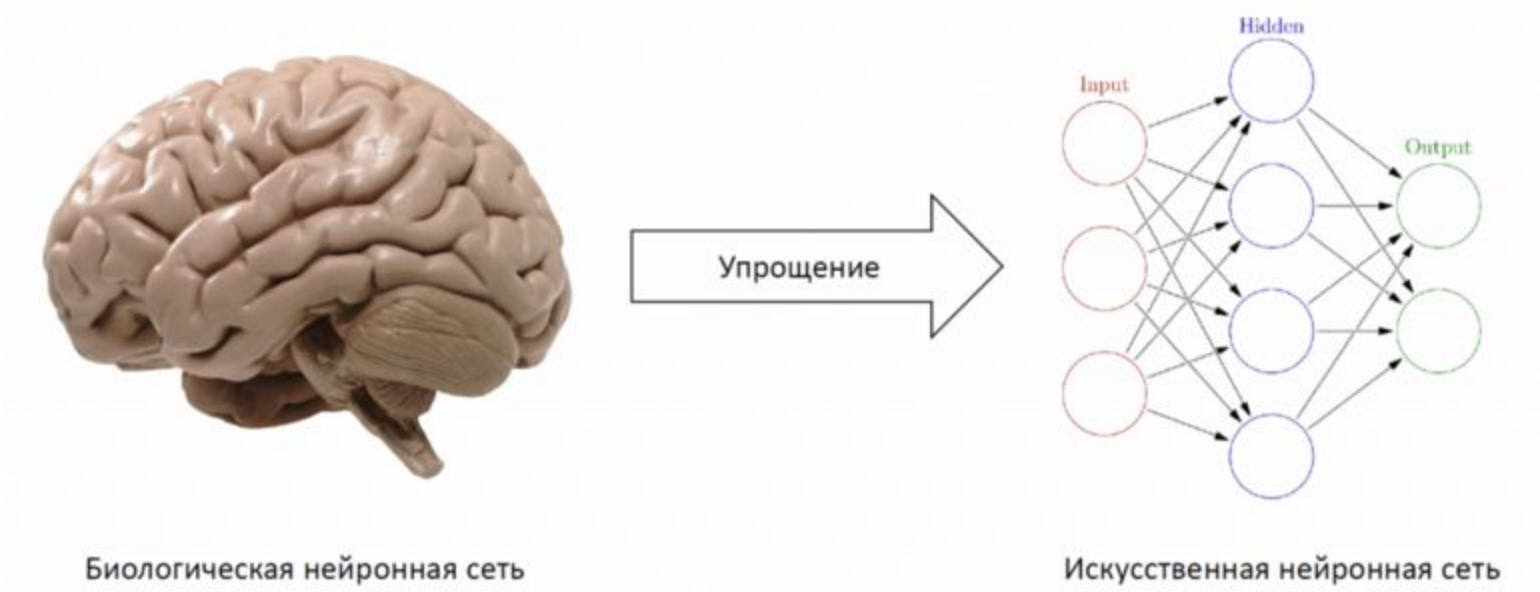
Введение в нейронные сети

Николаева Софья
НИУ ВШЭ
18 октября 2019

Что такое нейронная сеть?

Искусственная нейронная сеть (ИНС) (англ. Artificial neural network (ANN)) — упрощенная модель биологической нейронной сети, представляющая собой совокупность искусственных нейронов, взаимодействующих между собой.



Зачем нужны нейронные сети?

Нейронные сети используются для решения сложных задач, которые требуют аналитических вычислений подобных тем, что делает человеческий мозг.

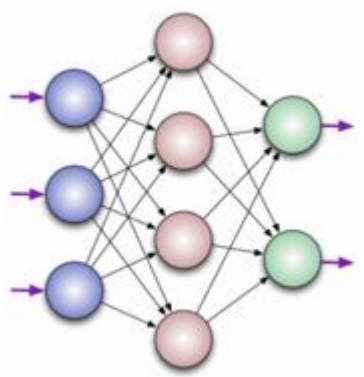
Самыми распространенными применениями нейронных сетей является:

классификация, предсказание и распознавание.

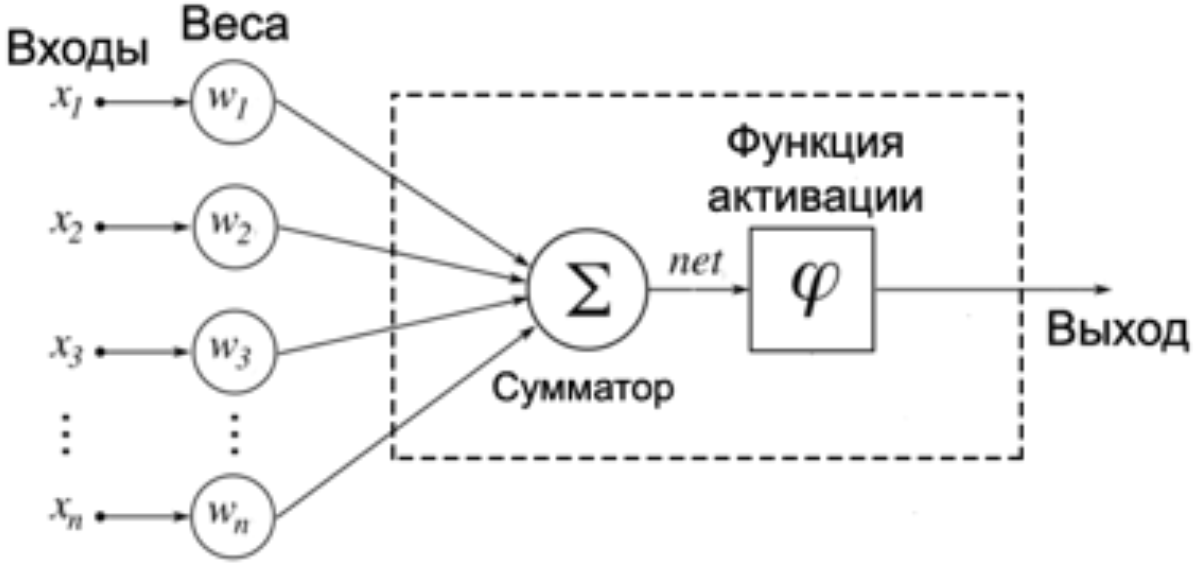
Структура нейронной сети

Нейрон — это вычислительная единица, которая получает информацию, производит над ней простые вычисления и передает ее дальше.

Они делятся на три основных типа: входной, скрытый и выходной.



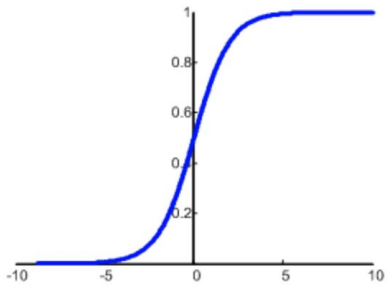
Структура нейронной сети



$$net = \sum_{i=1}^{i=n} w_i \cdot x_i = w^T \cdot x.$$

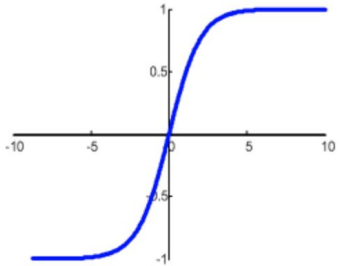
Функция активации

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



Сигмоид

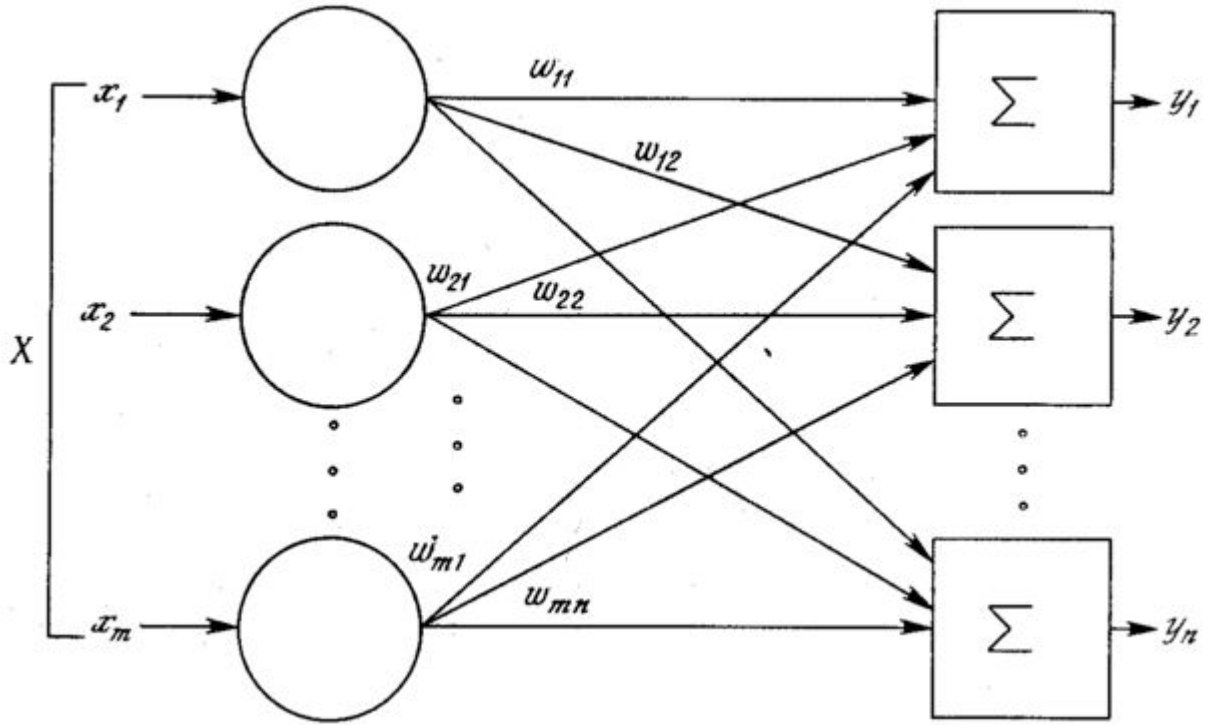
$$f(x) = \frac{e^{2x}-1}{e^{2x}+1}$$



Гиперболический тангенс

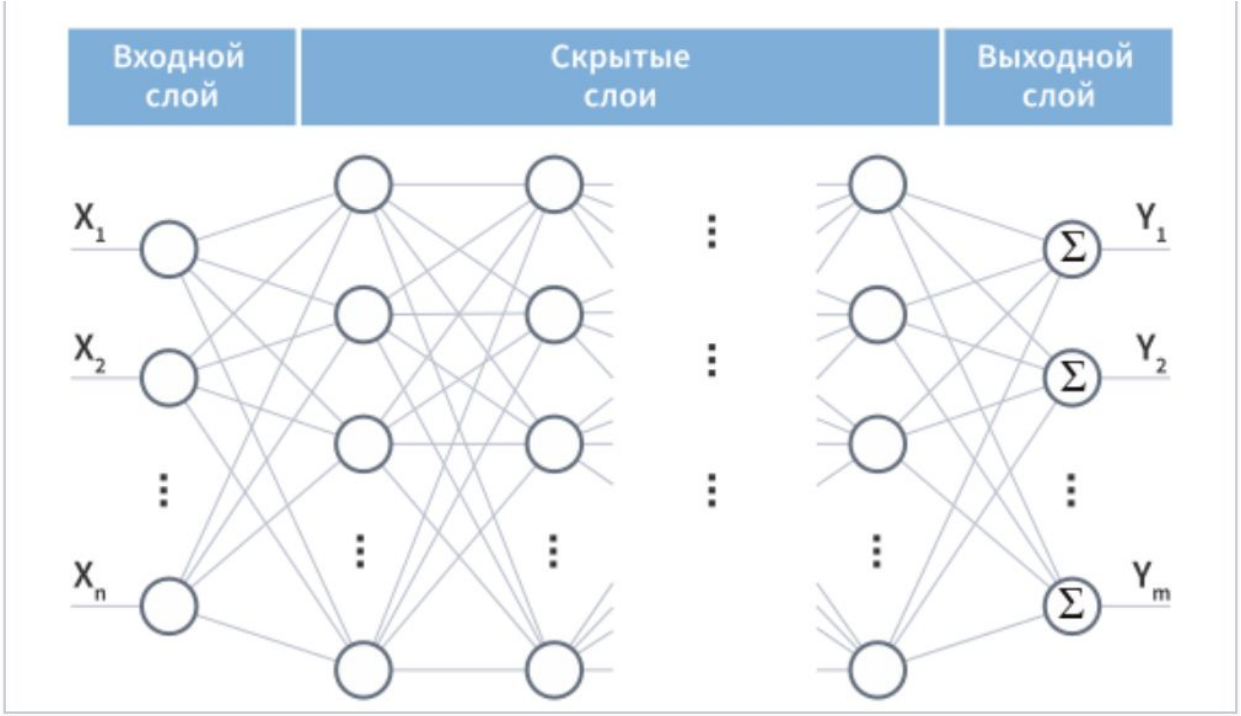
Однослойные нейронные сети

Однослойная нейронная сеть (англ. *Single-layer neural network*) — сеть, в которой сигналы от входного слоя сразу подаются на выходной слой, который и преобразует сигнал и сразу же выдает ответ.



Многослойные нейронные сети

Многослойная нейронная сеть (англ. *Multilayer neural network*) — нейронная сеть, состоящая из входного, выходного и расположенного (ых) между ними одного (нескольких) скрытых слоев нейронов.



Обучение нейронной сети

Обучение нейронной сети — поиск такого набора весовых коэффициентов, при котором входной сигнал после прохода по сети преобразуется в нужный нам выходной.

Обучающая выборка — конечный набор входных сигналов (иногда вместе с правильными выходными сигналами), по которым происходит обучение сети.

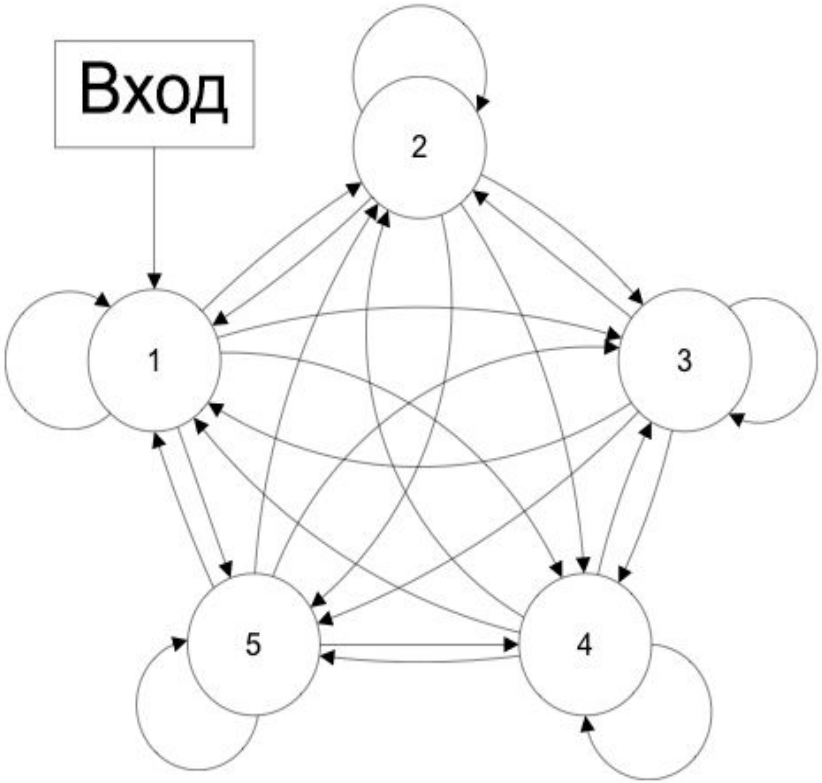
Тестовая выборка — конечный набор входных сигналов (иногда вместе с правильными выходными сигналами), по которым происходит оценка качества работы сети.

Feedforward neural networks and RNN

Сети прямого распространения (англ. *Feedforward neural network*) — искусственные нейронные сети, в которых сигнал распространяется строго от входного слоя к выходному.

Сети с обратными связями (англ. *Recurrent neural network*) — искусственные нейронные сети, в которых выход нейрона может вновь подаваться на его вход.

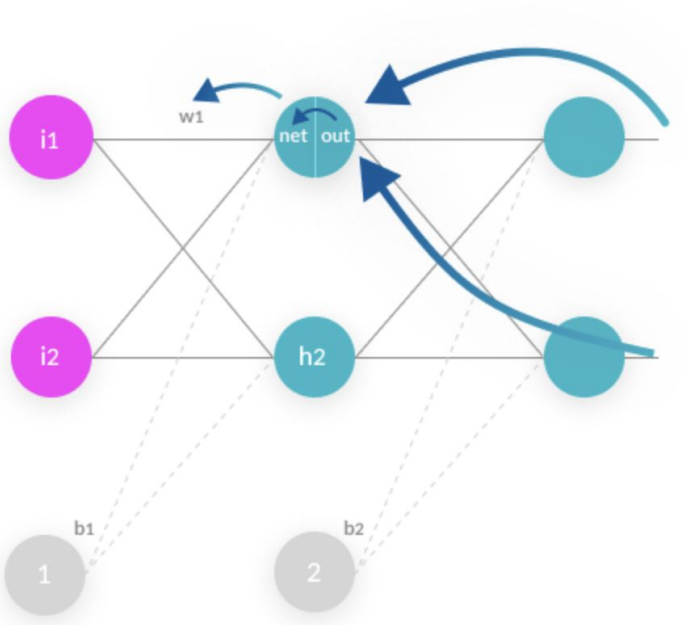
(рисунок справа)



Что такое **Backpropagation** (Метод Обратного Распространения)?

Это метод обучения нейронной сети, основная идея которого состоит в распространении сигналов ошибки от выходов к ее входам.

(чтобы было более понятно и наглядно будем использовать метод нахождения дельты)



Backpropagation

1 шаг. Считаем дельту δ output

$$1)\ \delta_O = (OUT_{ideal} - OUT_{actual}) * f'(IN)$$

2 шаг. Считаем δ hidden для каждого слоя

$$2)\ \delta_H = f'(IN) * \sum(w_i * \delta_i)$$

3 шаг. Считаем градиент (A - точка в начале синапса, B - в конце)

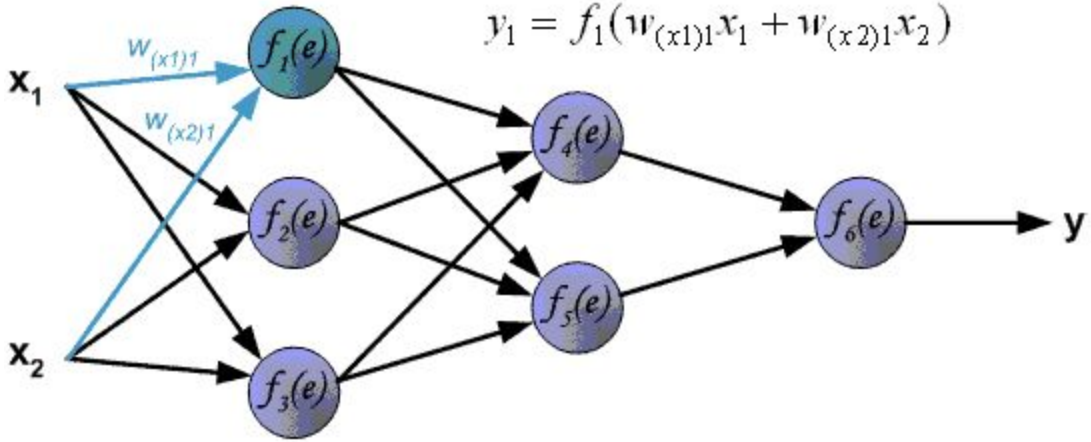
$$f'(IN) = \begin{cases} f_{sigmoid} = (1 - OUT) * OUT \\ f_{tangh} = 1 - OUT^2 \end{cases}$$

$$GRAD_B^A = \delta_B * OUT_A$$

4 шаг. Обновляем веса, E-скорость обучения, α-момент

$$\Delta w_i = E * GRADw + \alpha * \Delta w_{i-1}$$

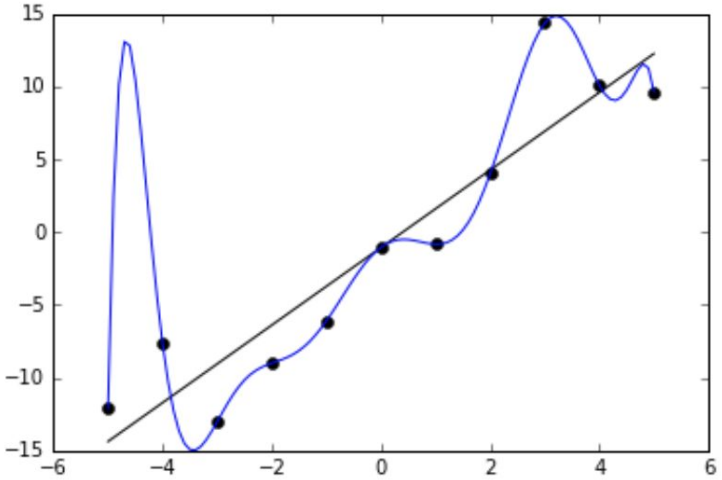
FP



Что такое переобучение и как с этим бороться?

Переобучение (overfitting) - это состояние нейросети, когда она перенасыщена данными.

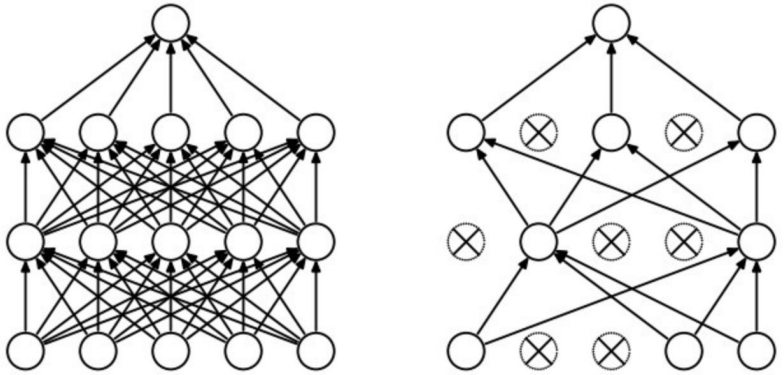
Решение проблемы: **Dropout**.



Dropout (метод исключения)

Главная идея *Dropout — вместо обучения одной НС обучить ансамбль нескольких НС, а затем усреднить полученные результаты.*

Сети для обучения получаются с помощью **исключения** из сети (dropping out) нейронов с вероятностью p, а вероятность того, что нейрон останется в сети, составляет $q=1-p$.



(a) Standard Neural Net

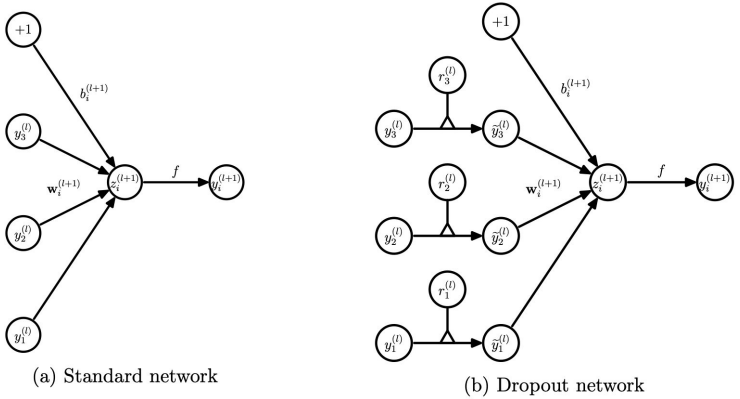
(b) After applying dropout.

Как работает dropout?

$$
\begin{aligned}
z_i^{(l+1)} &= \mathbf{w}_i^{(l+1)} \mathbf{y}^l + b_i^{(l+1)}, \\
y_i^{(l+1)} &= f(z_i^{(l+1)}),
\end{aligned}
$$

Let l ∈ {1, . . . , L} index the hidden layers of the network. Let z(l) denote the vector of inputs into layer l, y(l) denote the vector of outputs from layer l (y(0) = x is the input). W(l) and b(l) are the weights and biases at layer l. f is any activation function.

(The feed-forward operation of a standard neural network for l ∈ {0, . . . , L − 1} and any hidden unit i)



(a) Standard network

(b) Dropout network

$$
\begin{aligned}
r_j^{(l)} &\sim \text{Bernoulli}(p), \\
\widetilde{\mathbf{y}}^{(l)} &= \mathbf{r}^{(l)} * \mathbf{y}^{(l)}, \\
z_i^{(l+1)} &= \mathbf{w}_i^{(l+1)} \widetilde{\mathbf{y}}^{l} + b_i^{(l+1)}, \\
y_i^{(l+1)} &= f(z_i^{(l+1)}).
\end{aligned}
$$

With dropout, the feed-forward operation becomes (2 picture)

For any layer l, r(l) is a vector of independent Bernoulli random variables each of which has probability p of being 1.

Обратный (Inverted) Dropout

В данном случае мы умножаем функцию активации на коэффициент не во время тестового этапа, а во время обучения.

Коэффициент равен обратной величине вероятности того, что нейрон останется в сети:

(здесь X_i случайные величины, a - ф-я активации)

На этапе обучения: $O_i = \frac{1}{q} X_i a(\sum_{k=1}^{d_i} w_k x_k + b),$

На этапе тестирования: $O_i = a(\sum_{k=1}^{d_i} w_k x_k + b)$

Почему inverted чаще используют?

В случае прямого Dropout мы вынуждены изменять нейронную сеть для проведения тестирования, так как без умножения на q нейрон будет возвращать значения выше, чем те, которые ожидают получить последующие нейроны; именно поэтому реализация обратного Dropout встречается чаще.

Dropout и другие регуляризаторы

Dropout часто используется с L2-нормализацией и другими методами ограничения параметров. Методы нормализации помогают поддерживать невысокие значения параметров модели.

Dropout сам по себе не может предотвратить рост значений параметров на этапе обновления. Более того, обратный Dropout ведет к тому, чтобы шаги обновления стали даже больше, как показано ниже.

метода нормализации, ограничивающие значения параметров, могут упростить процесс выбора скорости обучения.

L2-нормализация

Вкратце, L2-нормализация представляет собой дополнительный элемент функции потерь, где $\lambda \in [0,1]$ — гиперпараметр, называемый “сила регуляризации” (regularization strength), F(W;x) — модель, а ϵ — функция ошибки между реальным значением y и предсказанным значением y:

$$\mathcal{L}(y, \hat{y}) = \epsilon(y, F(W; x)) + \frac{\lambda}{2} W^2$$

В итоге Если η — коэффициент скорости обучения, то параметр $w \in W$ обновляется на следующую величину:

$$w \leftarrow w - \eta(\frac{\partial F(W; x)}{\partial w} + \lambda w)$$

Обратный Dropout и L2-нормализация

$$w \leftarrow w - \eta(\frac{1}{q}\frac{\partial F(W;x)}{\partial w} + \lambda w)$$

Нетрудно заметить, что в случае обратного Dropout скорость обучения масштабируется с помощью коэффициента q.
Так как q принадлежит интервалу [0;1], отношение между η и q может принимать значения из следующего интервала:

$$r(q) = \frac{\eta}{q} \in [\eta = \lim_{q \to 1} r(q), +\infty = \lim_{q \to 0} r(q)]$$

Отсюда = > q будем называть ускоряющим множителем (**boosting factor**), так как он увеличивает скорость обучения. r(q) будем называть эффективной скоростью обучения (**effective learning rate**).

Batch normalization

Цель: ускорить обучение

— метод, который позволяет повысить производительность и стабилизировать работу искусственных нейронных сетей.

Суть данного метода: некоторым слоям нейронной сети на вход подаются данные, предварительно обработанные и имеющие **нулевое** математическое ожидание и **единичную** дисперсию.

Пакет

- Стохастический градиентный спуск (англ. stochastic gradient descent) — реализация, в которой на каждой итерации алгоритма из обучающей выборки каким-то (случайным) образом выбирается только один объект;
- Пакетный (батч) (англ. batch gradient descent) — реализация градиентного спуска, когда на каждой итерации обучающая выборка просматривается целиком, и только после этого изменяются веса модели.

Ковариантный сдвиг

Пакетная нормализация уменьшает величину, на которую смещаются значения узлов в скрытых слоях (т.н. **ковариантный сдвиг** (англ. covariance shift)).

Верхние две строки роз показывают первое подмножество данных, а нижние две строки показывают другое подмножество. Два подмножества имеют разные пропорции изображения роз. На графиках показано распределение двух классов в пространстве объектов с использованием красных и зеленых точек. Синяя линия показывает границу между двумя классами.

Свойства пакетной нормализации

- достигается более быстрая сходимость моделей
- пакетная нормализация позволяет каждому слою сети обучатся более независимо от других слоев
- становится возможным использование более высокого темпа обучения,
- пакетная нормализация в каком-то смысле также является механизмом регуляризации: данный метод привносит в выходы узлов скрытых слоев некоторый шум
- модели становятся менее чувствительны к начальной инициализации весов.

Алгоритм пакетной оптимизации

Вход: значения x из пакета $B = \{x_1, \dots, x_m\}$; настраиваемые параметры γ, β; константа ϵ для вычислительной устойчивости.
Выход: $\{y_i = BN_{\gamma,\beta}(x_i)\}$

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \quad \text{// математическое ожидание пакета}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \quad \text{// дисперсия пакета}$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad \text{// нормализация}$$

$$y_i = \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i) \quad \text{// сжатие и сдвиг}$$

сжатие и сдвиг нужны, т к нормализация входа слоя нейронной сети может изменить представление данных в слое.

Обучение

l - loss function

$$\frac{\partial l}{\partial \hat{x}_i} = \frac{\partial l}{\partial y_i} \cdot \gamma \, (1)$$

$$\frac{\partial l}{\partial \sigma_B^2} = \sum_{i=1}^{m} \frac{\partial l}{\partial \hat{x}_i} \cdot (x_i - \mu_B) \cdot \frac{-1}{2} (\sigma_B^2 + \epsilon)^{-3/2} \, (2)$$

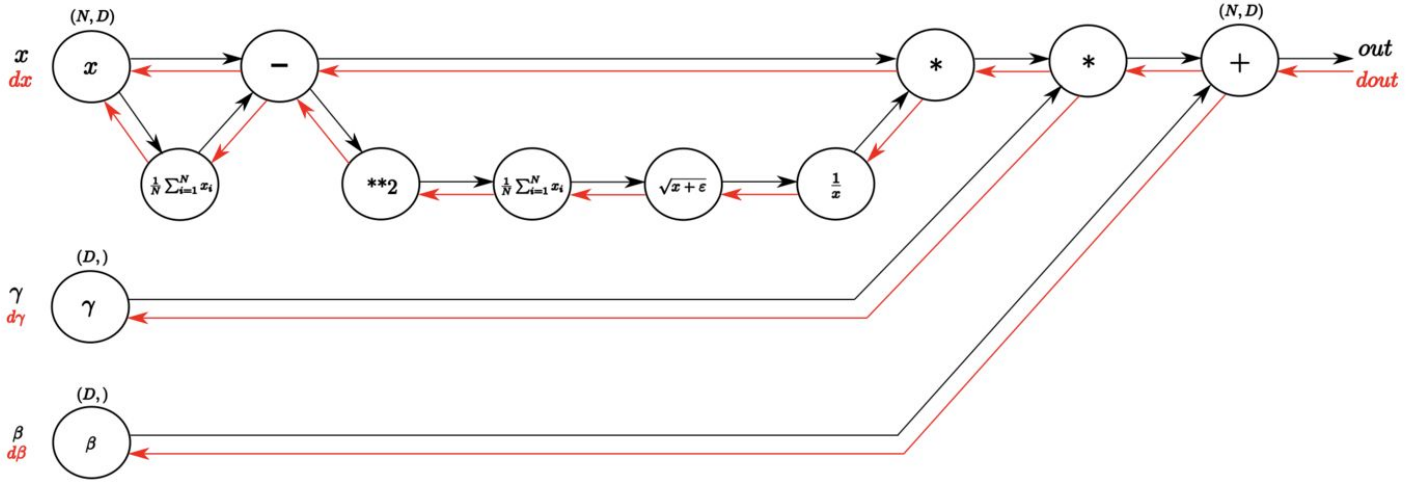
$$\frac{\partial l}{\partial \mu_B} = \left(\sum_{i=1}^{m} \frac{\partial l}{\partial \hat{x}_i} \cdot \frac{-1}{\sqrt{\sigma_B^2 + \epsilon}} \right) + \frac{\partial l}{\partial \sigma_B^2} \cdot \frac{\sum_{i=1}^{m} -2(x_i - \mu_B)}{m} \, (3)$$

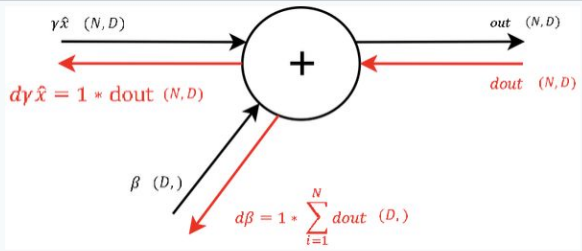
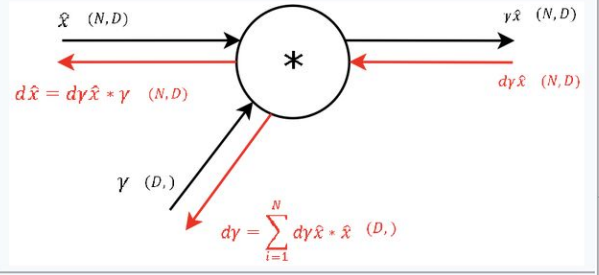
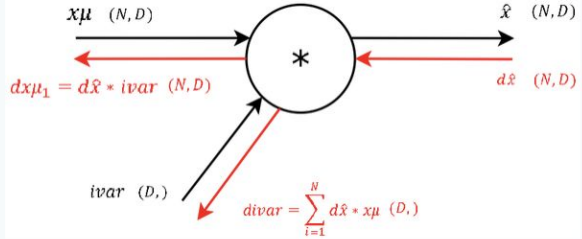
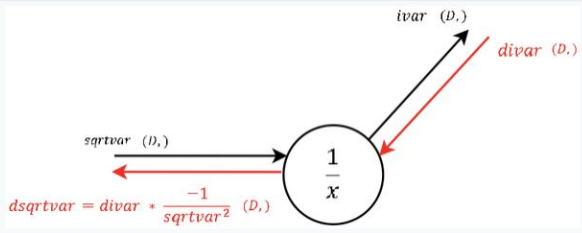
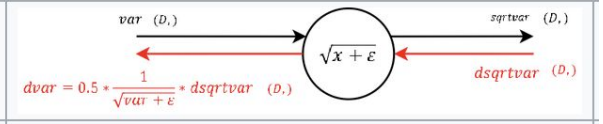
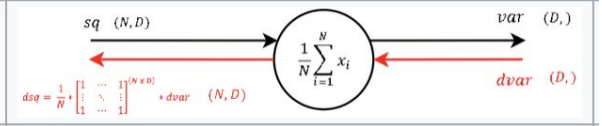
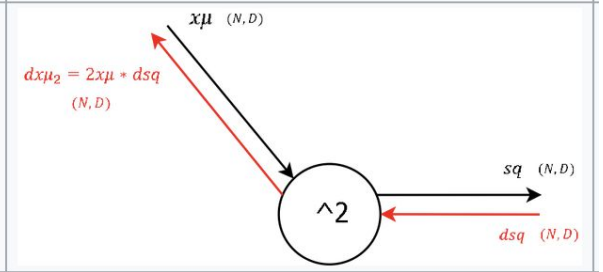
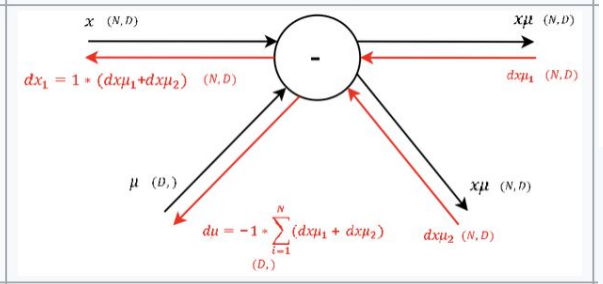
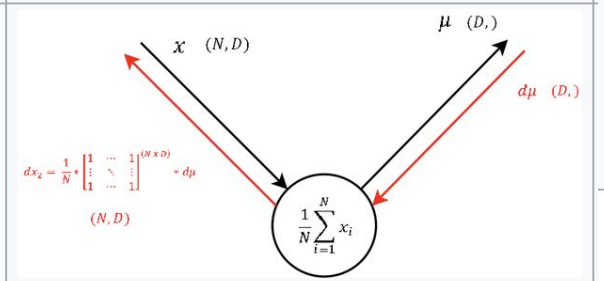
$$\frac{\partial l}{\partial x_i} = \frac{\partial l}{\partial \hat{x}_i} \cdot \frac{1}{\sqrt{\sigma_B^2 + \epsilon}} + \frac{\partial l}{\partial \sigma_B^2} \cdot \frac{2(x_i - \mu_B)}{m} + \frac{\partial l}{\partial \mu_B} \cdot \frac{1}{m} \, (4)$$

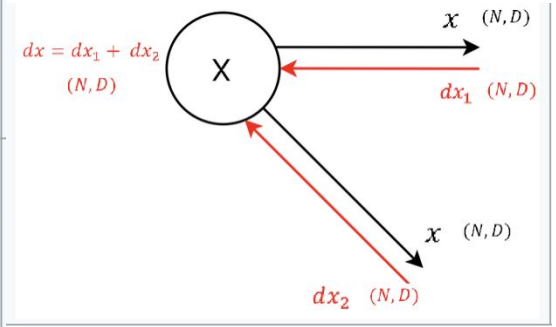
$$\frac{\partial l}{\partial \gamma} = \sum_{i=1}^{m} \frac{\partial l}{\partial y_i} \cdot \hat{x}_i \, (5)$$

$$\frac{\partial l}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial l}{\partial y_i} \, (6)$$

Обучение



Шаг	Изображение
1	
2	
3	
4	
5	
6	
7	
8	
9	



Еще о методах регуляризации

- Расширение обучающего множества (data augmentation)
- Ранняя остановка (early stopping)

Вопросы

1. В чем плюсы и минусы использования нейронных сетей?
2. В чем заключается алгоритм Backpropagation?
3. Как работает метод dropout? Почему и для чего его используют?

Источники

- http://neerc.ifmo.ru/wiki/index.php?title=Нейронные_сети,_перцептрон
- https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/
- https://habr.com/ru/company/wunderfund/blog/315476/
- http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf
- https://habr.com/ru/post/313216/
- http://neerc.ifmo.ru/wiki/index.php?title=Batch-normalization
- https://habr.com/ru/company/wunderfund/blog/330814/