Введение в нейронные сети

Куканов Виктор

НИУ ВШЭ

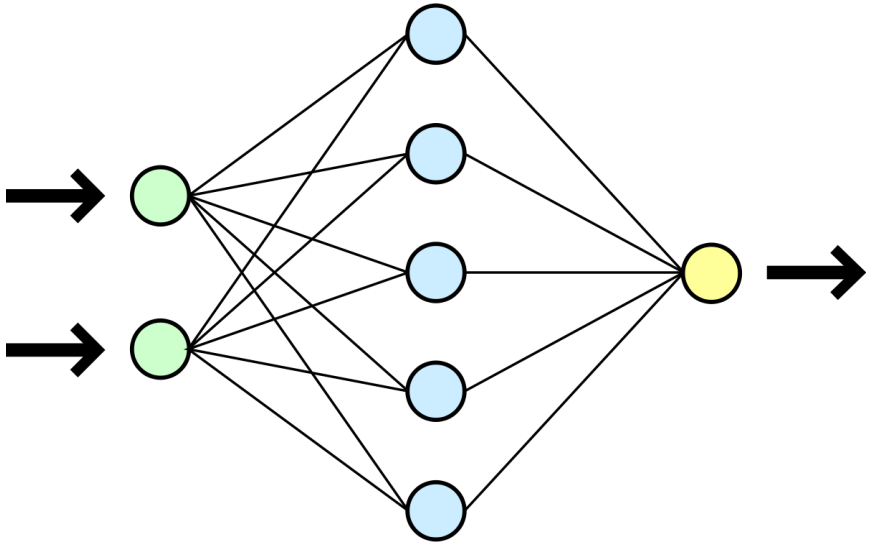
19 октября, 2018

Содержание

- Что такое нейросеть, основные определения
- История возникновения и развития
- Строение полносвязной нейронной сети
- Функции активации
- Обучение нейронных сетей
- Переобучение и методы борьбы с ним

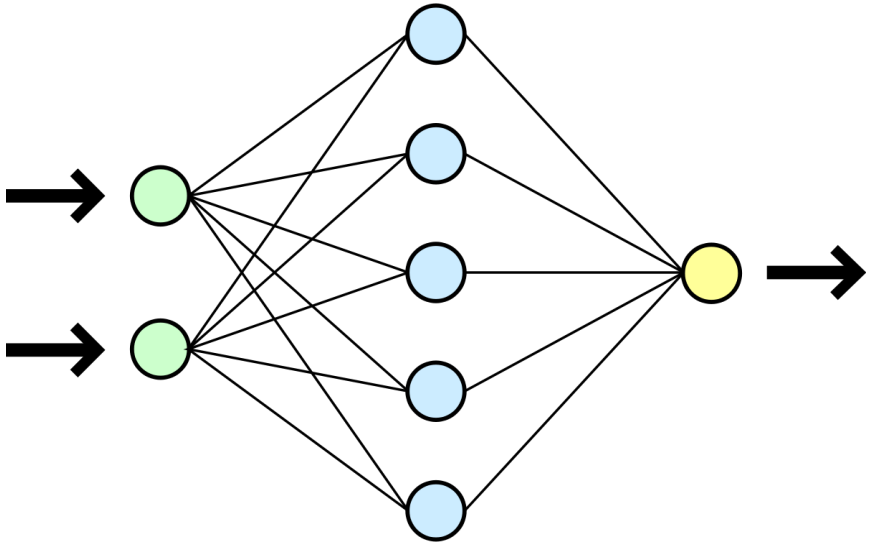
Что такое нейронная сеть?

- **Нейронная сеть** – последовательность нейронов, соединённых между собой синапсами
- Можно представить в виде графа, где вершины – этой нейроны, а рёбра – это синапсы



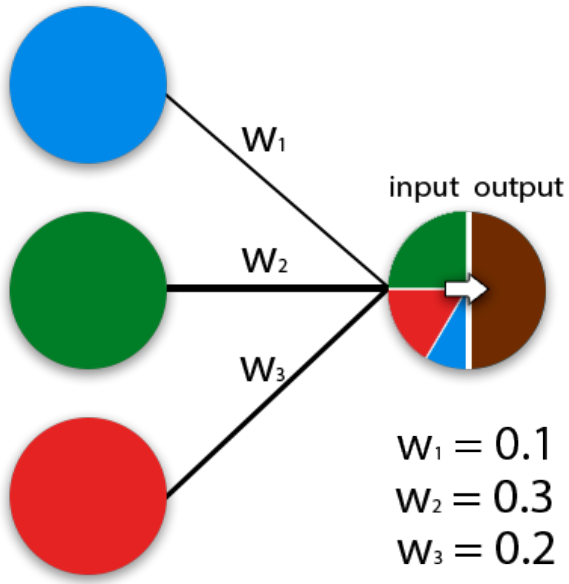
Нейроны

- **Нейрон** – некоторая функция, получающая на вход информацию с предыдущего слоя и производящая простейшие вычисления
- Выделяют 3 основных типа нейронов: входные, скрытые, выходные
- Группы нейронов объединяют в слои



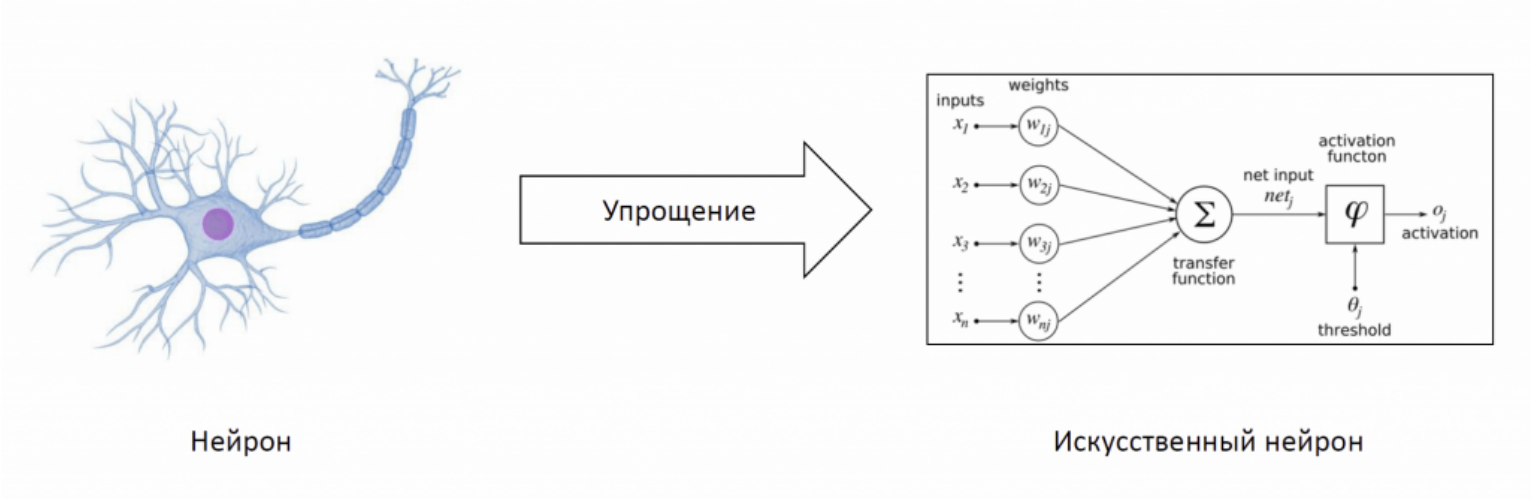
Синапсы

- **Синапс** – это связь между двумя нейронами
- У каждого синапса есть свой вес, с помощью которого регулируется “важность” связи между нейронами



Возникновение нейронных сетей

- В 1943 году разработана первая компьютерная модель нейронной сети
- Изначально учёные пытались смоделировать нейронную активность мозга человека
- Биология – основной источник идей новых архитектур нейросетей



Нейрон

Искусственный нейрон

Почему это работает?

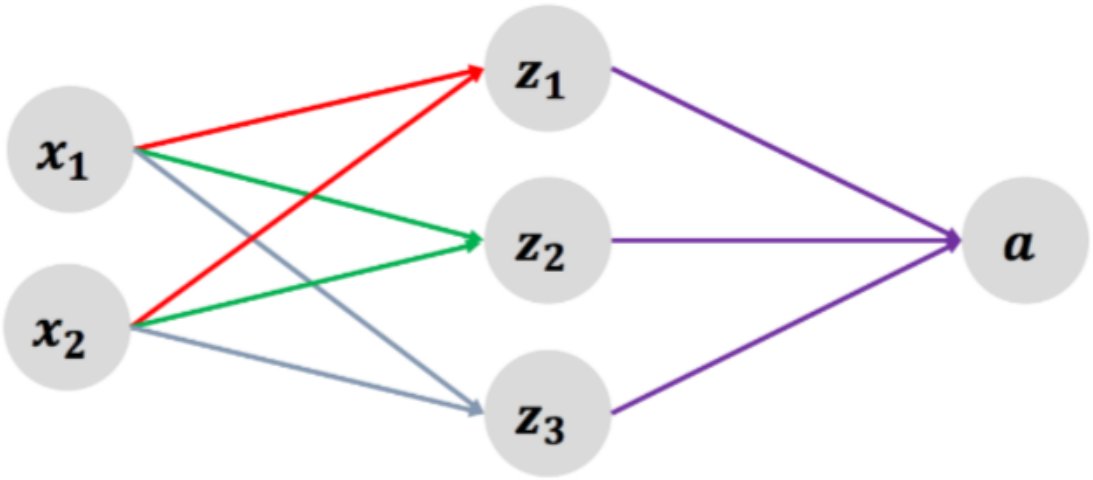
- В теории – теорема Цыбенко (универсальная теорема аппроксимации): нейронная сеть с одним скрытым слоем может аппроксимировать любую непрерывную функцию с любой точностью
- На практике – при большом объеме обучающей выборки и правильно подобранной архитектуре нейросеть способна выявлять сложные зависимости между выходными и выходными данными

Популярность

- Открытие **метода обратного распространения ошибки** (метода обучения нейросетей) вызвало практический интерес к нейронным сетям
- Для обучения требуется произвести большой объем вычислений, поэтому с увеличением производительности компьютеров росла популярность нейросетей
- С развитием интернета накапливалось всё больше данных, которые требуются для качественного обучения

Полносвязная нейронная сеть

- **Полносвязная сеть** – сеть, в которой каждый нейрон соединён синапсами со всеми нейронами следующего слоя

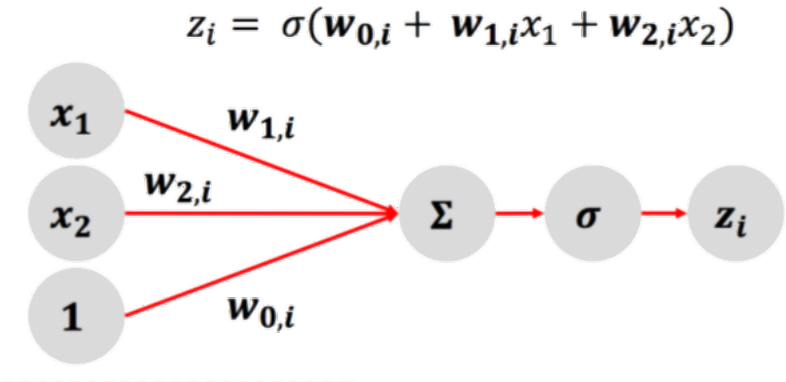


Полносвязная нейронная сеть

- Обычно нейроны представляются в виде:

$$z_i = \sigma\left(\sum_{j=1,\dots,n} x_j w_{j,i}\right) = \sigma(\langle w_i, x\rangle)$$

- $x = (x_1, x_2, \dots, x_n)$ – входные сигналы с прошлого слоя
- $w_i = (w_{1,i}, w_{2,i}, \dots, w_{n,i})$ – веса связей нейрона z_i с нейронами предыдущего слоя
- σ – **функция активации** для добавления нелинейности
- Обычно к каждому слою добавляется один нейрон с константным значением (bias)



Полносвязная нейронная сеть

- Применение полносвязного слоя можно представить в матричном виде:

$$\begin{bmatrix} z_1 \\ \cdots \\ z_m \end{bmatrix} = \begin{bmatrix} \sigma(\langle w_1, x \rangle) \\ \cdots \\ \sigma(\langle w_m, x \rangle) \end{bmatrix} = \sigma \begin{bmatrix} \langle w_1, x \rangle \\ \cdots \\ \langle w_m, x \rangle \end{bmatrix} = \sigma(Wx)$$

- То есть применение полносвязной сети N ко входу x выражается через обычное умножение матриц:

$$N(x) = \sigma(\dots \sigma(W_2 \sigma(W_1 x)))$$

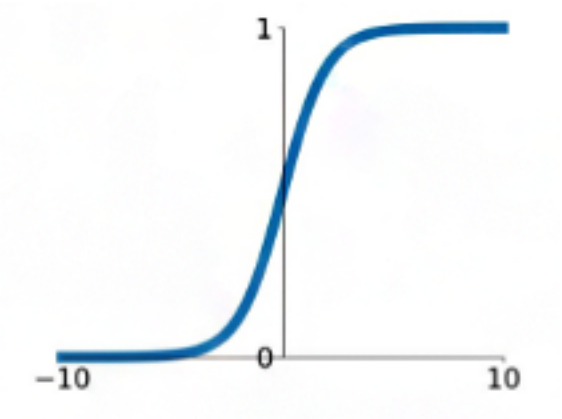
- Функция активации $\sigma(x)$ нужна для добавления нелинейности к выходам нейронов
- Если её убрать, то получим обычную линейную модель:

$$N(x) = W_k W_{k-1} \dots W_1 x = (W_k W_{k-1} \dots W_1)x = \overline{W}x$$

Сигмоида

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

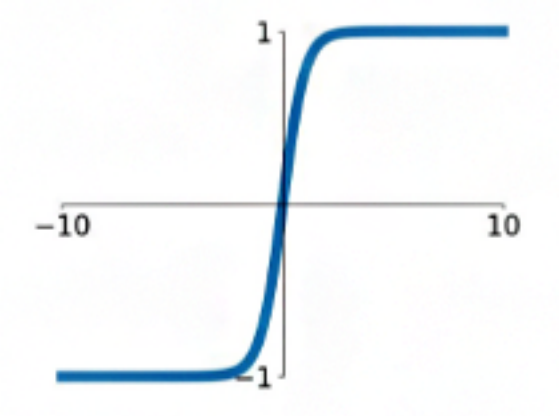
- Дифференцируема
- Выходные значения лежат в $[0,1]$
- Не проходит через начало координат
- Градиент очень мал при больших значениях x по модулю (что приводит к затуханию градиента при обучении)



Гиперболический тангенс

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

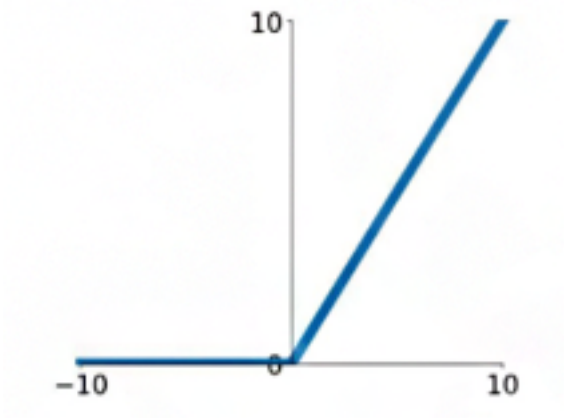
- Дифференцируема
- Выходные значения лежат в $[-1, 1]$
- Проходит через начало координат
- Также может приводить к затуханию градиента



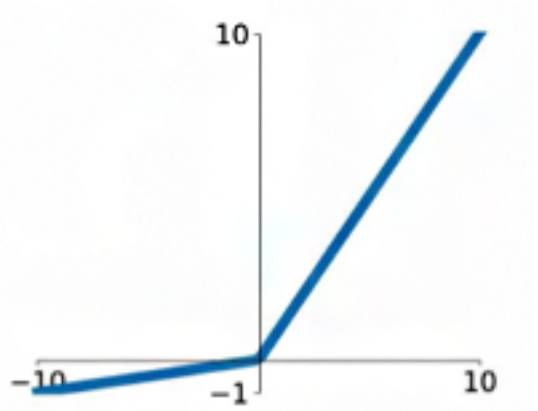
ReLU

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

- Легко вычислить
- Иногда повышает скорость сходимости из-за того, что производная в случае активации равна единице
- Нейроны могут “умирать” из-за нуля производной при $x < 0$
- Одна из модификаций – Leaky ReLU: $\sigma(x) = \max(\alpha x, x)$



ReLU



Leaky ReLU

Обучение

- x_0 – вход нейросети, x_n – значения выходного слоя, Q – функционал ошибки
- Каждый слой – некоторая функция от выходов с предыдущего слоя и весов связей, то есть i-ый слой можно записать как $x_i = f_i(x_{i-1}, w_{i-1})$
- Считаем градиенты функционала ошибки по весам каждого слоя
- Обновляем веса градиентным спуском

Обучение

- Если $z = g\big(f(x)\big), y = f(x), \frac{dy}{dx} = \left[\frac{\partial y_j}{\partial x_i}\right]_{i,j}$ – якобиан, где $x \in \mathbb{R}^n, y \in \mathbb{R}^m$, то (“правило дифференцирования сложной функции”):

$$\frac{\partial g}{\partial x} = \left(\frac{dy}{dx}\right)^T \frac{\partial g}{\partial y}$$

- Тогда можем последовательно вычислить все $\frac{\partial Q}{\partial w_i}$

$$\frac{\partial Q}{\partial w_{n-1}} = \left(\frac{dx_n}{dw_{n-1}}\right)^T \frac{\partial Q}{\partial x_n} \qquad \frac{\partial Q}{\partial x_{n-1}} = \left(\frac{dx_n}{dx_{n-1}}\right)^T \frac{\partial Q}{\partial x_n}$$

$$\frac{\partial Q}{\partial w_{n-2}} = \left(\frac{dx_{n-1}}{dw_{n-2}}\right)^T \frac{\partial Q}{\partial x_{n-1}} \qquad \frac{\partial Q}{\partial x_{n-2}} = \left(\frac{dx_{n-1}}{dx_{n-2}}\right)^T \frac{\partial Q}{\partial x_{n-1}}$$

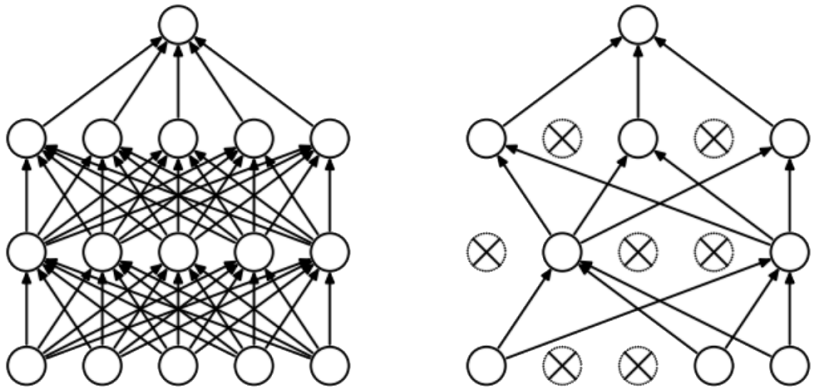
…

Проблемы с обучением

- С ростом количества слоёв растёт и количество возможных проблем, среди них “затухание” и “взрыв” градиента
- По мере распространения ошибки от выходного слоя к начальному, происходит домножение на производную функции активации
- Если производная меньше единицы, то чем дальше слой находится от выходного, тем ближе градиент к 0 (затухание)
- Если производная не ограничена, то можем получить экспоненциальный рост, который ведет к нестабильности в процессе обучения (взрыв)

Переобучение

- Если данных не очень много, то нейросеть легко переобучается, запоминая все объекты обучающей выборки
- Регуляризация – штрафуем за большую норму весов между слоями
- Ранняя остановка – прекращаем обучение до того момента, когда сеть начнёт подгоняться под выборку
- Dropout – на каждом шаге обучения с некоторой вероятностью выключаем нейрон



Batch normalization

- Обычно обучение происходит батчами
- Выходы нейронов могут иметь разный масштаб, что может приводить к медленному темпу обучения
- Можем нормализовать выходы слоя:

$X_B = \{x_1, \dots, x_m\}$ – выходы нейрона на текущем батче

$$\mu_B = \frac{1}{m}\sum_{x \in X_B} x \qquad \sigma_B^2 = \frac{1}{m}\sum_{x \in X_B} (x - \mu_B)^2$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

- Добавим гибкости выходам (разрешаем выучивать другое мат. ожидание и дисперсию)

$$y_i = \gamma \hat{x}_i + \beta$$

- Вносит некоторый шум в данные, тем самым может бороться с переобучением

Заключение

- Плюсы:
  - Нейросети – мощный и многофункциональный инструмент для анализа данных
  - Хорошо кастомизируются
  - Активно развиваются, появляется много статей/туториалов
- Минусы:
  - Нужно много данных
  - Легко переобучаются
  - Нужно много вычислительных ресурсов
  - Сложно интерпретировать результаты

Список литературы

- Batch normalization in neural nets
- How the backpropagation algorithm works
- Функции активации, регуляризация нейронных сетей
- Why are deep neural networks hard to train?