



# Искусственные нейронные сети

Лекция 2

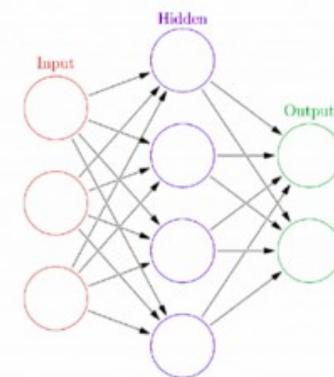
Основные понятия

# Искусственные нейронные сети (ИНС)

- Простейшая ИНС состоящая из входного слоя (сенсоры), слоя ассоциативных элементов и выходного слоя реагирующих элементов.

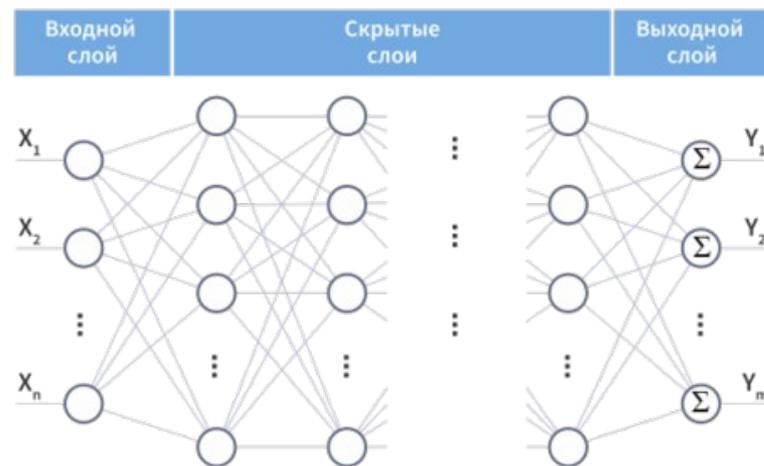


Биологическая нейронная сеть



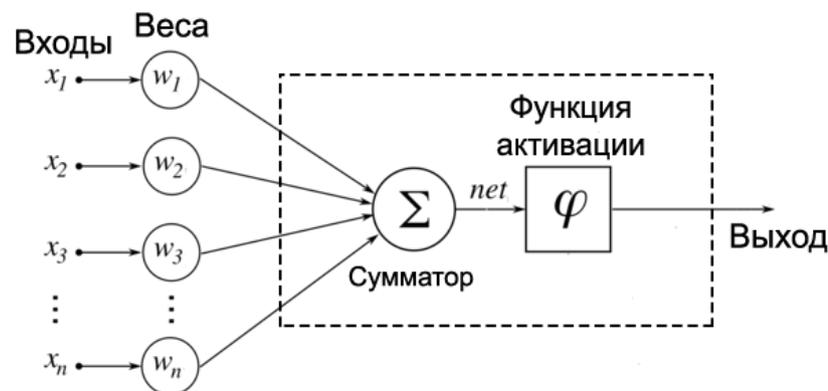
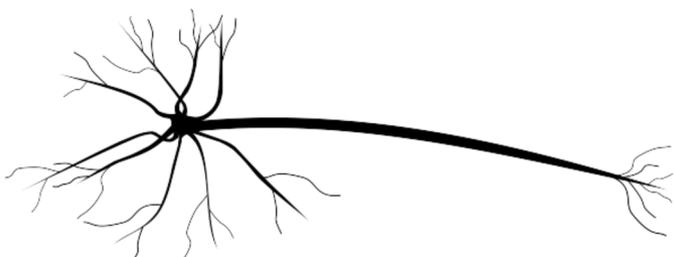
Искусственная нейронная сеть

- Сейчас в основном используют многослойные НС.



# Искусственные нейрон

Каждый узел — это искусственный нейрон, состоящий из сумматора и функции активации.



- Искусственный нейрон — это такая функция  $f: \mathbb{R}^n \rightarrow \mathbb{R}$ , которая преобразует несколько входных параметров в один выходной:

$$y = f(\sum_{i=1..n} w_i * x_i + b)$$

где  $n$  — число входов,  $w_i$  - веса,  $x_i$  - входные значения (сигналы),  $b$  — смещение (bias)

- В матричном виде это можно записать как

$$y = f(W^T * X + b), \quad W^T = \{w_1, \dots, w_n\}, \quad X^T = \{x_1, \dots, x_n\},$$



## Функции активации

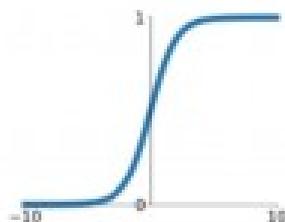
- Для разных типов нейронов используют самые разные функции активации, но одними из самых популярных являются:
  - Функция единичного скачка:  $f(x)=1$ , если  $x>0$ , а иначе 0
  - Сигмоидальная функция.  $f(x)=1/(1+\exp(-ax))$  где параметр  $a$  характеризует степень крутизны функции
  - Гиперболический тангенс.  $f(x)=\tanh(x/a)$ , где параметр  $a$  также определяет степень крутизны графика функции
  - Rectified linear units (ReLU).  $\text{ReLU}(x)=\max(0,x)$ 
    - Leaky ReLU:  $\text{LReLU}(x)=\max(0.1*x,x)$
    - ELU:  $\text{ELU}(x)=x$ , если  $x>0$  и  $a*(\exp(x)-1)$ , если  $x<0$



# Функции активации

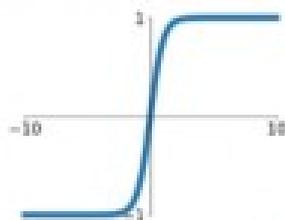
## Sigmoid

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



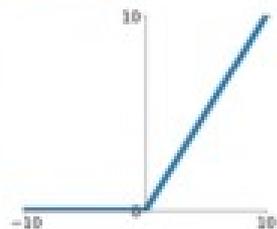
## tanh

$$\tanh(x)$$



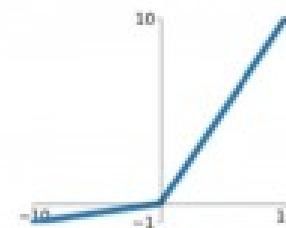
## ReLU

$$\max(0, x)$$



## Leaky ReLU

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

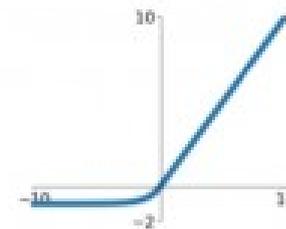


## Maxout

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

## ELU

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





# Метрики

- Метрика — это мера определяющая расстояние в некотором пространстве. Таким образом, это отображение  
 $r: M \rightarrow R$
- Как правило, в качестве пространства  $M$  у нас будет использоваться  $R^D$ , где  $D$  — размерность пространства.
- Наиболее популярные метрики:
  - $L_1 = |x-y|$ , где  $x, y \in R^D$
  - $L_2 = (x-y)^2$



# Многослойные нейронные сети

- Функция потерь для линейного классификатора

$$y = W^T * X, \text{ где } x \in \mathbb{R}^D, W \in \mathbb{R}^{C \times D}$$

- 2-слойная полносвязанная нейронная сеть

$$y = W_2^T * f(W_1^T X), \text{ где } x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$$

(мы опустили сдвигку для простоты)

- 3-слойная полносвязанная нейронная сеть

$$y = W_3^T * f(W_2^T * f(W_1^T X)), \text{ где}$$

$$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H_1 \times D}, W_2 \in \mathbb{R}^{H_2 \times H_1}, W_3 \in \mathbb{R}^{C \times H_2}$$



# Нелинейность функций активации

- Что произойдет, если использовать линейную функцию активации, например, для 3-слойной полносвязанной сети?

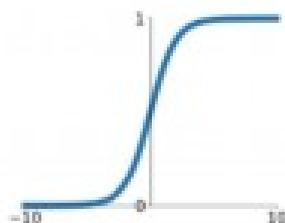
$$\begin{aligned}y &= W_3^T * f(W_2^T * f(W_1^T X)) \\ \implies y &= (W_3^T * W_2^T * W_1^T) * X \\ \implies y &= U^T * X\end{aligned}$$

- Такая сеть будет эквивалентна однослойной сети!
-

# Функции ошибок

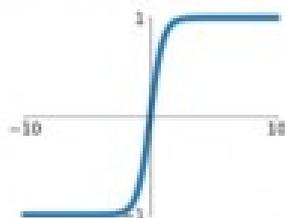
## Sigmoid

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



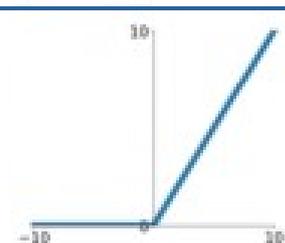
## tanh

$$\tanh(x)$$



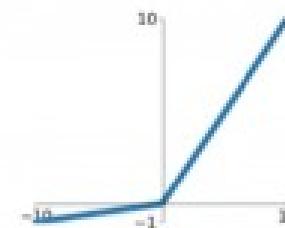
## ReLU

$$\max(0, x)$$



## Leaky ReLU

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

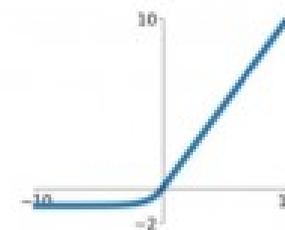


## Maxout

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

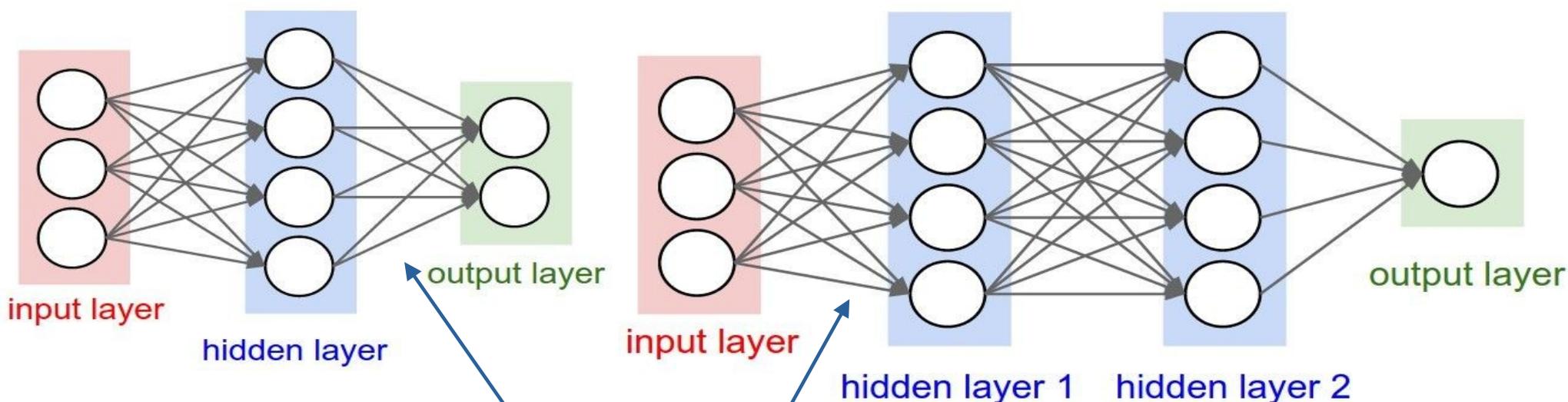
## ELU

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



- ReLU — хороший выбор для начала.

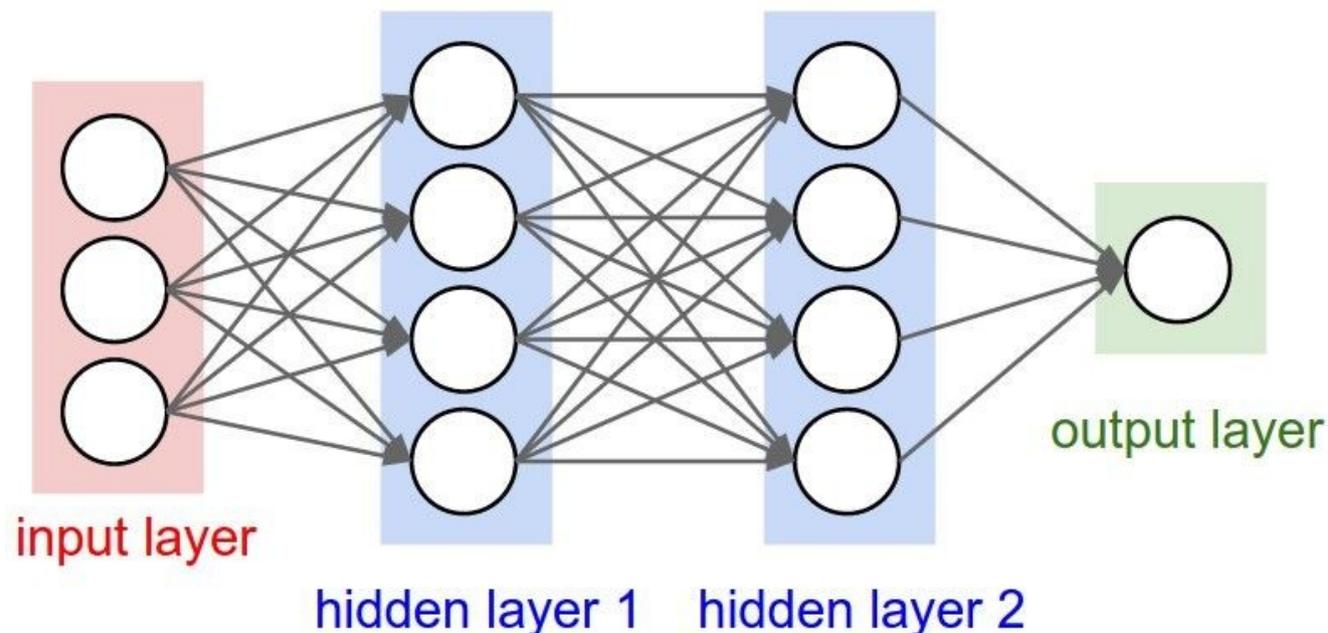
# Архитектура нейронных сетей



- 2-х слойные ИНН, или сети с одним скрытым слоем.

- 3-х слойные ИНН, или сети с двумя скрытыми слоями.
- Это сети с полностью связанными слоями.

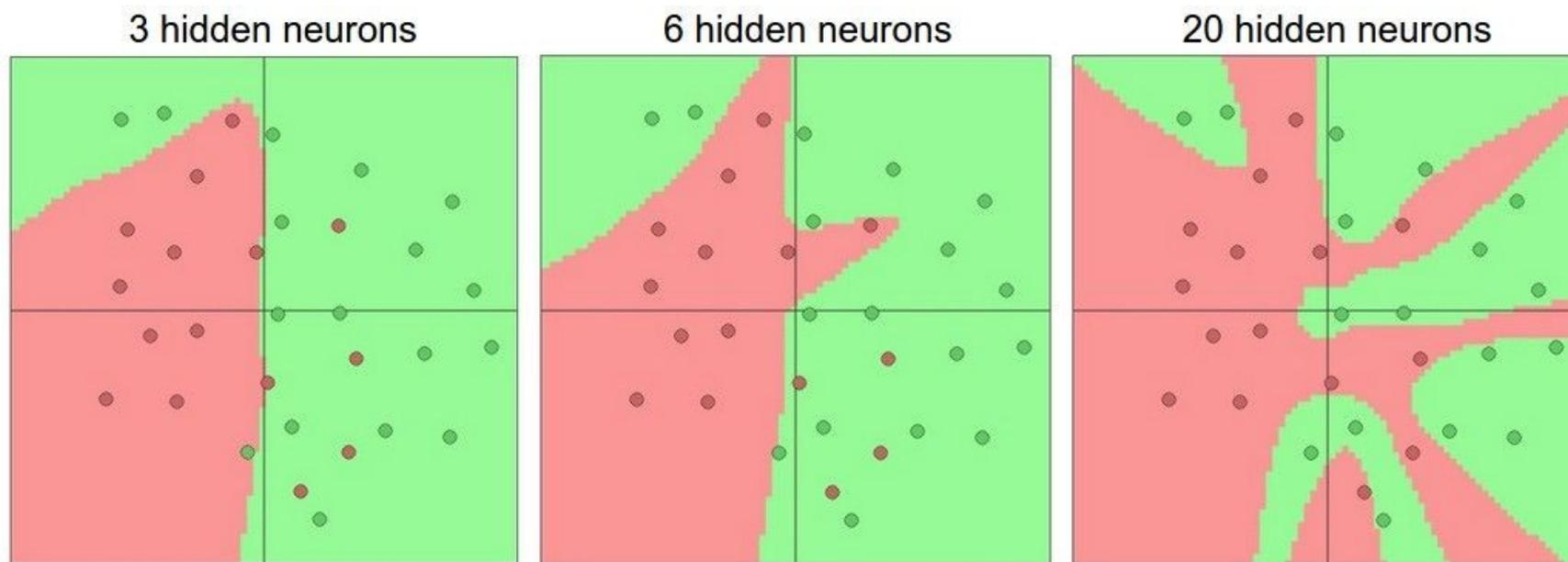
# Пример1



```
# 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)
```

- NumPy:  $\text{dot}(a, b)[i,j,k,m] = \text{sum}(a[i,j,:] * b[k,:,m])$

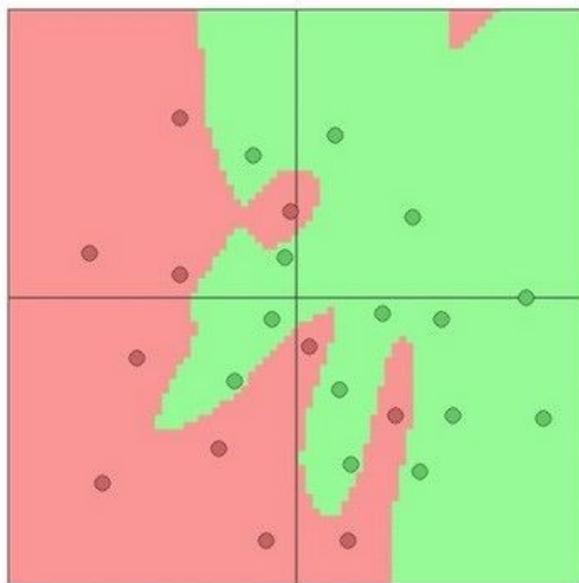
# Выбор числа слоев и нейронов в слое



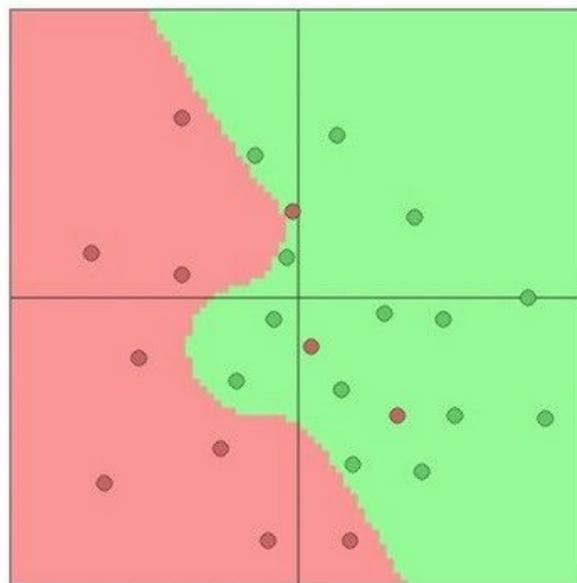
- Чем больше нейронов, тем более тонкие различия мы можем описать.

# Использование регуляризации

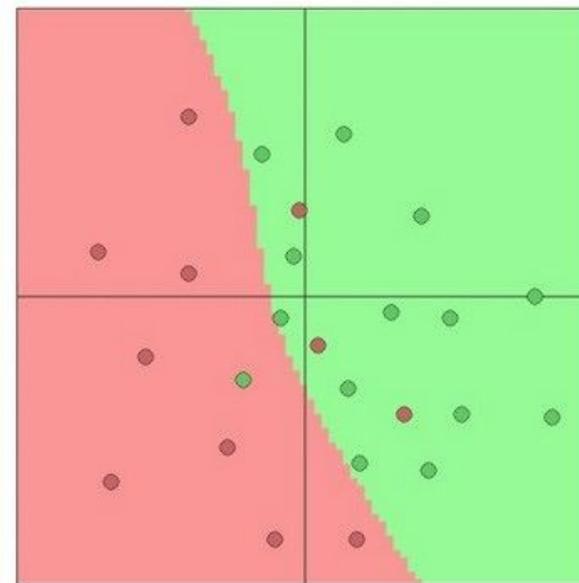
$\lambda = 0.001$



$\lambda = 0.01$



$\lambda = 0.1$



- Функция потерь с регуляризацией

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

- Увеличение вклада члена  $R$  делает процесс обучения более устойчивым к случайным выбросам в данных.



# Демонстрация



# Предыдущая лекция: Введение

([https://theory.sinp.msu.ru/doku.php/ml\\_lectures](https://theory.sinp.msu.ru/doku.php/ml_lectures))

Следующая лекция:

Обратное распространение ошибок