

# Нейросетевые методы. Свёрточные нейронные сети

—

Булат Ибрагимов

# План лекции

- Нейронные сети
- Обучение нейронных сетей
- Свёрточные нейронные сети
- Глубокие нейросети
- Пример: Autoencoder

# 2006 год: появление проекта Asirra

# starm




Please click on all the images that show cats:



Computer Vision: 60%

# 2014 год: соревнование Dogs VS Cats



## Dogs vs. Cats

Create an algorithm to distinguish dogs from cats  
215 teams · 5 years ago





[Overview](#) [Data](#) [Kernels](#) [Discussion](#) [Leaderboard](#) [Rules](#)

[Public Leaderboard](#) [Private Leaderboard](#)

The private leaderboard is calculated with approximately 70% of the test data.  
This competition has completed. This leaderboard reflects the final standings.

[Refresh](#)

■ Gold ■ Silver ■ Bronze

#	Δ <sub>pub</sub>	Team Name	Kernel	Team Members	Score 🏆	Entries	Last
1	—	Pierre Sermanet			<u>0.98914</u>	5	5y
2	▲ 4	orchid			0.98308	17	5y
3	—	Owen			0.98171	15	5y
4	—	Paul Covington			0.98171	3	5y

# Применение нейронных сетей

Яндекс Переводчик

ТЕКСТ САЙТ КАРТИНКА



АНГЛИЙСКИЙ

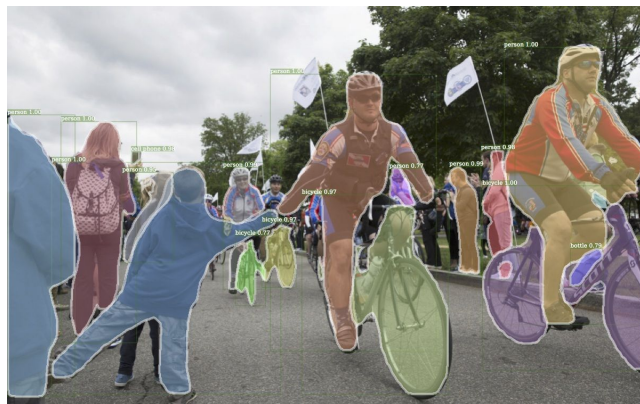


РУССКИЙ



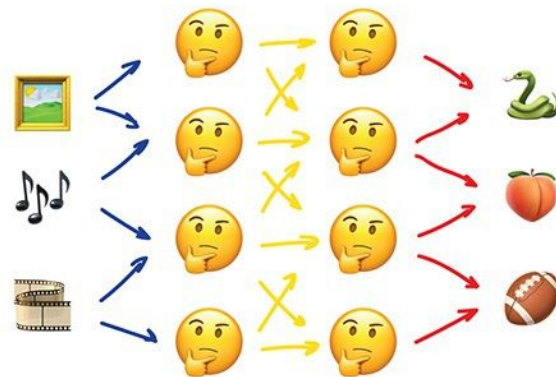
Введите текст или адрес сайта

0 / 10000



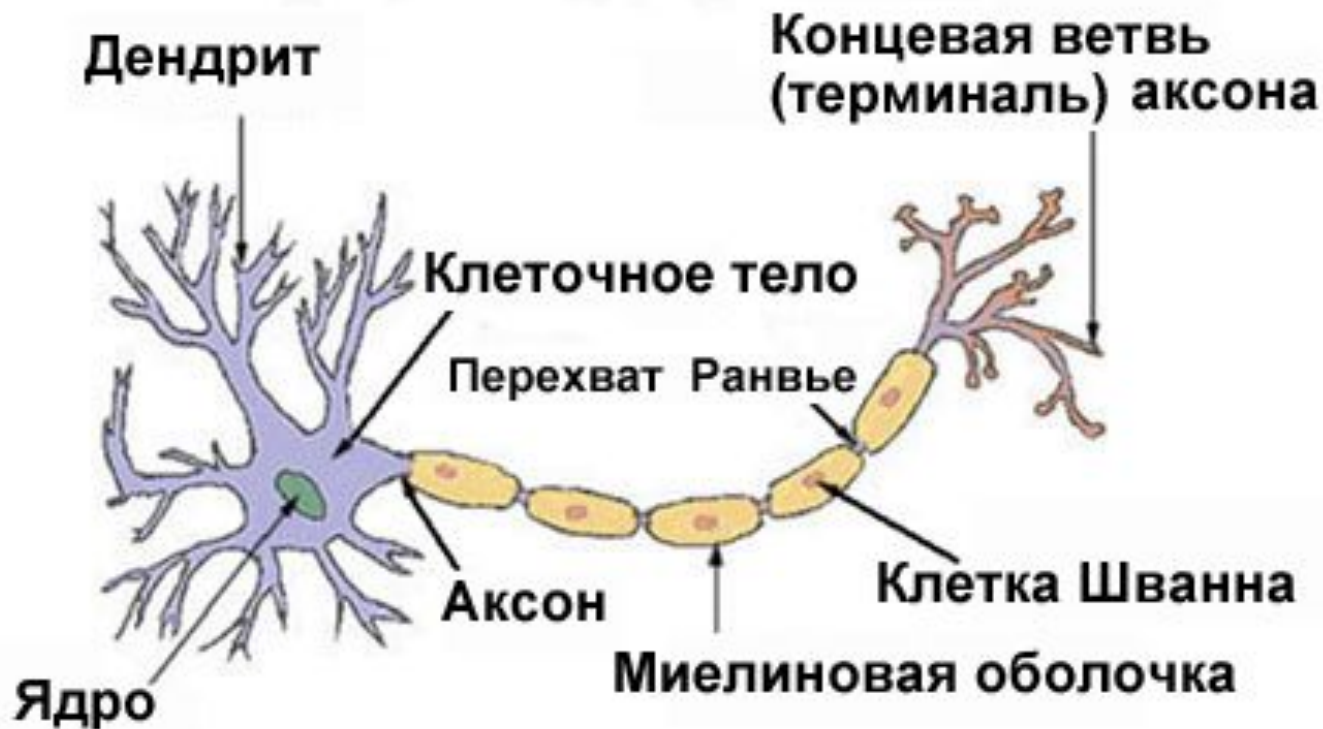
# Нейронные сети

- Самая быстроразвивающаяся и многообещающая ветвь машинного обучения
- Пригодна для обработки большого количества данных
- Берёт на себя процесс feature engineering
- Способна к “переносу знания”

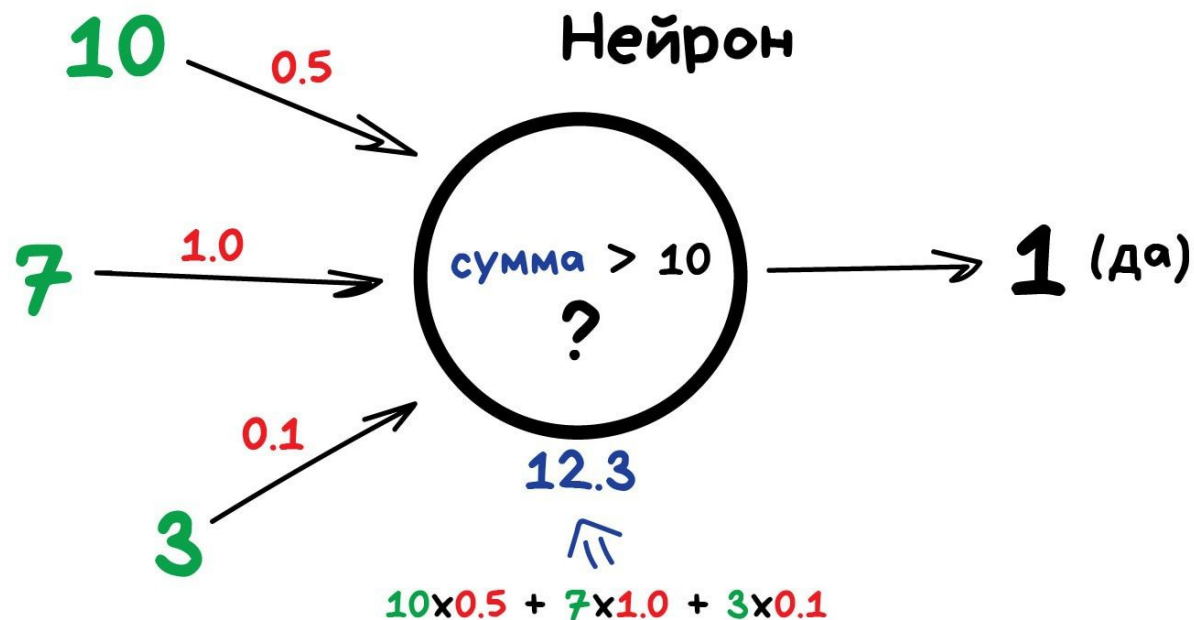


Neural Networks

# Устройство нейрона человека

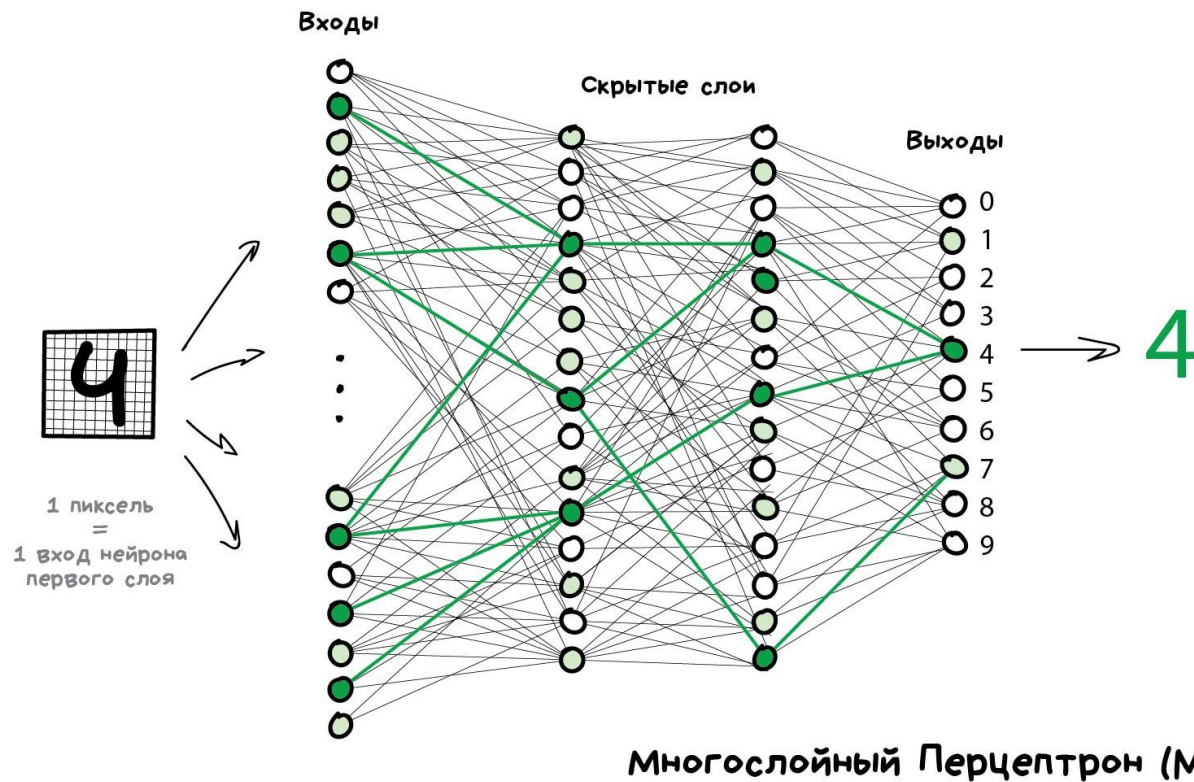


# Нейрон в нейронной сети





# Многослойный перцептрон



# Напоминание: логистическая регрессия

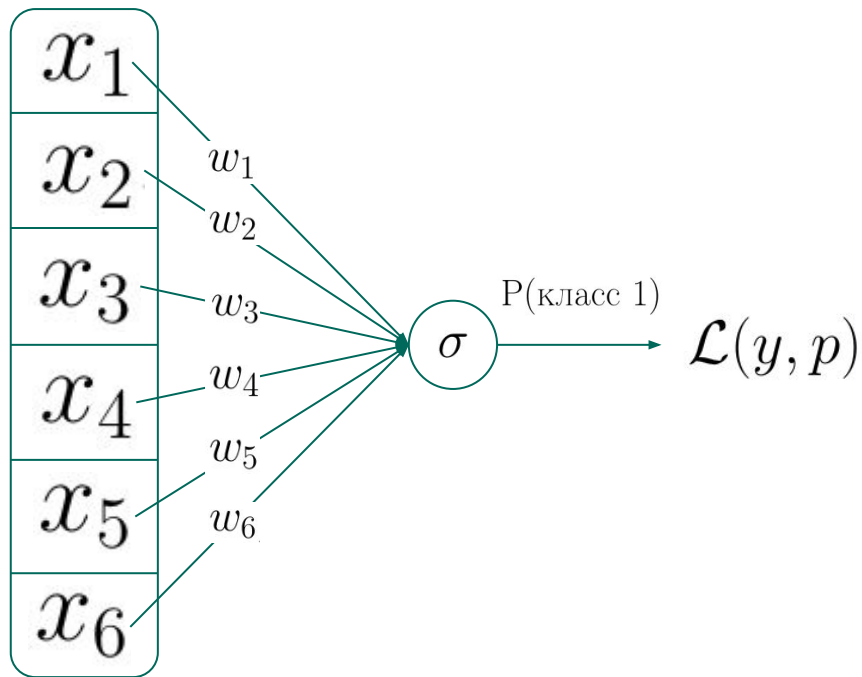
- Вход: вектор признаков  $x = (x_1, \dots, x_n) \in \mathbb{R}^n$
- Параметры: вектор весов  $w = (w_1, \dots, w_n) \in \mathbb{R}^n$
- Предсказывает вероятность класса 1 против класса -1 по формуле

$$P(\text{класс } 1) = \frac{1}{1 + e^{-\langle w, x \rangle}} = \sigma(\langle w, x \rangle)$$

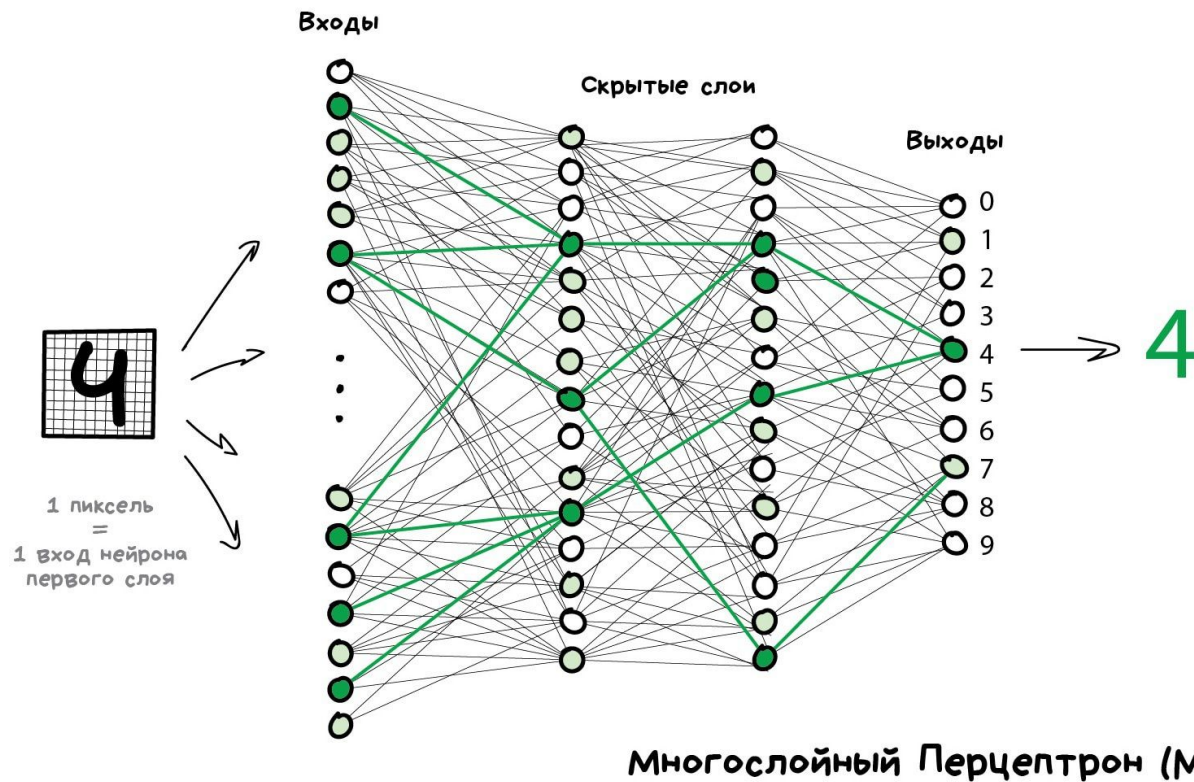
- Вектор весов  $w$  настраивается таким образом, чтобы минимизировать logloss:

$$L(x, y_{true}) = \sum_{i=1}^n \ln(1 + e^{-y_{true} \langle w, x \rangle}) \rightarrow \min_w$$

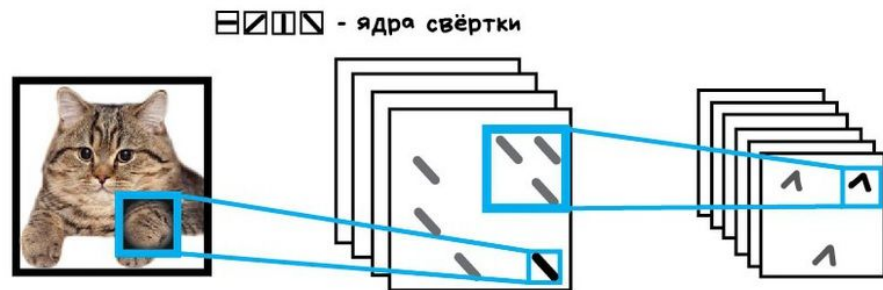
# Логистическая регрессия как нейронная сеть



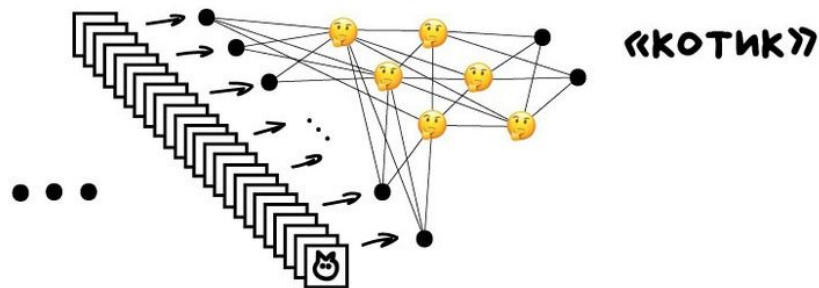
# Обучение нейронных сетей



# Свёрточные нейронные сети



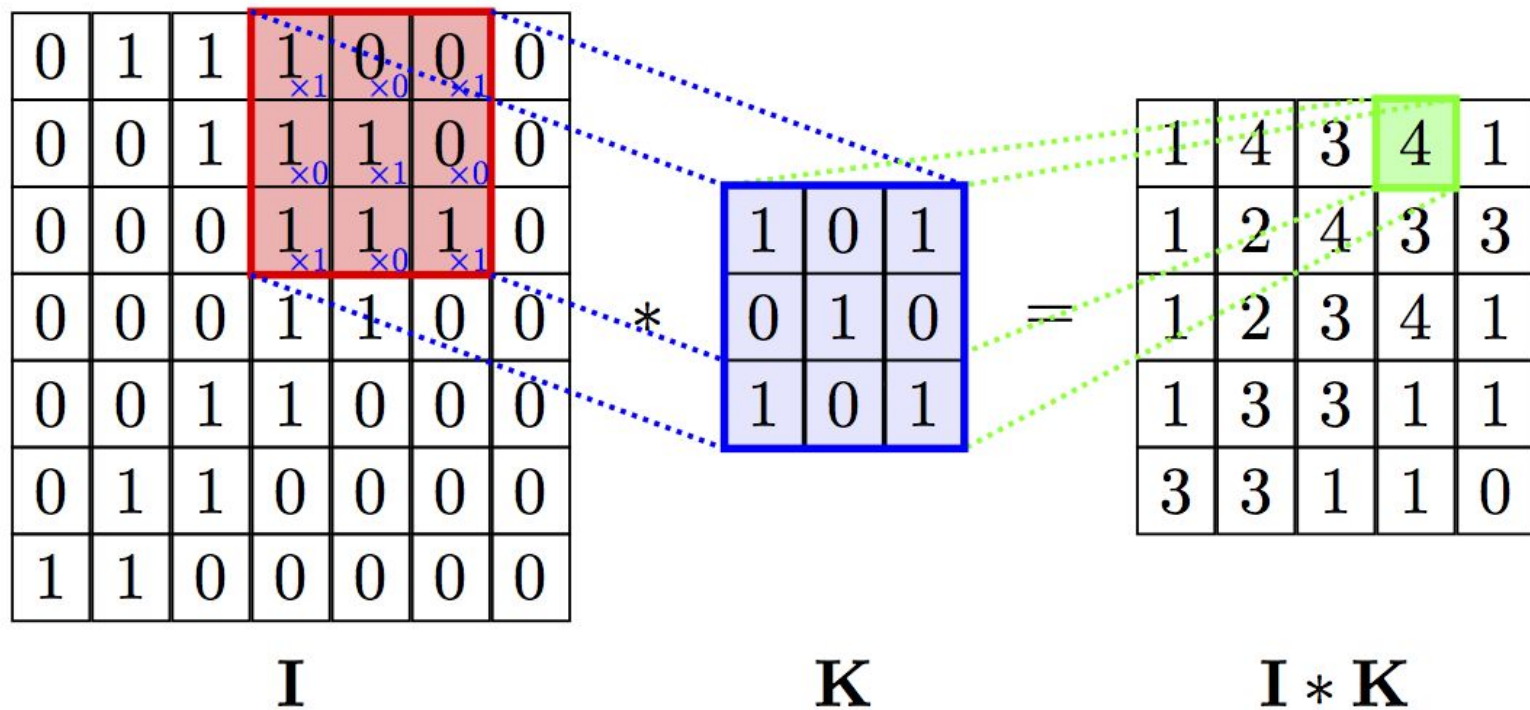
Сеть сама учится искать важные признаки,  
собирая их из простых палочек



Перцептрон уже находит  
важные конкретно  
для котика признаки

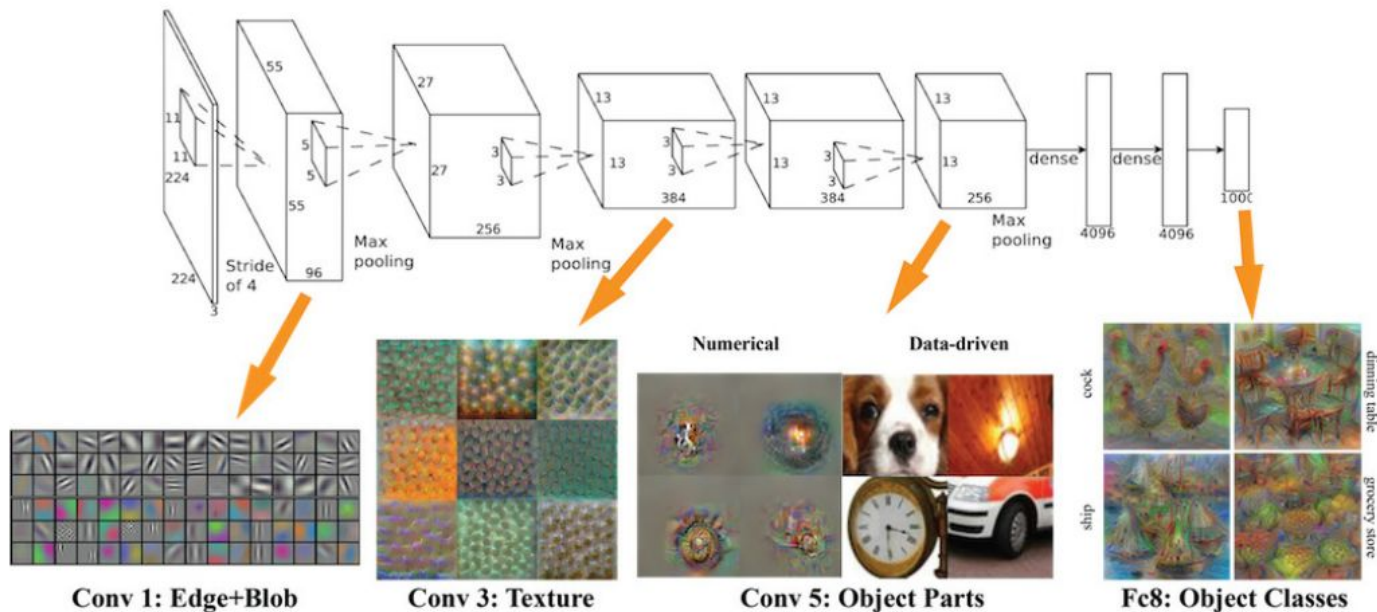
**Свёрточная Нейросеть (CNN)**

# Свёрточные нейронные сети

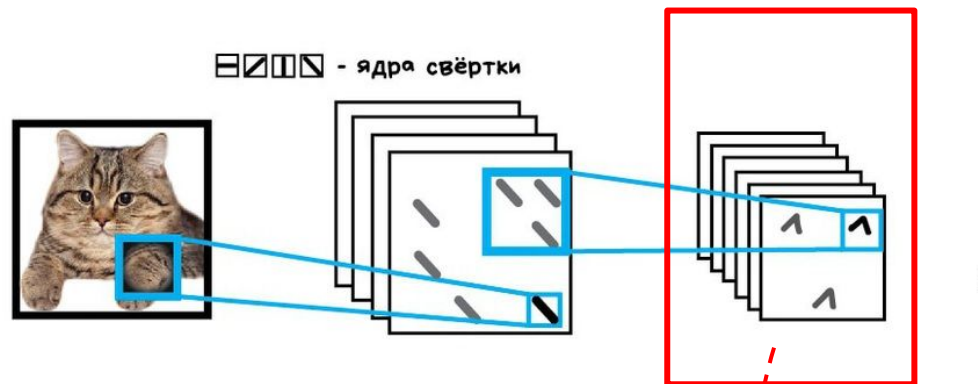


# Пример: топ-активации некоторых нейронов

Чем выше слой, тем более сложные закономерности распознаёт нейрон



# Transfer learning



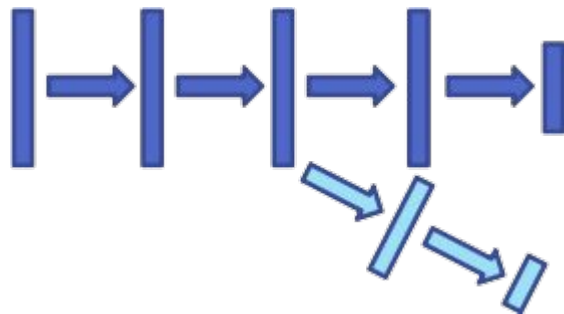
выученные слои можно использовать  
в других задачах



**Свёрточная Нейросеть (CNN)**



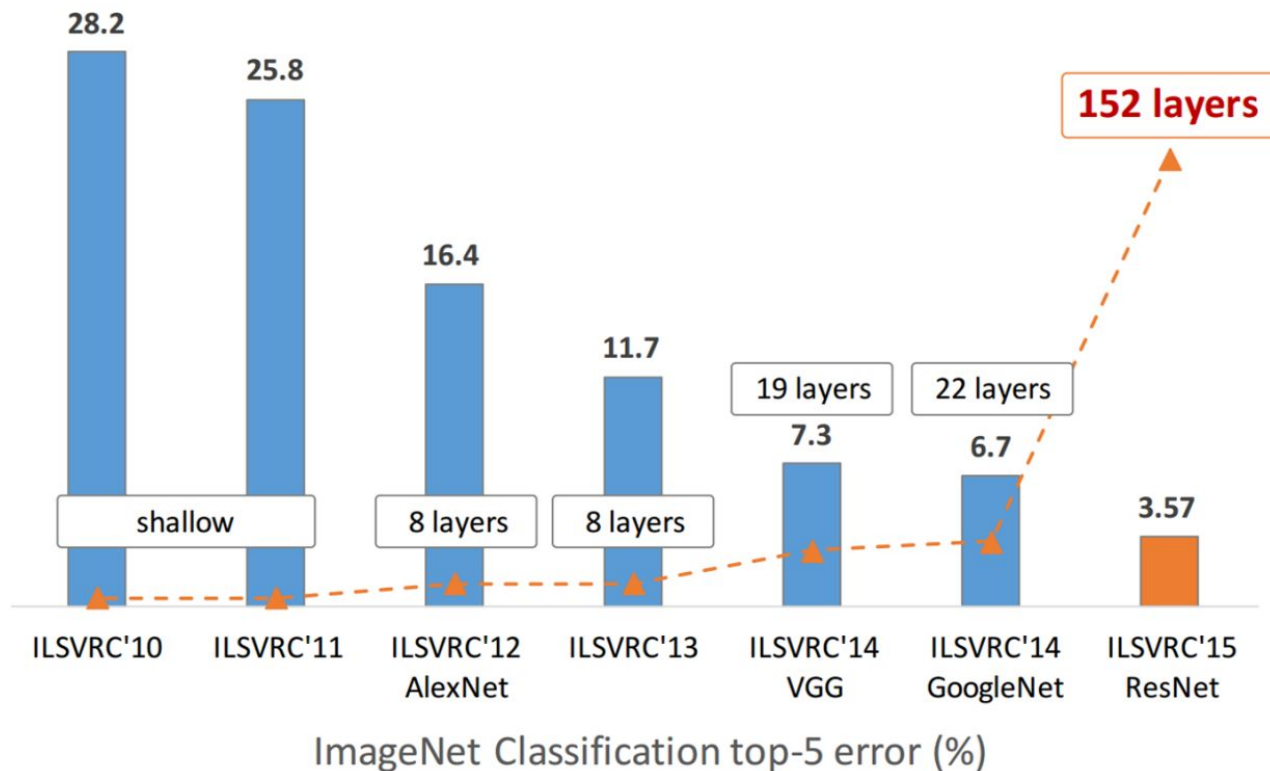
# Transfer learning



# Набор данных ImageNet



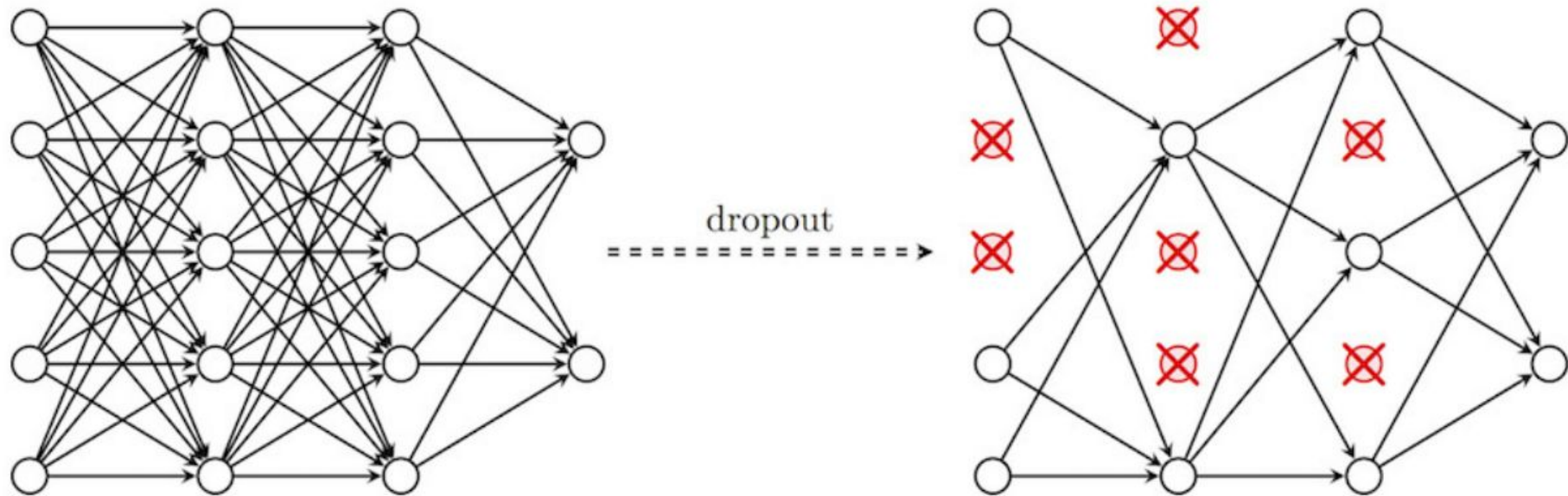
# Набор данных ImageNet



# Глубокие сети: борьба с переобучением

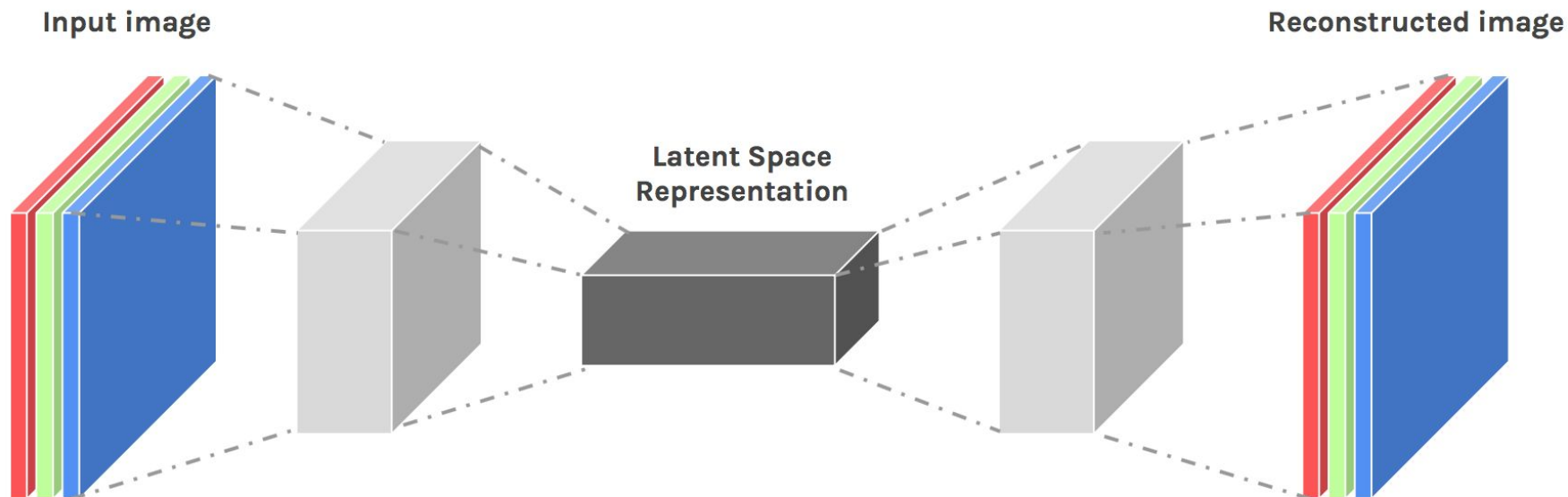
- Ввести в функцию потерь регуляризационное слагаемое на веса
- Dropout
- Batch Normalization

# Dropout





# Пример: Autoencoder



# Пример: Autoencoder

- Усредним представления всех улыбающихся людей:  $x_{\text{smile}}$

# Пример: Autoencoder

- Усредним представления всех улыбающихся людей:  $x_{\text{smile}}$
- Сделаем то же самое с грустными людьми:  $x_{\text{grumpy}}$



# Пример: Autoencoder

- Усредним представления всех улыбающихся людей:  $x_{\text{smile}}$
- Сделаем то же самое с грустными людьми:  $x_{\text{grumpy}}$
- Возьмём grumpy-фотографию и вычислим скрытое представление:  $x_{\text{Trump}}$



# Пример: Autoencoder

- Усредним представления всех улыбающихся людей:  $x_{\text{smile}}$
- Сделаем то же самое с грустными людьми:  $x_{\text{grumpy}}$
- Возьмём grumpy-фотографию и вычислим скрытое представление:  $x_{\text{Trump}}$
- Прибавим вектор “улыбки”:  $x_{\text{Trump}} + (x_{\text{smile}} - x_{\text{grumpy}})$



# Пример: Autoencoder

- Усредним представления всех улыбающихся людей:  $x_{\text{smile}}$
- Сделаем то же самое с грустными людьми:  $x_{\text{grumpy}}$
- Возьмём grumpy-фотографию и вычислим скрытое представление:  $x_{\text{Trump}}$
- Прибавим вектор “улыбки”:  $x_{\text{Trump}} + (x_{\text{smile}} - x_{\text{grumpy}})$



# Резюме: нейронные сети

- Множество применений, многие из которых ждут своего часа
- Разнообразие архитектур
- Близки к искусственному интеллекту: позволяют осуществлять накопление знаний
- Требуют большого количества данных
- Плохо интерпретируемы