

Глубокое обучение и вообще

Соловей Влад

1 декабря 2020 г.

Посиделка 4: эвристики для обучения сеток

Agenda

- И еще раз об backprop
- Какими бывают функции активации
- Инициализация весов в нейросетках
- Нормализация по батчам
- Dropout
- Другие эвристики, используемые при обучении нейронных сетей

Так как же обучить нейросетку?



Ты необучаем!

Нейросеть — сложная функция

- Прямое распространение ошибки (forward propagation):

$$X \Rightarrow X \cdot W_1 \Rightarrow f(X \cdot W_1) \Rightarrow f(X \cdot W_1) \cdot W_2 \Rightarrow \dots \Rightarrow \hat{y}$$

- Считаем потери:

$$Loss = \frac{1}{2}(y - \hat{y})^2$$

- Для обучения нужно использовать градиентный спуск

Как обучить нейросеть?

$$L(W_1, W_2) = \frac{1}{2} \cdot (y - f(X \cdot W_1) \cdot W_2)^2$$

Секрет успеха в умении брать производную и градиентном спуске.

$$f(g(x))' = f'(g(x)) \cdot g'(x)$$

$$\frac{\partial L}{\partial W_2} = -(y - f(X \cdot W_1) \cdot W_2) \cdot f(X \cdot W_1)$$

$$\frac{\partial L}{\partial W_1} = -(y - f(X \cdot W_1) \cdot W_2) \cdot W_2 f'(X \cdot W_1) \cdot W_1$$

Как обучить нейросеть?

$$L(W_1, W_2) = \frac{1}{2} \cdot (y - f(X \cdot W_1) \cdot W_2)^2$$

Секрет успеха в умении брать производную и градиентном спуске.

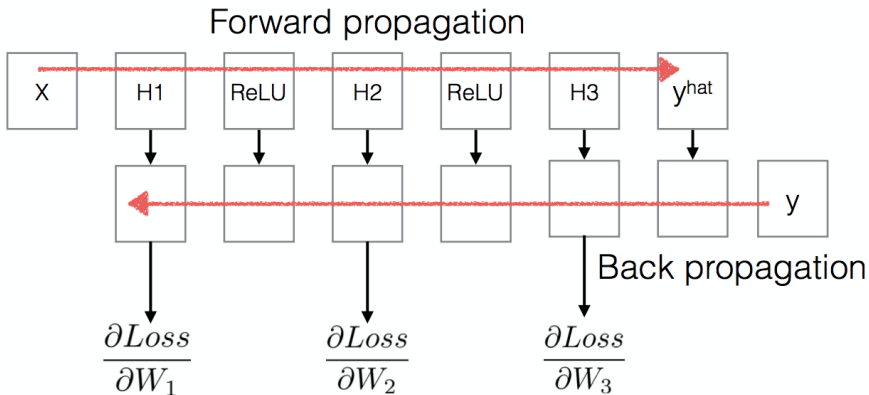
$$f(g(x))' = f'(g(x)) \cdot g'(x)$$

$$\frac{\partial L}{\partial W_2} = -(y - f(X \cdot W_1) \cdot W_2) \cdot f(X \cdot W_1)$$

$$\frac{\partial L}{\partial W_1} = -(y - f(X \cdot W_1) \cdot W_2) \cdot W_2 f'(X \cdot W_1) \cdot W_1$$

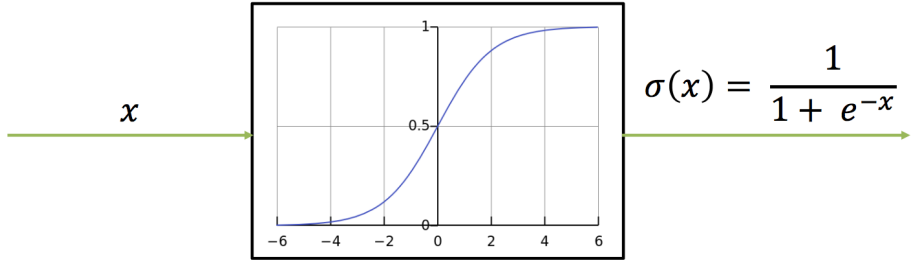
Дважды ищем одно и то же \Rightarrow оптимизация поиска производных даст нам алгоритм обратного распространения ошибки (back-propagation)

Back-propagation

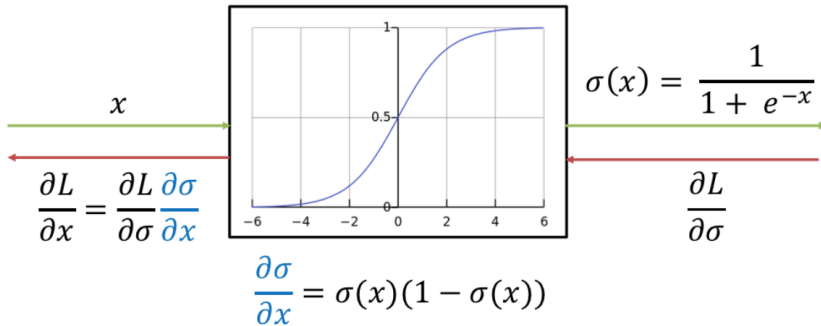


Какими бывают функции активации
и как через них пробросить градиент

Sigmoid activation



Sigmoid activation



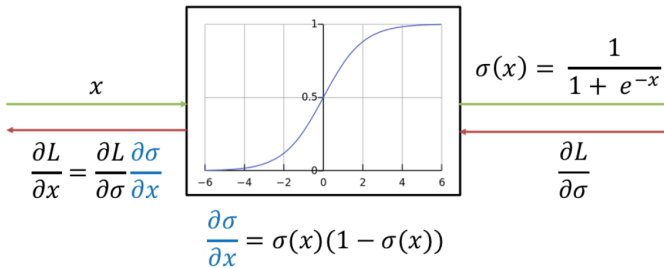
Паралич сети

- В случае сигмоиды $\sigma'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Сигмоида принимает значения на отрезке $[0; 1]$, значит максимальное значение её производной это $\frac{1}{4}$
- Если сеть очень глубокая, происходит **затухание градиента**
- Градиент затухает экспоненциально \Rightarrow сходимость замедляется, более ранние веса обновляются дольше, более глубокие веса быстрее \Rightarrow значение градиента становится ещё меньше \Rightarrow наступает **паралич сети**
- В сетях с небольшим числом слоёв этот эффект незаметен

Центрирование

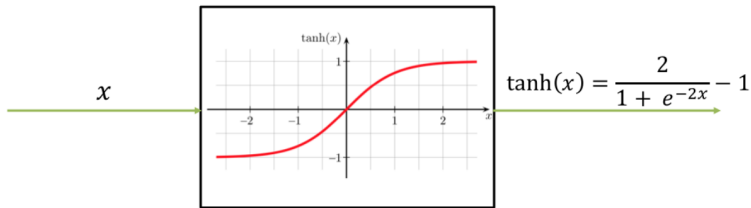
- Сигмоида не центрирована относительно нуля
- Выход слоя мы обычно находим как $o_i = \sigma(h_i)$, он всегда положительный, значит градиент по весам, идущим на вход в текущий нейрон тоже положительные \Rightarrow они обновляются в одинаковом направлении
- Сходимость идёт медленнее и зигзагообразно, но идёт

Sigmoid activation



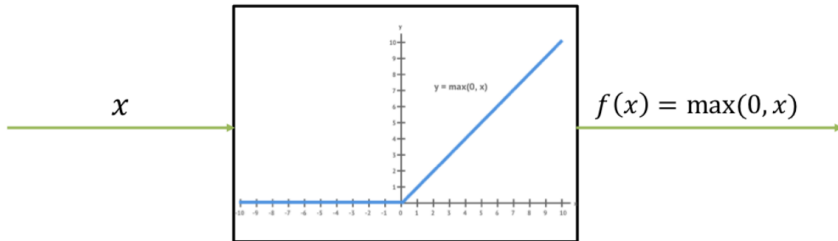
- Способствует затуханию градиента
- Не центрирована относительно нуля
- Вычислять e^x дорого

Tanh activation



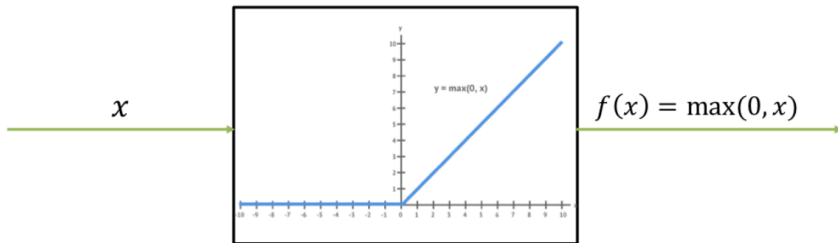
- Центрирован относительно нуля
- Всё ещё похож на сигмоиду
- $f'(x) = 1 - f(x)^2 \Rightarrow$ затухание градиента

ReLU activation



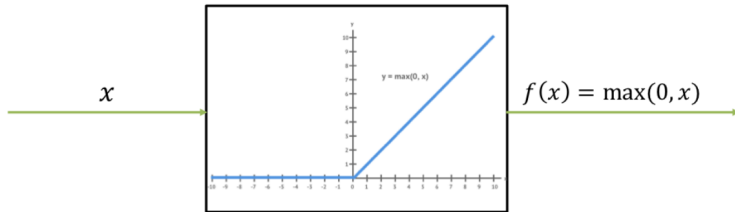
- Быстро вычисляется
- Градиент не затухает
- Сходимость сеток ускоряется

ReLU activation



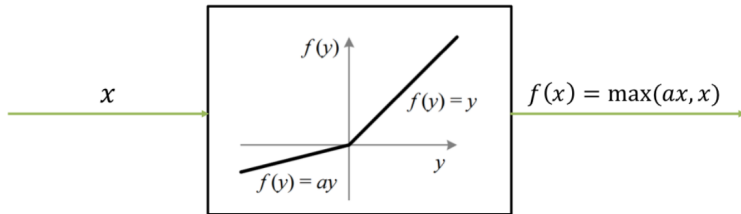
- Сетка может умереть, если активация занулитесь на всех нейронах
- Не центрирован относительно нуля

Зануление ReLU



- $f(x) = \max(0, w_0 + w_1 \cdot h_1 + \dots + w_k \cdot h_k)$
- Если w_0 инициализировано большим отрицательным числом, нейрон сразу умирает \Rightarrow надо аккуратно инициализировать веса

Leaky ReLU activation



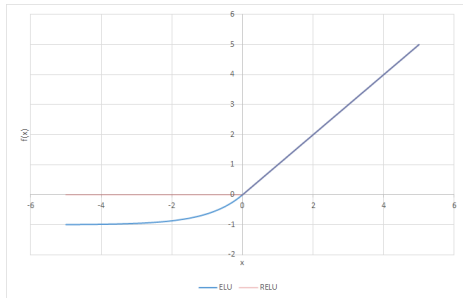
- Как ReLU, но не умирает, всё ещё легко считается
- Производная может быть любого знака
- Важно, чтобы $a \neq 1$, иначе линейность

Что же выбрать

- Обычно начинают с $ReLU$, если сетка умирает, берут $LeakyReLU$
- $ReLU$ — стандартный выбор для свёрточных сетей
- В рекуррентных сетках чаще всего предпочитается $tanh$
- На самом деле это не очень важно, нужно держать в голове свойства функций, о которых выше шла речь и понимать, что от перебора функций обычно выигрыш в качестве довольно низкий
- Но есть и исключения ...

Краткий обзор функций активаций: <https://arxiv.org/pdf/1804.02763.pdf>

ELU activation



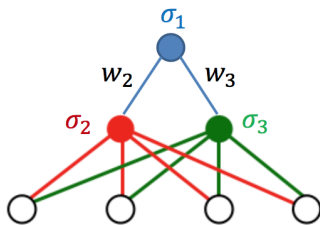
- ELU улучшает сходимость для глубоких сетей

$$f(x) = \begin{cases} x, & x \geq 0 \\ \alpha \cdot (e^x - 1), & x < 0 \end{cases}$$

<https://arxiv.org/pdf/1511.07289.pdf>

Инициализация весов

Инициализация весов

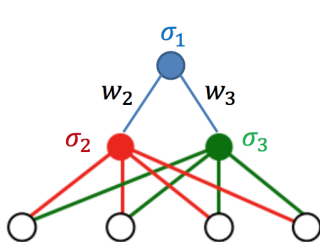


$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \sigma_1} \sigma_1 (1 - \sigma_1) \sigma_2$$

$$\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial \sigma_1} \sigma_1 (1 - \sigma_1) \sigma_3$$

- Что будет, если инициализировать веса нулями?

Инициализация весов

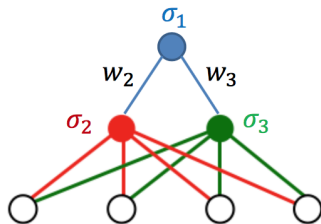


$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \sigma_1} \sigma_1 (1 - \sigma_1) \sigma_2$$

$$\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial \sigma_1} \sigma_1 (1 - \sigma_1) \sigma_3$$

- Что будет, если инициализировать веса нулями?
- σ_2 и σ_3 будут обновляться одинаково

Инициализация весов

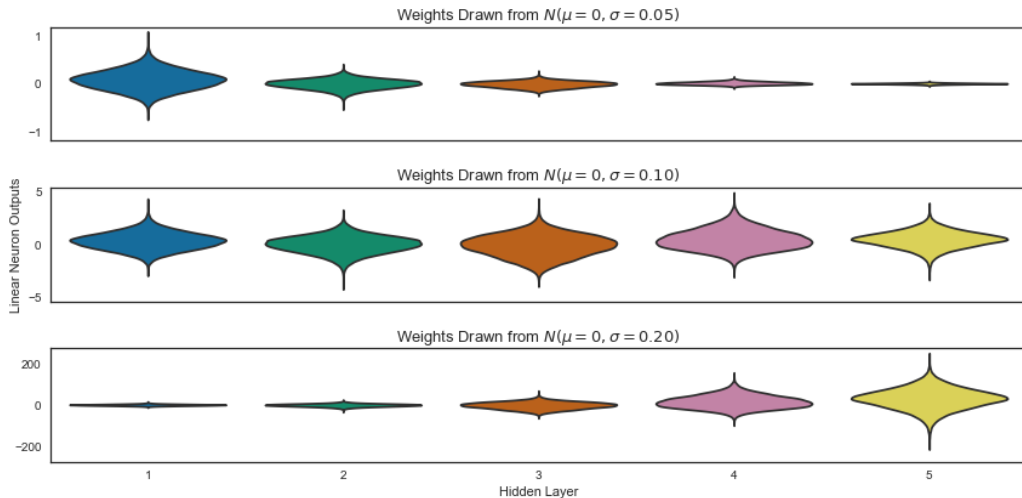


$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \sigma_1} \sigma_1 (1 - \sigma_1) \sigma_2$$

$$\frac{\partial L}{\partial w_3} = \frac{\partial L}{\partial \sigma_1} \sigma_1 (1 - \sigma_1) \sigma_3$$

- Хочется уничтожить симметрию
- Обычно инициализируют маленькими случайными числами из какого-то распределения (нормальное, равномерное)

Симметричный случай



Инициализация весов

- Наши признаки X пришли к нам из какого-то распределения
- Выход слоя $f(XW)$ будет принадлежать другому распределению
- Если инициализировать веса неправильно, дисперсия распределения може от слоя к слою затухать (сигнал будет теряться) либо наоборот, возрастать (сигнал будет рассеиваться)
- Эмпирически было выяснено, что это может портить сходимость для глубоких сетей
- Хочется контролировать дисперсию

Инициализация весов

- Посмотрим на выход нейрона перед активацией:

$$h_i = w_0 + \sum_{i=1}^{n_{in}} w_i x_i$$

- Дисперсия h_i выражается через дисперсии x и w
- Она не зависит от константы w_0
- Будем считать, что веса $w_1, \dots, w_k \sim iid$, наблюдения $x_1, \dots, x_n \sim iid$, а ещё x_i и w_i независимы между собой

Инициализация весов (симметричный случай)

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\ &= \sum_{i=1}^{n_{in}} [\text{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\text{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) =\end{aligned}$$

Инициализация весов (симметричный случай)

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\ &= \sum_{i=1}^{n_{in}} [\text{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\text{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) =\end{aligned}$$

- Если функция активации симметричная, тогда $E(x_i) = 0$. Будем инициализировать веса с нулевым средним, тогда $E(w_i) = 0$.

Инициализация весов (симметричный случай)

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\ &= \sum_{i=1}^{n_{in}} [\text{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\text{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) = \\ &= \sum_{i=1}^{n_{in}} \text{Var}(x_i) \cdot \text{Var}(w_i)\end{aligned}$$

- Если функция активации симметричная, тогда $E(x_i) = 0$. Будем инициализировать веса с нулевым средним, тогда $E(w_i) = 0$.

Инициализация весов (симметричный случай)

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\ &= \sum_{i=1}^{n_{in}} [\text{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\text{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) = \\ &= \sum_{i=1}^{n_{in}} \text{Var}(x_i) \cdot \text{Var}(w_i) = \text{Var}(x) \cdot [n_{in} \cdot \text{Var}(w)]\end{aligned}$$

Инициализация весов (симметричный случай)

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\ &= \sum_{i=1}^{n_{in}} [\text{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\text{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) = \\ &= \sum_{i=1}^{n_{in}} \text{Var}(x_i) \cdot \text{Var}(w_i) = \text{Var}(x) \cdot \underbrace{[n_{in} \cdot \text{Var}(w)]}_{=1}\end{aligned}$$

Плохая инициализация весов

Пусть

$$w_i \sim U \left[-\frac{1}{\sqrt{n_{in}}}; \frac{1}{\sqrt{n_{in}}} \right],$$

тогда

$$\text{Var}(w_i) = \frac{1}{12} \cdot \left(\frac{1}{\sqrt{n_{in}}} + \frac{1}{\sqrt{n_{in}}} \right)^2 = \frac{1}{3n_{in}} \Rightarrow \text{Var}(h_i) = \frac{1}{3}$$

Получаем затухание!

Немного лучше

Пусть

$$w_i \sim U \left[-\frac{\sqrt{3}}{\sqrt{n_{in}}}; \frac{\sqrt{3}}{\sqrt{n_{in}}} \right],$$

тогда

$$\text{Var}(w_i) = \frac{1}{12} \cdot \left(\frac{\sqrt{3}}{\sqrt{n_{in}}} + \frac{\sqrt{3}}{\sqrt{n_{in}}} \right)^2 = \frac{1}{n_{in}} \Rightarrow \text{Var}(h_i) = 1$$

Немного лучше

Пусть

$$w_i \sim U \left[-\frac{\sqrt{3}}{\sqrt{n_{in}}}; \frac{\sqrt{3}}{\sqrt{n_{in}}} \right],$$

тогда

$$\text{Var}(w_i) = \frac{1}{12} \cdot \left(\frac{\sqrt{3}}{\sqrt{n_{in}}} + \frac{\sqrt{3}}{\sqrt{n_{in}}} \right)^2 = \frac{1}{n_{in}} \Rightarrow \text{Var}(h_i) = 1$$

При forward pass на вход идёт n_{in} наблюдений, при backward pass на вход идёт n_{out} градиентов \Rightarrow **канал с дисперсией может быть непостоянным, если число весов от слоя к слою сильно колеблется**

Инициализация Ксавье (Глорота)

Для неодинаковых размеров слоёв невозможно удовлетворить обоим условиям, поэтому обычно усредняют:

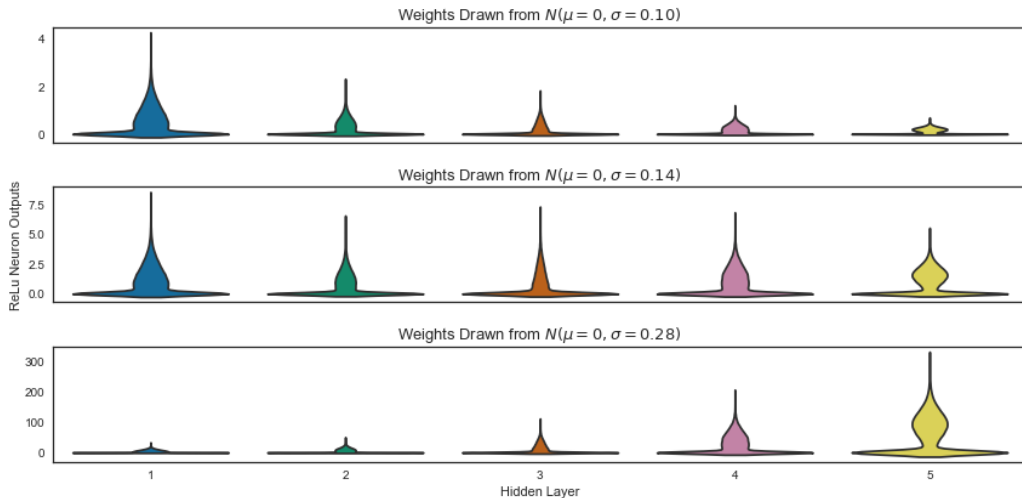
$$w_i \sim U \left[-\frac{\sqrt{6}}{\sqrt{n_{out} + n_{in}}}; \frac{\sqrt{6}}{\sqrt{n_{out} + n_{in}}} \right],$$

Такая инициализация называется **инициализацией Ксавье (или Глоро)**

По аналогии можно найти формулу для дисперсии нормального распределения, но это уже семинарская задача :)

<http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf>

Несимметричный случай



Инициализация Хе

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\ &= \sum_{i=1}^{n_{in}} [\mathbf{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\mathbf{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i)]\end{aligned}$$

- Когда нет симметрии, можно занулить только второе слагаемое

Инициализация Хе

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\ &= \sum_{i=1}^{n_{in}} [\text{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\text{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) = \\ &= \sum_{i=1}^{n_{in}} [\text{E}(x_i)]^2 \cdot \text{Var}(w_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) = \sum_{i=1}^{n_{in}} \text{Var}(w_i) \cdot E(x_i^2)\end{aligned}$$

- Когда нет симметрии, можно занулить только второе слагаемое

Инициализация Хе

$$\begin{aligned}\text{Var}(h_i) &= \text{Var}\left(\sum_{i=1}^{n_{in}} w_i x_i\right) = \sum_{i=1}^{n_{in}} \text{Var}(w_i x_i) = \\&= \sum_{i=1}^{n_{in}} [\mathbf{E}(x_i)]^2 \cdot \text{Var}(w_i) + [\mathbf{E}(w_i)]^2 \cdot \text{Var}(x_i) + \text{Var}(x_i) \cdot \text{Var}(w_i)] = \\&= \sum_{i=1}^{n_{in}} [\mathbf{E}(x_i)]^2 \cdot \text{Var}(w_i) + \text{Var}(x_i) \cdot \text{Var}(w_i) = \sum_{i=1}^{n_{in}} \text{Var}(w_i) \cdot E(x_i^2) = \\&= E(x^2) \cdot [n_{in} \cdot \text{Var}(w)]\end{aligned}$$

Инициализация Хе

$$\begin{aligned}\text{Var}(h_i) &= E(x_i^2) \cdot [n_{in} \cdot \text{Var}(w)] \\ x_i &= \max(0; h_{i-1})\end{aligned}$$

Инициализация Хе

$$\begin{aligned}\text{Var}(h_i) &= E(x_i^2) \cdot [n_{in} \cdot \text{Var}(w)] \\ x_i &= \max(0; h_{i-1})\end{aligned}$$

Если h_{i-1} симметрично распределён относительно нуля, тогда:

$$E(x_i^2) = \frac{1}{2} \cdot \text{Var}(h_{i-1})$$

Инициализация Хе

$$\begin{aligned}\text{Var}(h_i) &= E(x_i^2) \cdot [n_{in} \cdot \text{Var}(w)] \\ x_i &= \max(0; h_{i-1})\end{aligned}$$

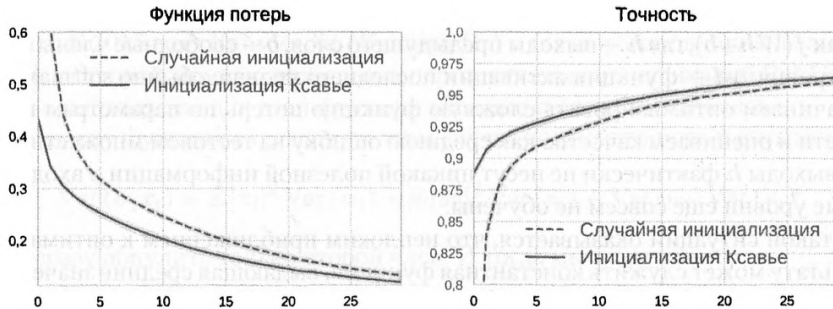
Если h_{i-1} симметрично распределён относительно нуля, тогда:

$$\begin{aligned}E(x_i^2) &= \frac{1}{2} \cdot \text{Var}(h_{i-1}) \\ \text{Var}(h_i) &= \frac{1}{2} \cdot \text{Var}(h_{i-1}) \cdot [n_{in} \cdot \text{Var}(w)] \\ \text{Var}(w_i) &= \frac{2}{n_{in}}\end{aligned}$$

Краткие итоги

- Для симметричных функций с нулевым средним используйте инициализацию Ксавье `init="glorot_uniform"` или
- Для ReLU и им подобным инициализацию Хе `init="he_uniform"` или `init="he_nomal"`
- Эти две инициализации корректируют параметры распределений в зависимости от входа и выхода слоя так, чтобы поддерживать дисперсию равной единице

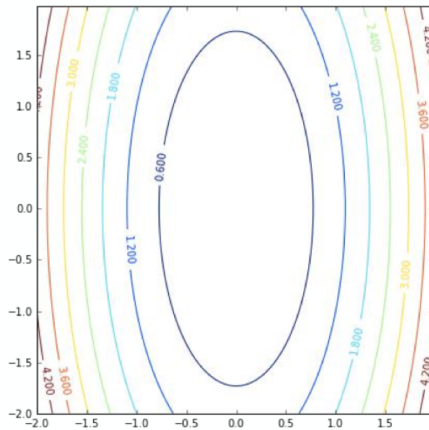
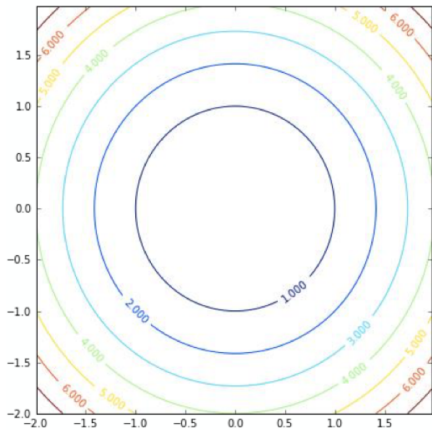
Эксперимент с MNIST



Источник: Николенко, страница 149

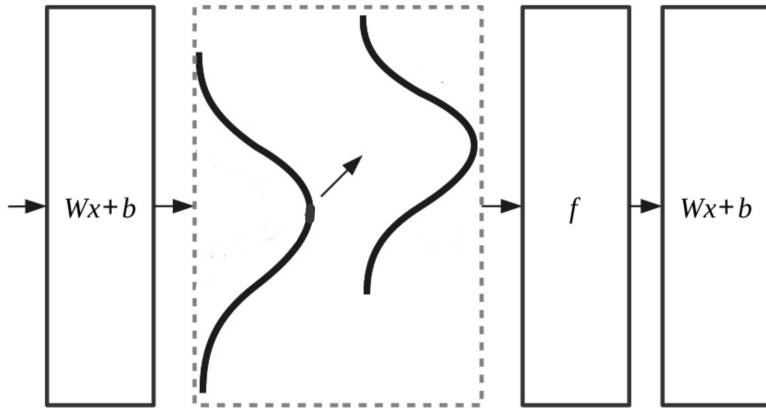
Батч-нормализация

Стандартизация

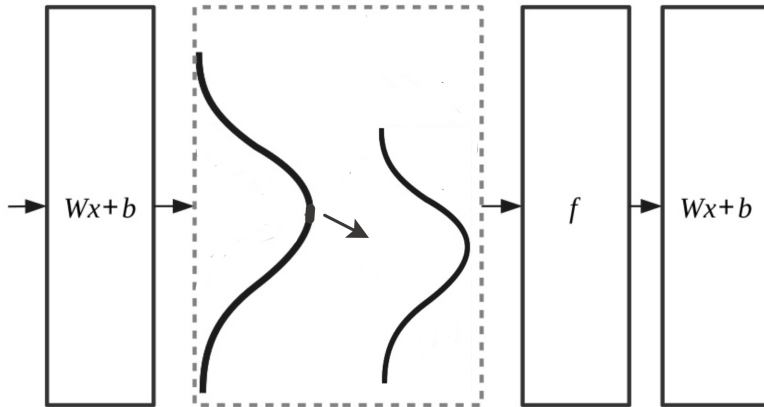


Какая из ситуаций лучше для SGD?

А что внутри?



А что внутри?



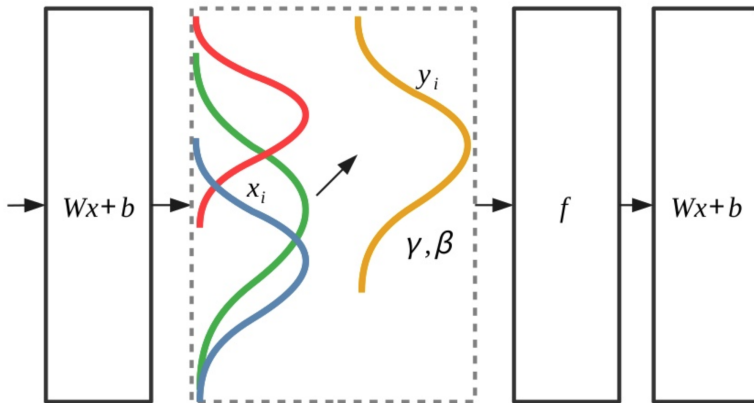
Проблема

- Давайте вместо X на входе использовать $\frac{X - \mu_X}{\sigma_X}$
- Даже если мы стандартизовали вход X , внутри сетки может произойти несчастье и скрытый слой окажется нестандартизован
- Скрытые представления $h = f(XW)$ могут менять своё распределение в процессе обучения, это усложняет его

Проблема

- Давайте вместо X на входе использовать $\frac{X - \mu_X}{\sigma_X}$
- Даже если мы стандартизовали вход X , внутри сетки может произойти несчастье и скрытый слой окажется нестандартизован
- Скрытые представления $h = f(XW)$ могут менять своё распределение в процессе обучения, это усложняет его
- Давайте на каждом слое вместо h использовать $\hat{h} = \frac{h - \mu_h}{\sigma_h}$
- На выход будем выдавать $\beta \cdot \hat{h} + \gamma$, для того, чтобы у нас было больше свободы, параметры β и γ тоже учим

Batch norm (2015)



Batch norm (2015)

- Откуда взять μ_h и σ_h ?
- Оценить по текущему батчу!

$$\mu_h = \alpha \cdot \bar{x}_{batch} + (1 - \alpha) \cdot \mu_h$$

$$\sigma_h = \alpha \cdot \hat{s}_{batch} + (1 - \alpha) \cdot \sigma_h$$

- Коэффициенты β и γ оцениваются в ходе обратного распространения ошибки
- Обучение довольно сильно ускоряется, сходимость улучшается

<https://arxiv.org/pdf/1502.03167.pdf>

Forward pass

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\dots m}\}$;

Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

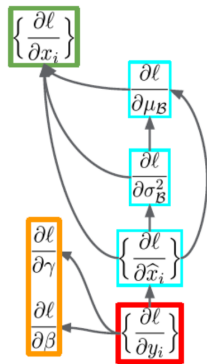
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

Backward pass



$$\frac{\partial \ell}{\partial \hat{x}_i} = \frac{\partial \ell}{\partial y_i} \cdot \gamma$$

$$\frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2} = \sum_{i=1}^m \frac{\partial \ell}{\partial \hat{x}_i} \cdot (x_i - \mu_{\mathcal{B}}) \cdot \frac{-1}{2} (\sigma_{\mathcal{B}}^2 + \epsilon)^{-3/2}$$

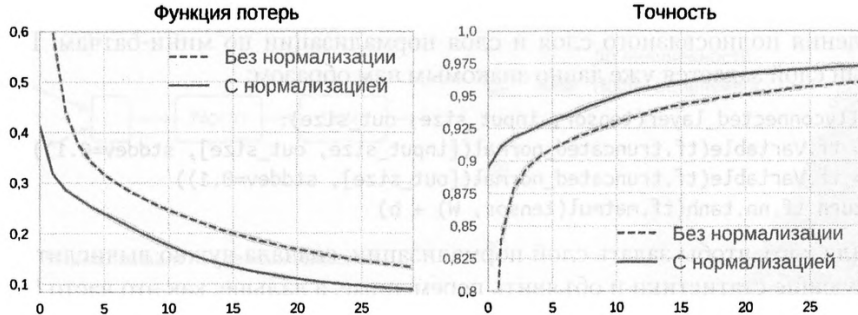
$$\frac{\partial \ell}{\partial \mu_{\mathcal{B}}} = \left(\sum_{i=1}^m \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{-1}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \right) + \frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2} \cdot \frac{\sum_{i=1}^m -2(x_i - \mu_{\mathcal{B}})}{m-1}$$

$$\frac{\partial \ell}{\partial x_i} = \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{1}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} + \frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2} \cdot \frac{2(x_i - \mu_{\mathcal{B}})}{m-1} + \frac{\partial \ell}{\partial \mu_{\mathcal{B}}} \cdot \frac{1}{m}$$

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^m \frac{\partial \ell}{\partial y_i} \cdot \hat{x}_i$$

$$\frac{\partial \ell}{\partial \beta} = \sum_{i=1}^m \frac{\partial \ell}{\partial y_i}$$

Эксперимент с MNIST



Источник: Николенко, страница 160

Трюки

- С батч-нормализацией нужно уменьшить силу Dropout и регуляризацию
- Не забывайте перемешивать обучающую выборку перед каждой новой эпохой, чтобы батчи были разнообразными

http://openaccess.thecvf.com/content_CVPR_2019/papers/Li_Understanding_the_Disharmony_Between_Dropout_and_Batch_Normalization_by_Variance_CVPR_2019_paper.pdf



Dropout



Overfitting
?!?



Too many
neurons



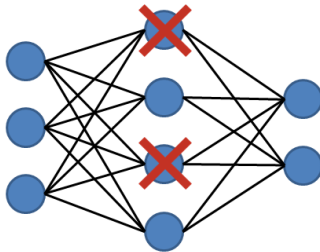
Not
enough
DATA



BAD
Network

Dropout

- С вероятностью p отключаем нейрон
- Делает нейроны более устойчивыми к случайным возмущениям
- Борьба с ко-адаптацией, не все соседи похожи, не все дети похожи на родителей



Dropout в формулах

- forward pass:

$$o = f(X \cdot W + b)$$

Dropout в формулах

- forward pass:

$$o = f(X \cdot W + b)$$

$$o = D \cdot f(X \cdot W + b), \quad D = (D_1, \dots, D_k) \sim iidBern(p)$$

Dropout в формулах

- forward pass:

$$o = f(X \cdot W + b)$$

$$o = D \cdot f(X \cdot W + b), \quad D = (D_1, \dots, D_k) \sim iidBern(p)$$

$$o_i = D_i \cdot f(wx_i^T + b) = \begin{cases} f(wx_i^T + b), p \\ 0, 1 - p \end{cases}$$

Дропаут — это просто небольшая модификация функции активации

Dropout в формулах

- forward pass:

$$o = f(X \cdot W + b)$$

$$o = D \cdot f(X \cdot W + b), \quad D = (D_1, \dots, D_k) \sim iidBern(p)$$

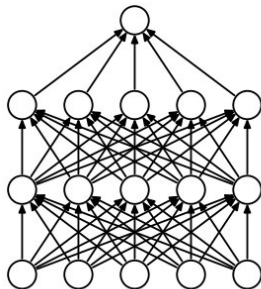
- backward pass:

$$d = f'(h) \cdot W \cdot d$$

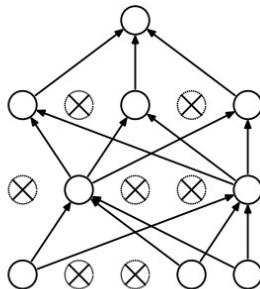
$$d = D \cdot f'(h) \cdot W \cdot d$$

Dropout

- При обучении мы домножаем часть выходов на D_i , тем самым мы изменяем только часть параметров и нейроны учатся более независимо
- Dropout эквивалентен обучению 2^n сетей



(a) Standard Neural Net



(b) After applying dropout.

Dropout

- При обучении мы домножаем часть выходов на D_i , тем самым мы изменяем только часть параметров и нейроны учатся более независимо
- Dropout эквивалентен обучению 2^n сетей
- Что делать на стадии тестирования?

Dropout

- При обучении мы домножаем часть выходов на D_i , тем самым мы изменяем только часть параметров и нейроны учатся более независимо
- Dropout эквивалентен обучению 2^n сетей
- Нам надо симитировать работу такого ансамбля: можно отключать по очереди все возможные комбинации нейронов, получить 2^n прогнозов и усреднить их

Dropout

- При обучении мы домножаем часть выходов на D_i , тем самым мы изменяем только часть параметров и нейроны учатся более независимо
- Dropout эквивалентен обучению 2^n сетей
- Нам надо симитировать работу такого ансамбля: можно отключать по очереди все возможные комбинации нейронов, получить 2^n прогнозов и усреднить их
- Но лучше просто брать по дропауту математическое ожидание

$$o = p \cdot f(X \cdot W + b)$$

Обратный Dropout

- На тесте ищем математическое ожидание:

$$o = p \cdot f(X \cdot W + b)$$

Обратный Dropout

- На тесте ищем математическое ожидание:

$$o = p \cdot f(X \cdot W + b)$$

- Это неудобно! Надо переписывать функцию для прогнозов!

Обратный Dropout

- На тесте ищем математическое ожидание:

$$o = p \cdot f(X \cdot W + b)$$

- Это неудобно! Надо переписывать функцию для прогнозов!
- Давайте лучше будем домножать на $\frac{1}{p}$ на этапе обучения:

$$\text{train: } o = \frac{1}{p} \cdot D \cdot f(X \cdot W + b)$$

$$\text{test: } o = f(X \cdot W + b)$$

Другие эвристики для обучения сеток

Предобучение

- Обучаем каждый нейрон на случайной подвыборке, каждый нейрон впитывает какие-то отдельные её особенности, после скрепляем все нейроны вместе и продолжаем обучение на всей выборке
- **На будущее:** обучаем на корпусе картинок автокодировщик, encoder благодаря этому учится выделять наиболее важные фичи, которые позволяют эффективно сжимать изображения. После срезаем decoder и на его месте достраиваем слои для решения нашей задачи, запускаем обычное дообучение.

Динамическое наращивание сети

- Обучение сети при заведомо недостаточном числе нейронов N
- После стабилизации функции потерь — добавление нового нейрона и его инициализация путём обучения
 - либо по случайной подвыборке
 - либо по объектам с наибольшими значениями потерь
 - либо по случайному подмножеству входов
 - либо из различных случайных начальных приближений
- Снова итерации BackProp

Эмпирический опыт: Общее время обучения обычно лишь в 1.5 — 2 раза больше, чем если бы в сети сразу было итоговое число нейронов. Полезная информация, накопленная сетью не теряется при добавлении нейронов.

Прореживание сети

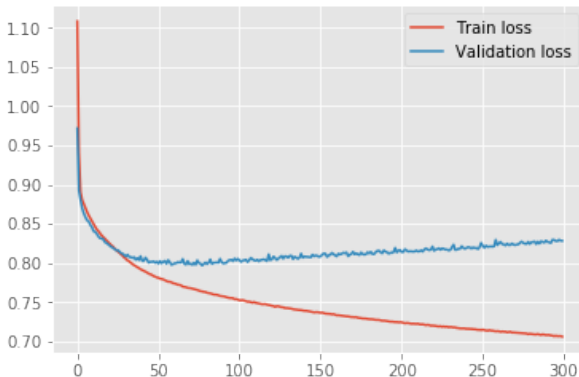
- Начать с большого количество нейронов и удалять незначимые по какому-нибудь критерию
- **Пример:** обнуляем вес, смотрим как сильно упала ошибка, сортируем все связи по этому критерию, удаляем N наименее значимых
- После прореживания снова запускаем backprop
- Если качество модели сильно упала, надо вернуть последние удалённые связи

Другие хаки

Уже обсуждали

- l_1 и l_2 регуляризация
- Ранняя остановка
- Различные новые градиентные спуски, ускоряющие процедуру сходимости

Early stopping



- Будем останавливать обучение, когда качество на валидации начинает падать

Регуляризация

- L_2 : приплюсовываем к функции потерь $\lambda \cdot \sum w_i^2$
- L_1 : приплюсовываем к функции потерь $\lambda \cdot \sum |w_i|$
- Можно регуляризовать не всю сетку, а отдельный нейрон или слой
- Не даёт нейрону сфокусироваться на слишком выделяющемся входе

Регуляризация

- В keras можно добавить для каждого слоя на три вида связей:
- **kernel_regularizer** - на матрицу весов слоя;
- **bias_regularizer** - на вектор свободных членов;
- **kernel_regularizer** - на вектор выходов.
- **Делается примерно так:**

```
model.add(Dense(256, input_dim = 32,  
                kernel_regularizer = regularizers.l1(0.001),  
                bias_regularizer = regularizers.l2(0.1),  
                activity_regularizer = regularizers.l2(0.01)))
```


Взаимосвязи

- На практике обычно используют Dropout. Действия всех этих регуляризаторов оказывается схожим:
- Например, в [1] написано:

«We show that the dropout regularizer is first-order equivalent to an L_2 regularizer applied after scaling the features by an estimate of the inverse diagonal Fisher information matrix»
- У Гудфеллоу в Глубоком обучении на стр. 218 можно найти, что ранняя остановка для линейных моделей эквивалентна l_2 регуляризации с MSE, обучаемой SGD.

[1] <https://arxiv.org/abs/1307.1493>

Ещё обсудим

- Скип-конекшены
- Аугментация данных
- Более забубуенистые архитектуры