

Лаборатория компьютерной графики и  
мультимедиа  
ВМК МГУ имени М.В. Ломоносова

*Курс «Компьютерное зрение»*

Лекция №4  
«Задача классификации изображений.  
Введение в нейросети»

Антон Конушин

2024 год

# План лекции

---



1. Задача классификации изображений и датасеты
2. Линейная классификация и перспептрон
3. Свёрточные нейронные сети
4. Ключевой этап: модель AlexNet



# 1. Задача классификации изображений и датасеты

# Бинарная классификация

---



- Есть ли на этом изображении пешеход?
- Бинарный ответ  $y \in [0,1]$ , 1 – да, 0 – нет
- Альтернативные формулировки
  - Оценка вероятности положительного ответа  $p_{yes} \in [0,1]$
  - Оценки вероятности обоих ответов  $p_{yes} \in [0,1]$ ,  $p_{no} \in [0,1]$ ,  
 $p_{yes} + p_{no} = 1$

# Многоклассовая классификация

---



- Какой объект показан на этом изображении?
- Список  $s$  классов задан изначально
- Эталонный ответ - метка класса  $y \in [1, S]$
- Альтернативный вариант, список оценённых (estimated) вероятностей:
  - $p_i \in [0,1], i = [1, S]$  ,  $\sum p_i = 1$

# Распознавание свойств (атрибутов) объектов

---



Мужчина

Азиат

Бородат

Улыбается

- Атрибуты – «типичные» характеристики объекта
- Для человека - пол, возраст, раса, борода, усы, улыбка, очки и т.д.
- Можем свести к задачам классификации
  - Определение пола – бинарная классификация
  - Определение расы – многоклассовая классификация
  - Определение возраста – либо классификация (возрастные группы), либо регрессия (определение числового параметра)

# Показатели качества («метрики»)



## 1) % правильно классифицированных изображений

Dataset	CNN	Original	BP[23]	CBP[11]	KP	Others
CUB [43]	VGG-16 [38]	73.1*	84.1	84.3	<b>86.2</b>	82.0 84.1
	ResNet-50 [15]	78.4	N/A	81.6	84.7	[18] [16]
Stanford Car [19]	VGG-16	79.8*	91.3	91.2	<b>92.4</b>	<b>92.6</b> 82.7
	ResNet-50	84.7	N/A	88.6	91.1	[18] [14]
Aircraft [27]	VGG-16	74.1*	84.1	84.1	<b>86.9</b>	80.7
	ResNet-50	79.2	N/A	81.6	85.7	[14]
Food-101 [4]	VGG-16	81.2	82.4	82.4	84.2	50.76
	ResNet-50	82.1	N/A	83.2	<b>85.5</b>	[4]

Table 2. Performance comparisons among all baselines, where KP is the proposed kernel pooling method with learned coefficients. Following the standard experimental setup, we use the input size of  $448 \times 448$  for CUB, Stanford Car and Aircraft datasets except the original VGG-16 (marked by an asterisk \*), which requires a fixed input size of  $224 \times 224$ . For Food-101, we use the input size of  $224 \times 224$  for all the baselines.

## 2) Rank X

- Если классификация многоклассовая, мы ранжируем ранжируем все выходы по качеству
- Если истинный ответ попадает в первые X выходов, тогда результат считается верным (часто  $X=5$ )

# Домены данных



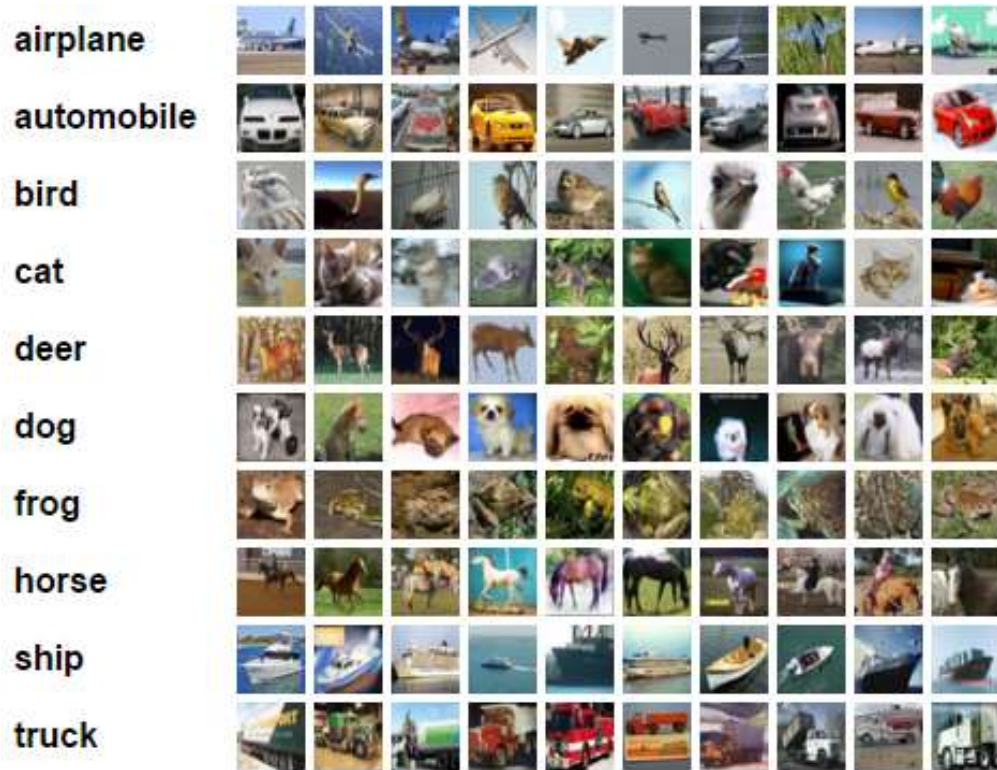
- Каждый алгоритм компьютерного зрения разрабатывается для работы с определённой выборкой изображений из некоторого распределения (statistical population) изображений (ещё называется *доменом*)
- Распределение всех «валидных» (для алгоритма) изображений:
  - $img \sim P(I), I \subseteq R^{H \times W \times C}$
- Алгоритмы работают, используя свойства и закономерности в рассматриваемой выборке данных
- Примеры доменов – рентгены лёгких, данные видеонаблюдения, лица людей, и т.д.



# CIFAR-10 и CIFAR-100



## Выборки из TinyPictures



<http://www.cs.toronto.edu/~kriz/cifar.html>

- CIFAR-10
  - 10 классов
  - 60000 изображений
  - 5000 обучающих и 1000 тестовых на класс
- CIFAR-100
  - 100 классов
  - 60000 изображений
  - 500 обучающих и 100 тестовых на класс



# ImageNet

Цель: собрать коллекцию с 1000 изображений на каждый из 117000 категорий

~14 000 000 изображений (~1 000 000 с аннотацией ограничивающими прямоугольниками)

~ 22 000 не пустых класса (~10 000 классов с более чем 1000 примерами)



IMAGENET

14,197,122 images, 21041 synsets indexed

SEARCH

Home Explore  
About Download

Not logged in. [Login](#) | [Signup](#)

Start exploring here

[WordNet Structure](#) [Cloud map](#) [Most popular](#)

1-20 of 735 most popular synsets in current ImageNet ([How do we measure popularity?](#))

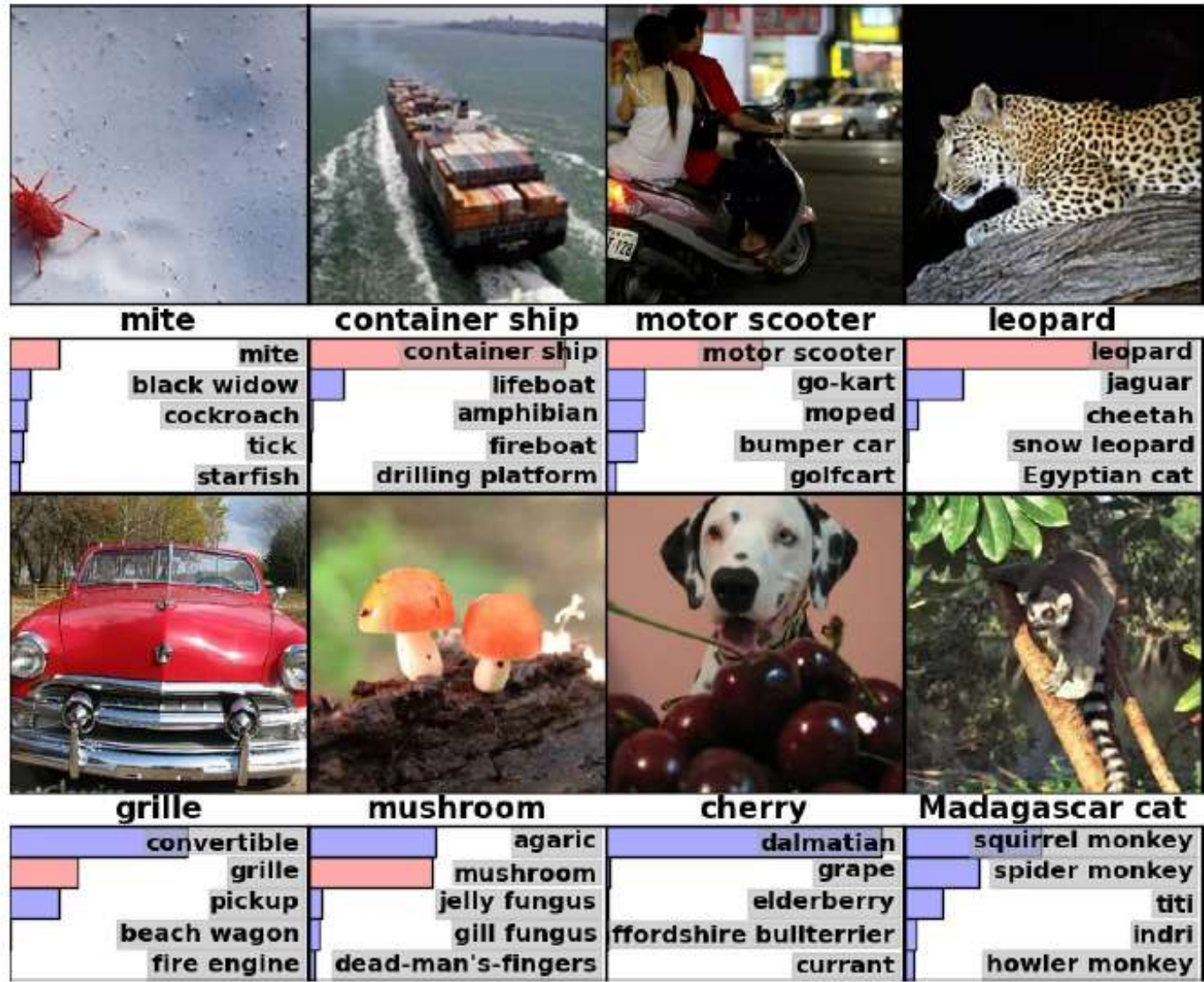
**Synset: people** has bounding box  
Definition: (plural) any group of human beings (men or women or children) collectively; "old people"; "there were at least 200 people in the audience".

**Synset: homo, man, human, being, human** has bounding box  
Definition: any living or extinct member of the family Hominidae characterized by superior intelligence, articulate speech, and erect carriage.

**Synset: child, kid** has bounding box  
Definition: a human offspring (son or daughter) of any age; "they had three children"; "they were able to send their kids to college".

Source: <http://image-net.org>

# Проблемы с разметкой в ImageNet





Пример microwave oven

- Цель – сделать самую большую открытую коллекцию изображений из реальной жизни с разнообразной разметкой
- 9 млн. изображений с лицензией CC BY 2.0
- 59,919,574 меток для 19,957 категорий
- Для тех же изображений есть много других видов разметок, например, локализованные текстовые описания

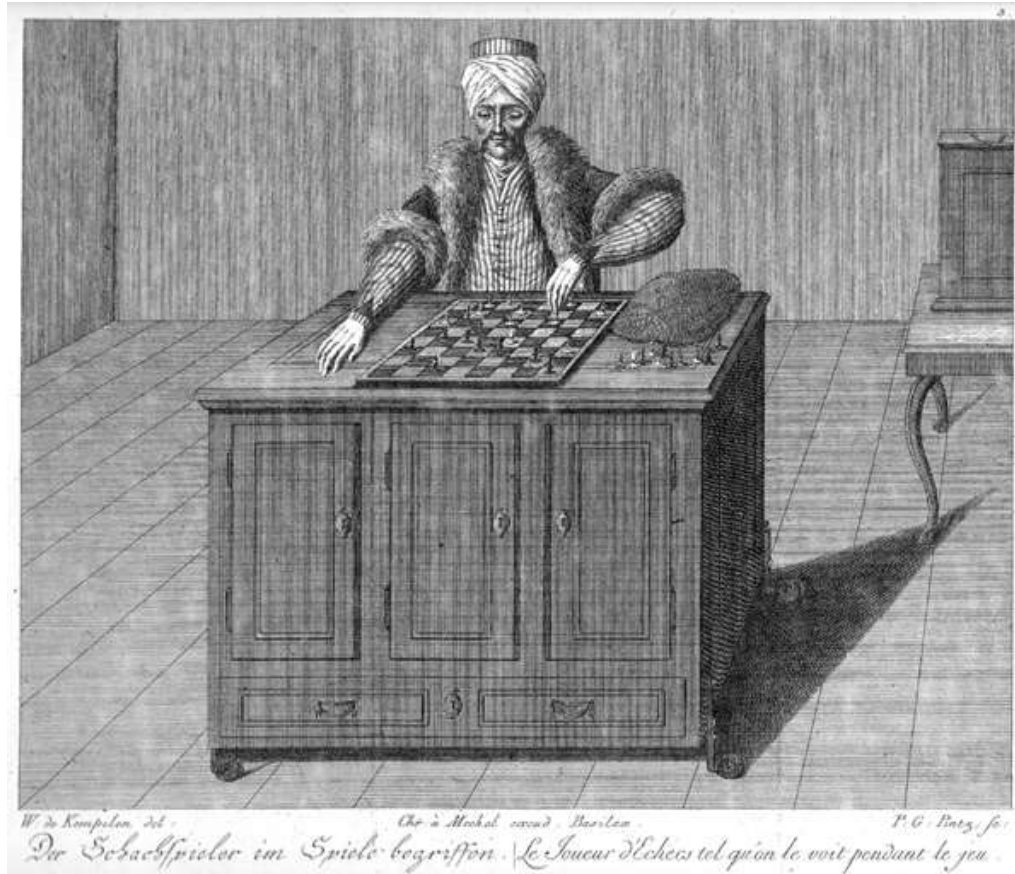
# Fine-grained classification



- Конкретные экземпляры (виды) в рамках одной категории

Figure 7. Images we used for visual recognition. From left to right, each column contains examples from CUB Bird [43], Stanford Car [19], Aircraft [27] and Food-101 [4].

# Как готовить коллекции?



- Mechanical Turk - Automaton Chess Player – робот, игравший в шахматы
  - Автоматон двигает фигуры, говорит «Чек» и обыгрывает всех!
- С 1770 по 1854 развлекал публику, только в 1820 году раскрыли обман



<http://www.galaxyzoo.org/>

- Классификация изображений галактик
- Первый масштабный проект такого рода
- Более 150000 волонтеров за первый год бесплатно сделали более 60 млн. меток



Какие объекты присутствуют на картинке?

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10

Сделано: 0/10

- 1 Светофоры
- 2 Люди
- 3 Машины
- 4 Велосипеды
- 5 Мотоциклы
- 6 Рельсовый
- 7 Ничего

Для переключения между изображениями используйте кнопки в верхней части экрана или клавиши "←" и "→" на клавиатуре.

Выбор варианта ответа можно сделать с помощью клавиатуры. Клавиши от 1 до 9, соответствующие подсказкам.

Для масштабирования пользуйтесь колёсиком мыши или клавишами "+" и "-".

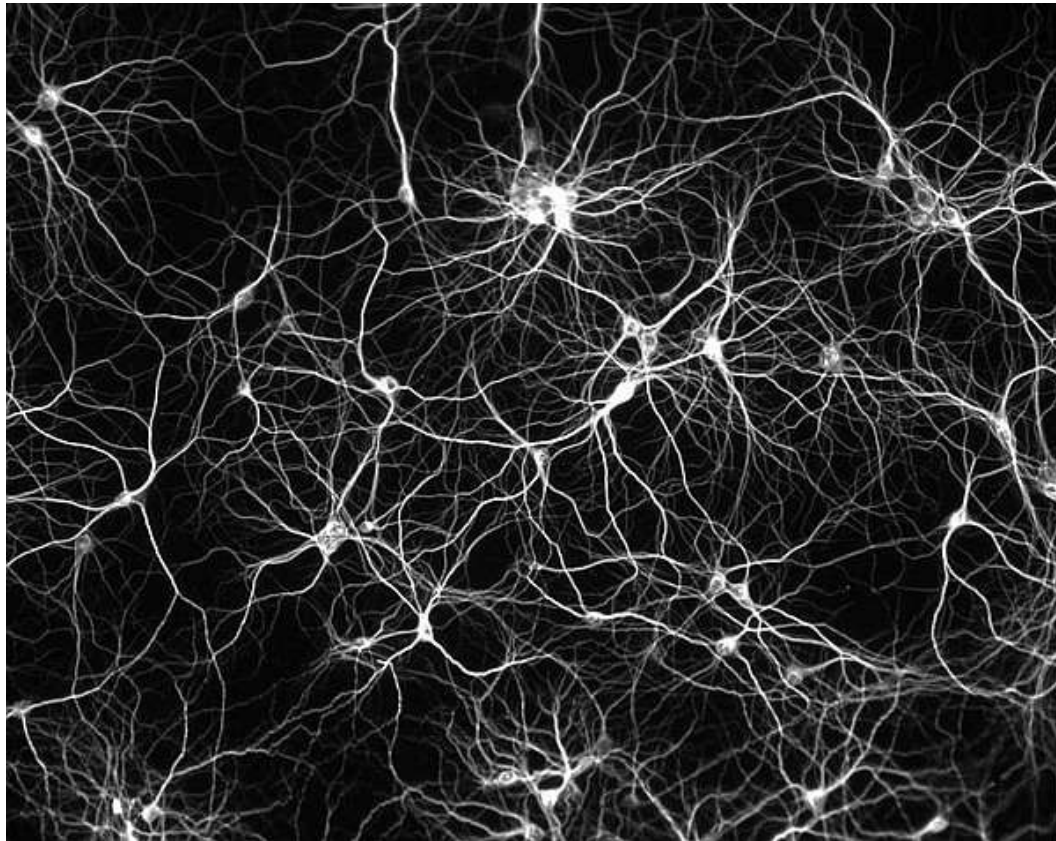






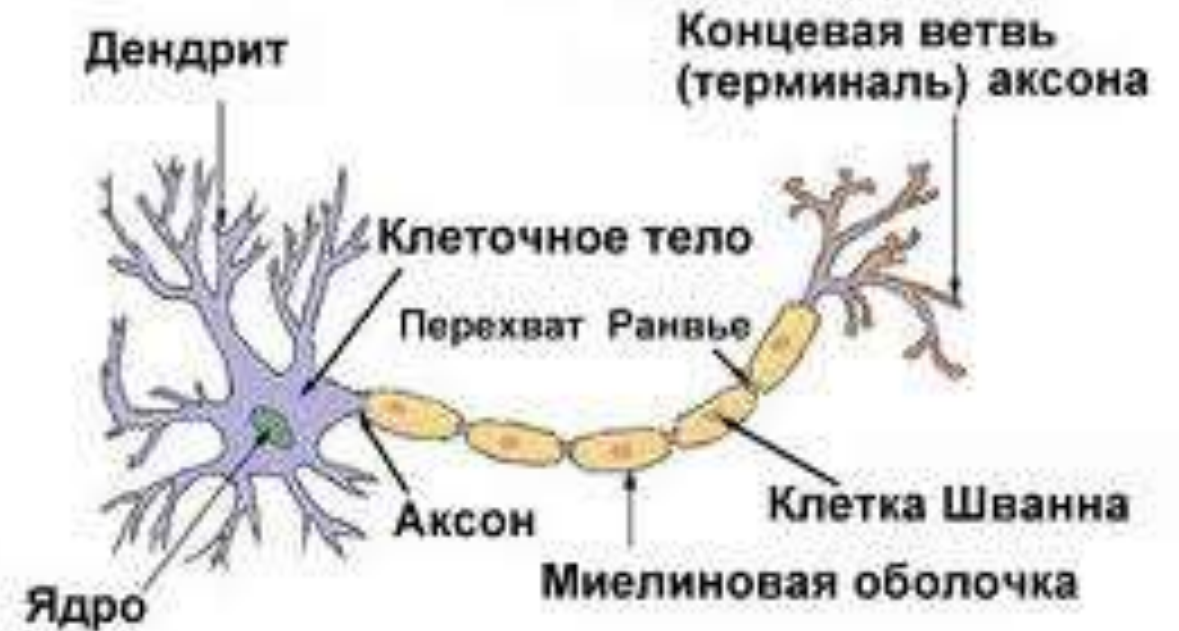
## 2. Линейная классификация и персептрон

# Структура мозга человека



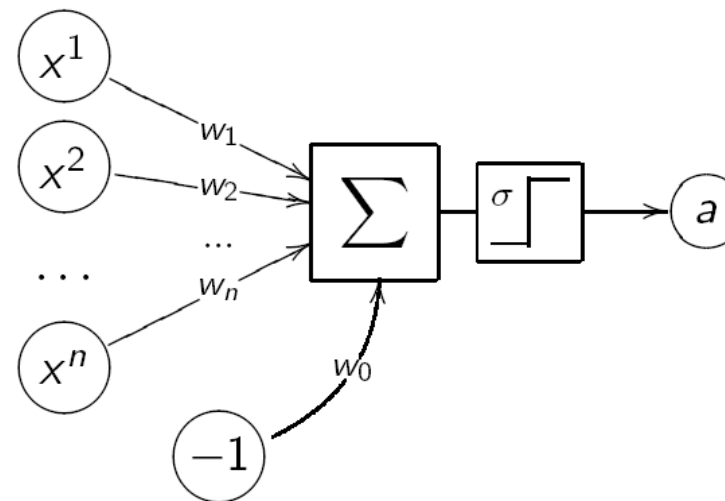
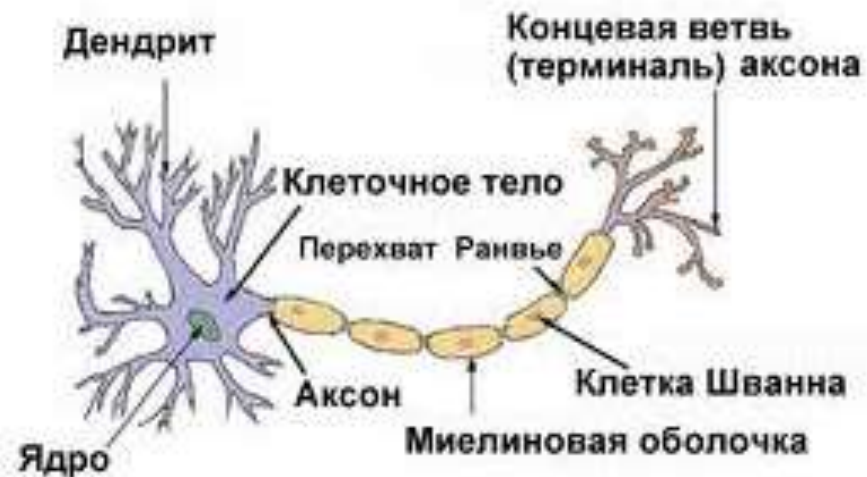
Нейросеть

## Типичная структура нейрона



Отдельный нейрон

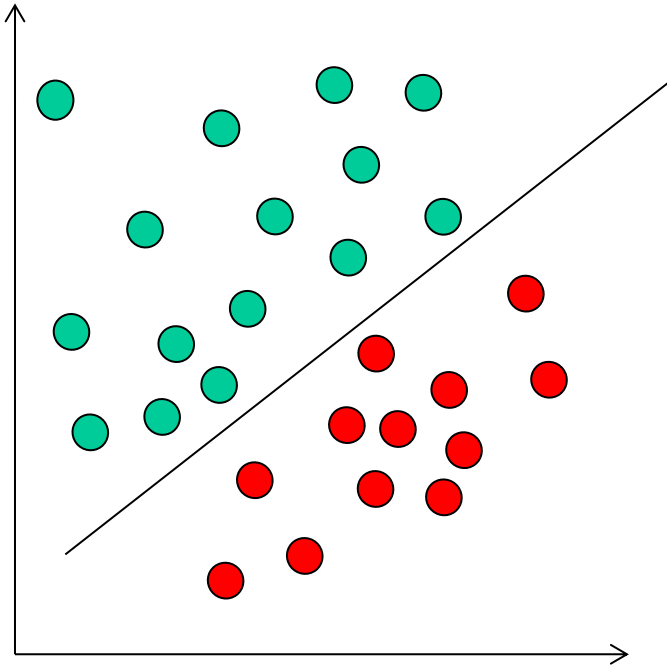
# Линейная модель МакКаллока-Питтса



$$a(x, w) = f\left(\sum_{i=1}^n w_i x_i + b\right)$$

- $w_i$  - весовые коэффициенты синаптических связей
- $b$  – bias (иногда  $w_0$  - порог активации)
- $f()$  – функция активации

# Нейрон как линейный классификатор



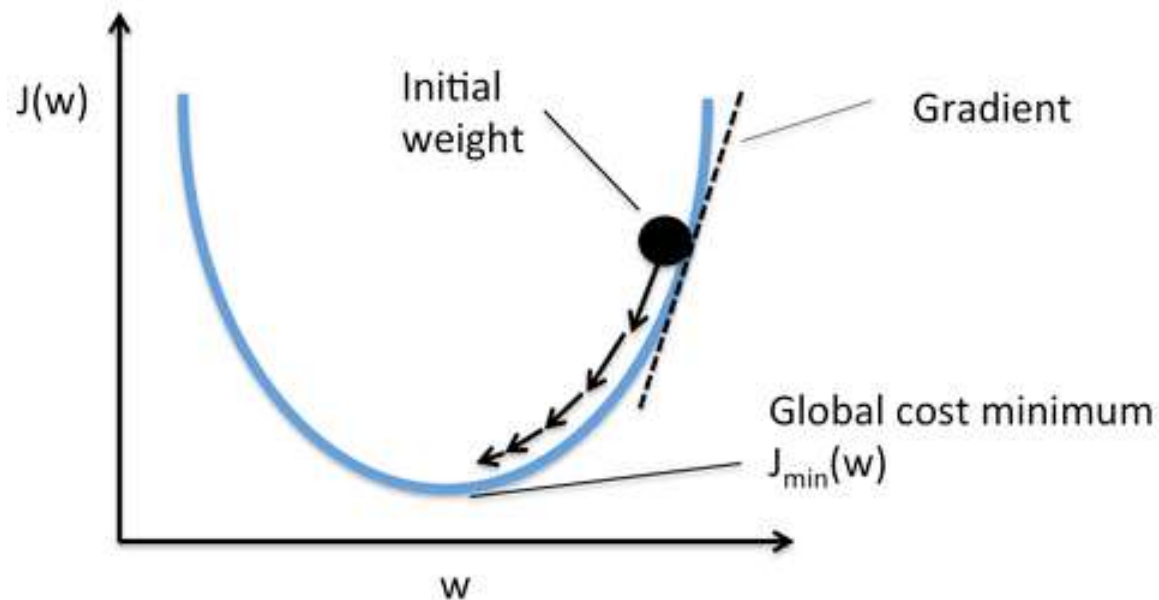
$$a(x, w) = f\left(\sum_{i=1}^n w_i x_i + b\right)$$

$x_i$  положительные:  $x_i \cdot w + b \geq 0$

$x_i$  отрицательные:  $x_i \cdot w + b < 0$

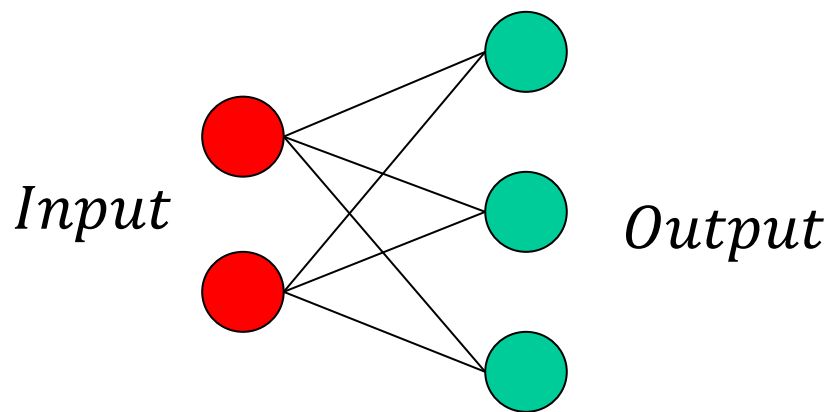
- Нейрон с функцией активации `sign` задаёт бинарный линейный классификатор (гиперплоскость в  $x$ )
- «Обучение» нейрона = настройка весов  $w_j$  и  $b$
- Настраивать веса линейного классификатора можем по обучающей выборке с использованием градиентного спуска, минимизируя выбранную функцию потерь

# Напоминание про градиентный спуск

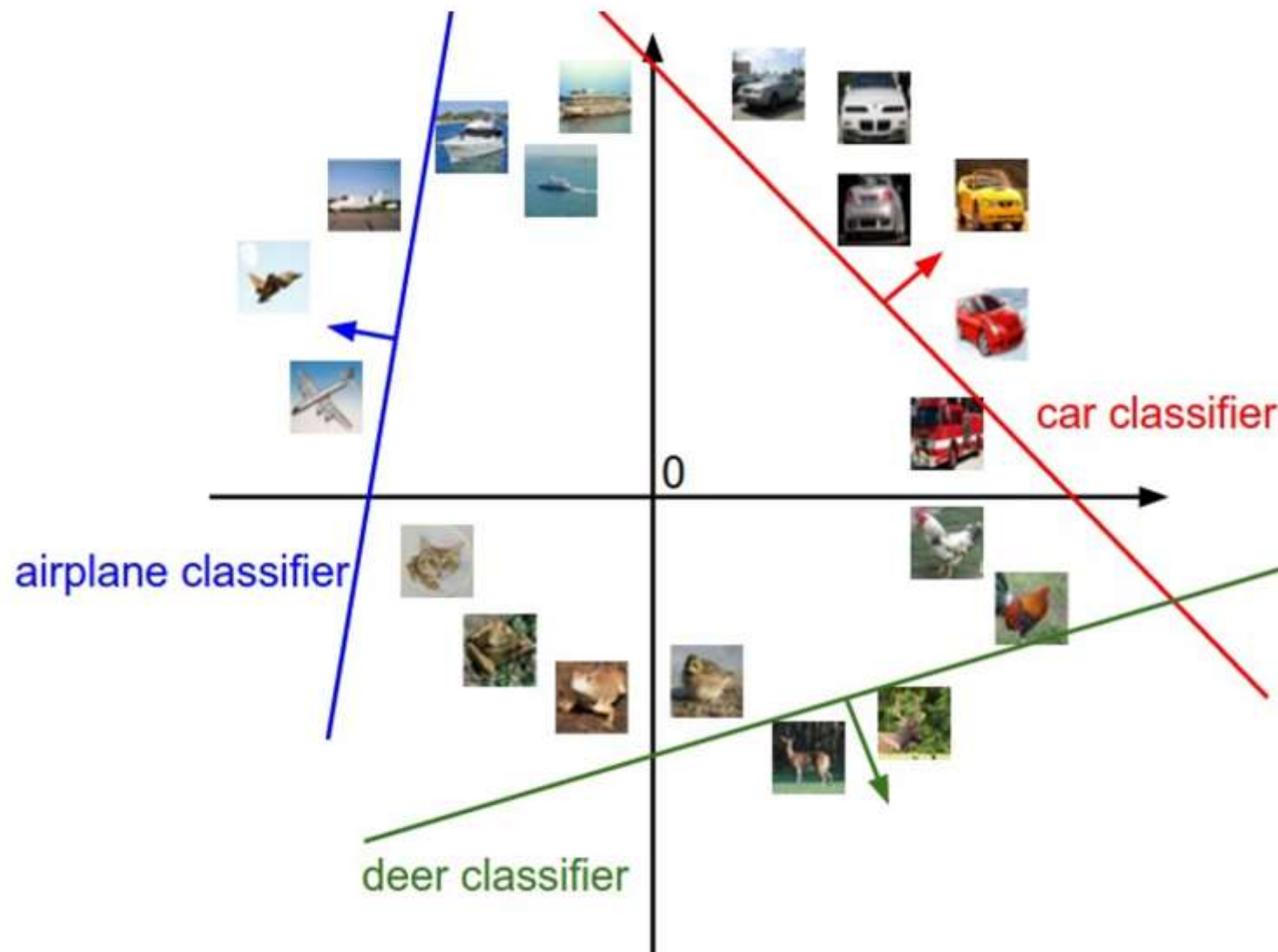


- Есть функция стоимости от параметров  $w$ , нужно найти параметры, при которых она достигает минимума
- Считаем градиент функции с точки начального приближения и сдвигаем  $w$  в сторону уменьшения стоимости
- Повторяем до сходимости
- Попадаем в локальный минимум (который может быть глобальным, или нет)

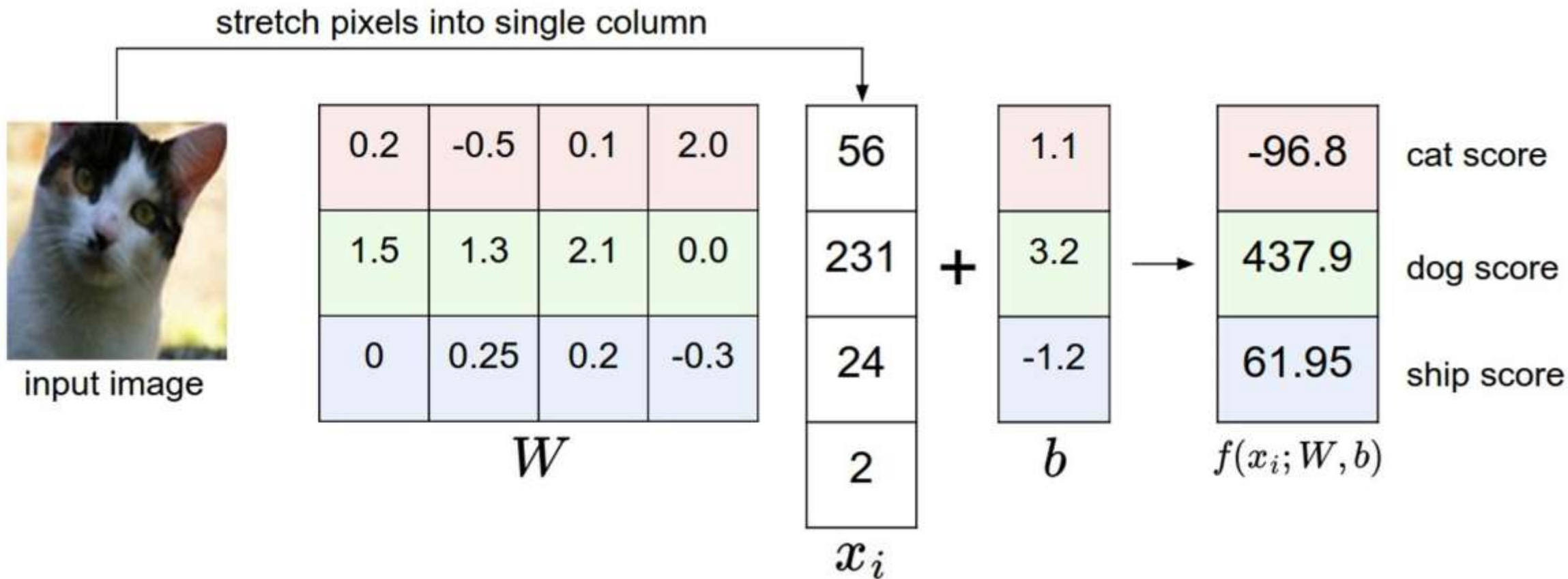
# Многоклассовая линейная классификация



Линейный персептрон



# Многоклассовая линейная классификация



# Функция потерь для многоклассовой классификации



- Categorical cross-entropy loss
- Измеряет близость истинного и оцененного распределения меток

$$L(\hat{y}, y) = - \sum_k^K y^{(k)} \log \hat{y}^{(k)}$$

- Будем интерпретировать выходы (score) как ненормализованный логарифм вероятности и подадим на вход softmax преобразованию

$$x = x_1, \dots, x_k$$
$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^k e^{x_j}}$$

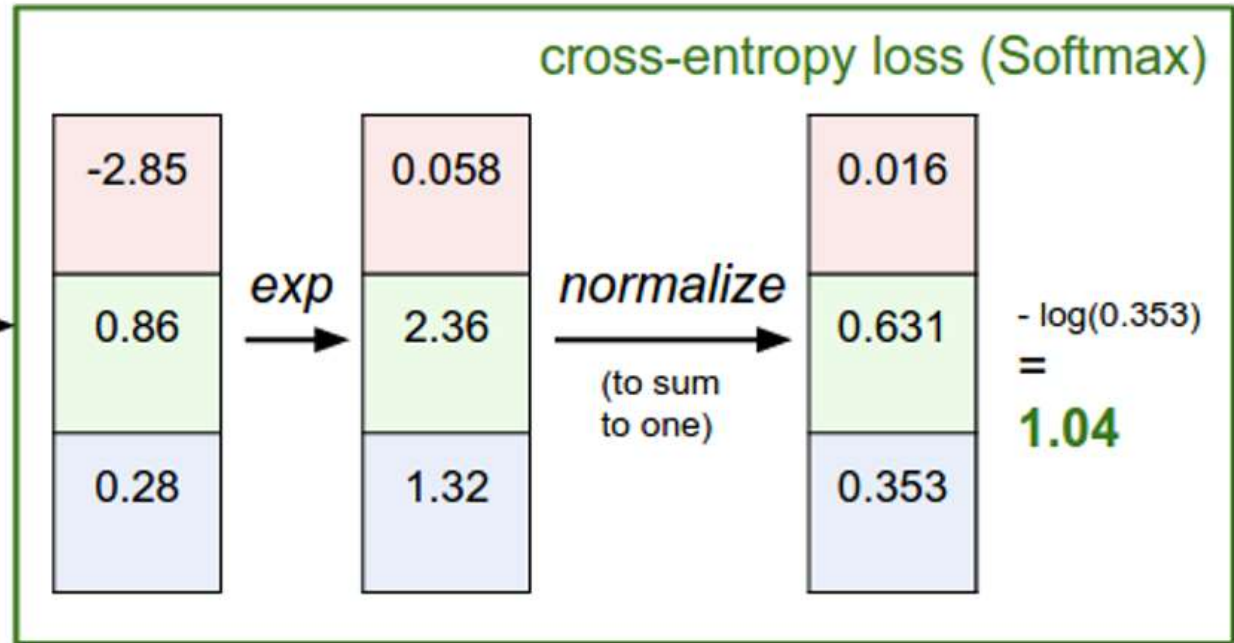
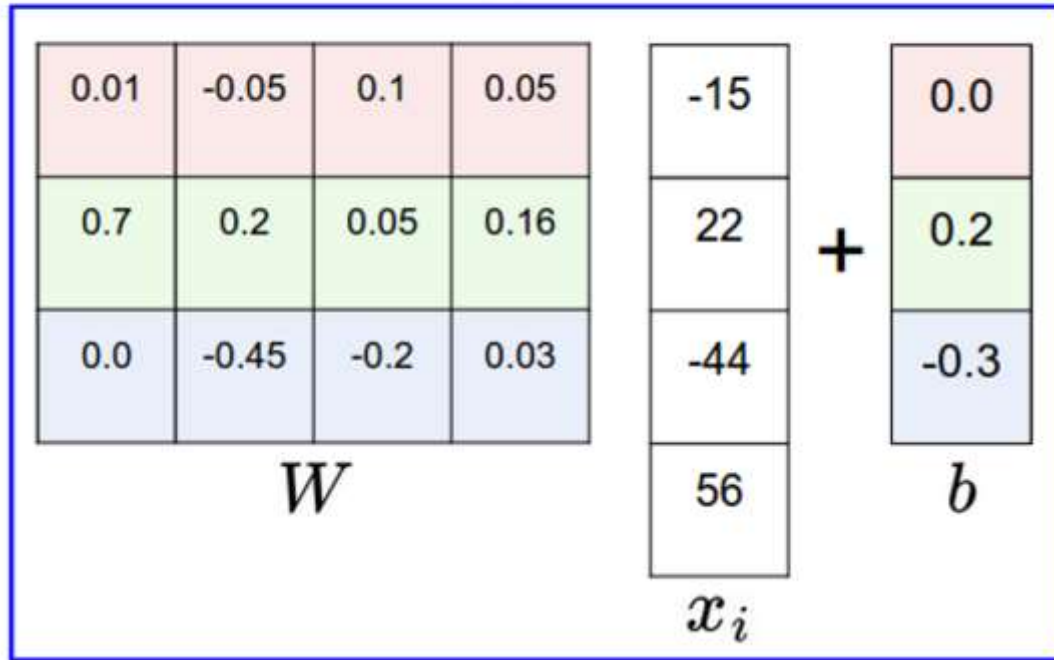
- После этого получаем метки – оценку вероятности принадлежности классу



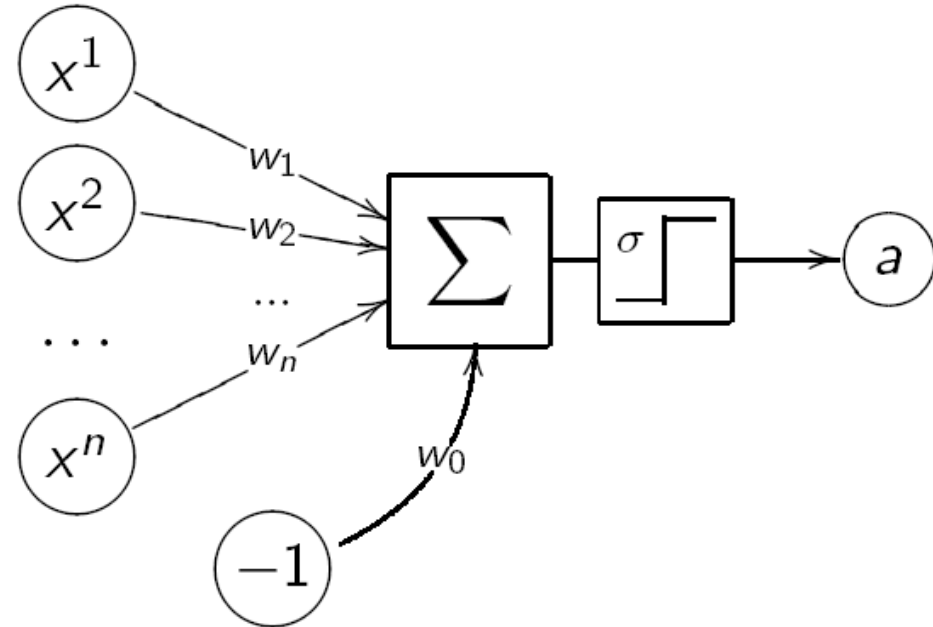
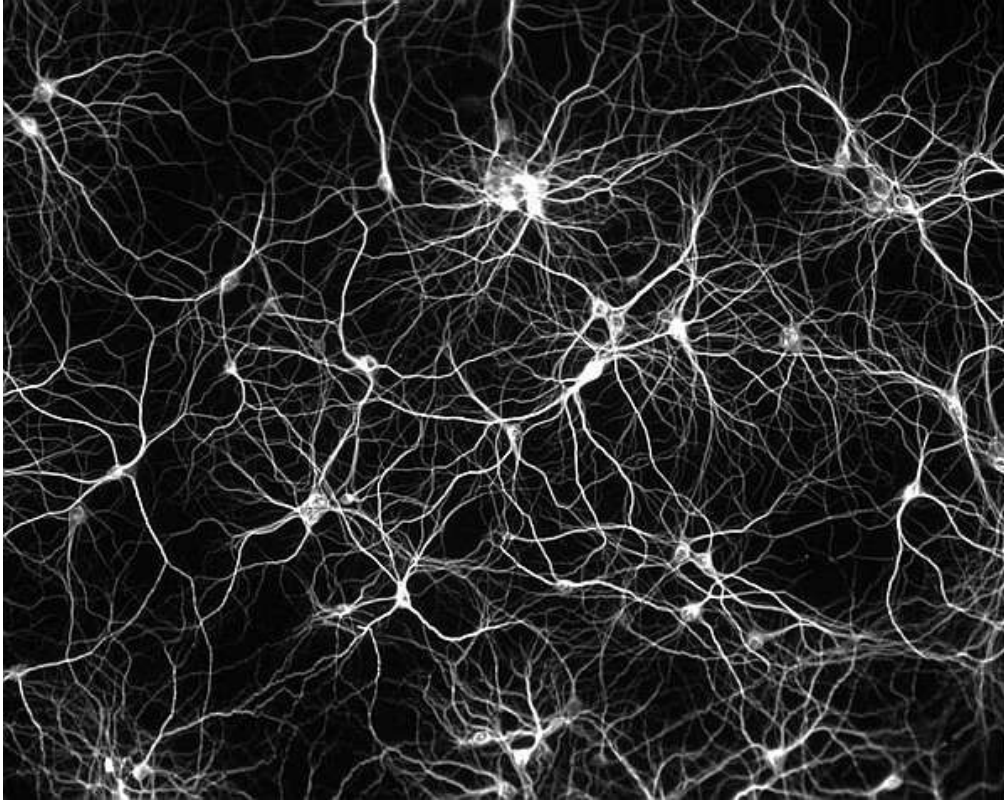
# Многоклассовая линейная классификация



matrix multiply + bias offset



# Представимость функций нейросетями



- Обычный перспептрон и нейрон реализуют линейную классификацию
- А суперпозицией (сетью) нейронов?

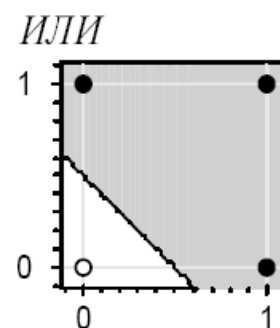
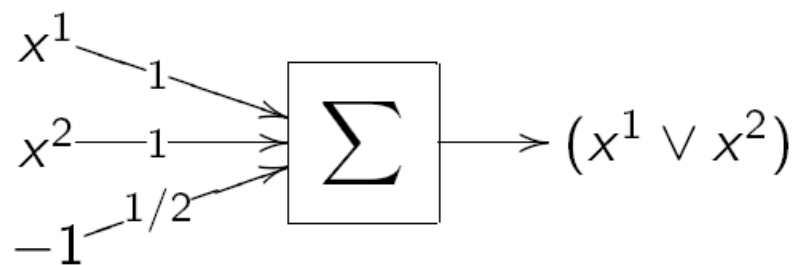
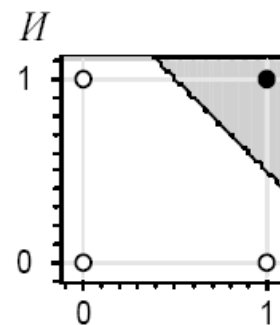
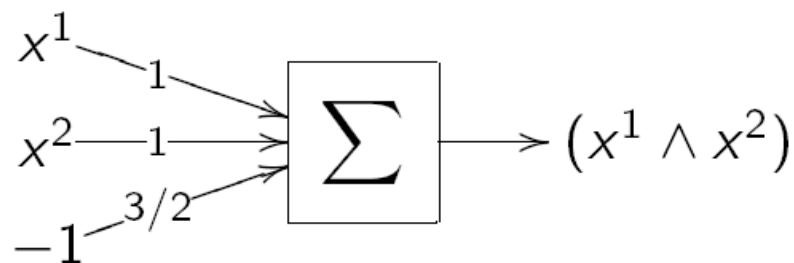


Функции И, ИЛИ, НЕ от бинарных переменных  $x^1$  и  $x^2$ :

$$x^1 \wedge x^2 = [x^1 + x^2 - \frac{3}{2} > 0];$$

$$x^1 \vee x^2 = [x^1 + x^2 - \frac{1}{2} > 0];$$

$$\neg x^1 = [-x^1 + \frac{1}{2} > 0];$$



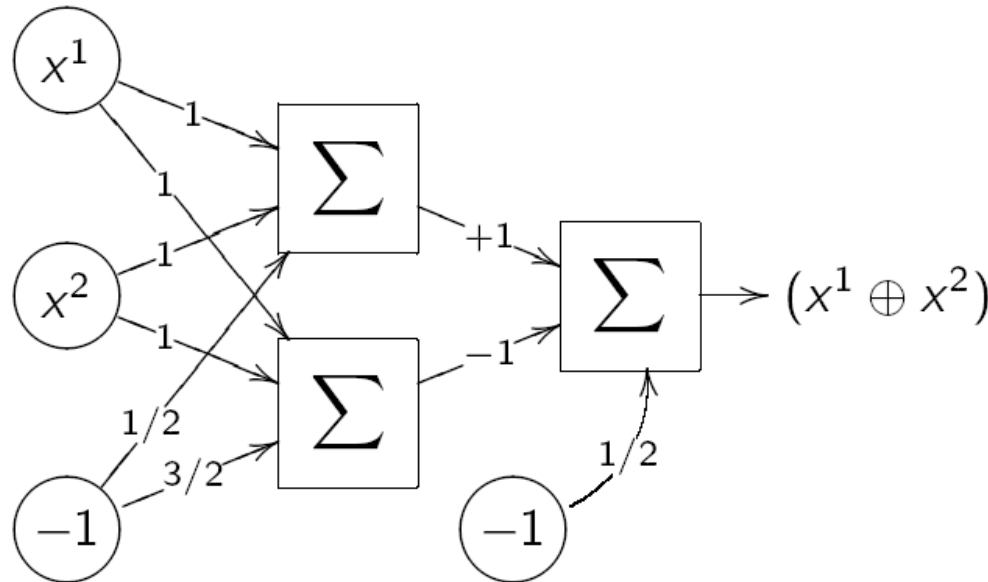
# Исключающее ИЛИ (XOR)



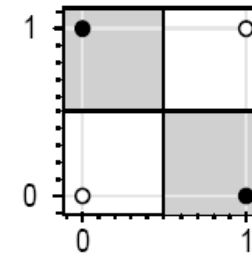
Функция  $x^1 \oplus x^2 = [x^1 \neq x^2]$  не реализуема одним нейроном.

Два способа реализации:

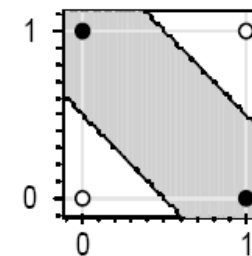
- Добавлением нелинейного признака:  
 $x^1 \oplus x^2 = [x^1 + x^2 - 2x^1x^2 - \frac{1}{2} > 0]$ ;
- **Сетью** (двухслойной суперпозицией) функций И, ИЛИ, НЕ:  
 $x^1 \oplus x^2 = [(x^1 \vee x^2) - (x^1 \wedge x^2) - \frac{1}{2} > 0]$ .



1-й способ



2-й способ





## Утверждение

Любая булева функция представима в виде ДНФ, следовательно, и в виде двухслойной сети.

Решение тринадцатой проблемы Гильберта:

## Теорема (Колмогоров, 1957)

Любая непрерывная функция  $n$  аргументов на единичном кубе  $[0, 1]^n$  представима в виде суперпозиции непрерывных функций одного аргумента и операции сложения:

$$f(x^1, x^2, \dots, x^n) = \sum_{k=1}^{2n+1} h_k \left( \sum_{i=1}^n \varphi_{ik}(x^i) \right),$$

где  $h_k, \varphi_{ik}$  — непрерывные функции, и  $\varphi_{ik}$  не зависят от  $f$ .

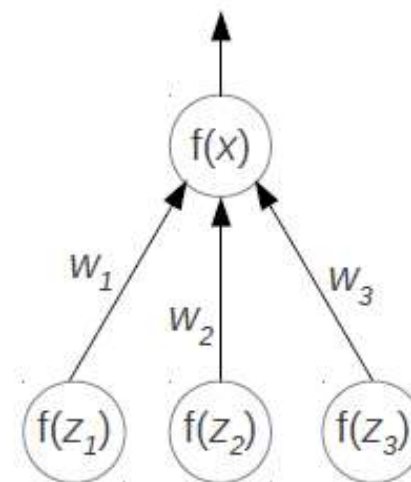
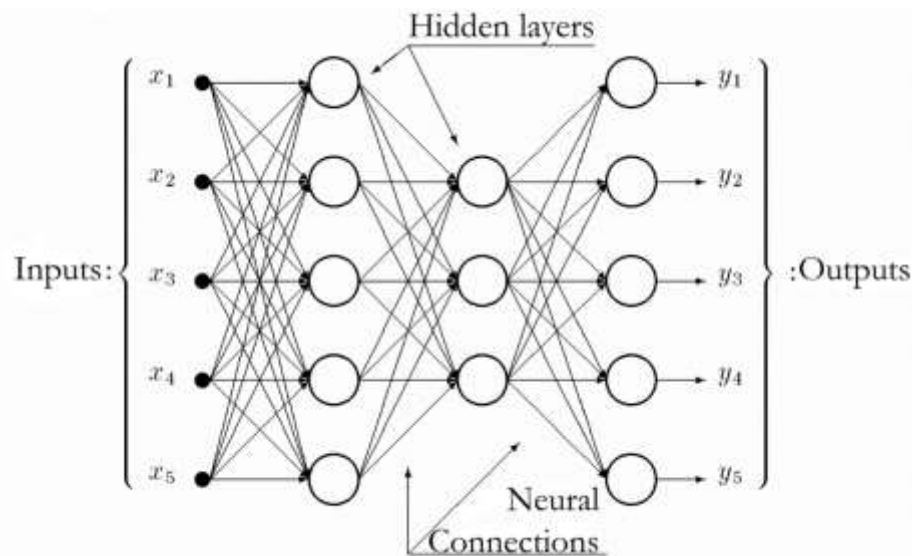
# Представимость функций

---



- Итого, теоретически доказано, что с помощью линейных операций и одной нелинейной функции активации можно вычислить любую непрерывную функцию с любой желаемой точностью
- Однако из доказательств не следует, как должна быть устроена сеть, сколько в ней должно быть нейронов, какие у них должны быть веса

# Задание нейросети

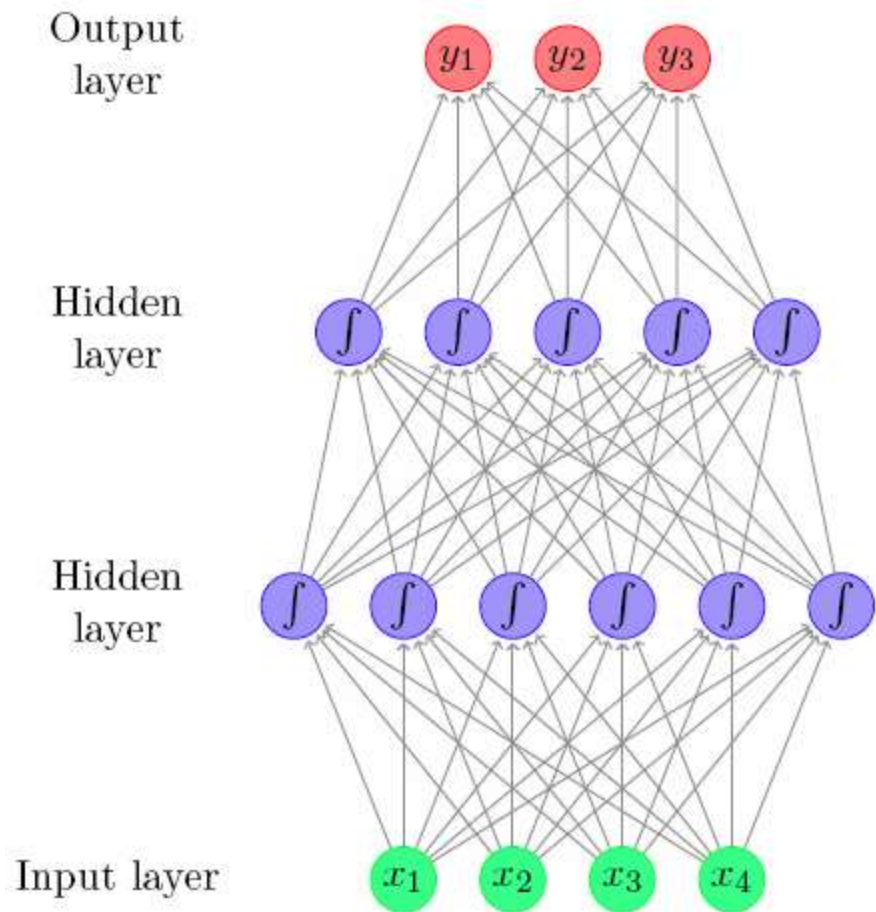


$$x = w_1 f(z_1) + w_2 f(z_2) + w_3 f(z_3)$$

$x$  is called the total input to the neuron, and  $f(x)$  is its output

- **Архитектура нейросети** – взвешенный ориентированный граф, в котором вершины – нейроны, ребра – связи
  - Архитектуру обычно задаёт разработчик на основании опыта и «лучших примеров»
  - Архитектуру только недавно начали «учить» (Neural Architecture Search)
- **Веса нейросети** – совокупность весов всех рёбер (веса каждого нейрона)
  - Настройка весов – «обучение» нейросети, для этого предложено несколько подходов

# Многослойный персептрон



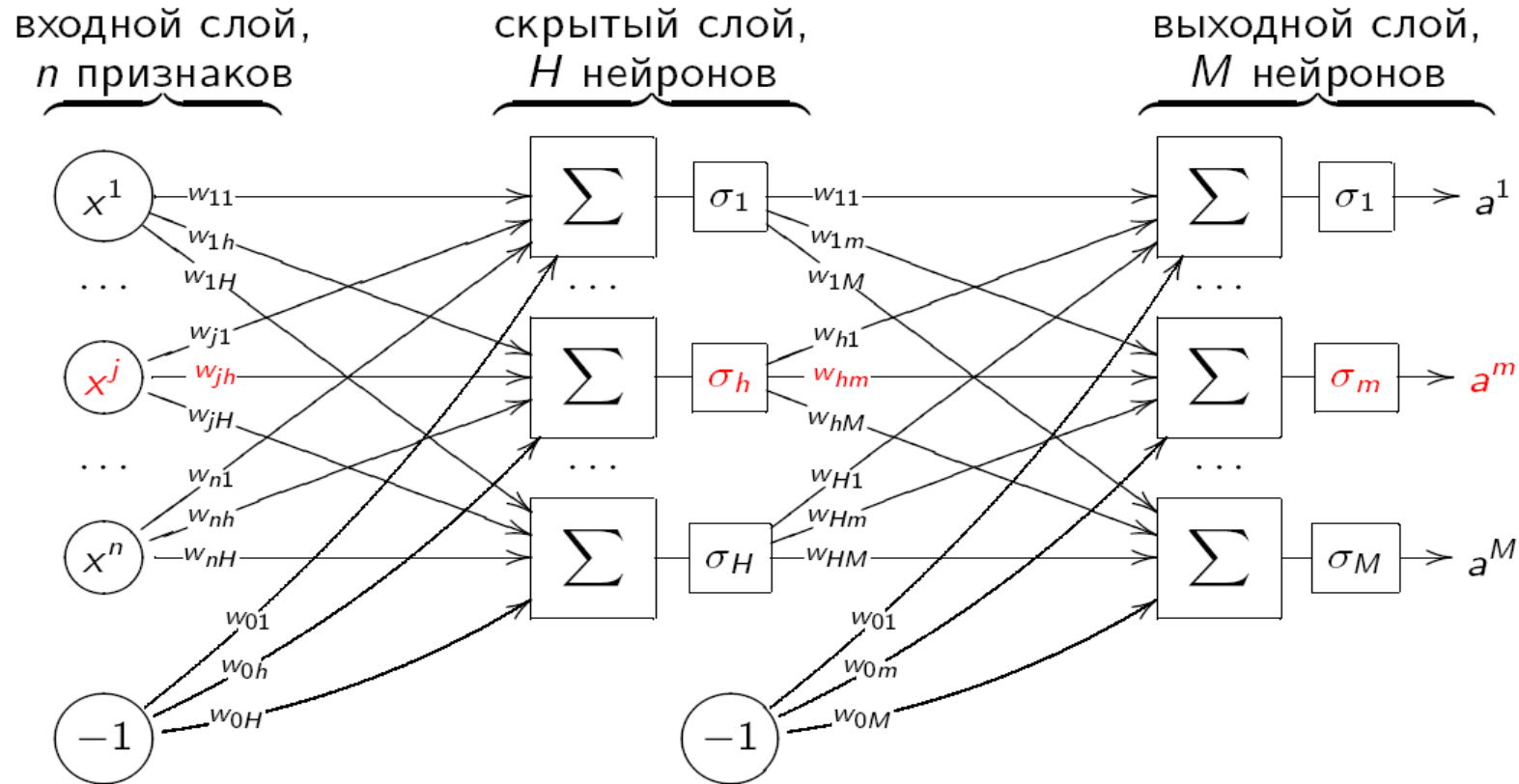
- Промежуточные слои – «скрытые»
- Если каждый нейрон слоя связан с каждым нейроном предыдущего слоя, то такой слой – полносвязанный (fully connected)
- Нейроны скрытых слоёв обычно имеют функцию активации (нелинейную)



# Многослойная feed-forward нейросеть

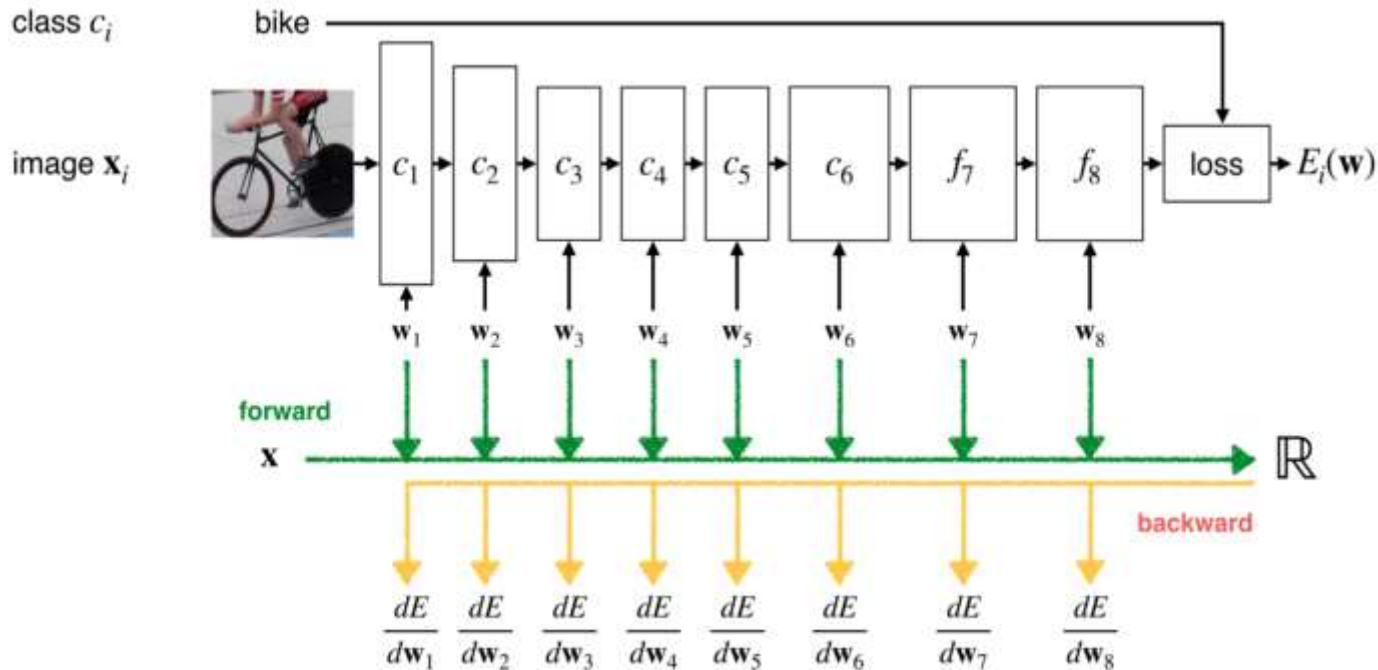


Пусть для общности  $Y = \mathbb{R}^M$ , для простоты слоёв только два.



- Передача сигналов идёт в одном направлении (feed-forward)
- Сеть можно разделить на «слои нейронов», по числу предшествующих нейронов на пути сигнала

# Обучение многослойных нейросетей



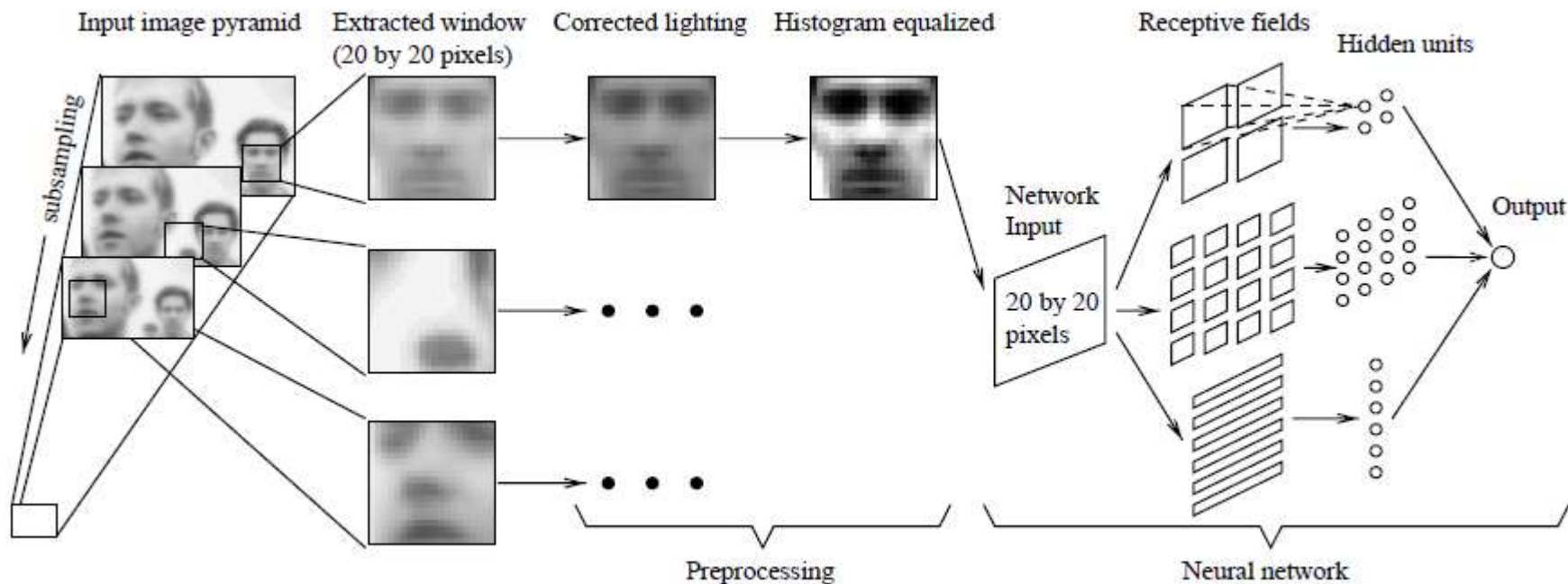
- Нейросеть вычисляет дифференцируемую функцию от своих входов
- Можем последовательно применять правило дифференцирования сложных функций для вычисления производных по каждому параметру нейросети
- Метод получил название «обратное распространение ошибки» и имеет длинную историю

- [↑](#) Галушкин А. И. Синтез многослойных систем распознавания образов. — М.: «Энергия», 1974.
- [↑](#) Werbos P. J., Beyond regression: New tools for prediction and analysis in the behavioral sciences. Ph.D. thesis, Harvard University, Cambridge, MA, 1974.
- [↑](#) [1](#) [2](#) Rumelhart D.E., Hinton G.E., Williams R.J., Learning Internal Representations by Error Propagation. In: Parallel Distributed Processing, vol. 1, pp. 318—362. Cambridge, MA, MIT Press. 1986.

# Rowley face detector (1998)



- Метод обратного распространения ошибки оказался очень эффективным
- Пример – детектор лица, лучший до Viola-Jones



B. Rowley, T. Kanade. Neural Network-Based Face Detection. PAMI, 1998.



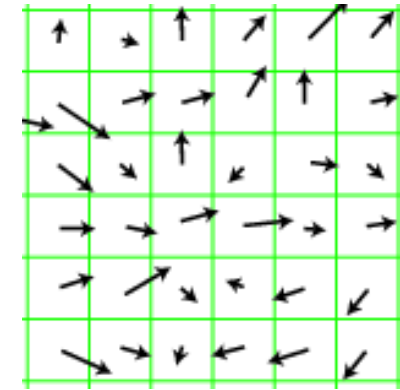
# Свёрточные нейронные сети

# Нейросети для обработки картинок



- Персептрон работает с векторами
- Мы хотим обрабатывать изображение, т.е. на вход подавать 3D матрицу  $X \in \mathbb{R}^{m \times n \times k}$
- Как должна быть устроена нейросеть, чтобы обрабатывать изображения?

- Мы также хотим учесть знания об обработке изображений в мозге человека и эвристическими методами распознавания
- Края, градиенты и т.д. хорошо помогают!



# Простые и сложные клетки визуальной коры



Хубель и Визель (Hubel & Wiesel)

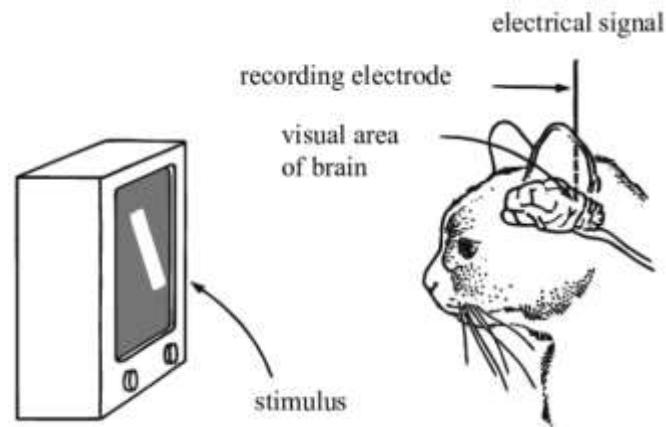
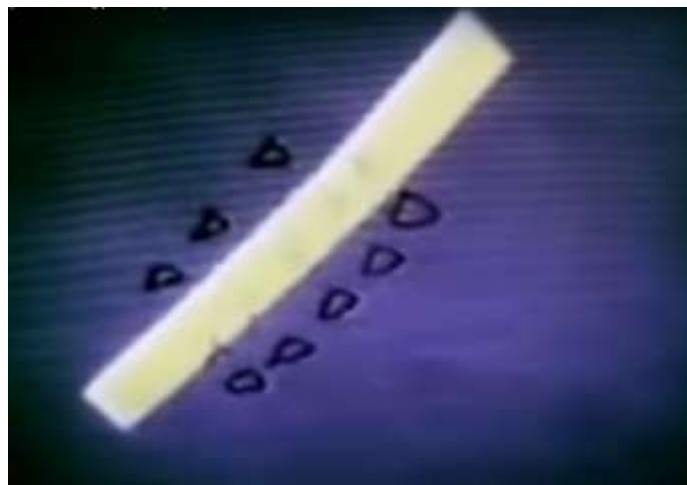
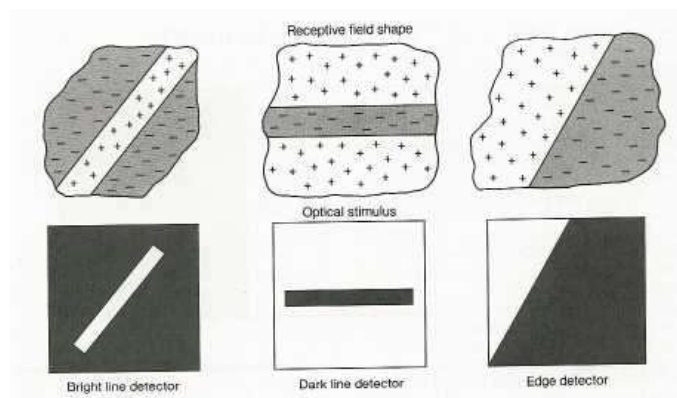


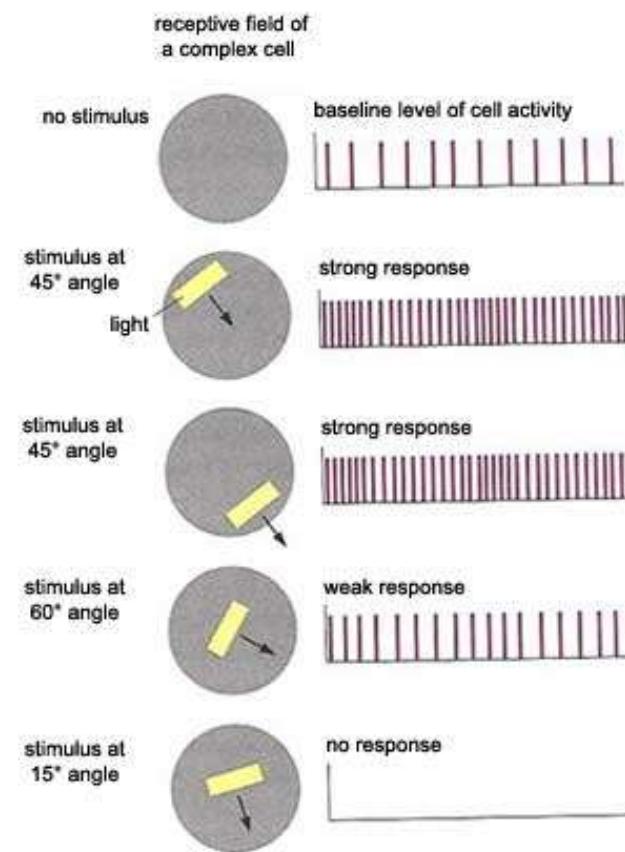
Схема эксперимента



Визуальный стимул



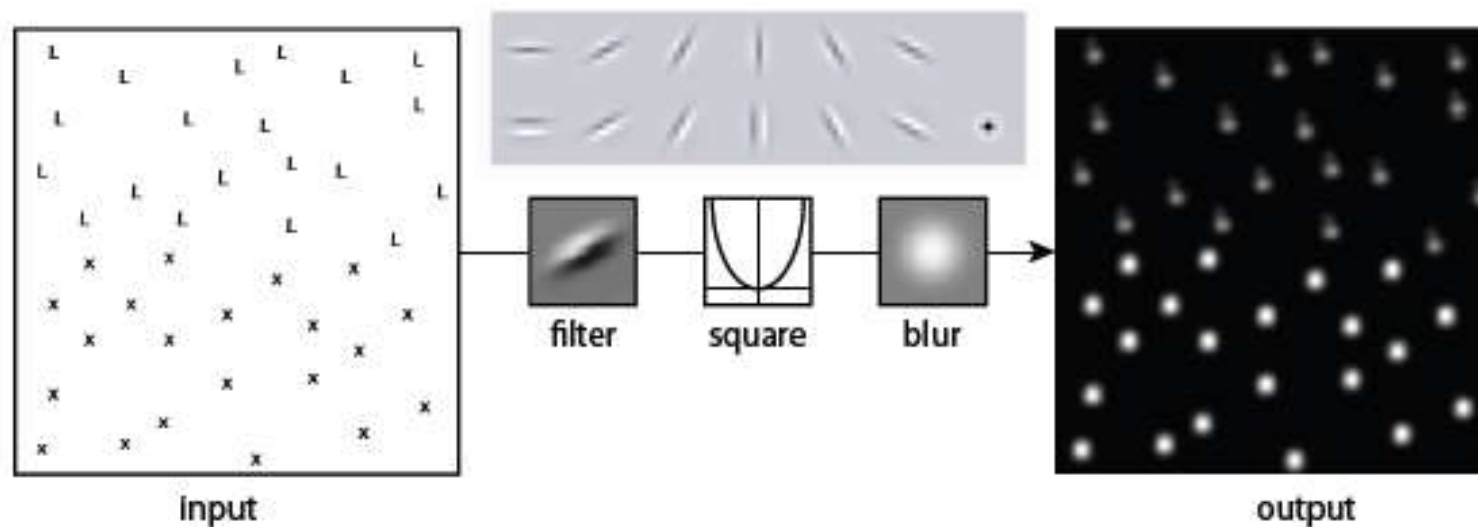
Простые клетки (S) - чувствительны к контрастным объектам определённого размера, ориентации и положения



Сложные клетки (C) - **инвариантны** к сдвигам в небольшой окрестности

Как S и C смоделировать?

# Банки текстурных фильтров



- Выберем набор (банк) фильтров, каждый из которых чувствителен к краю определенной ориентации и размера
- Каждый пиксель изображения после обработки банком фильтров даёт вектор признаков
- Этот вектор признаков эффективно описывает локальную текстуру окрестности пикселя

Pietro Perona and Jitendra Malik «Detecting and Localizing edges composed of steps, peaks and roofs», ICCV 1990



# Фильтры Габора как модель простых клеток

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi\frac{x'}{\lambda} + \psi\right)$$

$$x' = x \cos(\theta) + y \sin(\theta)$$

$$y' = -x \sin(\theta) + y \cos(\theta)$$

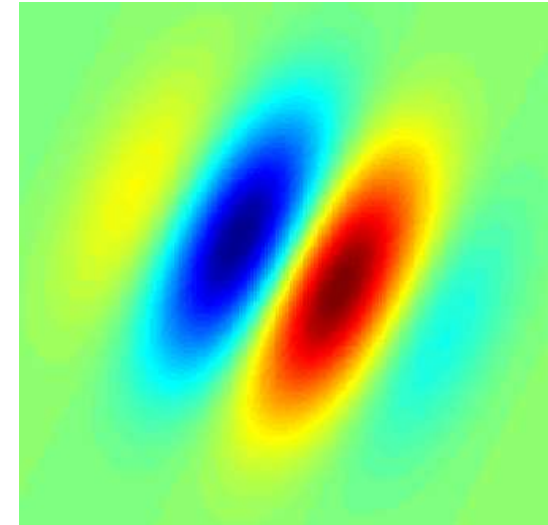
$\theta$  - ориентация

$\lambda$  - длина волны

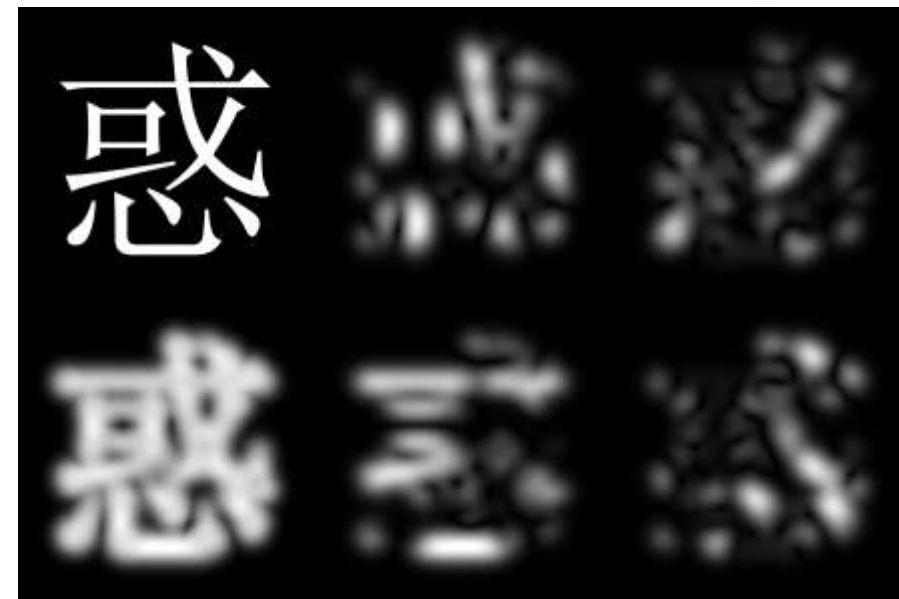
$\sigma$  - сигма гауссиана

$\gamma$  - соотношение размеров (aspect ratio), «эллиптичность фильтра»

$\psi$  - сдвиг фазы

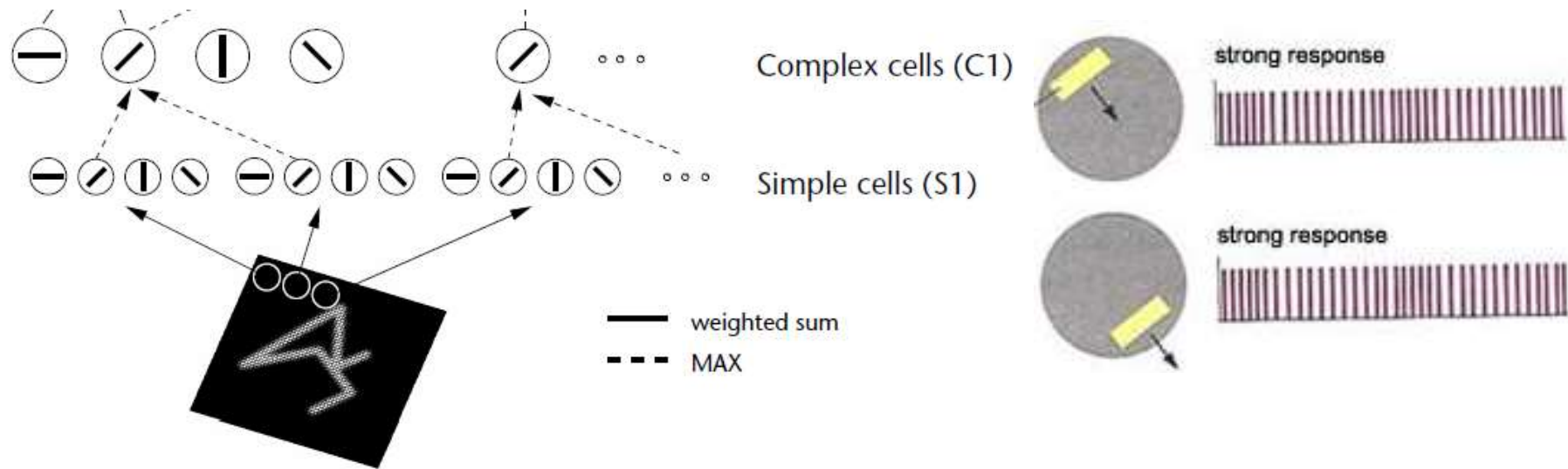


- 2D фильтр Габора – ядро гауссиана, домноженное на синусоиду
- Предложены в 1947 Денисом Габором (нобелевским лауреатом), независимо переоткрыты в 1980 году
- Позволяет сделать банк фильтров, для выделения краёв разной ориентации, масштаба и положения в окрестности



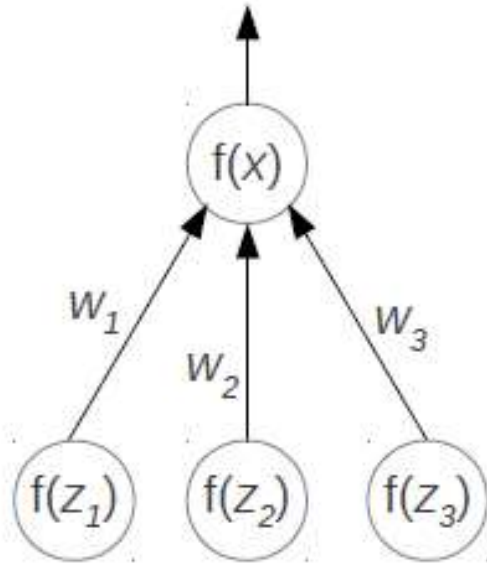


# MAX-pooling как модель сложных клеток

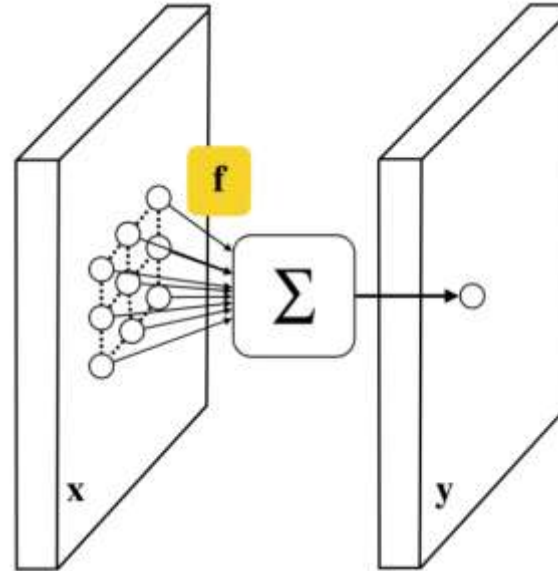


Инвариантность можно обеспечить за счёт применения оператора MAX на выходах набора «простых» клеток, чувствительных к краю одной ориентации, но в разных точках одной области

# Свёрточный слой

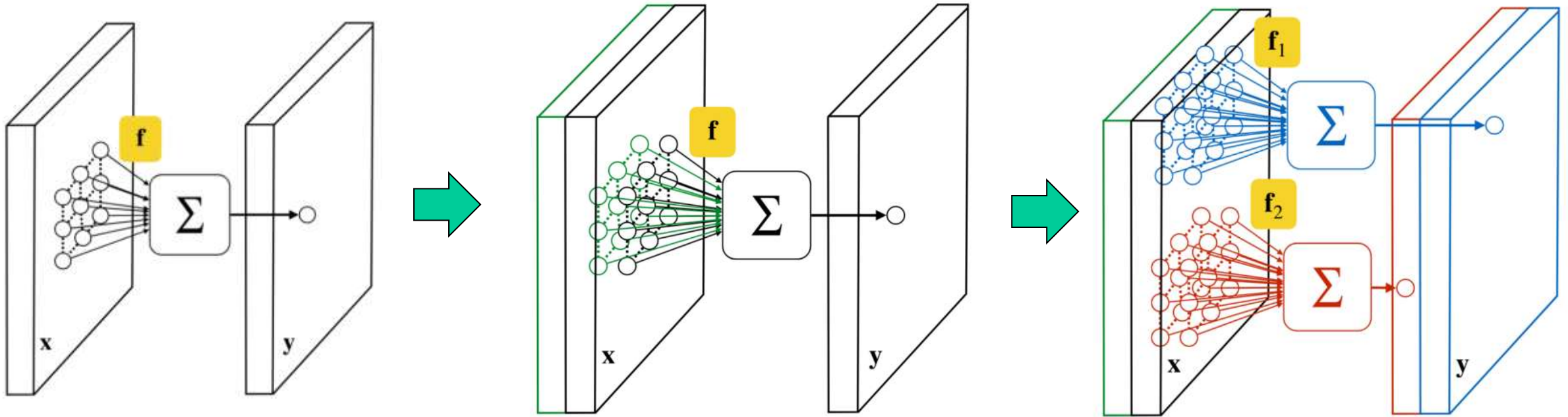


$$x = w_1 f(z_1) + w_2 f(z_2) + w_3 f(z_3)$$



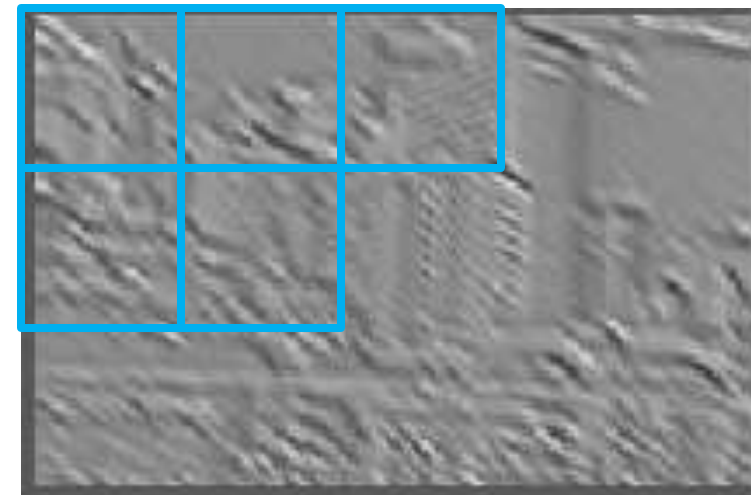
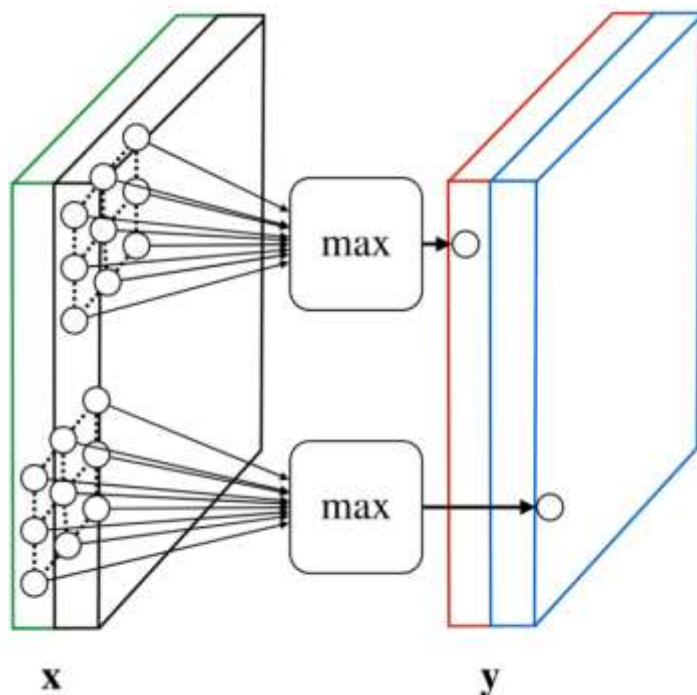
- Операцию линейной фильтрации (свёртки) для одного пикселя можно реализовать одним нейроном
- Свёртку изображения целиком можно реализовать как «слой» нейронов, веса которых одинаковы (shared weights)
- Обычно под «свёрточным слоем» понимают набор свёрток, применяемых к одному входу («банк фильтров»)

# Свёрточный слой



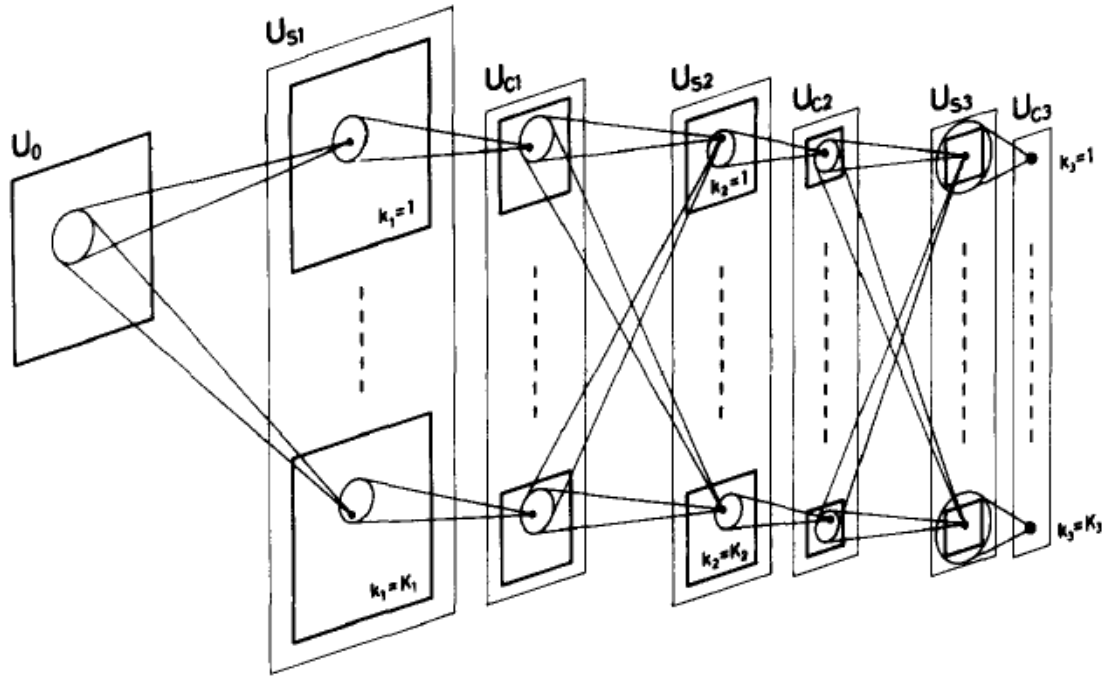
- Мы рассматривали свёртку как операцию над одноканальным 2D изображением, теперь расширим на 3D матрицу
- Обычно под «свёрточным слоем» понимают набор свёрток, применяемых к одному входу («банк фильтров»)
- Результаты свёрток объединяют в один выход  $X \in \mathbb{R}^{m \times n \times k}$ , где  $k$  – число свёрток в слое

# Pooling слой



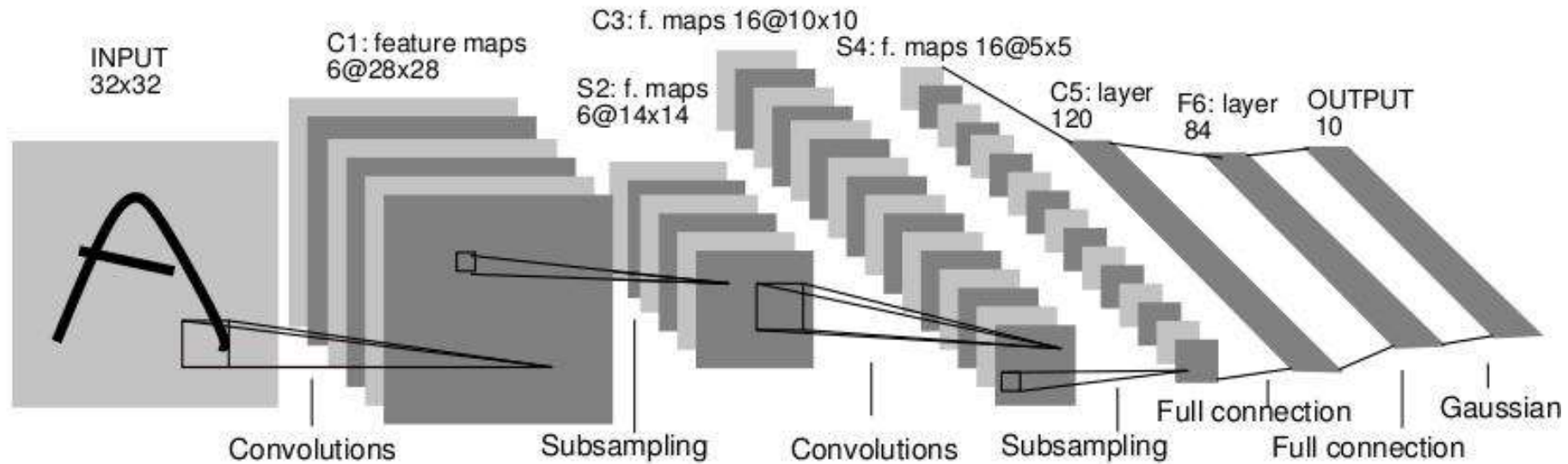
- Можно сделать специальный слой, который реализует операцию max-pooling
- Мы просто применяем операцию max по выбранным областям (по сетке)
- На вход получаем 3D матрицу, и выдаем 3D матрицу меньшего (обычно) разрешения
- При расчёте градиентов ошибки пробрасываем в тот нейрон предыдущего слоя, который дал max
- Можем для pooling применять операции max, sum, average, etc.

# Neocognitron (1980)



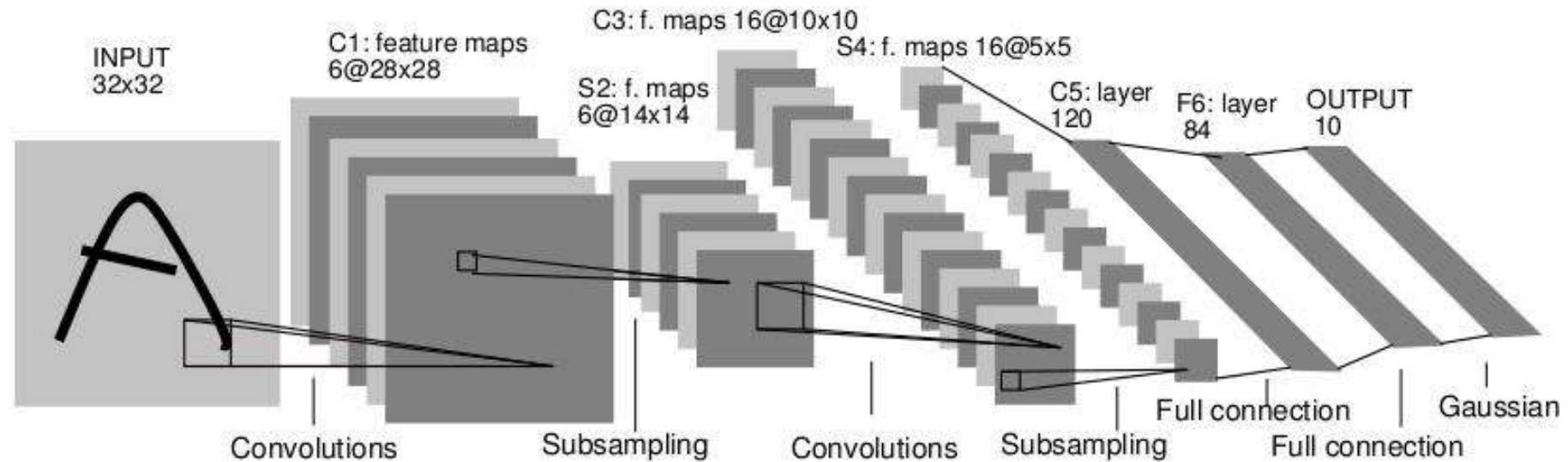
- Многослойная нейросеть с чередующимися S и C слоями
  - S-слои – линейные фильтры изображения («свёрточный слой»)
  - C-слои – MAX операторы, дающие инвариантность
- На верхнем уровне обеспечивается инвариантность по положению по всему изображению

# Свёрточные сети



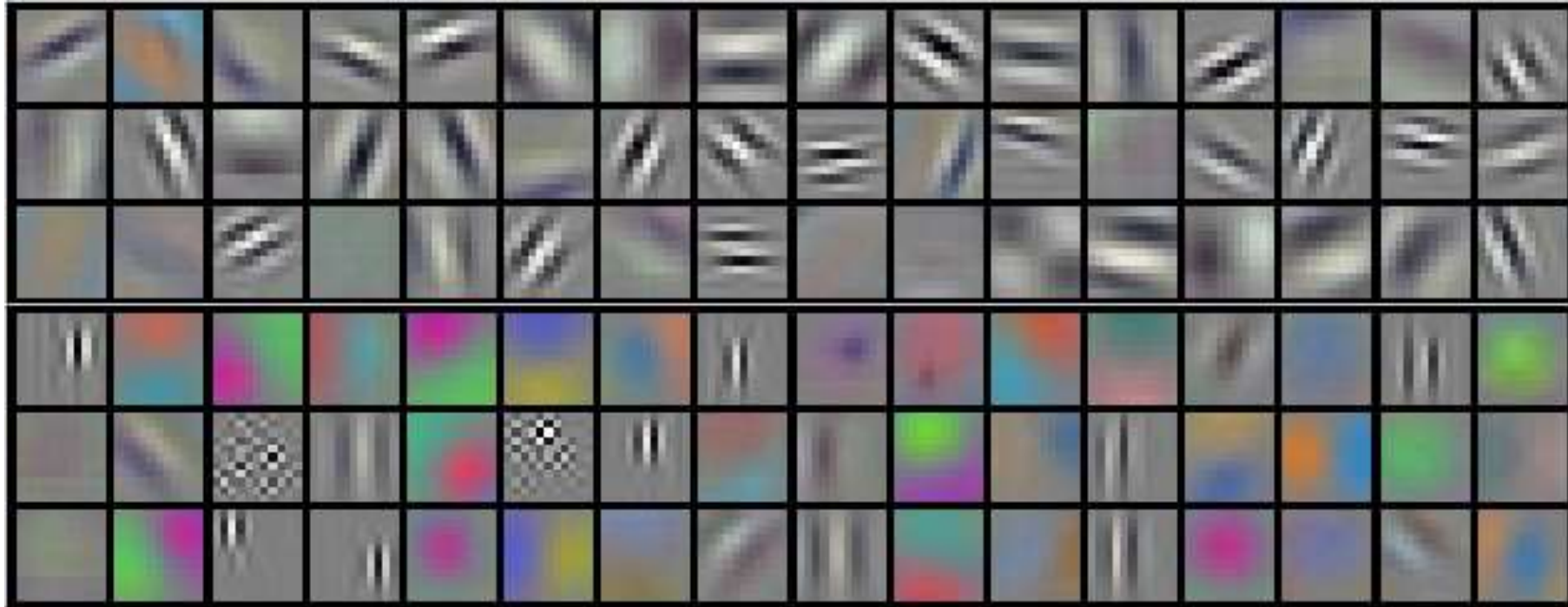
- Неокогнитрон + обратное распространение ошибки = свёрточная сеть (Convolutional Neural Network, CNN)
- Поскольку для сверточного слоя нужно задать параметры только всех свёрток, что число параметров заметно меньше общего числа весов слоя
- Очень эффективная архитектура для распознавания изображений

# Подсчёт параметров



- Каковы размеры фильтров свёрток на разных слоях?
  - $5 \times 5 \times 1$  на первом слое
  - $5 \times 5 \times 6$  на втором свёрточном слое
  - «Глубина» тензора на выходе свёрточного слоя равна числу свёрток в свёрточном слое
  - 3е измерение свёртки равно «толщине» входного тензора
- Число весов и параметров второго слоя:
  - $\text{Параметров} = 16 \text{ свёрток } 5 \times 5 \times 6 = 150 \times 16 + 16 = 2416$
  - $\text{Весов} = (\text{примерно}) \text{Параметров} \times 10 \times 10 = 240000$

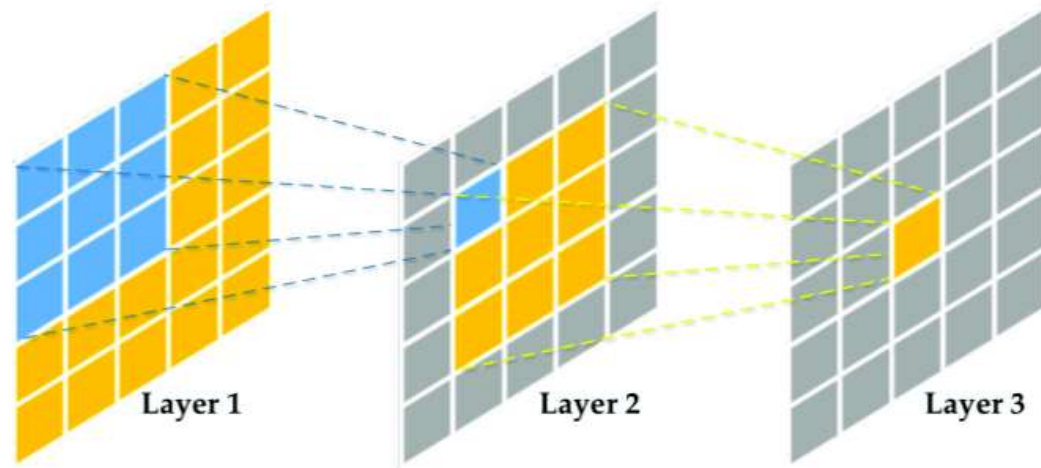
# Фильтры первого уровня



- Визуализируем веса фильтров
- Поскольку сворачиваем RGB изображение, то визуализация весов в RGB
- Обратите внимание на вычисление градиентов цветов



# Рецептивное поле (Receptive field)



- Receptive field нейрона – область изображения, от которой зависит выход этого нейрона
- Размер и положения поля определяется глубиной нейрона, размерами свёрток, размерами областей пулинга



## 4. Ключевой этап: модель AlexNet

# Large-scale visual recognition 2012

---



- LSVR – конкурс на базе датасета ImageNet
- AlexNet вышла победителем конкурса 2012 года
- Ошибка упала в 2 раза по сравнению с соперниками



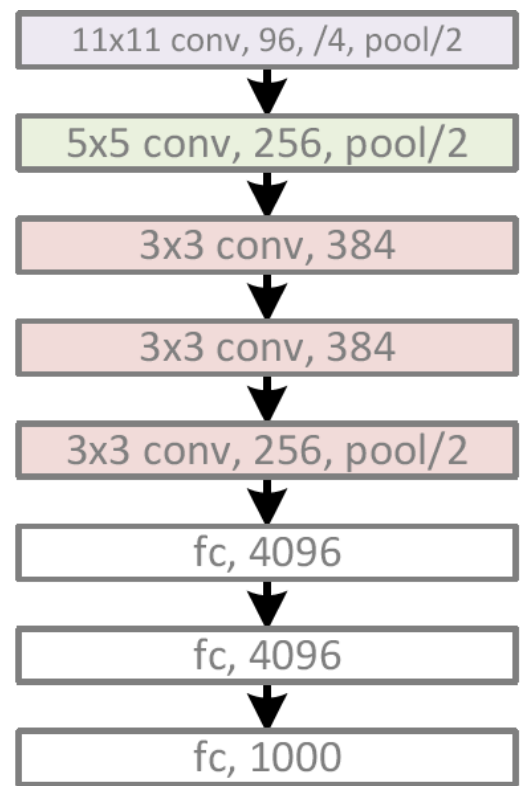
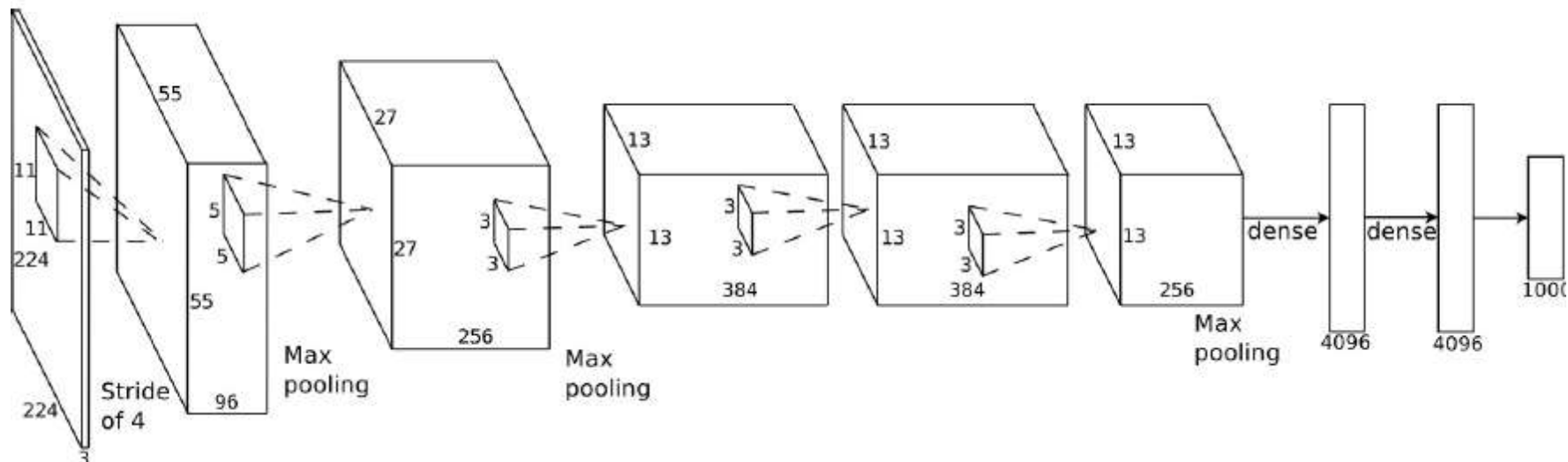
Car

**Winner**

SuperVision

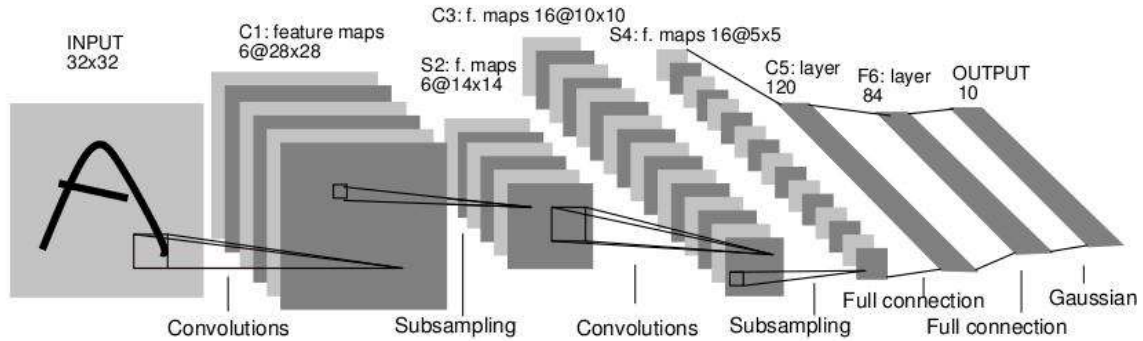
Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton  
University of Toronto

# SuperVision



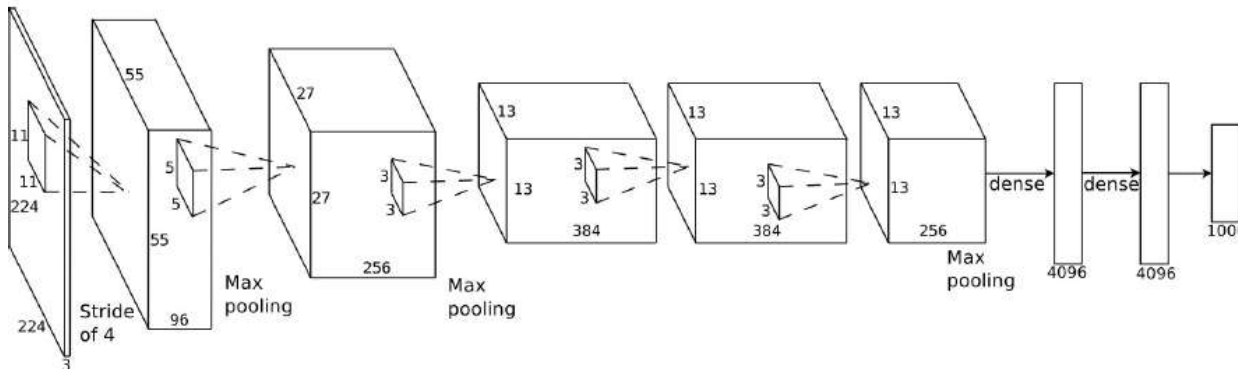
- 650,000 neurons
- 60,000,000 parameters
- 630,000,000 connections
- 1 машина, 2 GPU по 2Gb, 5GB Ram, 27Gb HDD, 1 неделя на обучение

# Сравнение LeNet и AlexNet



1998 год

- 2 свёрточных слоя (6 и 6 фильтров)
- 2 полносвязанных (120 и 84 нейрона)

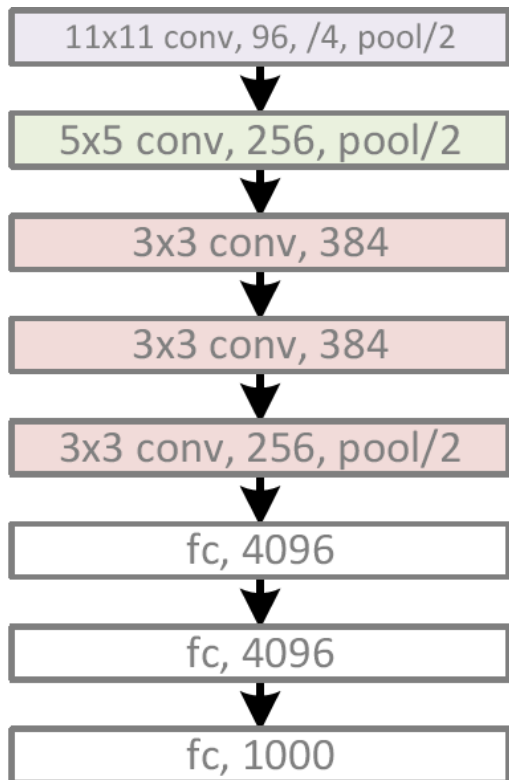


2012 год

- 5 свёрточных слоёв (96, 256, 384, 384, 256 фильтров)
- 2 полносвязанных (4096 и 4096 нейрона)

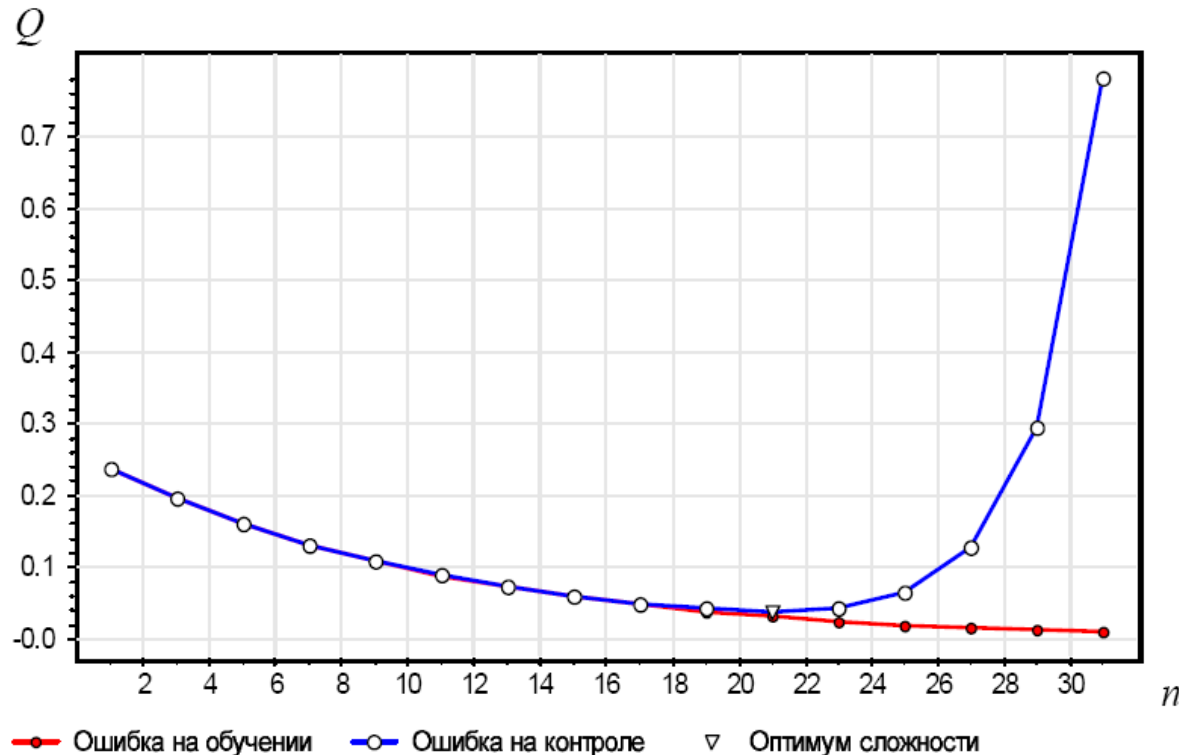
- Больше слоёв, фильтров, нейронов
- Какие ещё изменения произошли между LeNet и AlexNet?

# Важные замечания по AlexNet vs LeNet



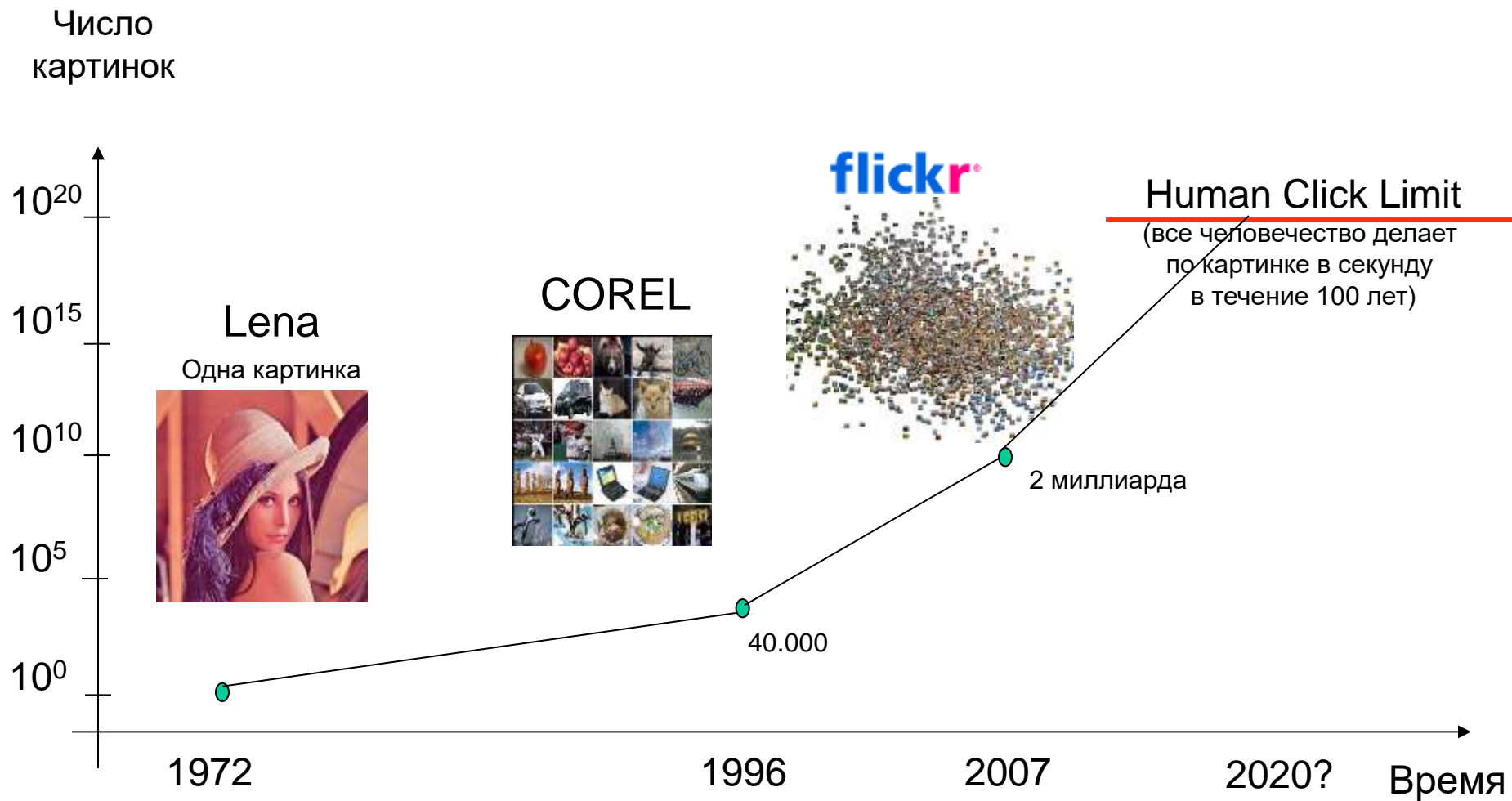
- Больше данных для обучения (ImageNet)
- Вычислительные мощности для обучения (GPU)
- Активация ReLU
- Аугментации изображений
- Регуляризация dropout

# Переобучение



- Чем сложнее задача – тем более сложная нейросеть нужна
- Но параметров нейросети очень много, и нейросеть быстро «переобучалась»
- Происходило «запоминание» всей обучающей выборки без её «обобщения»

# «Интернет-бум» + «Закон Мура»





# Функции активации



- **Tanh**

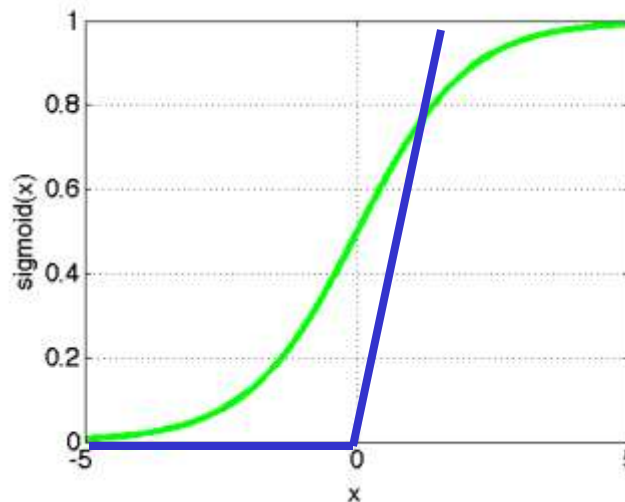
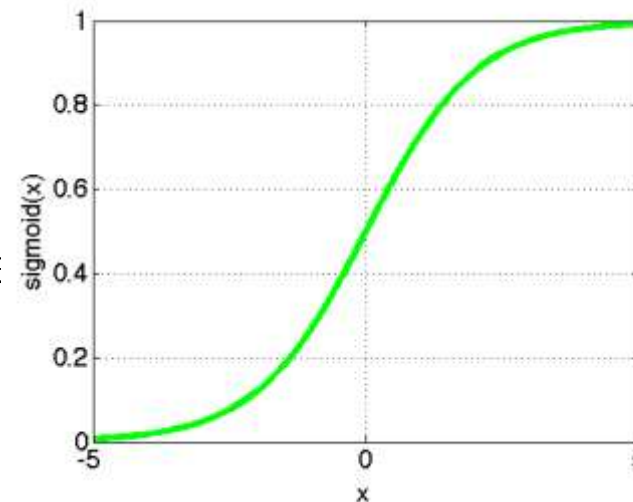
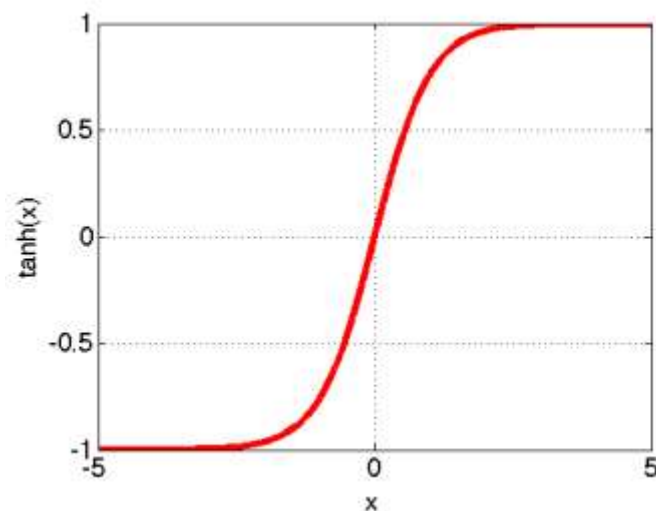
- $\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$

- **Sigmoid:**

- $\text{sigm}(x) = \frac{1}{1+e^{-x}}$  - преобразует все значения в интервал  $[0,1]$

- **Rectified linear**

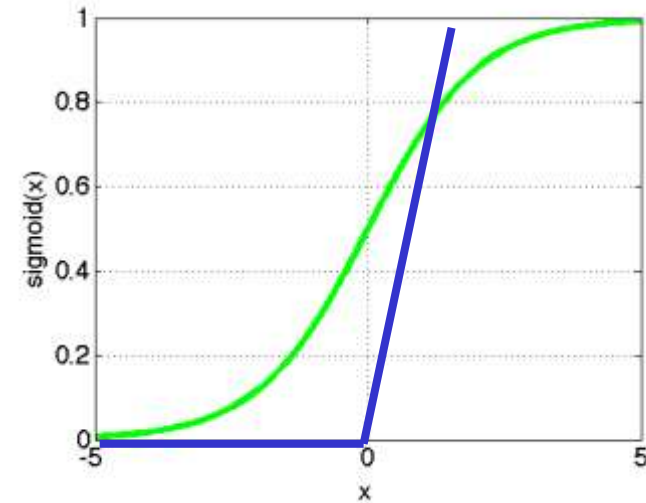
- $\text{ReLU}(x) = \max(0,x)$





# Проблемы обучения

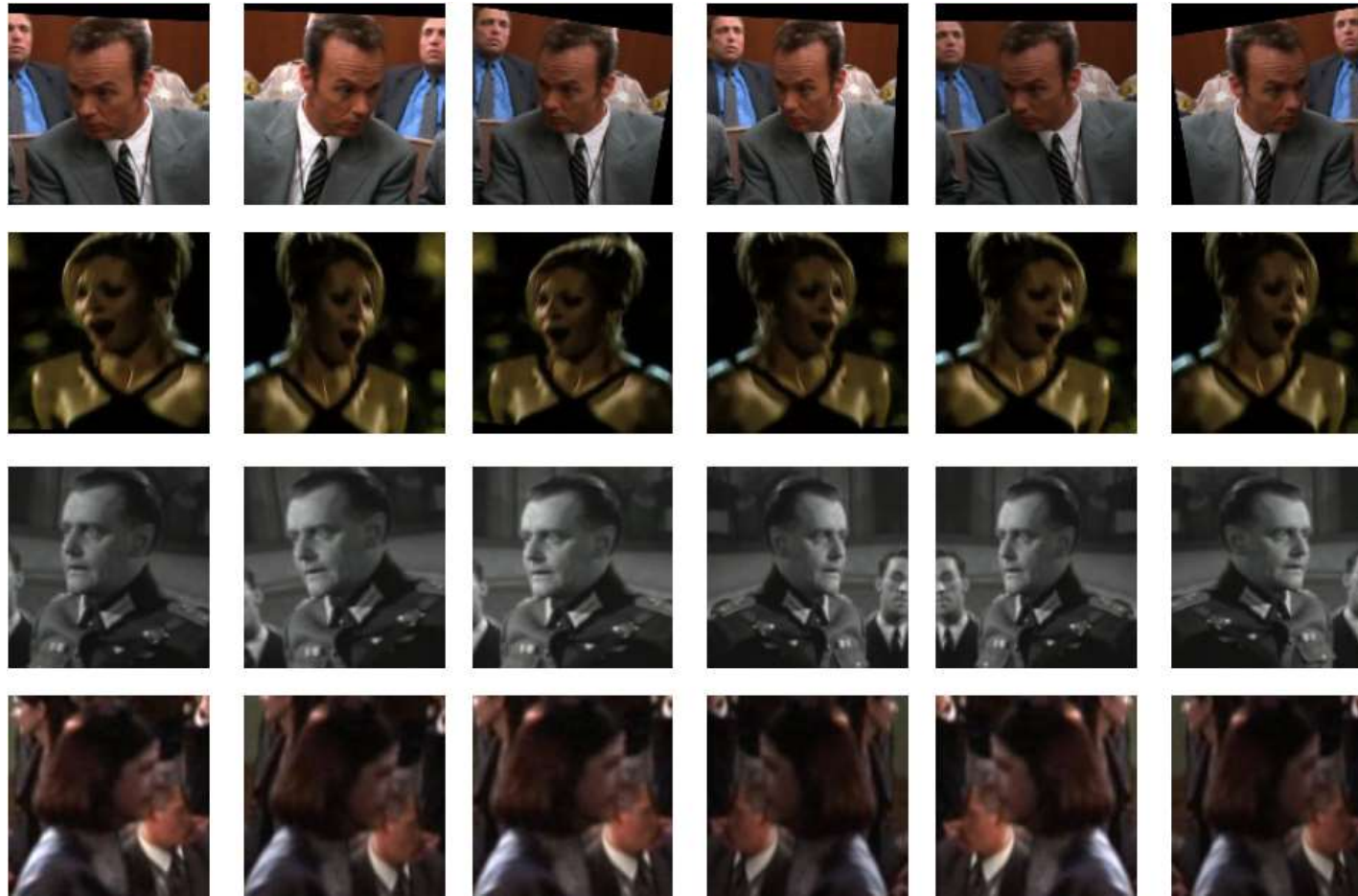
- «Затухание» (saturation) градиентов, если попадаем на область низкого градиента функции активации
  - Sigmoid очень страдает от этого
  - ReLU не страдает
- «Мёртвые» (dead) нейроны, которые, так получилось, получают только отрицательные или нулевые входы
- Исчезающие или взрывные градиенты (vanishing or exploding gradients) в глубоких сетях





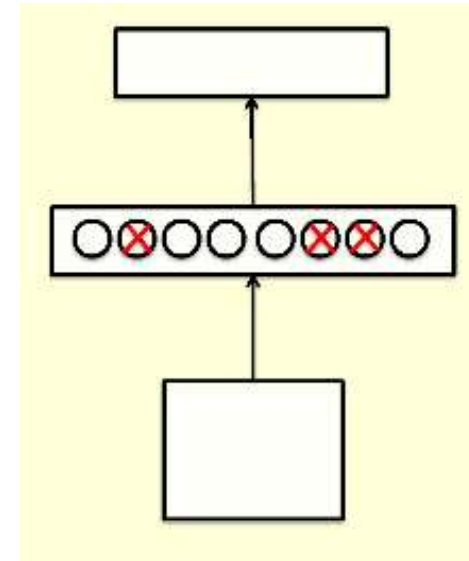
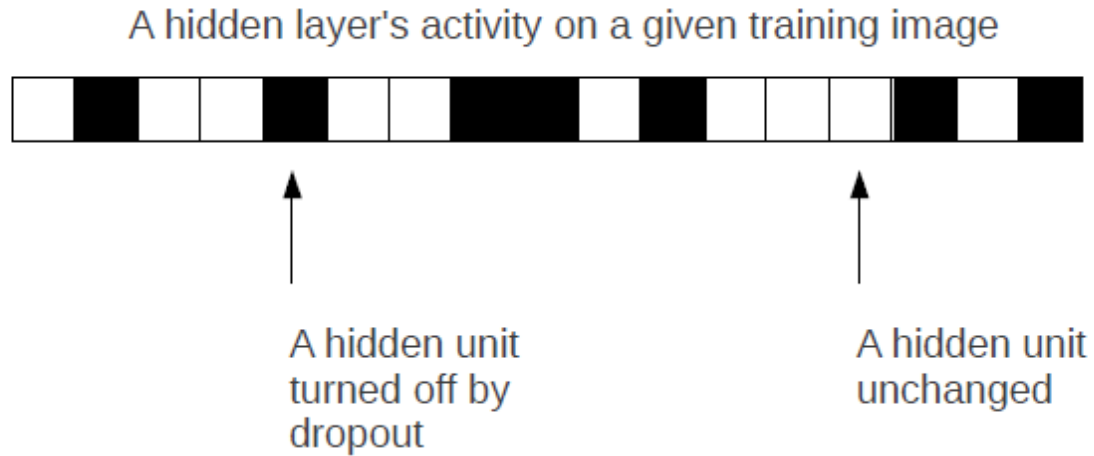
- «Data augmentation»
- Борьба с переобучением
- Из 256x256 случайно выбираем фрагменты 224x224 и их отражения
- Добавляем цветовые искажения

# Варианты размножения данных



Небольшие сдвиги, отображения, повороты,  
изменения масштаба

# Dropout



- Отключаем половину нейронов в каждом слое
- Получаем случайную выборку из множества сетей
- Во время тестирования используем «среднюю» сеть с уполовиненными весами

Nitish Srivastava Improving Neural Networks with Dropout.  
Master Thesis, 2013



- Рассмотрели постановку задачи классификации, примеры датасетов и как их собирать
- Основной способ решения задачи классификации изображений – свёрточные нейросети
- Концептуально нейросети остались такими же, как в 1990х, но было предложено множество относительно небольших изменений, которые в совокупности с ростом доступных данных позволили сети эффективно обучать