Искусственные нейронные сети

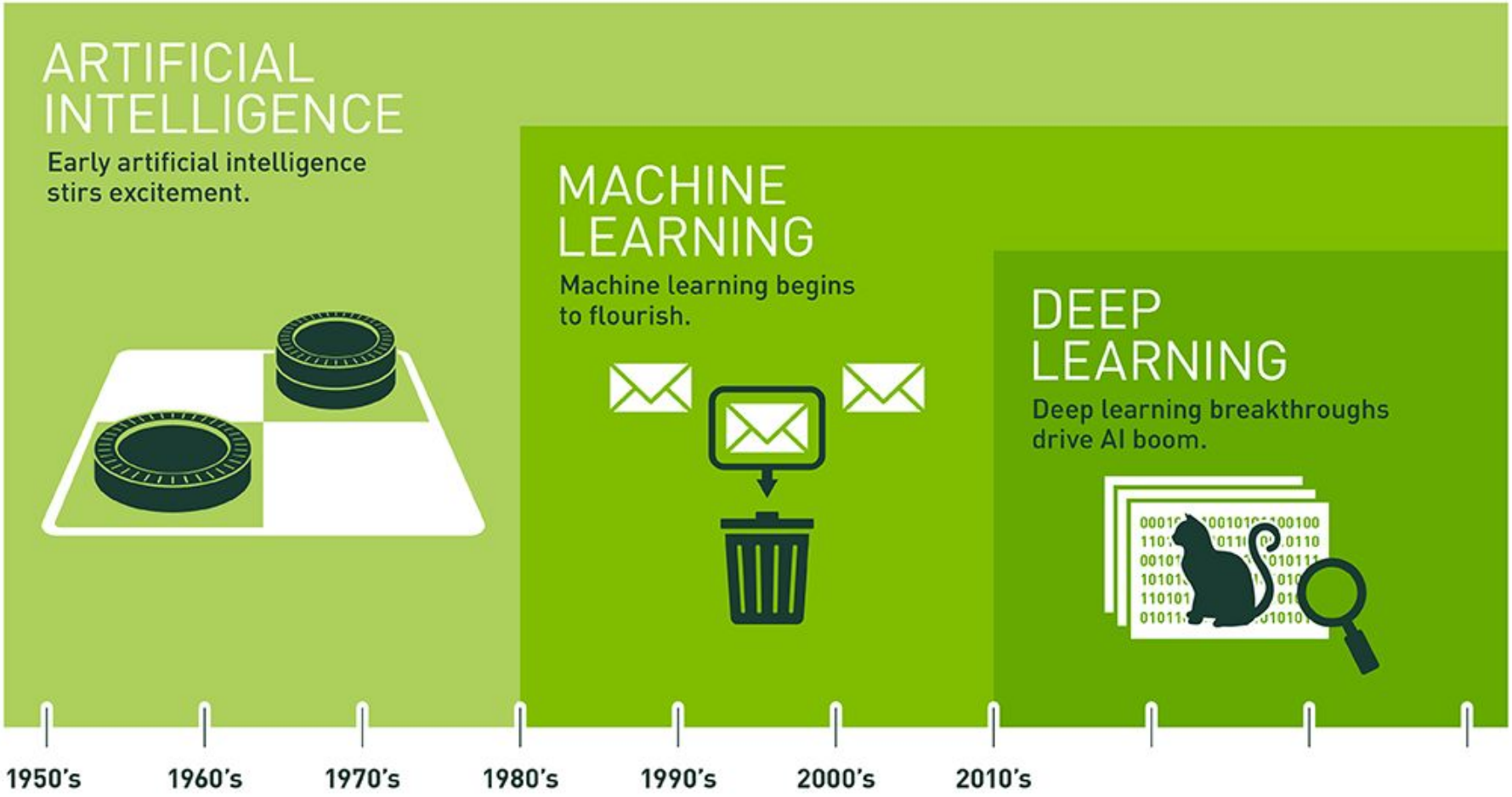
Сугоняев Андрей, группа 331

СПбГУ,
2018

Что такое нейронные сети

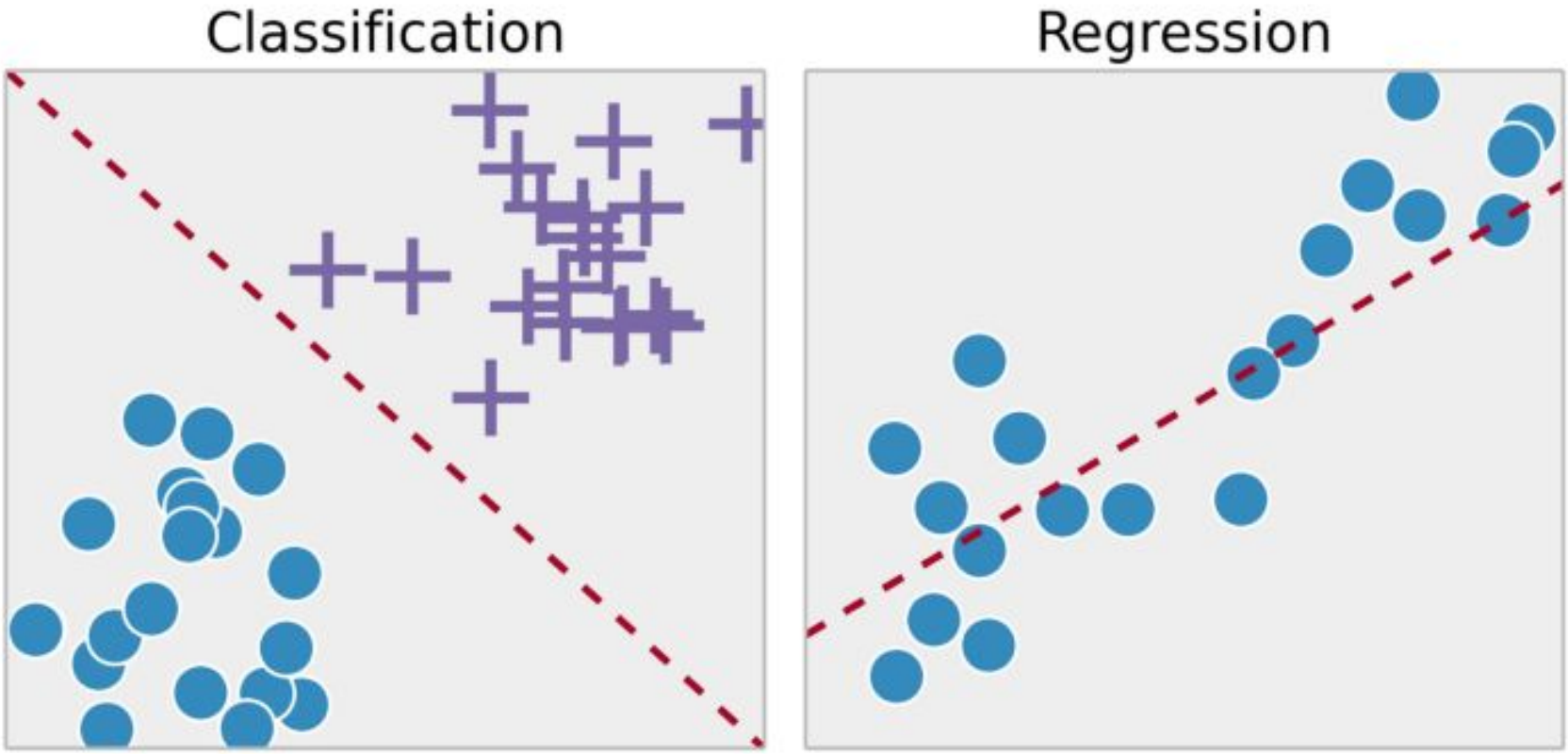
Искусственная нейронная сеть — математическая модель, построенная по принципу организации и функционирования биологических нейронных сетей.

Какое место занимают нейросети в Computer Science

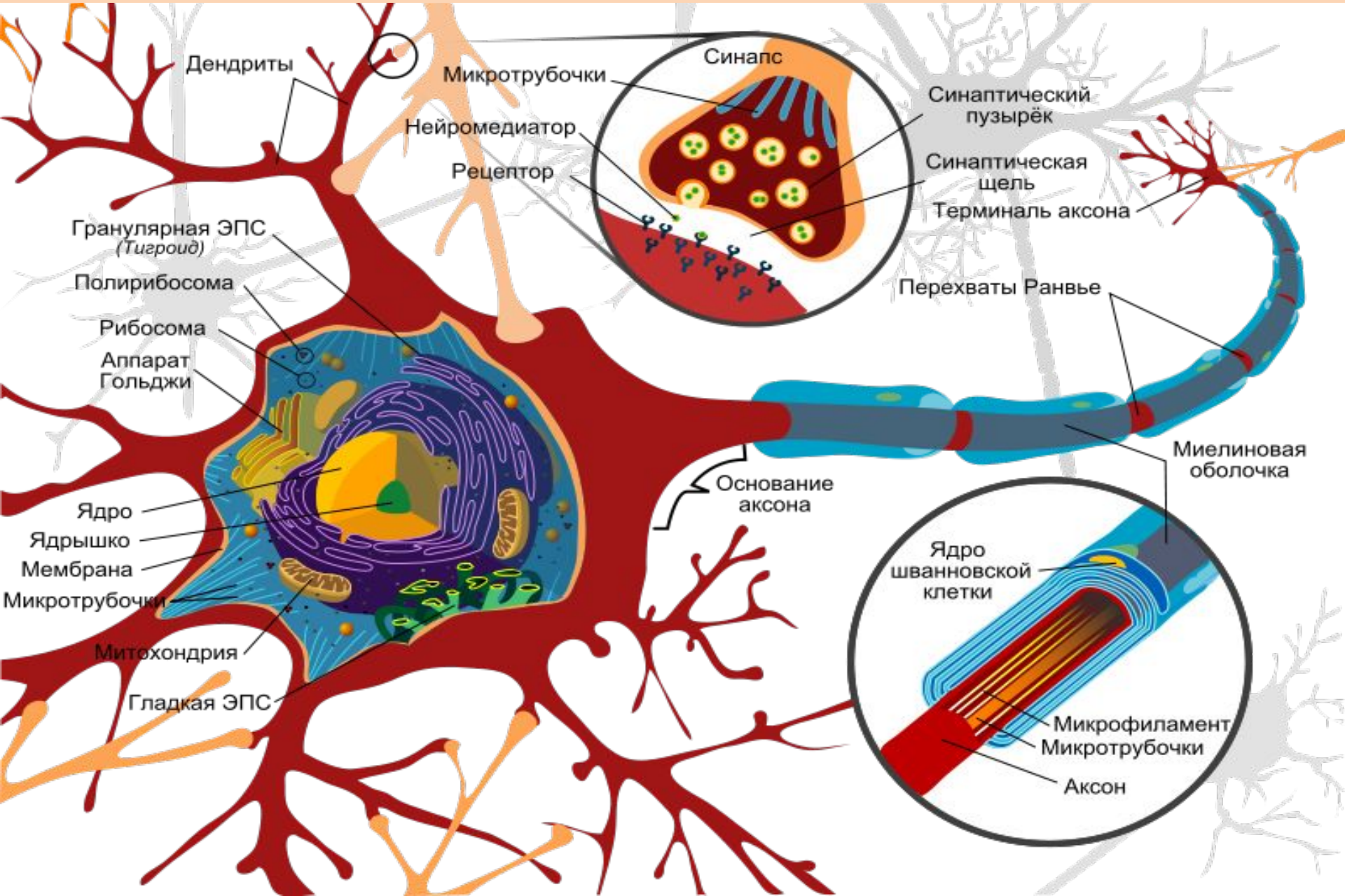


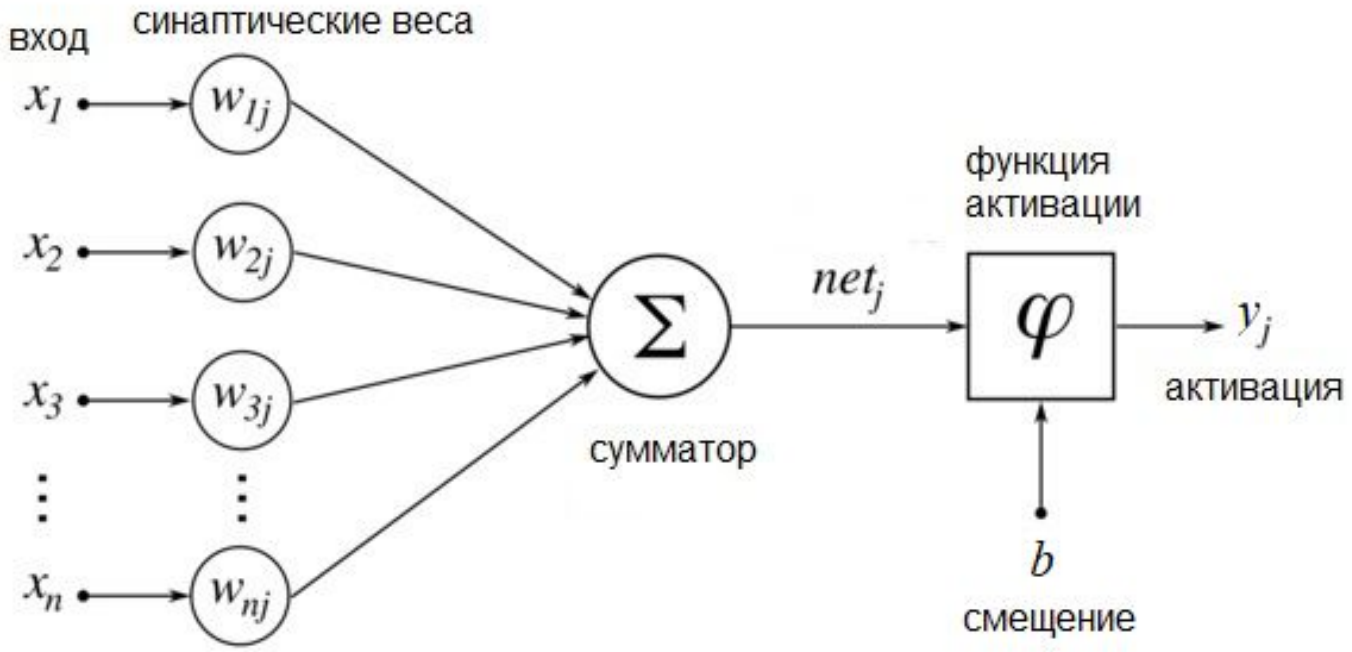
Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Нейронные сети принадлежат к классу алгоритмов, обучающихся с учителем (supervised learning), и решает типовые задачи этого класса:



Искусственный нейрон



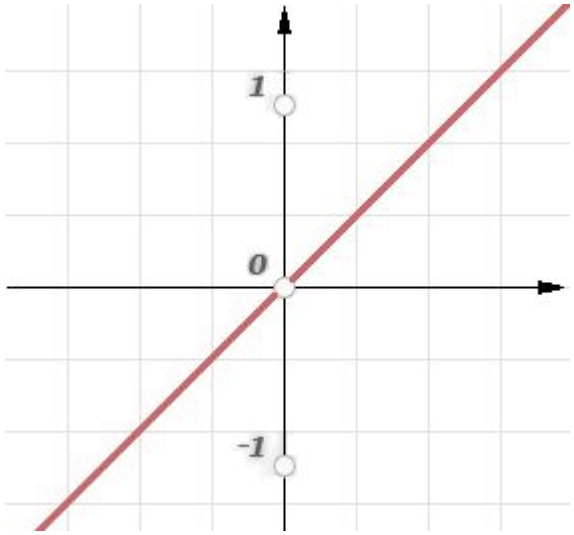


ИН – формализованная модель биологического нейрона, предложенная в 1943 году У. Маккалоком и У.Питтсом.

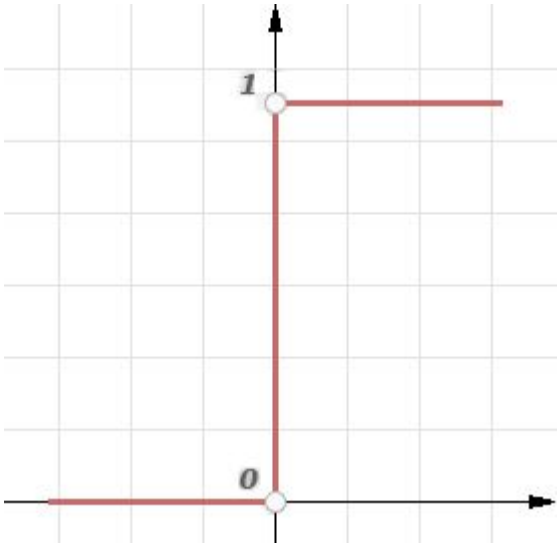
Он работает следующим образом:

- Нейрон получает на вход вектор $\boldsymbol{x}$
- Его компоненты умножаются на соответствующий вес и складываются. Также прибавляется смещение $\boldsymbol{b}$.
- К взвешенной сумме применяется функция активации.
- $Y = \varphi(\sum w_i x_i + b)$

Функции активации



- Линейная
  - Выходы сети являются линейными комбинациями входов

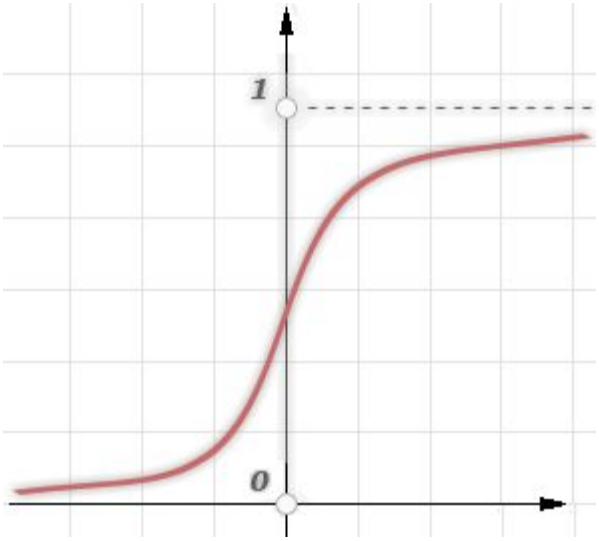


- Пороговая

  Эта функция использовалась в оригинальной модели ИН.

  \+ имеет центрированный аналог (sign x)

  – не дифференцируема

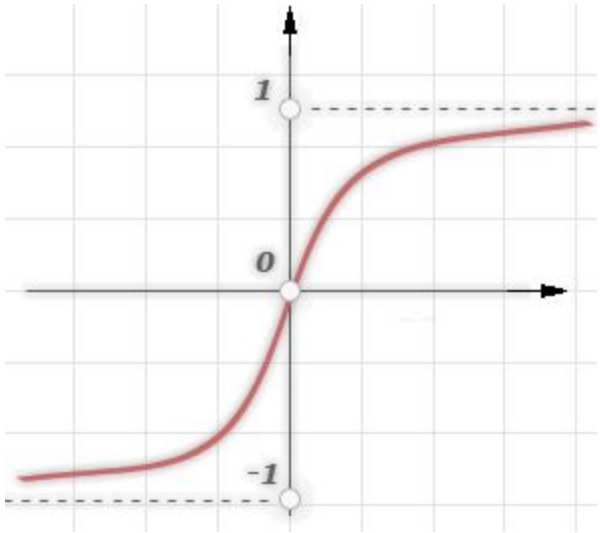


- Сигмоида $f(x) = \dfrac{1}{1+e^{-\alpha x}}$

  Долгое время считалась функцией, лучше всего описывающей работу нейрона.

  \+ дифференцируема

  – имеет порог насыщения

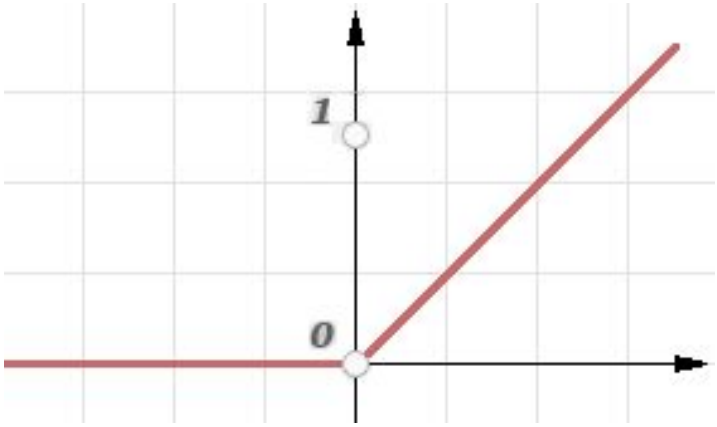


- Tanh $f(x) = \tanh\left(\dfrac{\alpha x}{2}\right) = \dfrac{1-e^{-\alpha x}}{1+e^{-\alpha x}}$

  \+ дифференцируема

  \+ центрована

  – имеет порог насыщения



- ReLU (rectified linear unit)

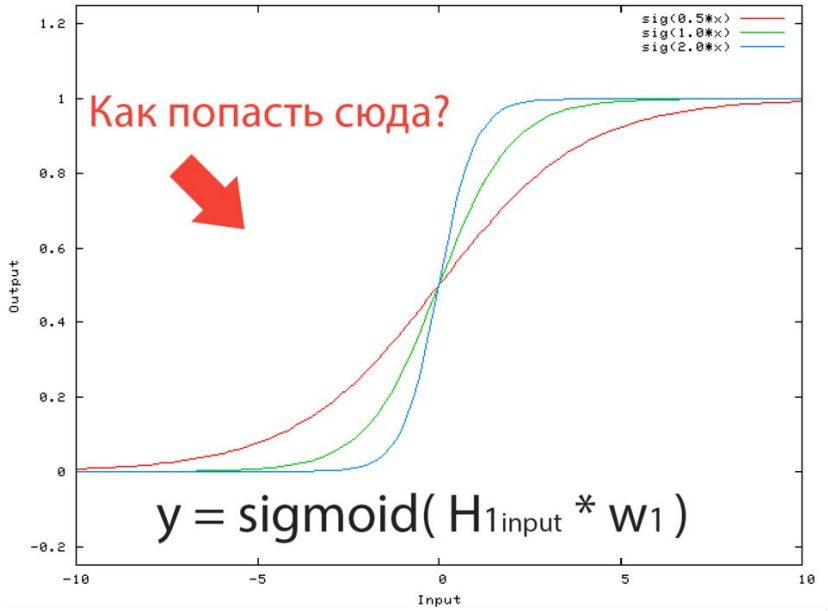
$$f(x) = \max(0, x)$$

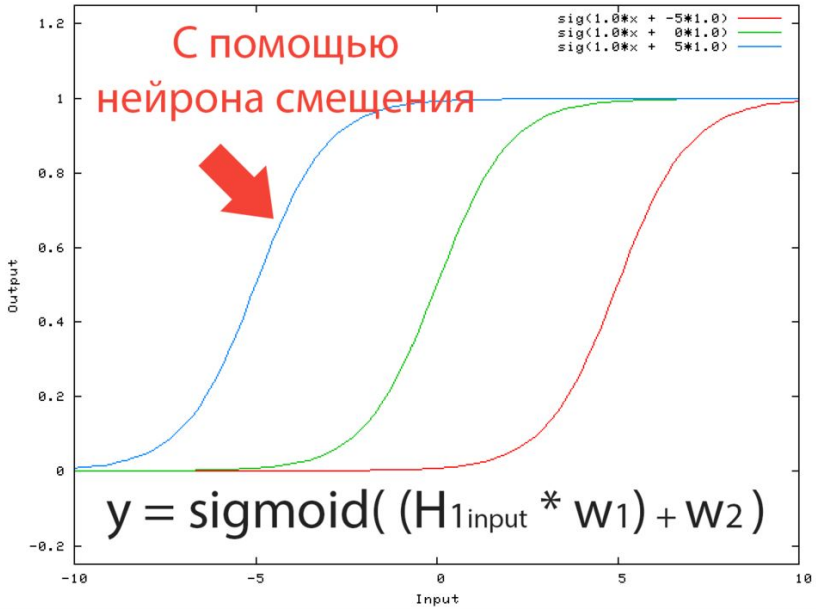
В настоящее время самая широко используемая функция активации в силу своей простоты.

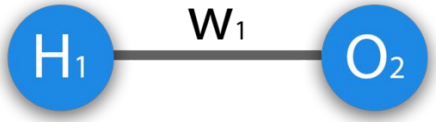
Также недавние исследования показывают, что она правильнее описывает работу биологических нейронов

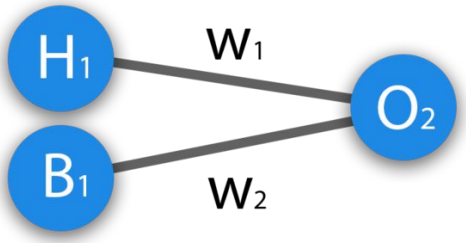
+ дифференцируема
+ не имеет порога насыщения
+ быстро вычисляется
– не центрована
– чувствительна к инициализации

Зачем нужно смещение





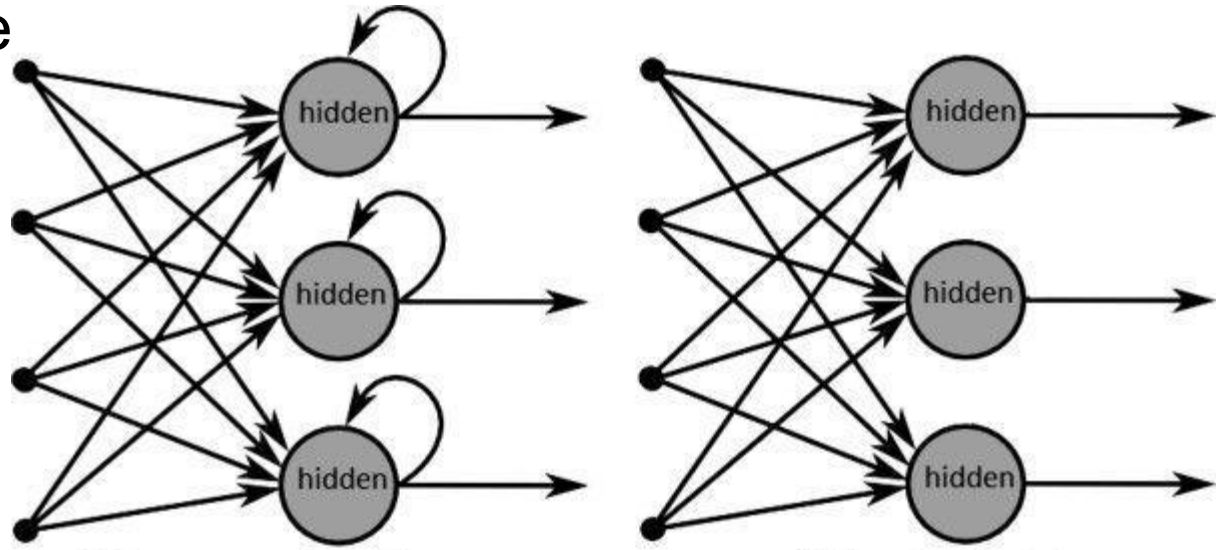




Замечание: сдвиг *b* можно также считать отдельным нейроном, на который всегда подается значение 1. Такой нейрон называется **нейроном смещения**.

Слои

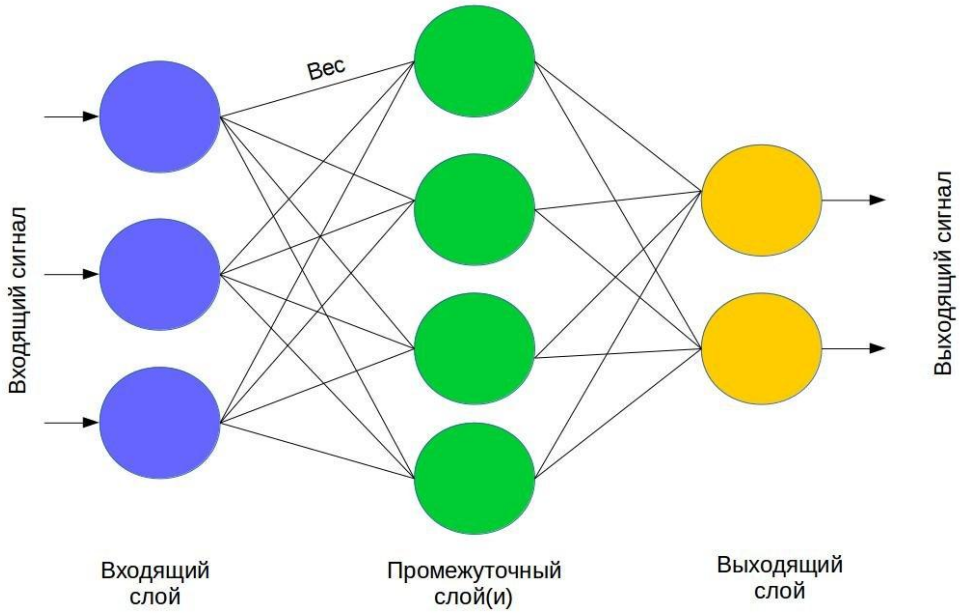
- Слой - совокупность нейронов сети, объединяемых по особенностям их функционирования. В плоскослоистых сетях это группа нейронов, имеющих один и тот же набор входов и не соединенных между собой.
- По виду связи между слоями сети делят на
  - Сети прямого распространения (FFNN)
  - Ре



(a) Recurrent neural network

(b) Forward neural network

Персептрон



$$y = f(\sum w_2 f(\sum w_1 x))$$

Перцептрон - одна из первых моделей нейронных сетей, предложенная в 1957 году Ф. Розенблаттом как средство решения задач классификации.

Формальное определение задачи классификации

- Имеется множество *объектов*, разделённых некоторым образом на *классы*. Задано конечное множество объектов, для которых известно, к каким классам они относятся (*выборка)*. Классовая принадлежность остальных объектов неизвестна. Требуется построить алгоритм, способный *классифицировать* произвольный объект из исходного множества.
- Построить алгоритм, который по признаковому описанию объекта (вектору $\boldsymbol{x} = (x_1, \ldots, x_n)$) правильно определит метку класса.

Разделяющая гиперплоскость

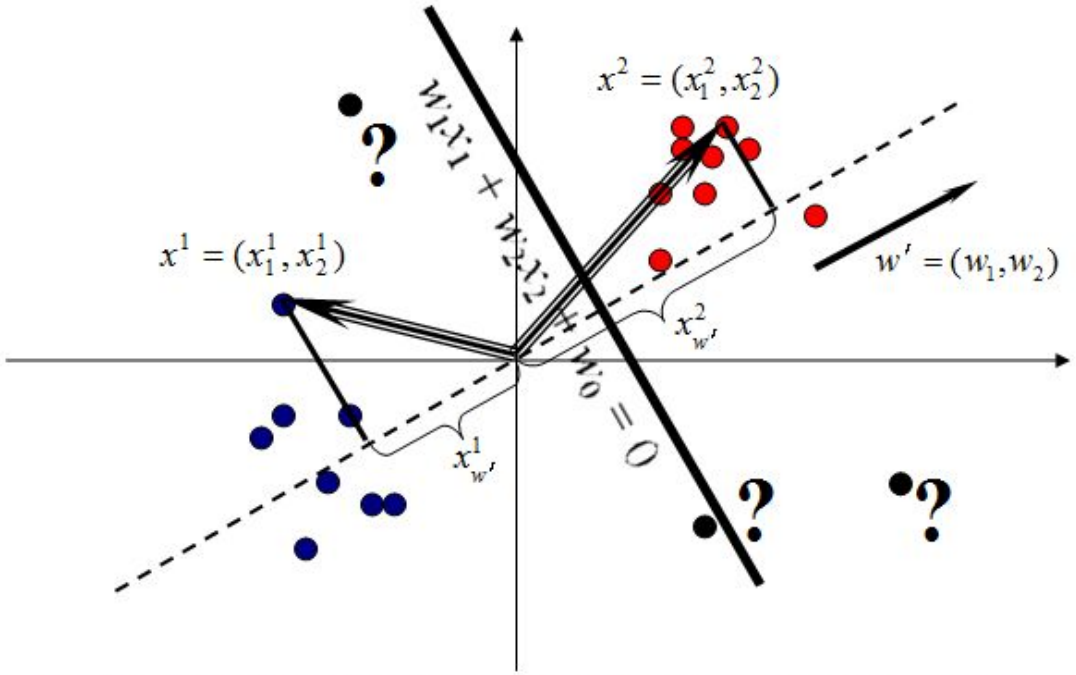


Рис. 3. Геометрическая интерпретация задачи разделения элементов множества на два класса в двумерном пространстве.

В задаче классификации однослойный персептрон строит в R^n гиперплоскость (или поверхность, если функция активации нелинейна), разделяющую объекты на 2 класса.

Булевы функции

- Как пример задачи классификации рассмотрим булевы функции, в которых признаковому описанию, состоящему из значений двух булевых переменных, сопоставляется метка класса – «истина» или «ложь».

Персептроны, реализующие булевы функции

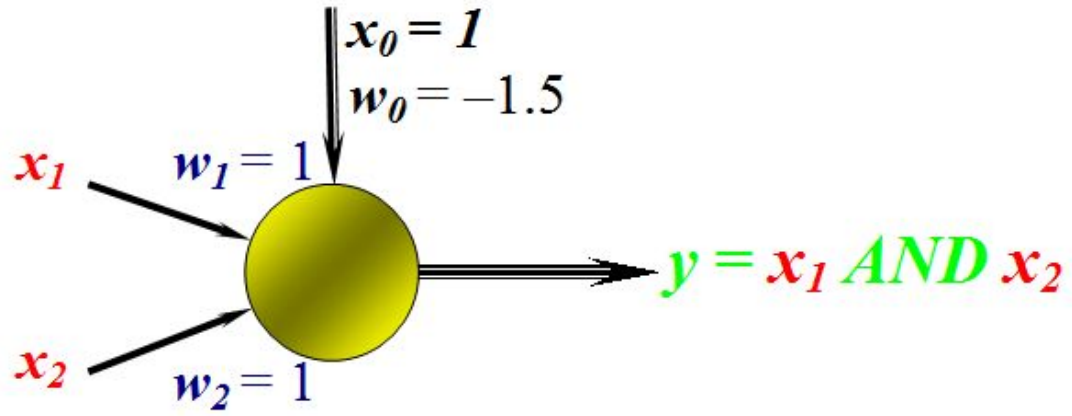


Рис. 6. Однослойный персептрон реализующий булеву функцию AND.

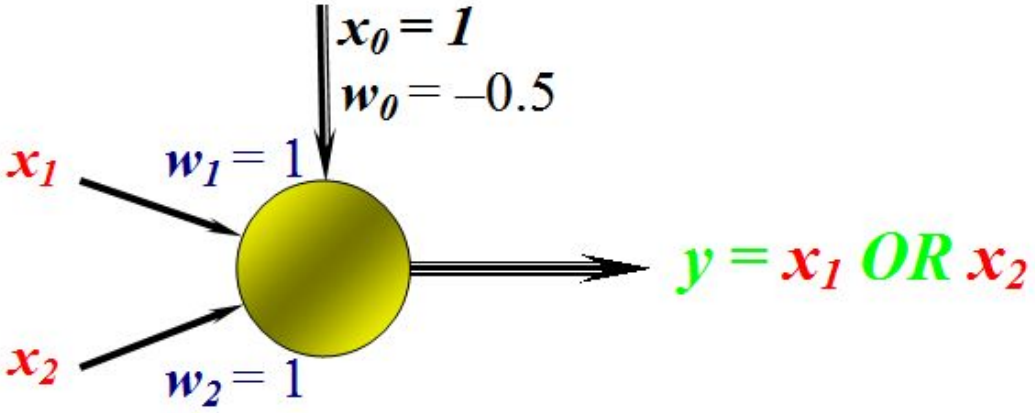


Рис. 8. Однослойный персептрон реализующий булеву функцию OR.

Соответствующие разделяющие гиперплоскости

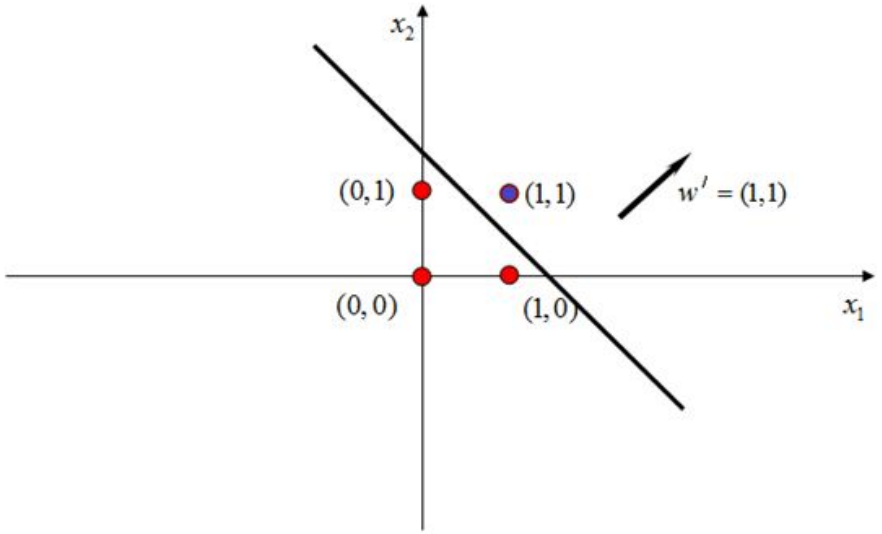


Рис. 5. Гиперплоскость $x_1 + x_2 = 1.5$ реализующая булеву функцию AND.

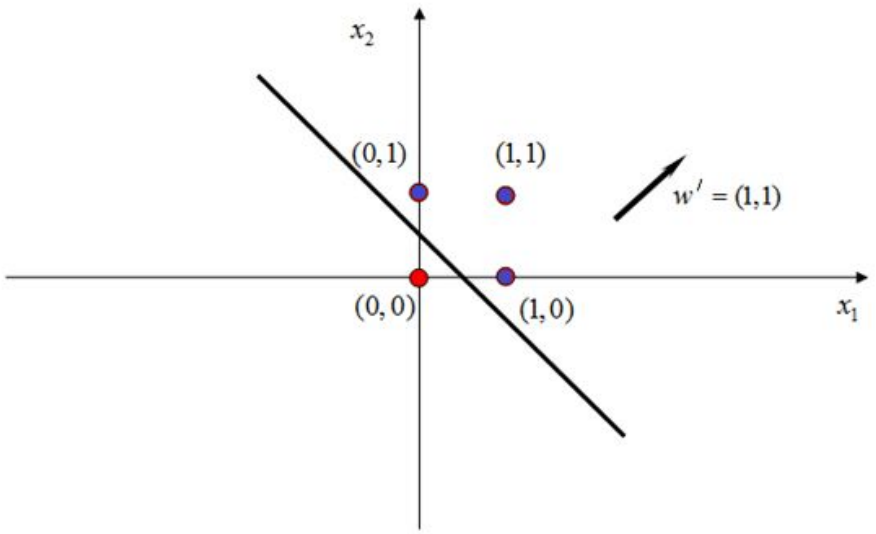
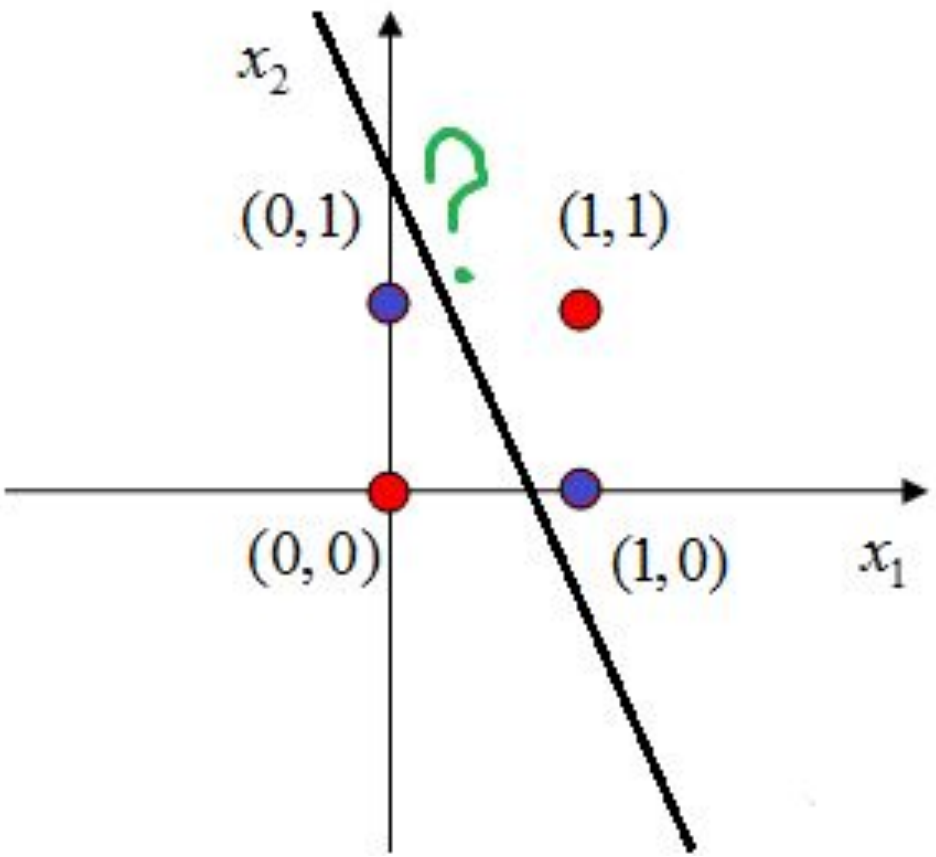


Рис. 7. Гиперплоскость $x_1 + x_2 = 0.5$ реализующая булеву функцию OR.

Проблема XOR



- Научное сообщество на долгое время потеряла интерес к нейронным сетям после выхода в 1969 году статьи Марвина Минского и Сеймура Паперта, в которой утверждалось, что персептрон не способен обучиться функции XOR.

Решение проблемы

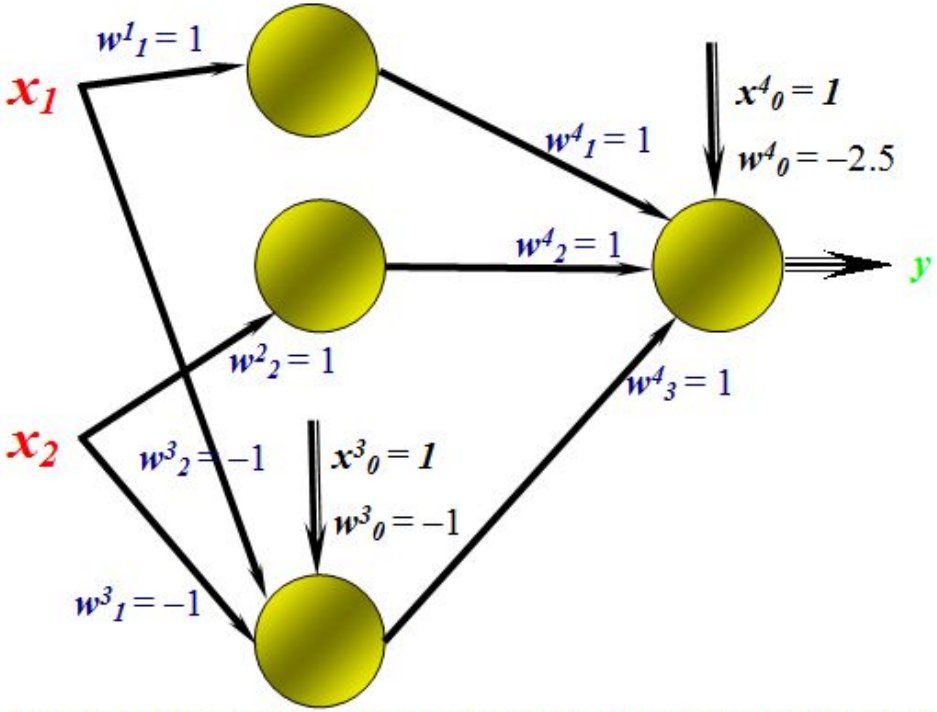


Рис. 12. Двухслойная сеть выделяющая область треугольной формы.

Для упрощения схемы нулевые входы и веса опущены.

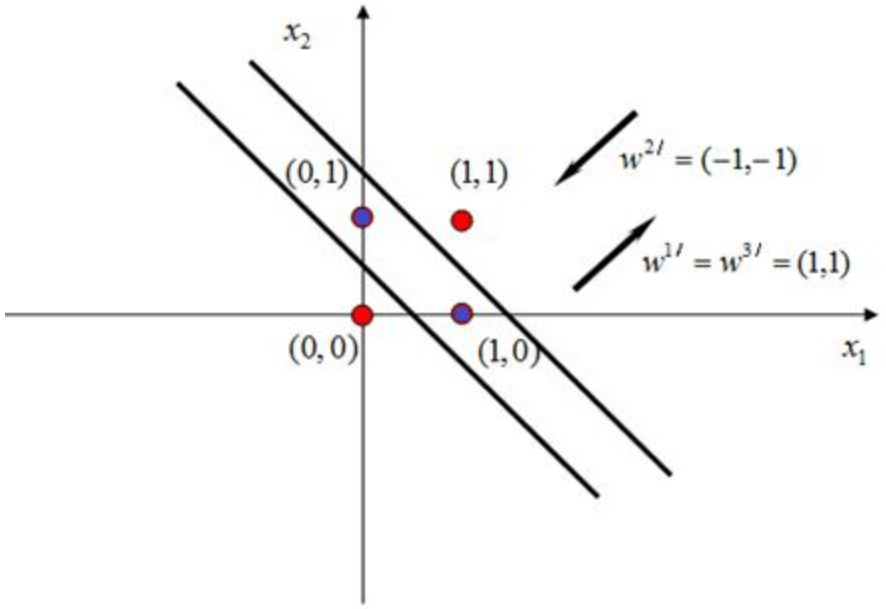


Рис. 9. Гиперплоскости реализующие функцию XOR.

Теорема Колмогорова-Арнольда

Любая непрерывная функция любого количества переменных представляется в виде суперпозиции непрерывных функций одной и двух переменных (и, более того, что в таком представлении можно обойтись, в дополнение к непрерывным функциям одной переменной, единственной функцией двух переменных — сложением):

$$f(x_1,\ldots,x_n)=\sum_{q=0}^{2n}\Phi_q\left(\sum_{p=1}^{n}\psi_{q,p}(x_p)\right)$$

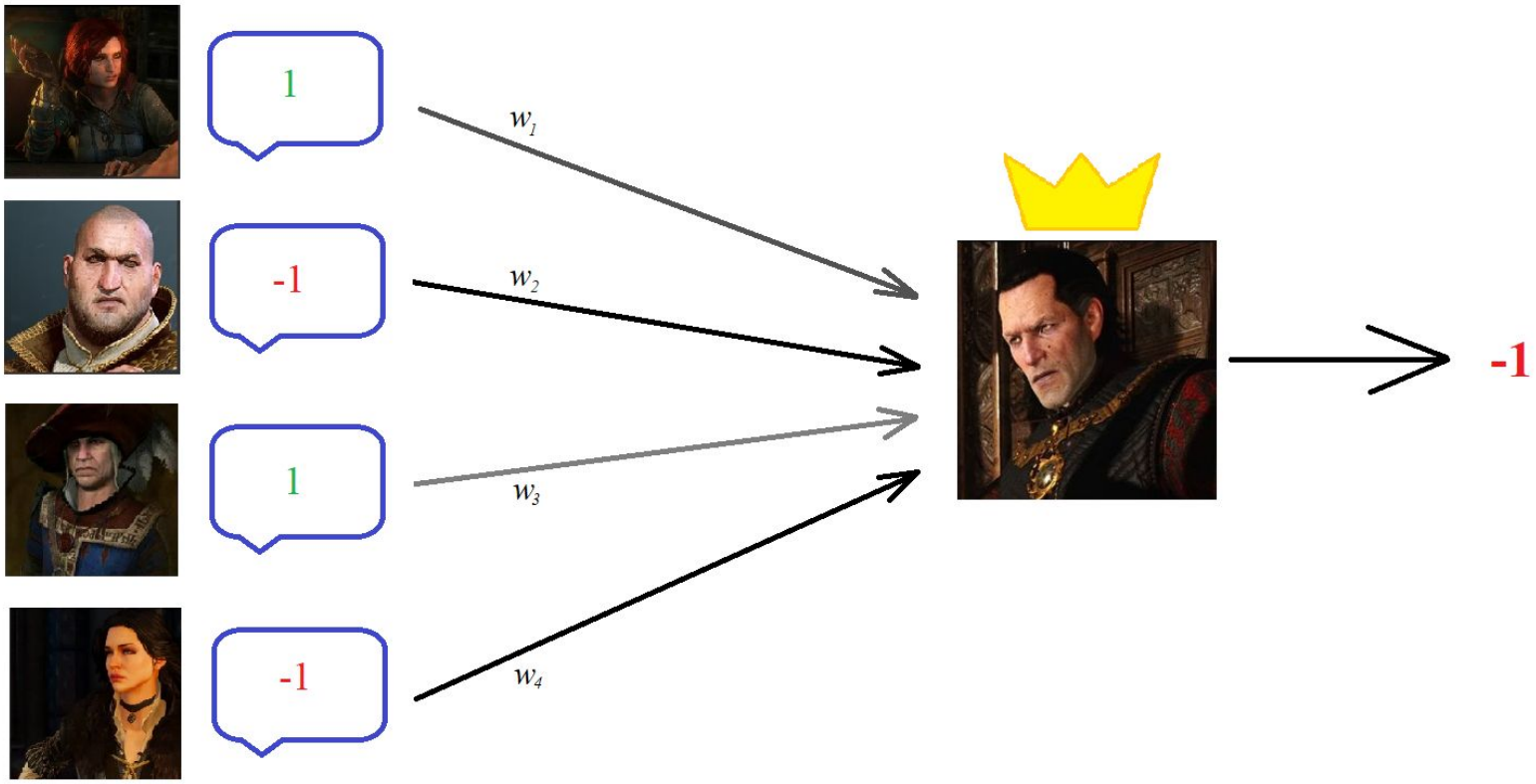
Обучение сети

- Наиболее распространенный метод обучения нейронной сети – **метод обратного распространения ошибки**.

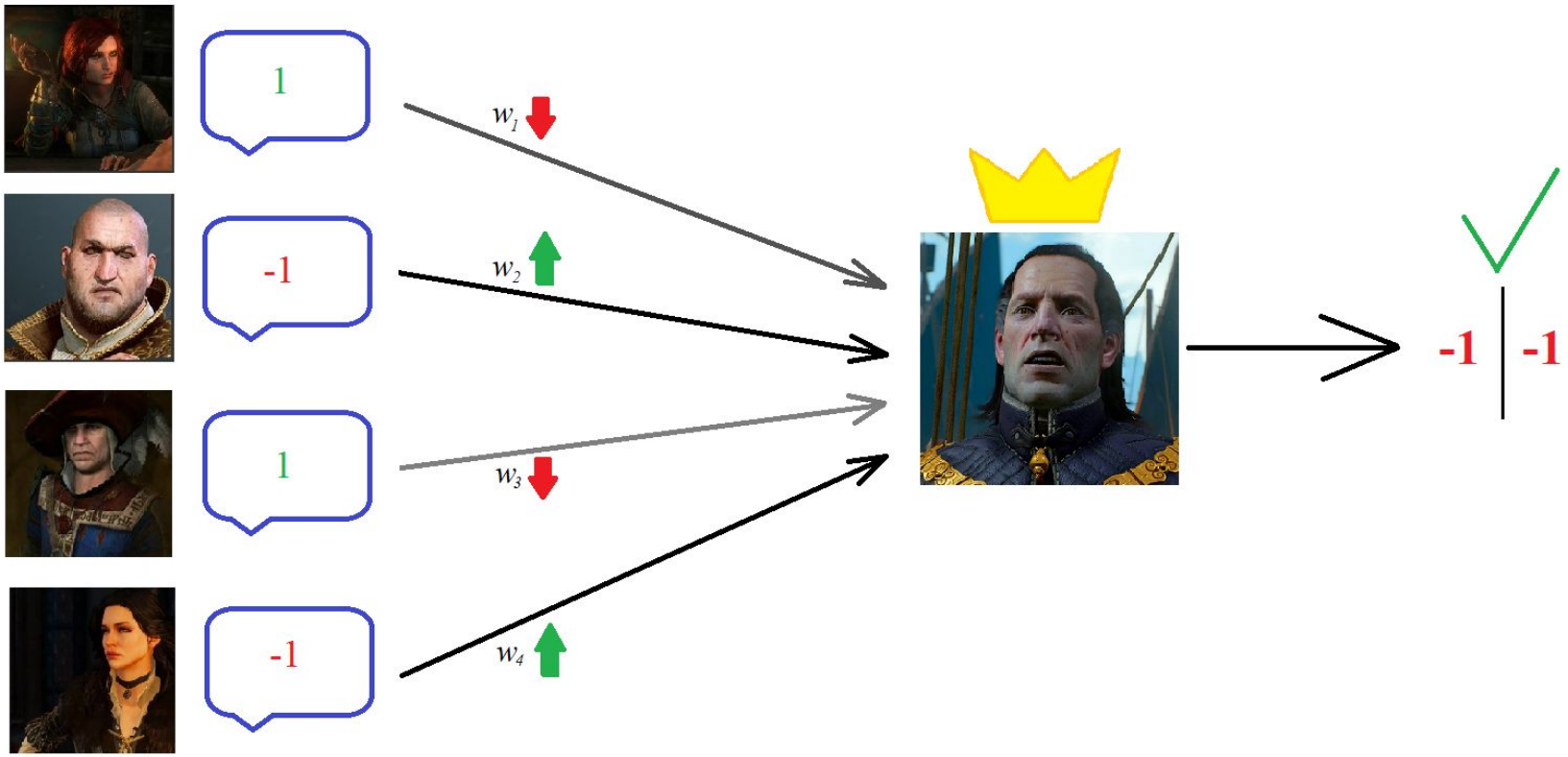
  Он был впервые описан в 1974 г. А.И. Галушкиным.

- Основная идея этого метода состоит в распространении сигналов ошибки от выходов сети к её входам, в направлении, обратном прямому распространению сигналов в обычном режиме работы.

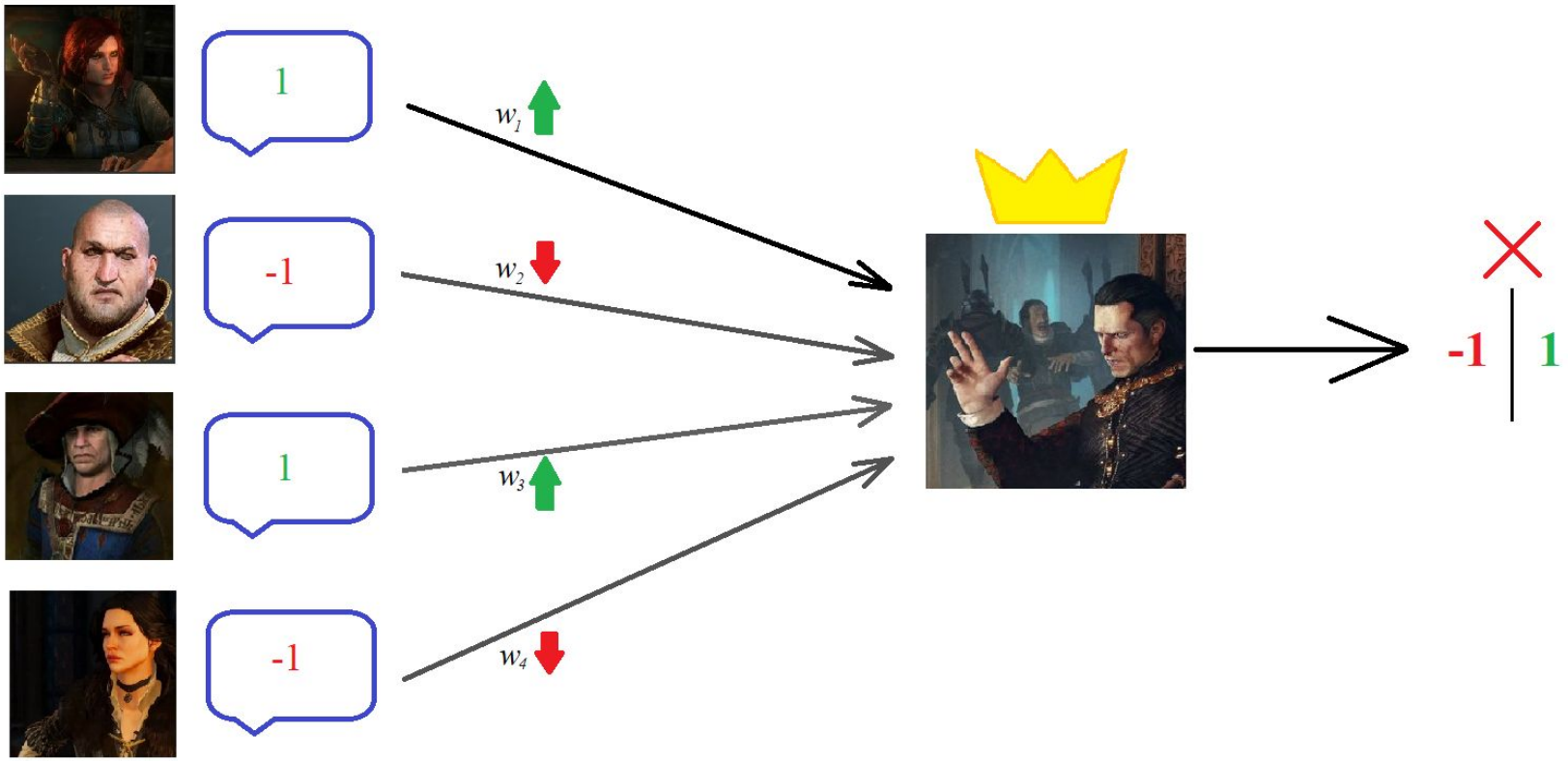
Аналогия для понимания
(дельта-правило)



Аналогия для понимания
(дельта-правило)

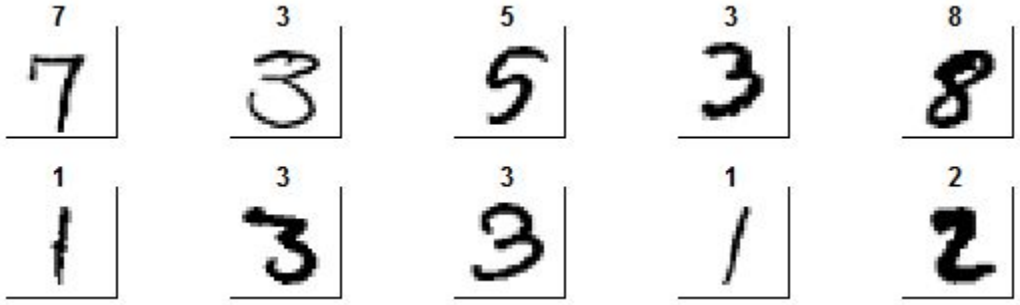


Аналогия для понимания
(дельта-правило)



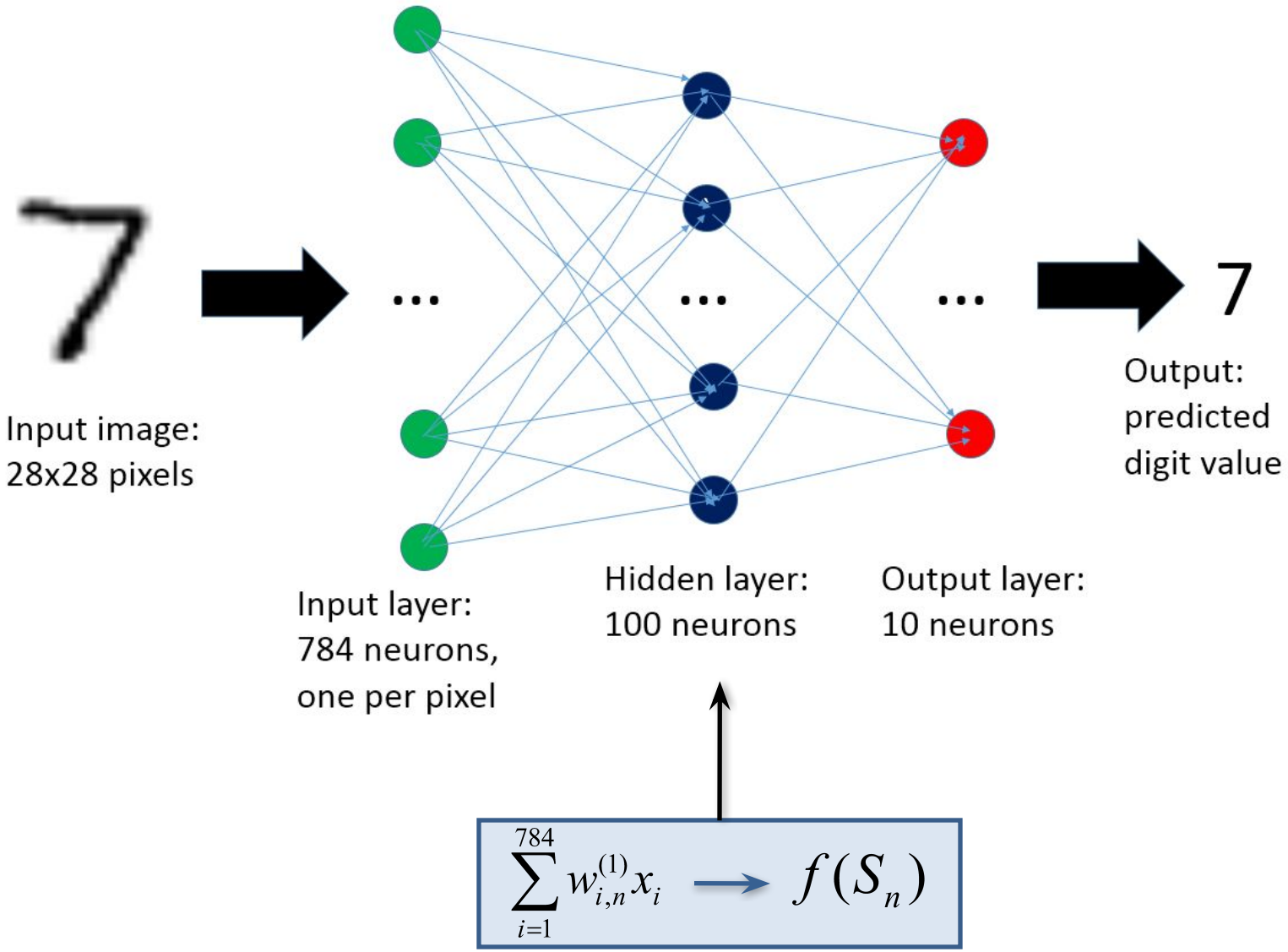
Обучающая выборка

$$D \left\{ \begin{array}{|c|c|} \hline x_1 & \tilde{y}_1 \\ \hline x_2 & \tilde{y}_2 \\ \hline \vdots & \\ \hline x_N & \tilde{y}_N \\ \hline \end{array} \right.$$



Выборка – набор размеченных входных векторов (т. е. таких, для которых известен правильный ответ), по которому производится настройка сети.

Прямой ход


7
Input image:
28x28 pixels
Input layer:
784 neurons,
one per pixel
Hidden layer:
100 neurons
Output layer:
10 neurons
7
Output:
predicted
digit value
$\sum_{i=1}^{784} w_{i,n}^{(1)} x_i \rightarrow f(S_n)$

Функция потерь

Функция потерь — функция, по значению которой можно оценить работу сети.

Две наиболее часто используемых функции потерь:

– среднеквадратичная ошибка (MSE):

$$E = \frac{1}{2}\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2$$

– логистическая (log loss):

$$E = -\frac{1}{N}\sum_{i=1}^{N}(\tilde{y}_i \cdot \log(y_i) + (1 - \tilde{y}_i) \cdot \log(1 - y_i))$$

Обратный ход

Будем минимизировать функцию потерь

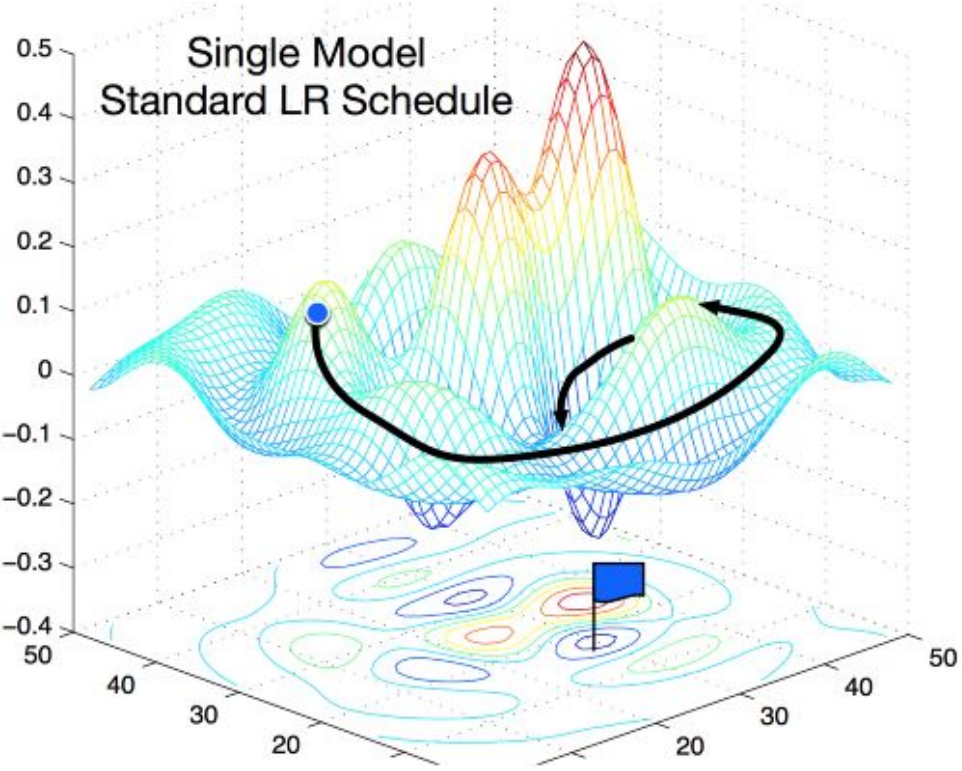
$$E = \frac{1}{2}\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2$$

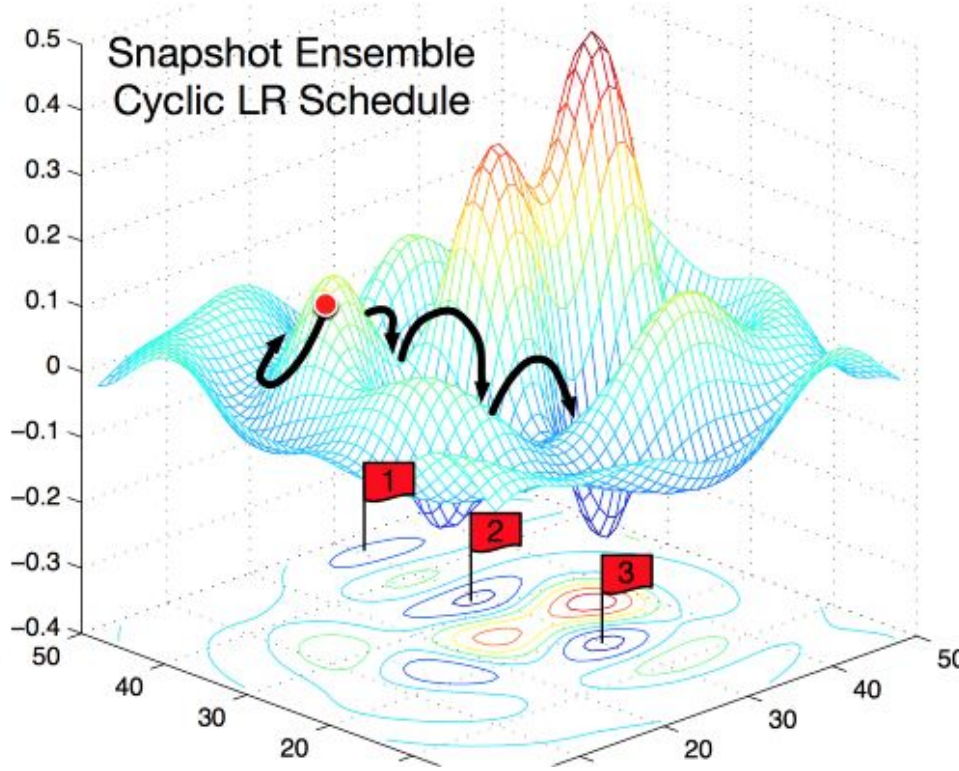
методом стохастического градиентного спуска

$$w_{ij} = w_{ij} + \Delta w_{ij}$$

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}$$

«Спуск» по поверхности ошибки





Гиперпараметры

η - шаг обучения. Он является гиперпараметром, то есть он настраиваются вручную до начала обучения.

$$0<\eta<1$$

Также гиперпараметрами являются глубина сети (количество слоев), ширина слоев, количество эпох обучения.

Рассмотрим, как будут меняться веса перед выходным слоем.

w_{ij} влияет на выход сети только как часть суммы

$$S_j = \sum_i w_{ij} x_i$$

Поэтому

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial S_j} \frac{\partial S_j}{\partial w_{ij}} = x_i \frac{\partial E}{\partial S_j}$$

S_j влияет на общую ошибку только в рамках выхода j-го узла y_j (являющегося выходом сети):

$$y_j = f(S_j)$$

Поэтому

$$\frac{\partial E}{\partial S_j} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j} = \left(\frac{\partial}{\partial y_j}\frac{1}{2}\sum_k (y_k - \tilde{y}_k)^2\right)\left(\frac{\partial f(S)}{\partial S}\Big|_{S=S_j}\right) =$$

$$= \left(\frac{1}{2}\frac{\partial}{\partial y_j}(y_j - \tilde{y}_j)^2\right) f'(S_j) = (y_j - \tilde{y}_j) f'(S_j)$$

Следовательно

$$\Delta w_{ij} = -\eta x_i (y_j - \tilde{y}_j) f'(S_j)$$

Рассмотрим теперь, как будут меняться веса между скрытыми слоями.

S_j влияет на выход сети через всех «детей» j-того нейрона.

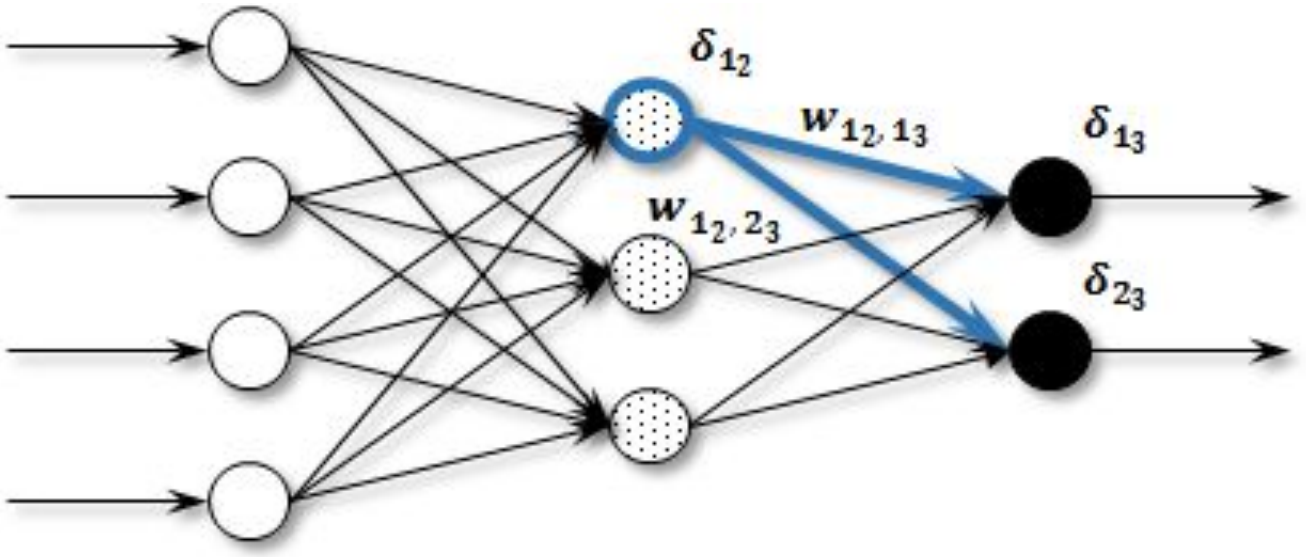
$$\frac{\partial E}{\partial S_j} = \sum_{k \in дети(j)} \frac{\partial E}{\partial S_k} \frac{\partial S_k}{\partial S_j}$$

и

$$\frac{\partial S_k}{\partial S_j} = \frac{\partial S_k}{\partial y_j} \frac{\partial y_j}{\partial S_j} = w_{jk} \frac{\partial y_j}{\partial S_j} = w_{jk} f'(S_j)$$

а $\frac{\partial E}{\partial S_k}$ - это аналогичная поправка, но вычисленная для *k*-того нейрона следующего слоя.

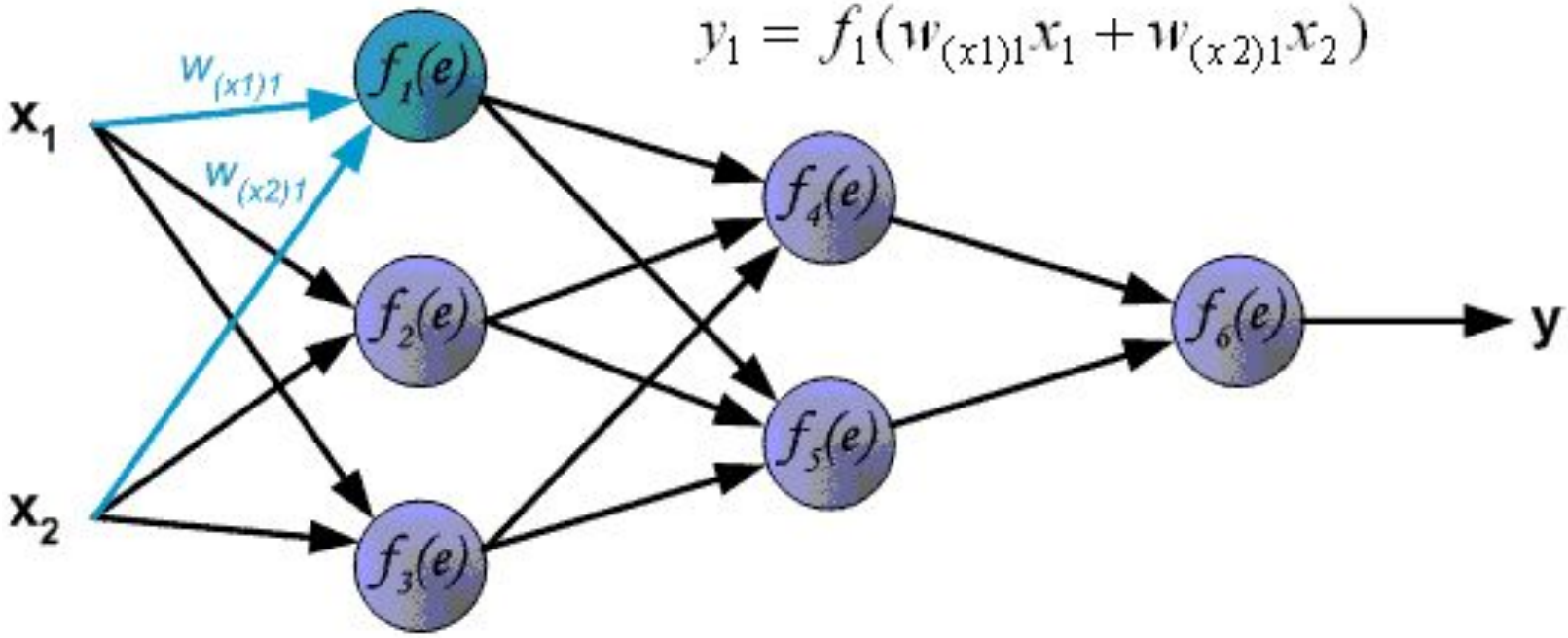
Итого:



- Для последнего слоя: $\delta_j = (y_j - \tilde{y}_j) f'(S_j)$
- Для внутренних слоев: $\delta_j = \left(\sum_k \delta_k w_{jk} \right) f'(S_j)$
- Для всех: $\Delta w_{ij} = -\eta \delta_j y_i$

$$w_{ij} = w_{ij} + \Delta w_{ij}$$

FP



Проблемы обучения

- Паралич сети – сеть перестает обучаться.
- Переобучение
- Недообучение

Причины:

- затухающий градиент
- взрывающийся градиент
- неправильный выбор гиперпараметров

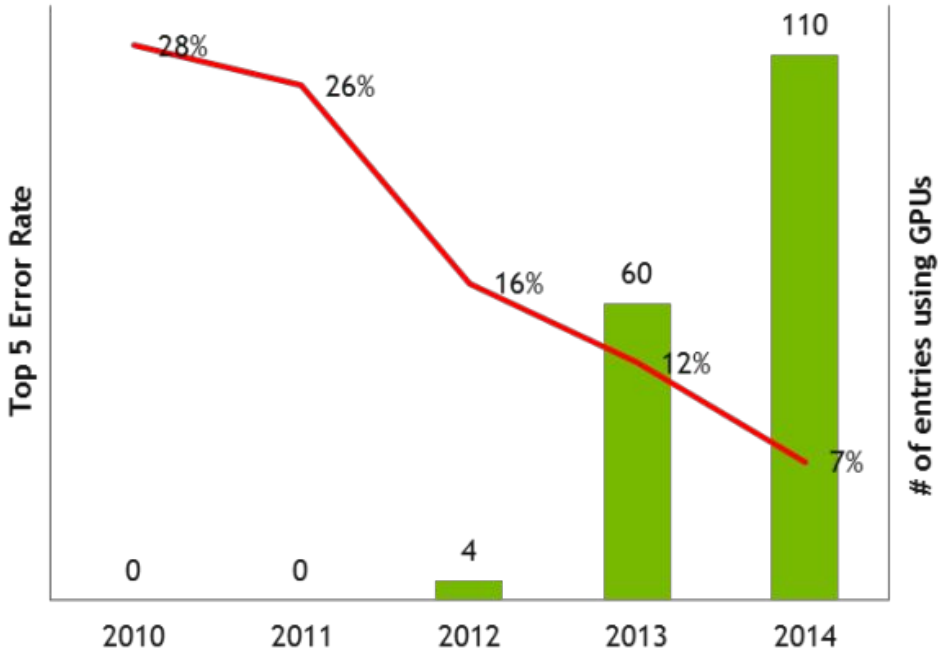
Контроль обучения

- Кросс-валидация
- Регуляризация
  - штраф за большие веса
  - dropout
  - batch norm
- Работа с обучающей выборкой

Применения персептрона

- Распознавание образов и классификация
- Анализ данных
- Принятие решений и управление
  - в нейроинформатике
  - в химии (хемоинформатике)
- Прогнозирование временных рядов
  - в экономике

Когда все поменялось

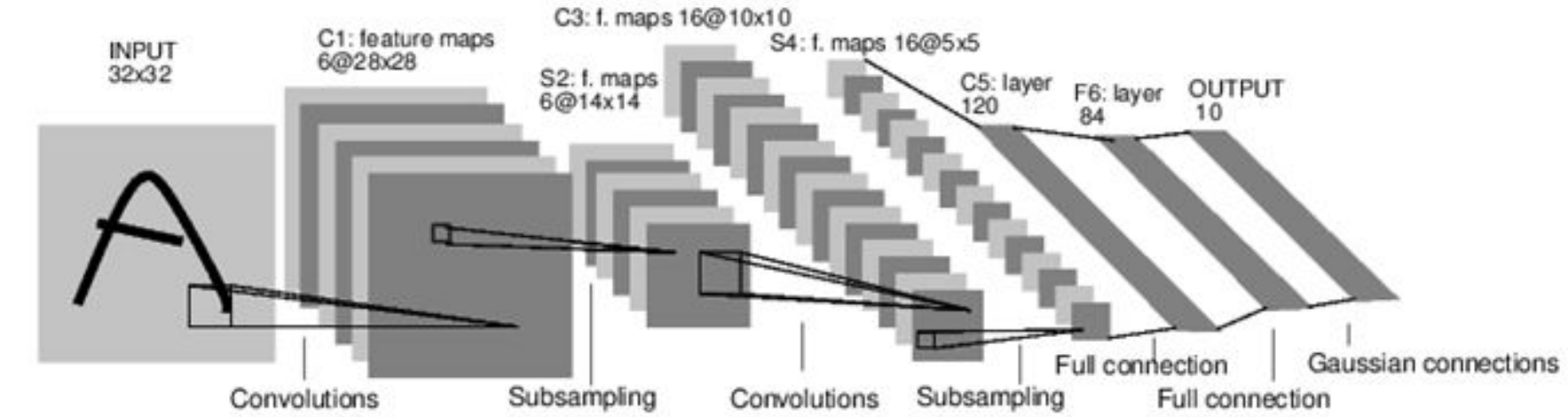


В 2012 году нейросеть впервые выиграла соревнование по распознаванию IMAGENET с 10% отрывом.

Причины:

- появление больших датасетов
- обучение на GPU
- использование сверточной архитектуры

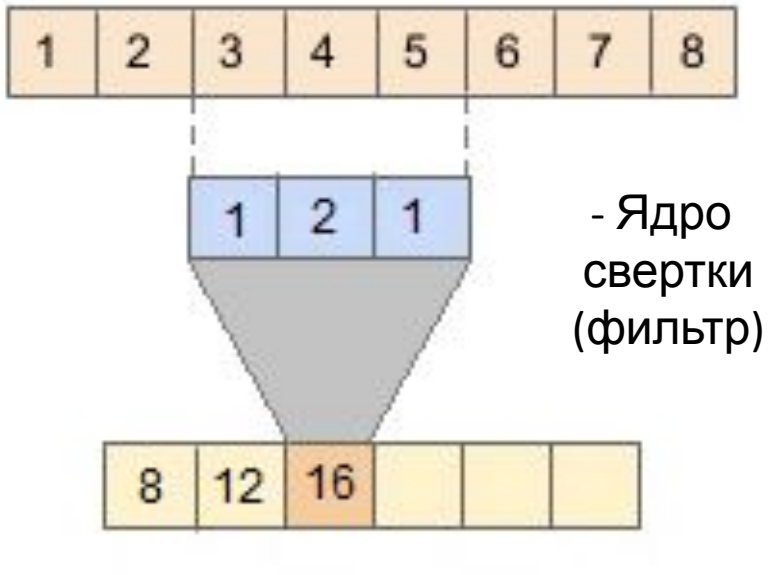
Сверточные нейронные сети



A Full Convolutional Neural Network (LeNet)

Сверточная нейронная сеть (*CNN*) — специальная архитектура нейронных сетей, предложенная Яном Лекуном в 1988 году и нацеленная на эффективное распознавание изображений. Идея заключается в чередовании сверточных слоев и субдискретизирующих слоев

Свертка

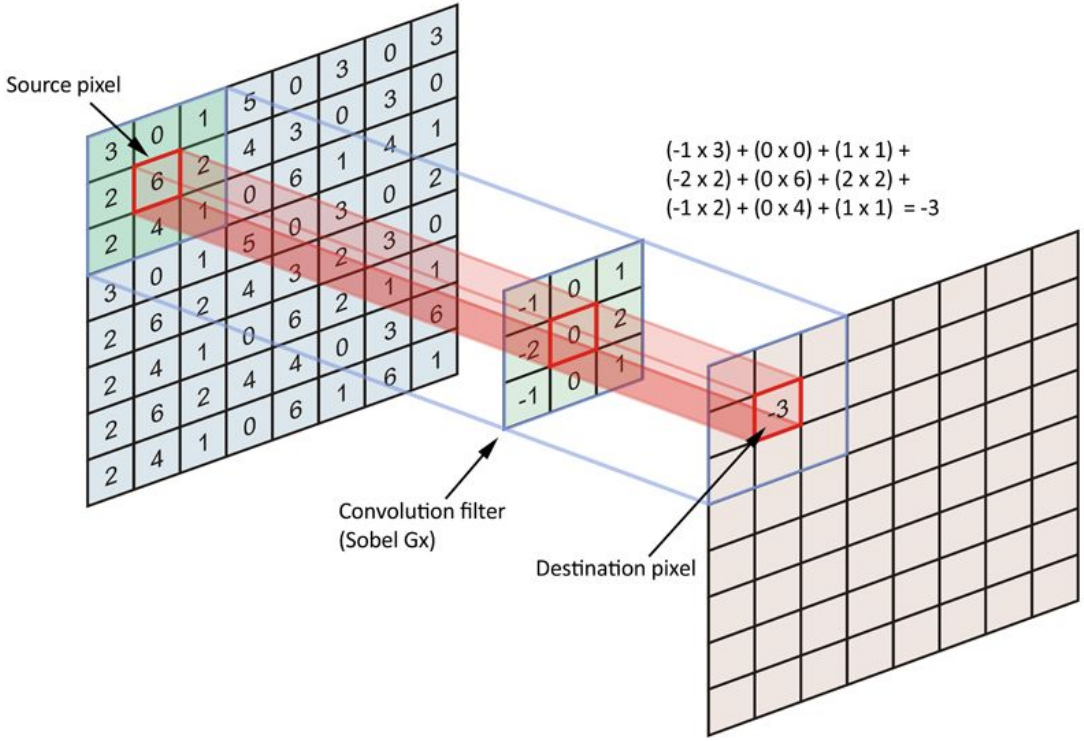


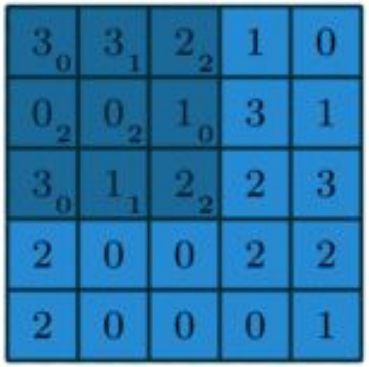
Свертка – операция, применяемая к двум массивам, которая заключается в следующем:

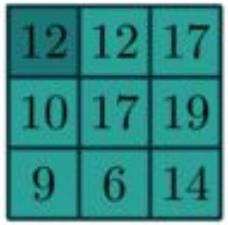
– фильтр «скользит» по входному массиву, и каждый элемент выходного массива равен скалярному произведению фильтра и соответствующей области входного массива.

$$n[k] = \sum_{i=-w}^{w} m[k+i]a[-i]$$

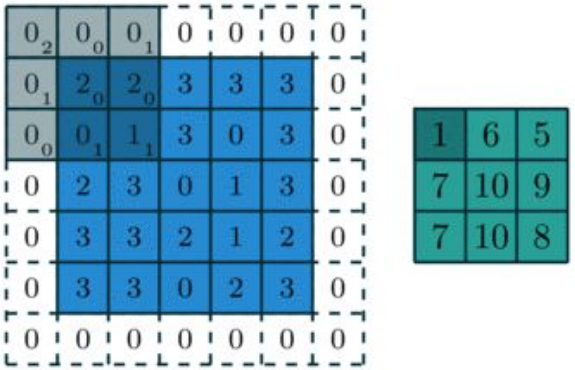
Двумерная свертка

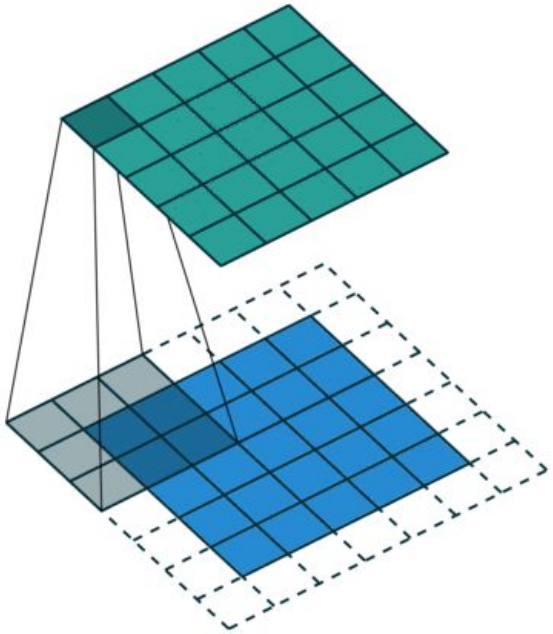




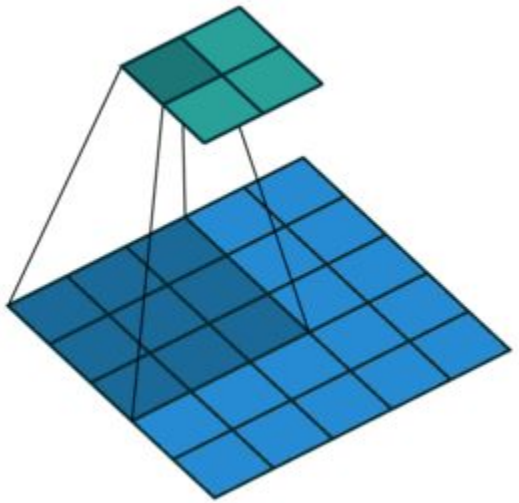


Шаг и padding





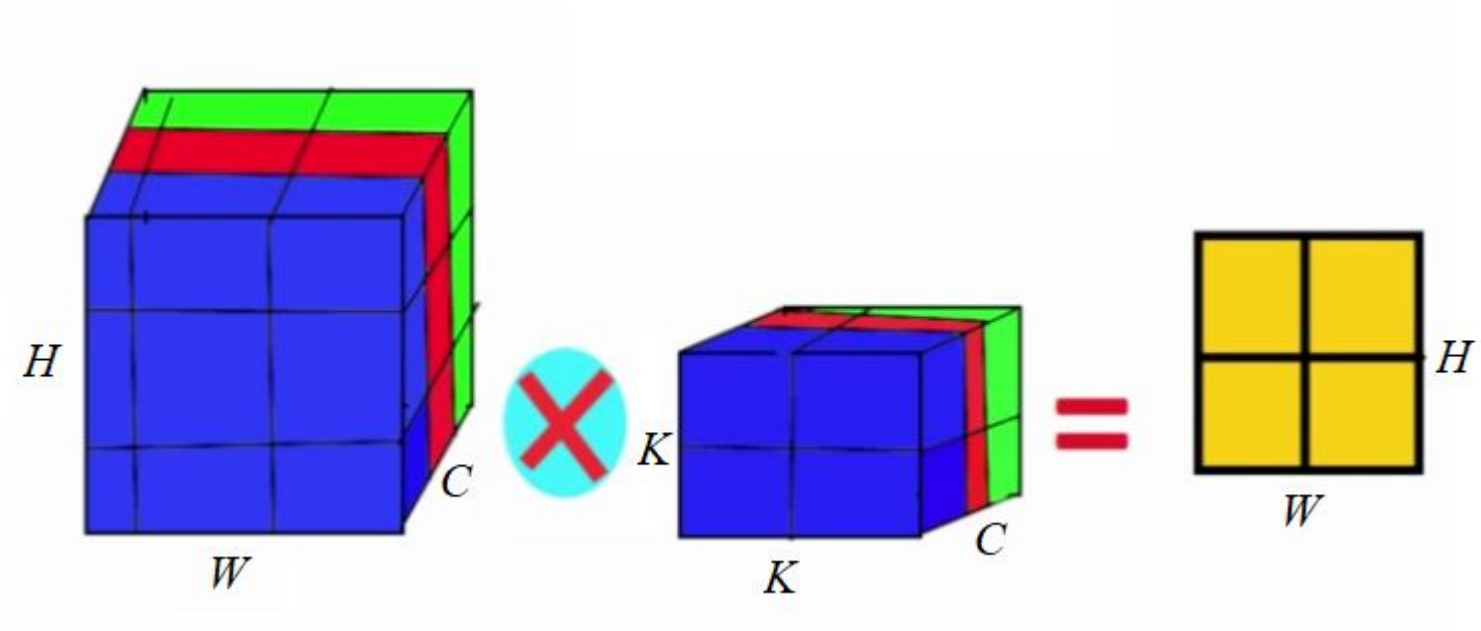
Единичный 0-padding



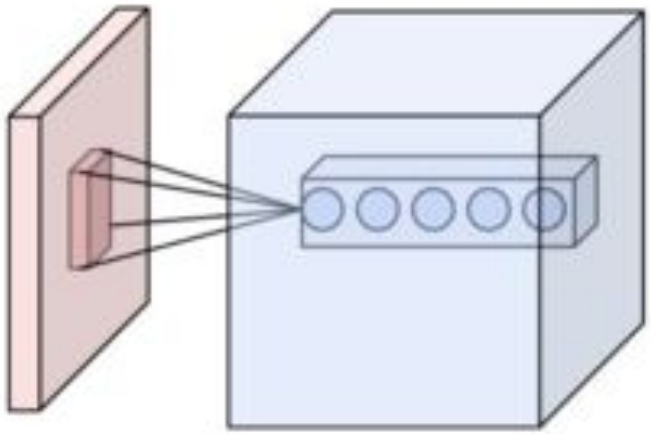
Свертка с шагом 2

Свертка в нейронных сетях

- Изображение – это трехмерный тензор (массив), с размерностями: *WxHxC*.
- К нему применяется фильтр – тензор *KxKxC*.
- Результат – матрица активации *WxH*.

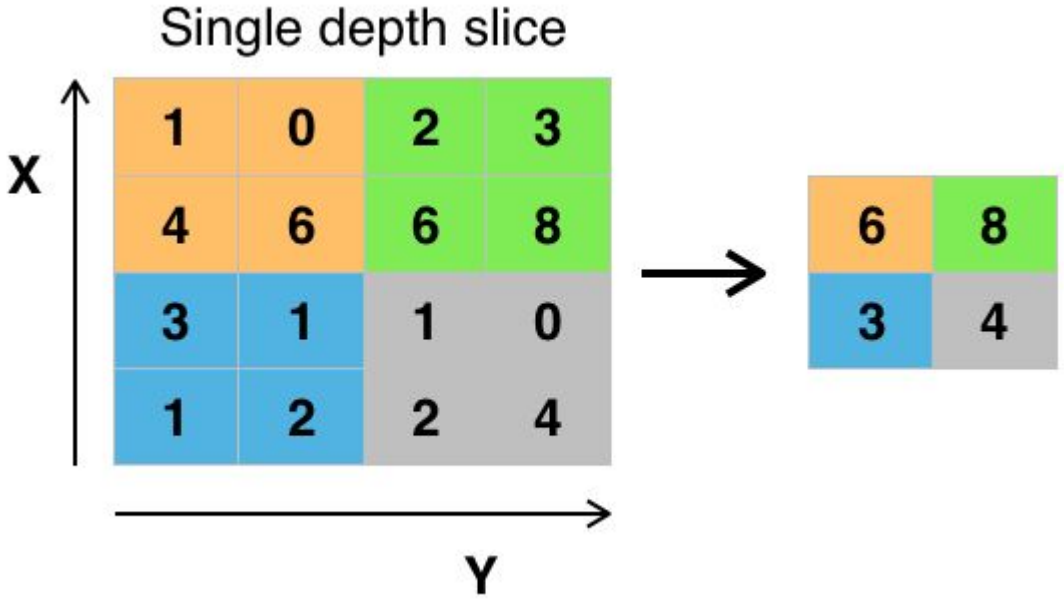


Сверточный слой



- Будем использовать не один, а F фильтров.
- Сверточный слой принимает на вход тензор $W_x H_x D$ и производит свертку набором из F фильтров $K_x K_x D$. Каждый фильтр дает двумерную матрицу активации, следовательно на выходе получается тензор $W_x H_x F$.
- Каждый фильтр ищет в окрестности своего пикселя некоторый паттерн.

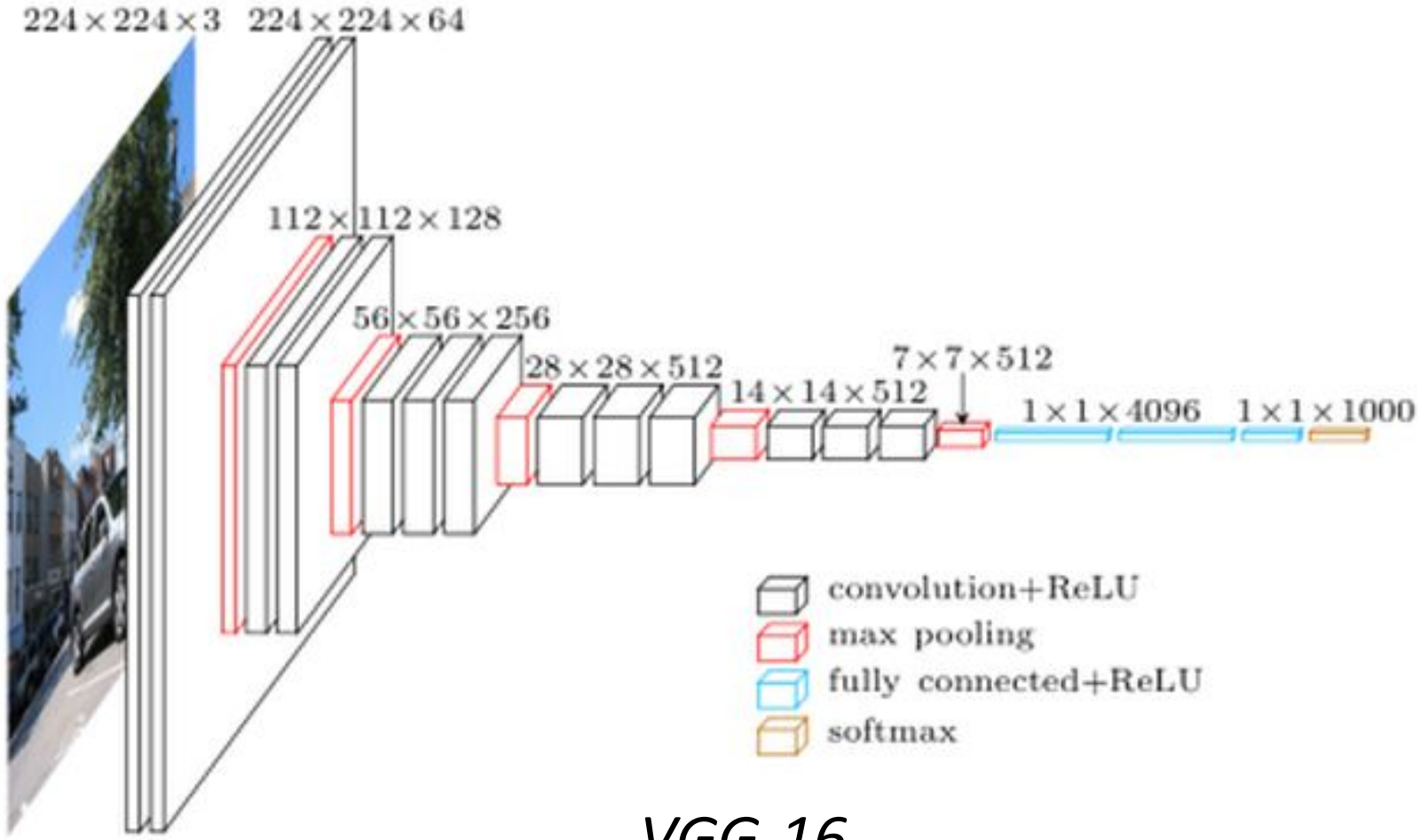
Субдискритизация (pooling)



Изображение делится на регионы (напр. квадраты 2x2), и каждый регион заменяется на максимальное значение в этом регионе.

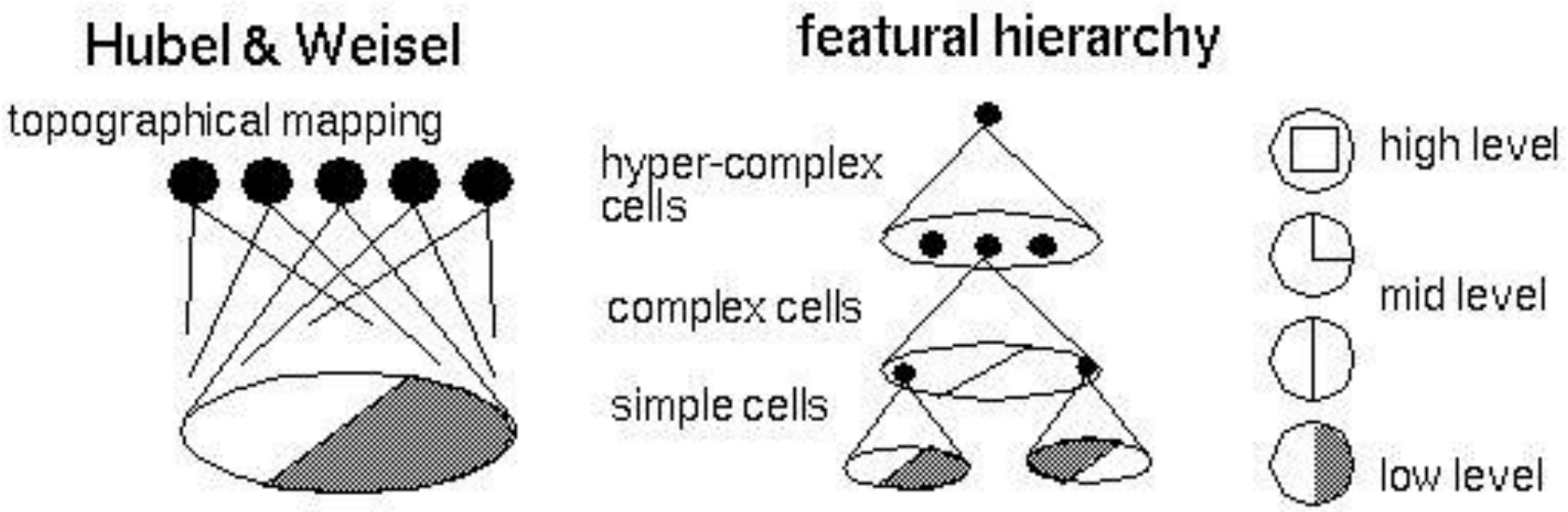
- Вырабатывается инвариативность к небольшим сдвигам
- Увеличение рецептивной области
- Уменьшение вычислительных затрат

Примеры



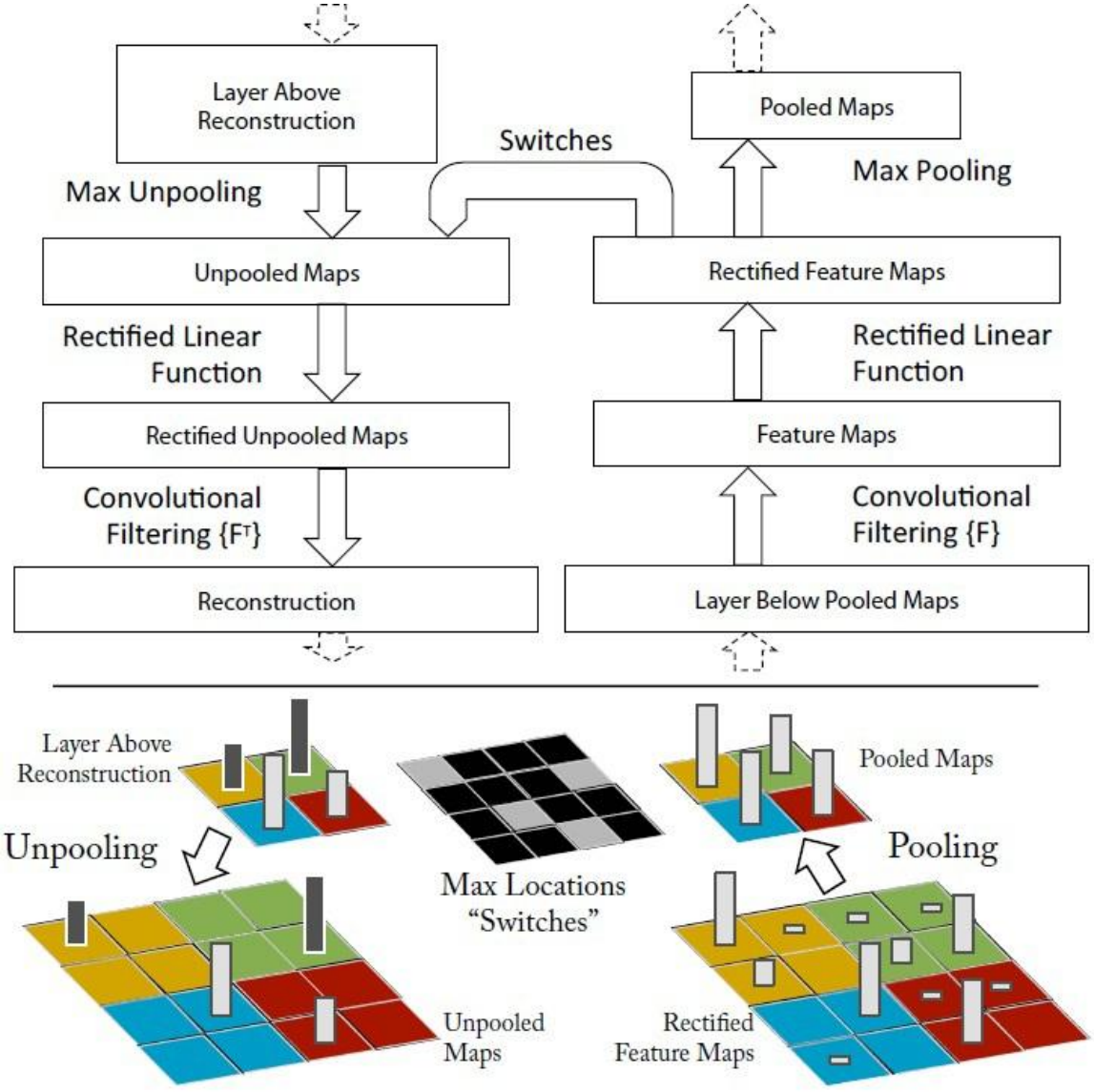
VGG-16

Понимание работы CNN



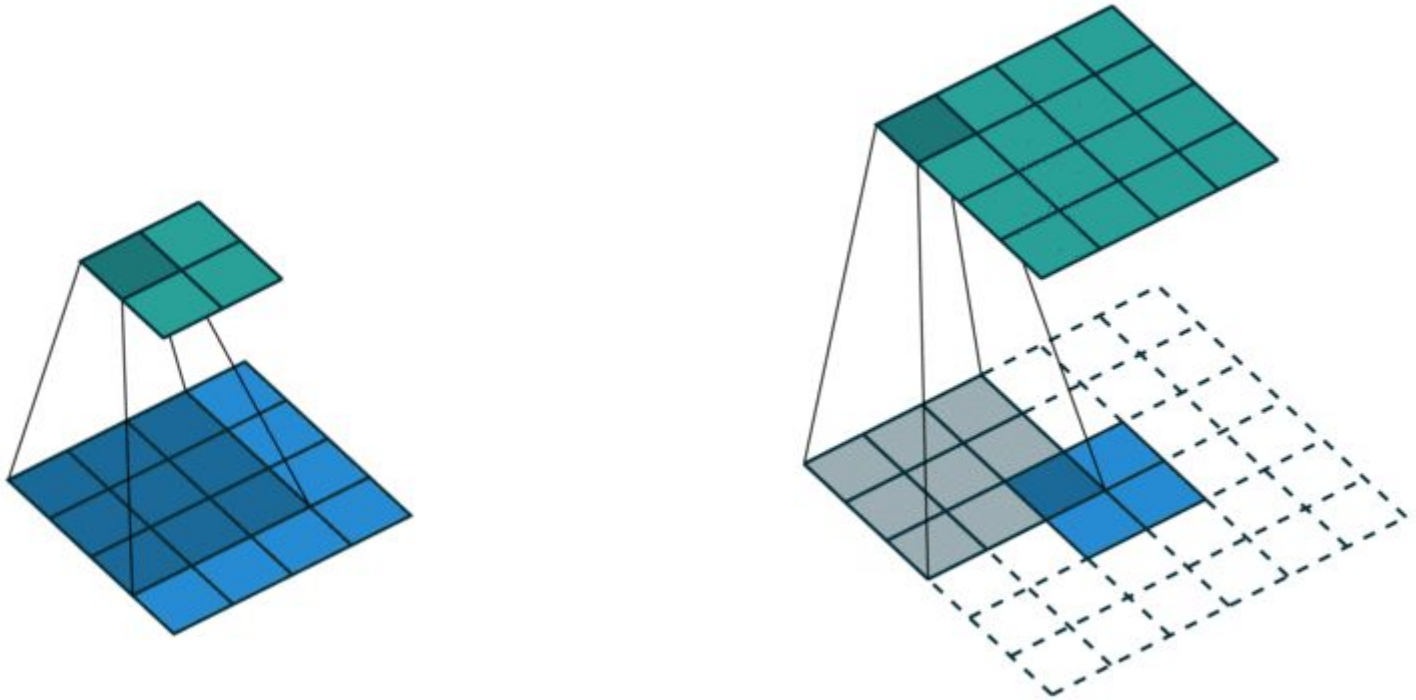
Показано, что мозг обрабатывает визуальную информацию иерархически: сначала находят границы, углы, а на более глубоких слоях – сложные объекты.

Deconvolutional network



– это сеть, которая интерпретирует CNN, т.е. показывает, какие пиксели повлияли на активацию тех или иных выходов.

Транспонированная свертка

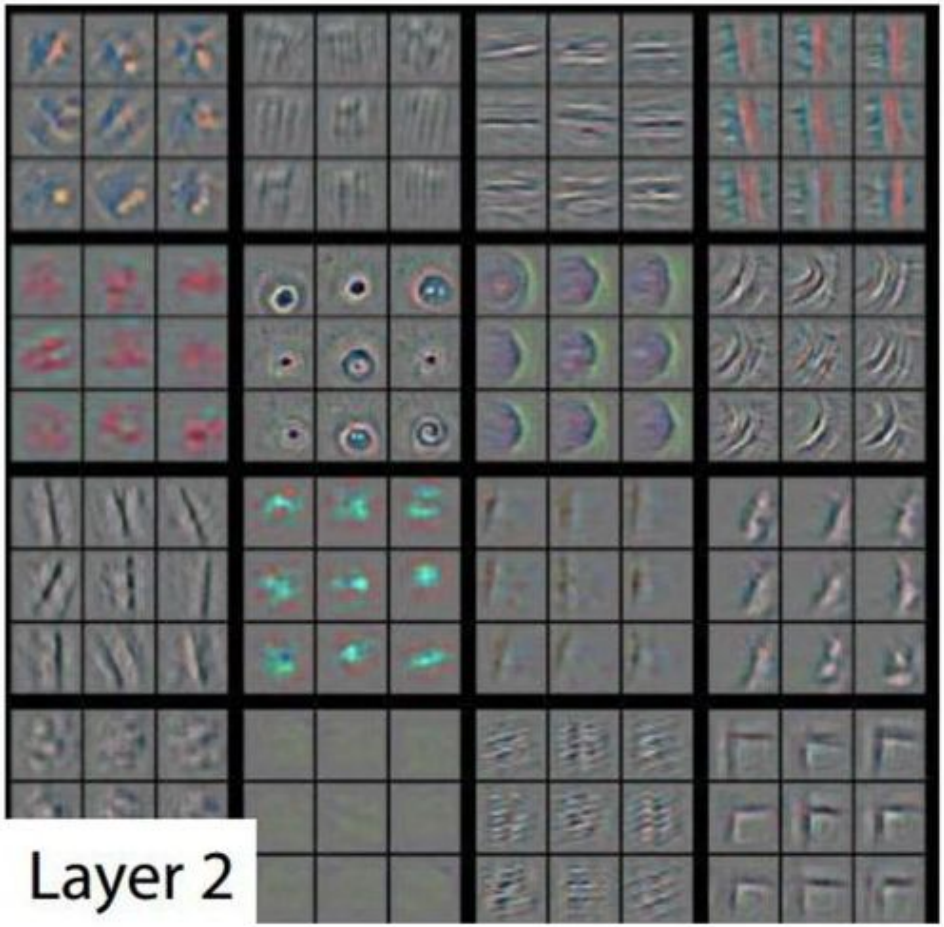


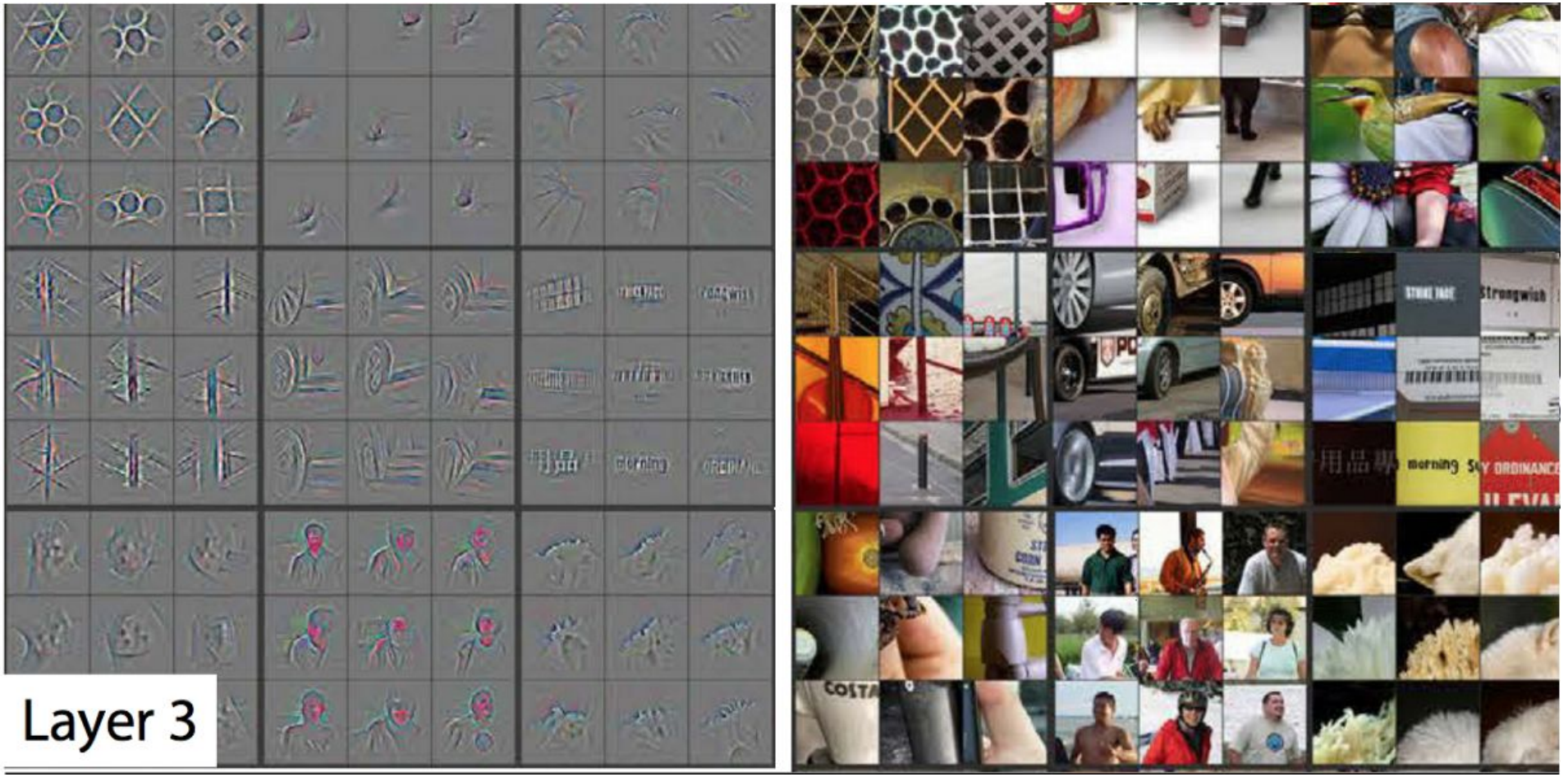
Свертка и соответствующая ей транспонированная свертка

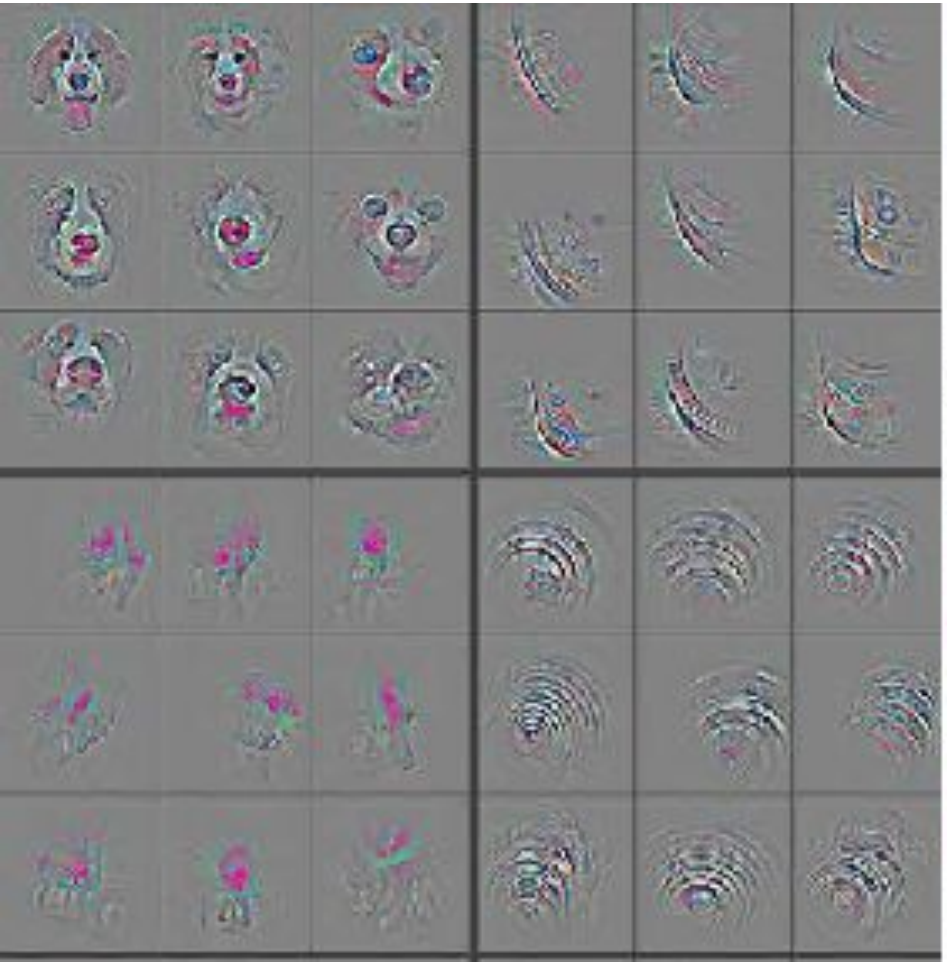
Выучиваемые признаки

Layer 1

На рисунке показаны куски изображения, которые больше всего были ответственны за то, чтобы активировать нейрон на первом слое.


Layer 2


Layer 3



Layers 4, 5

CNN для распознавания звуков и текстов

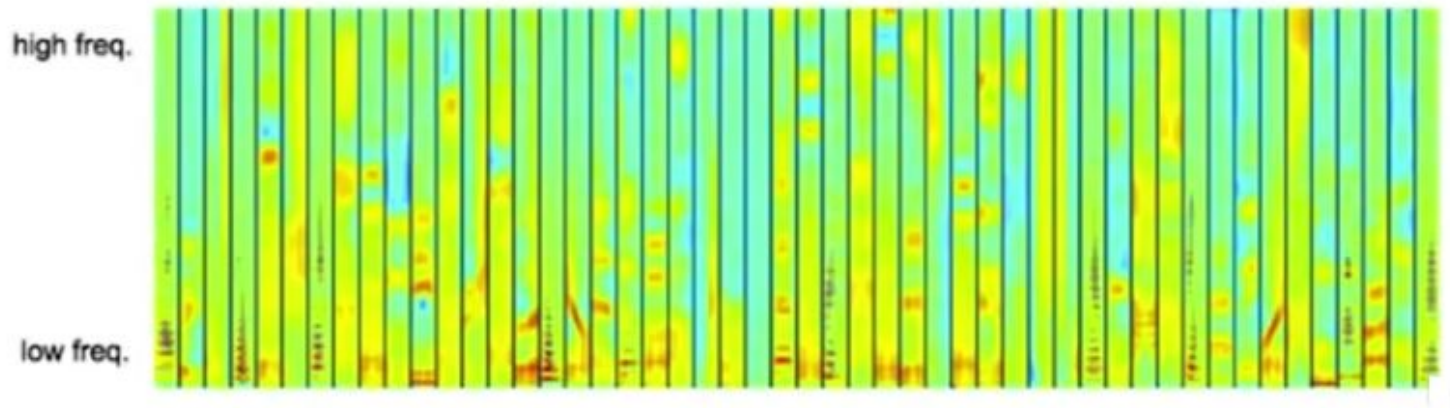


Рис.: Спектрограмма голосового сигнала

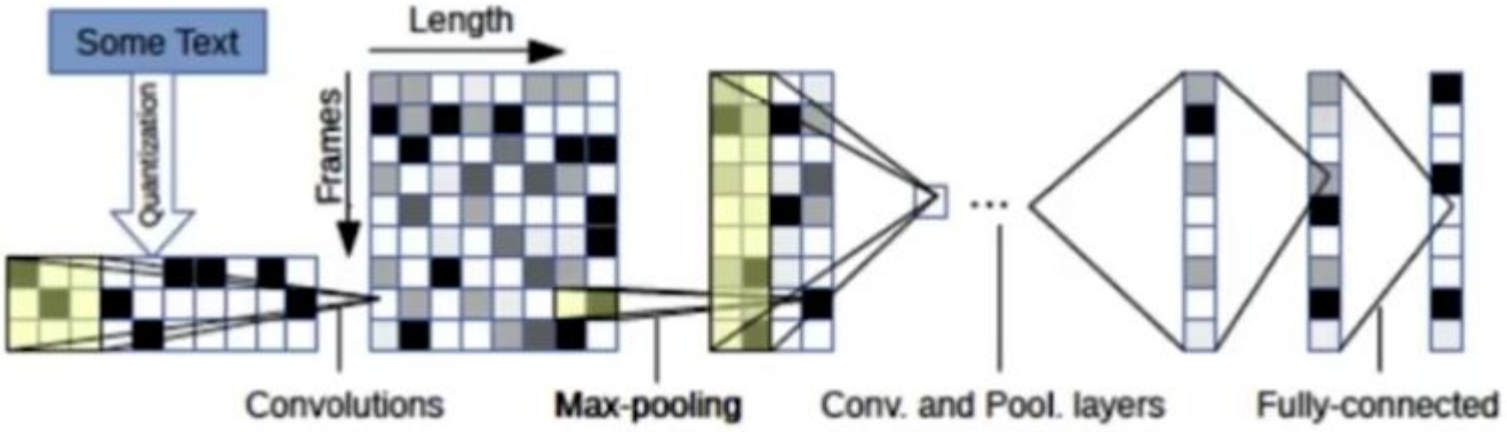
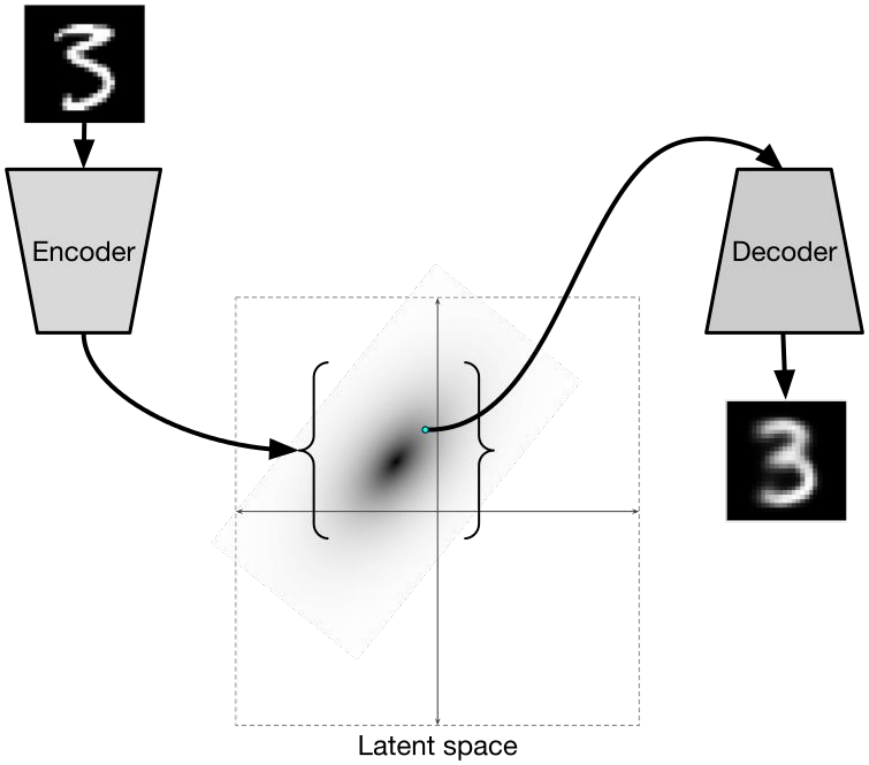
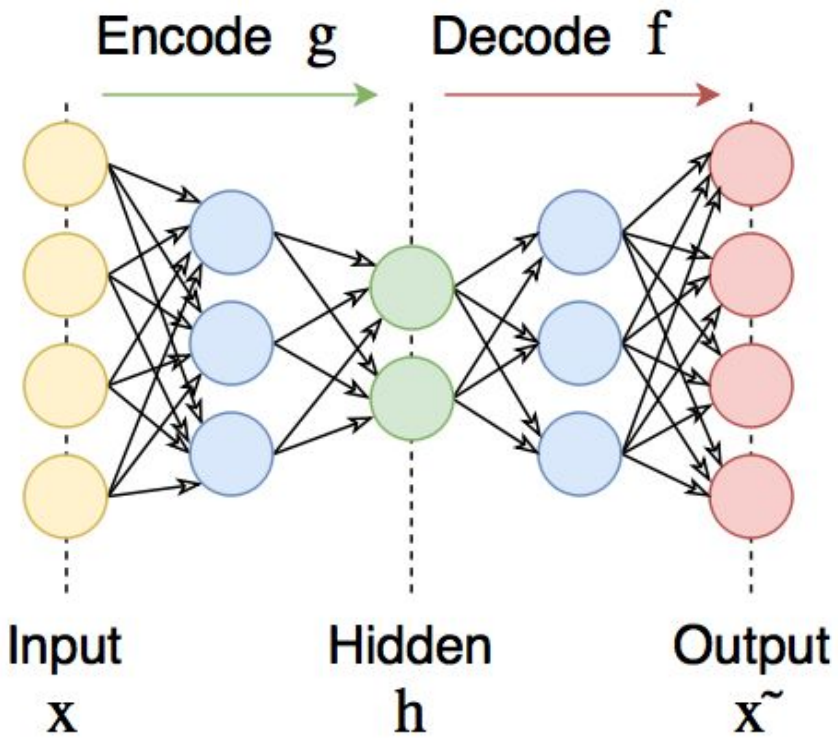


Рис.: Обработка изображения представляющего текст

Автоэнкодеры



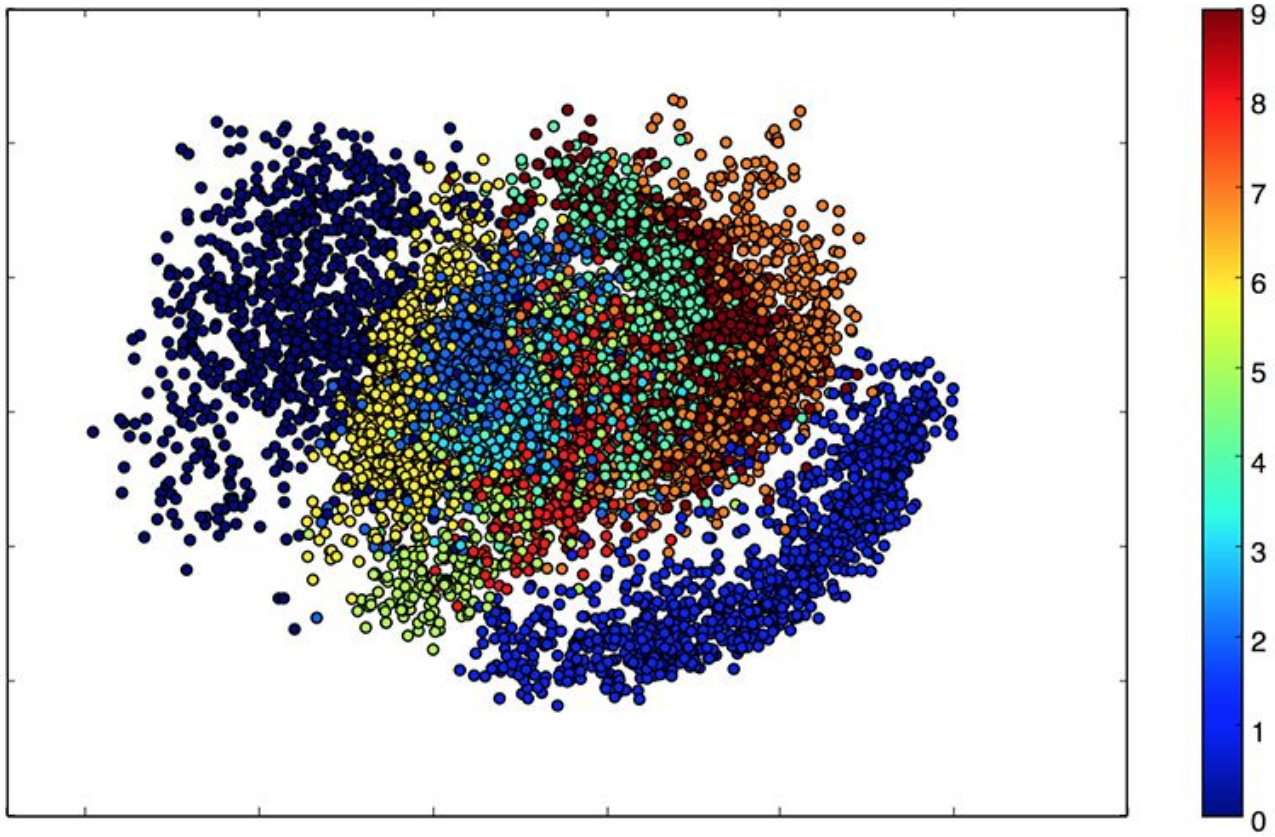
Предположим, наша задача не классифицировать картинки, а получить для них какое-то малоразмерное представление. Тогда для обучающей выборки нет меток класса, и необходимо применять обучение без учителя.



$$L(x, f(g(x))) \rightarrow \min$$

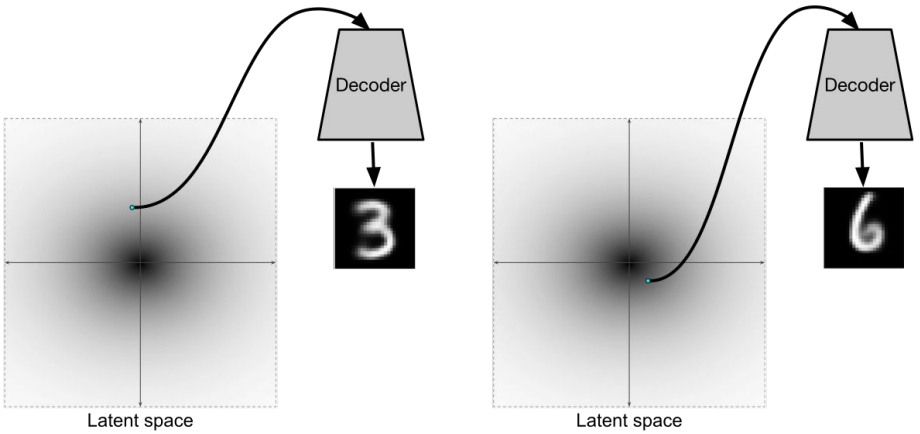
- Автоэнкодер – это специальная архитектура нейросети, состоящая из *кодировщика* и *декодировщика*. На месте их стыка образуется «*бутылочное горлышко*», на котором собираются наиболее важные признаки.
- Автоэнкодер пытается выучить тождественное преобразование, т.е. минимизировать разницу между входом и выходом

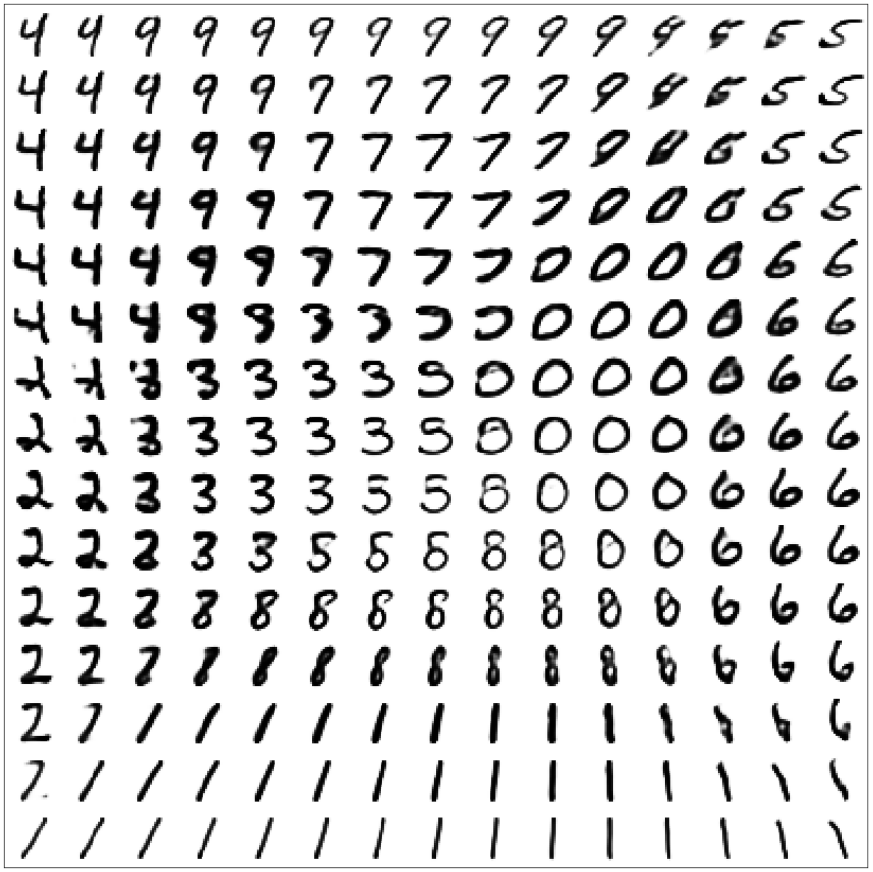
Скрытое пространство



Скрытое пространство – маломерное пространство, в которое кодировщик отображает данные. Его визуализация позволяет получать проекции, лучшие чем PCA или какой-либо другой классический метод

Движение в скрытом пространстве





Интересный эффект получается, если подавать на декодировщик значения, полученные при движении от признаков одной цифры к

Рекуррентные нейронные сети

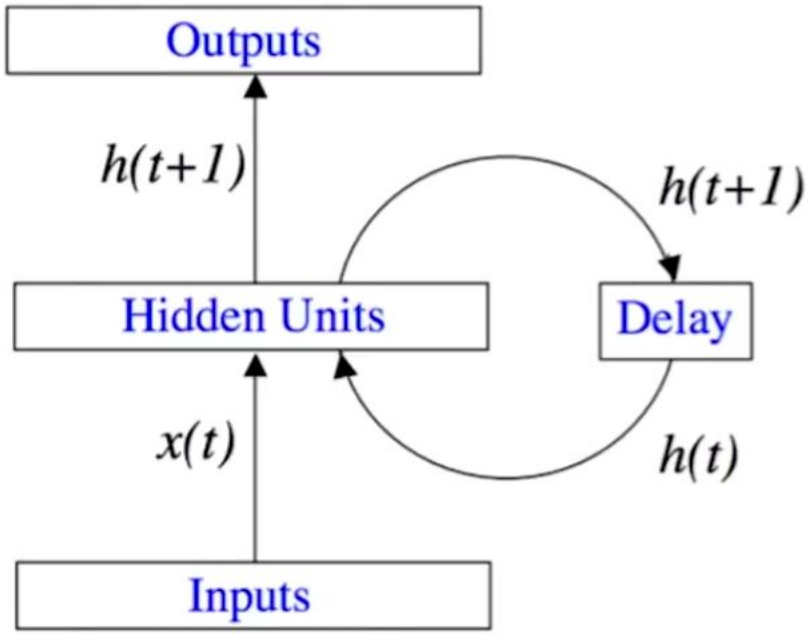
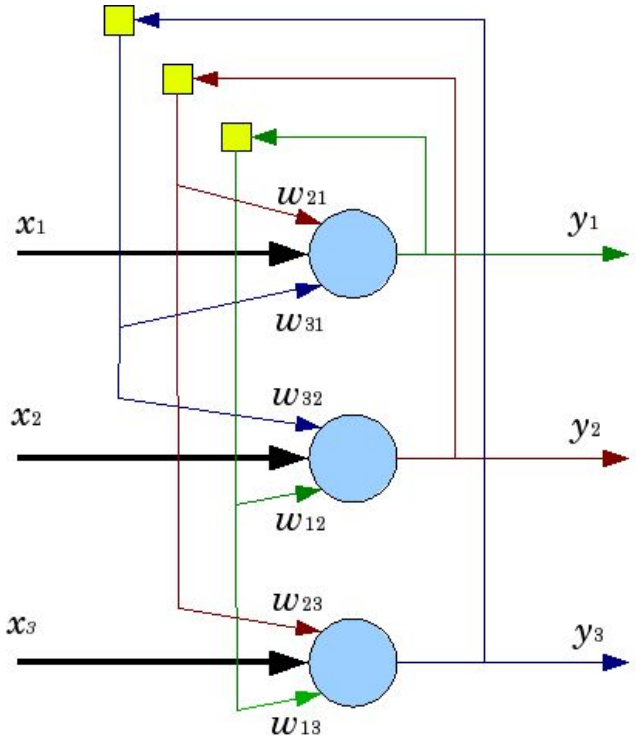


Рис.: RNN с задержкой на скрытом слое

- Рекуррентная нейронная сеть (RNN) — архитектура нейронных сетей, где связи между элементами образуют направленный цикл.
- Наличие таких циклов делает эту архитектуру идеальной для обработки последовательностей и данных, распределенных во

- Все биологической нейронной сети – рекуррентные
- RNN моделирует динамическую систему
- Универсальная теорема аппроксимации говорит, что с помощью RNN можно смоделировать поведение любой динамической системы
- Существует много алгоритмов обучения RNN без явного лидера.

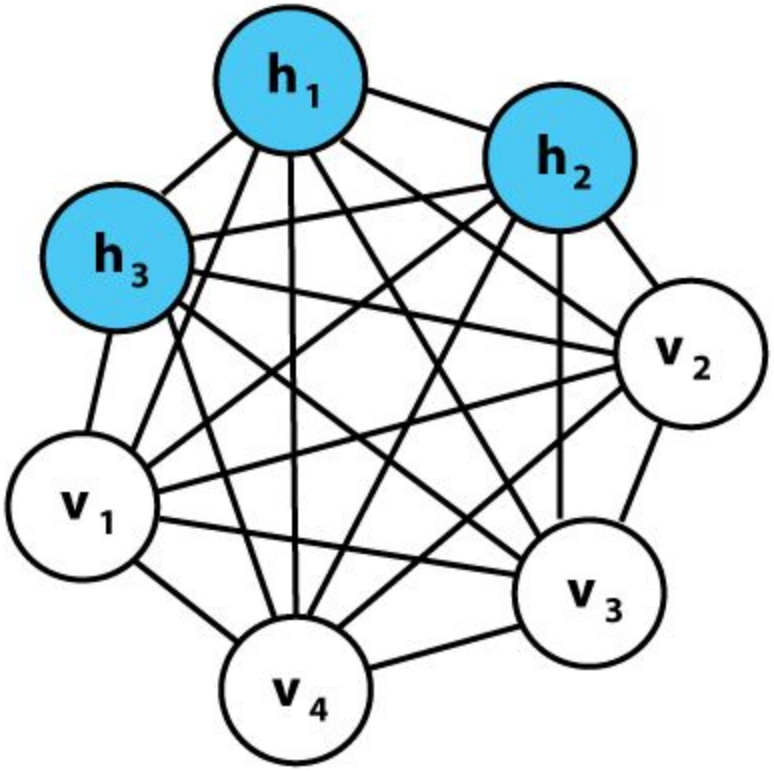
Сеть Хопфилда



- Однослойная RNN с пороговой функцией активации.
- Она моделирует ассоциативную память – она «запоминает» какой-то набор образов и потом способна восстановить его из памяти.
- Сеть обучается по следующему правилу:

$$X_i = WX_i \Rightarrow W = \frac{1}{N}\sum_i X_i X_i^T$$

Машина Больцмана

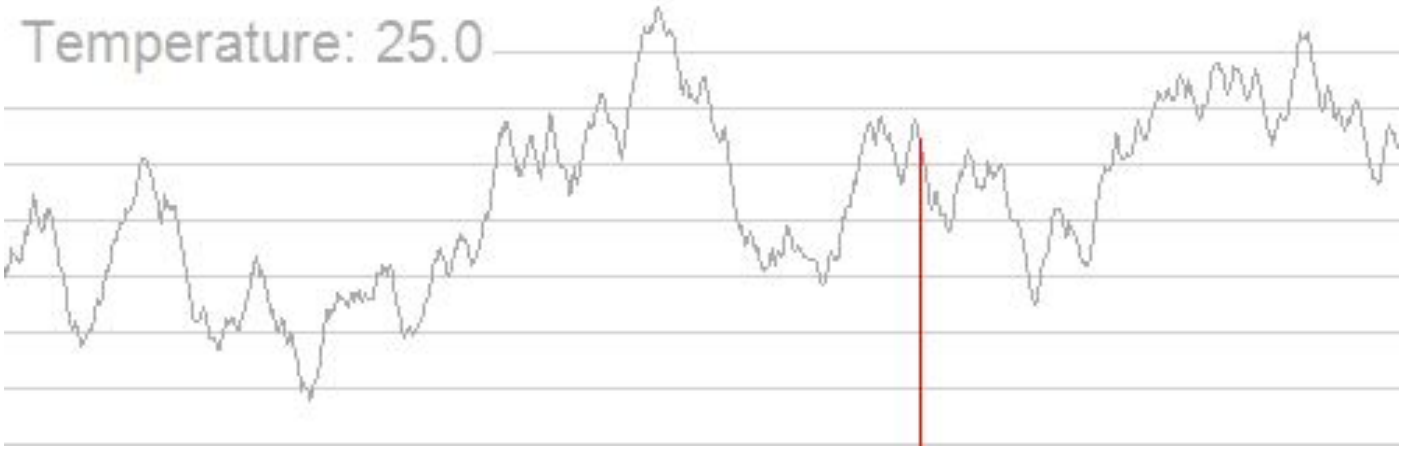


- Стохастическая аналог сети Хопфилда, придуманный Дж. Хинтоном в 1985г.
- Для нее определено понятие «энергии»:

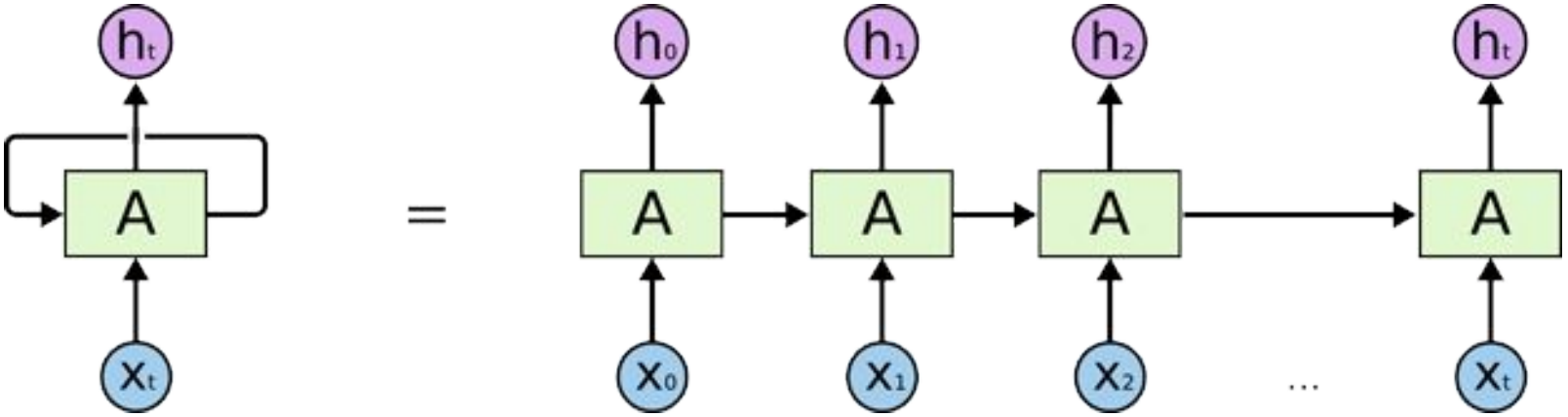
$$E = -\sum_{i<j} w_{ij} s_i s_j - \sum_i \theta_i s_i$$

- Она была первой нейронной сетью, способной обучаться внутренним представлениям и решать

- Машина Больцмана обучается алгоритмом имитации обжига:
  - система вычисляет значение энергии в некотором случайном состоянии. Если оно меньше текущего, то система переходит в это состояние, иначе остается в текущем. Вероятность перехода со временем уменьшается до нуля.



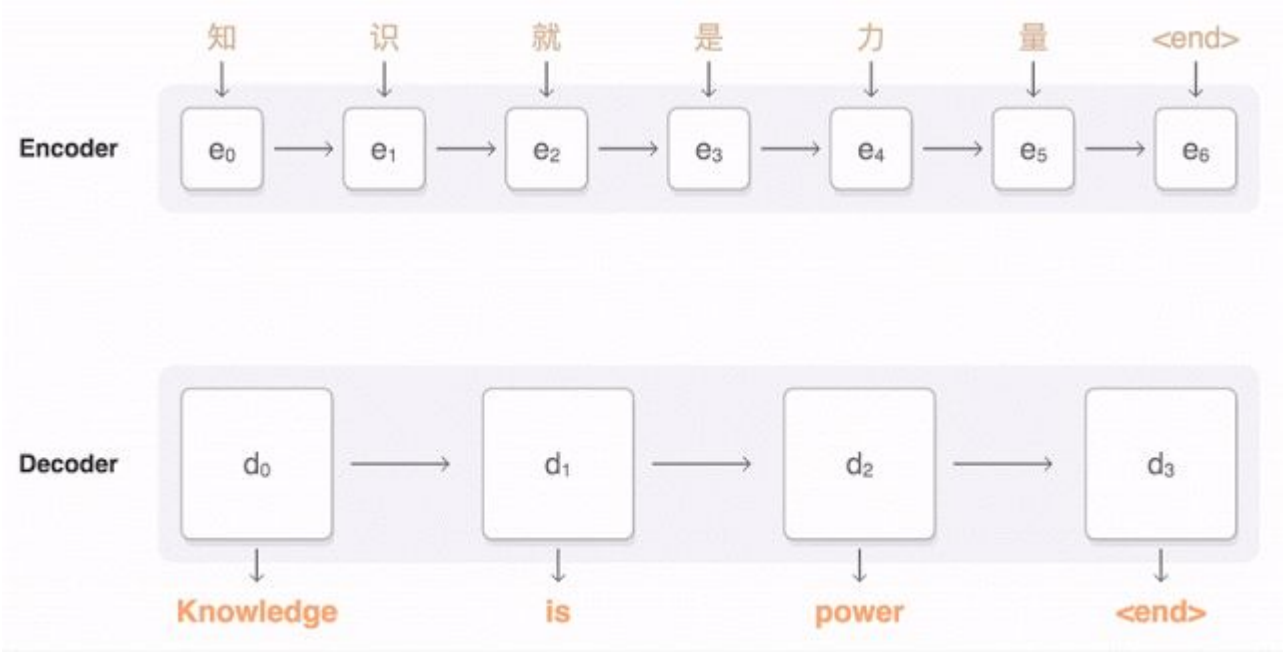
Общий случай



- В общем случае RNN может запоминать некоторый «контекст» на скрытых слоях
- Для этого она обучается методом обратного распространения ошибки развёрнутого во времени.

Применение RNN

- Моделирование последовательностей
  - преобразования (напр. из звука в текст)
  - предсказание следующего элемента последовательности (напр. следующего слова в предложении)
- Анализ контекста и внутренней структуры



Пример использования RNN в задаче машинного перевода

Вопросы

- Что такое нейронная сеть и что она моделирует? [1]
- В чем основная идея метода обратного распространения ошибки? [2]
- Какие две операции, используемые в сверточных сетях делают их идеальными для работы с изображениями? [3]
- Какую задачу решает автокодировщик? Какие особенности его архитектуры помогают ее решить? [4]
- В чем принципиальное отличие рекуррентных сетей от сетей прямого распространения? Какой вид данных это отличие позволяет эффективно обрабатывать? [5]

Источники

- www.deeplearningbook.org
- www.coursera.org/learn/neural-networks
- Andrej Karpathy «Connecting images and natural language»

- wikipedia.org
- habrahabr.ru
- ulearn.me
- Лекции Техносферы. Нейронные сети в машинном обучении