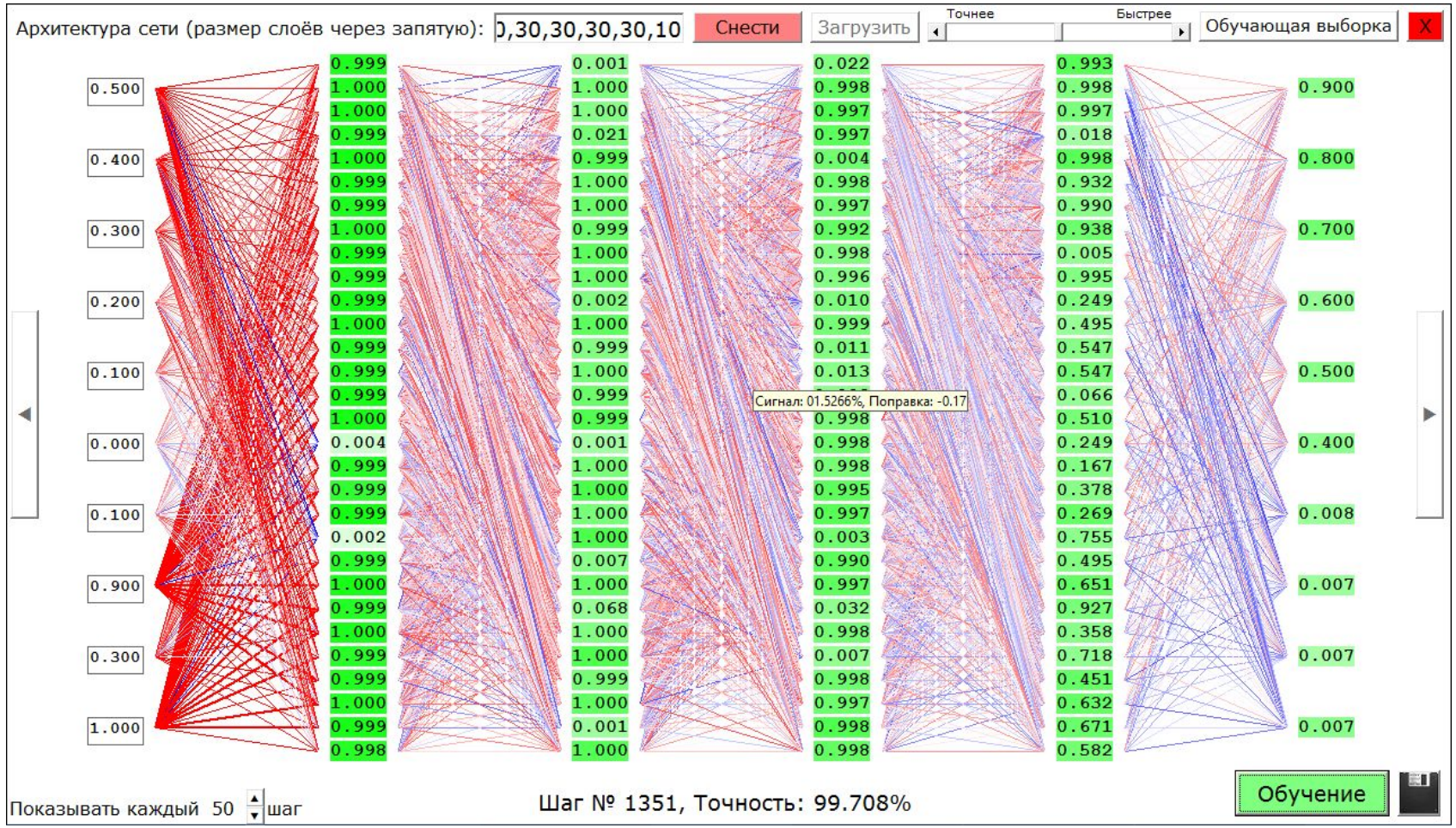
Персептрон и n-слойные сети

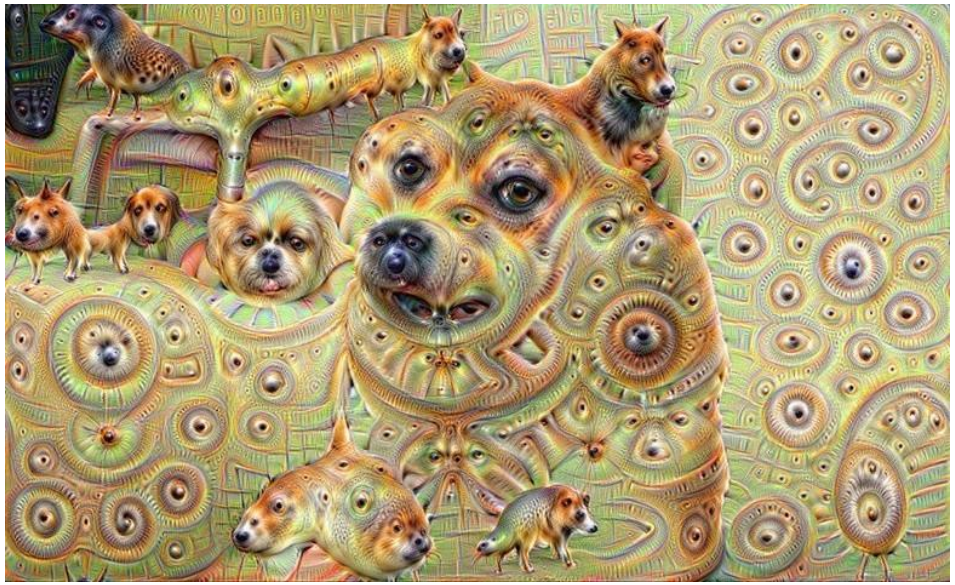




Источник: https://habr.com/post/312450/

Персептрон и n-слойные сети



Источник: https://habr.com/post/312450/

Спасибо за внимание!