Лекция 8

Методы оптимизации в нейронных сетях

Машинное обучение

Сергей Муравьёв

23.10.2020

План лекции

- Проблема исчезающих градиентов
- Функции активации
- Методы второго порядка
- Улучшения градиентного спуска
- Предобработка данных и инициализация весов
- Батчевая нормализация

- Слайды доступны: **shorturl.at/ltVZ3**
- Видео доступны: **shorturl.at/hjyAX**

План лекции

- **Проблема исчезающих градиентов**
- Функции активации
- Методы второго порядка
- Улучшения градиентного спуска
- Предобработка данных и инициализация весов
- Батчевая нормализация

Стохастический градиентный спуск (напоминание)

Стохастический градиентный спуск:

$w_{(0)}$ – **некоторое начальное значение**;

$x_{(1)}, \dots, x_{(|\mathcal{D}|)}$ – **некоторый порядок объектов**;

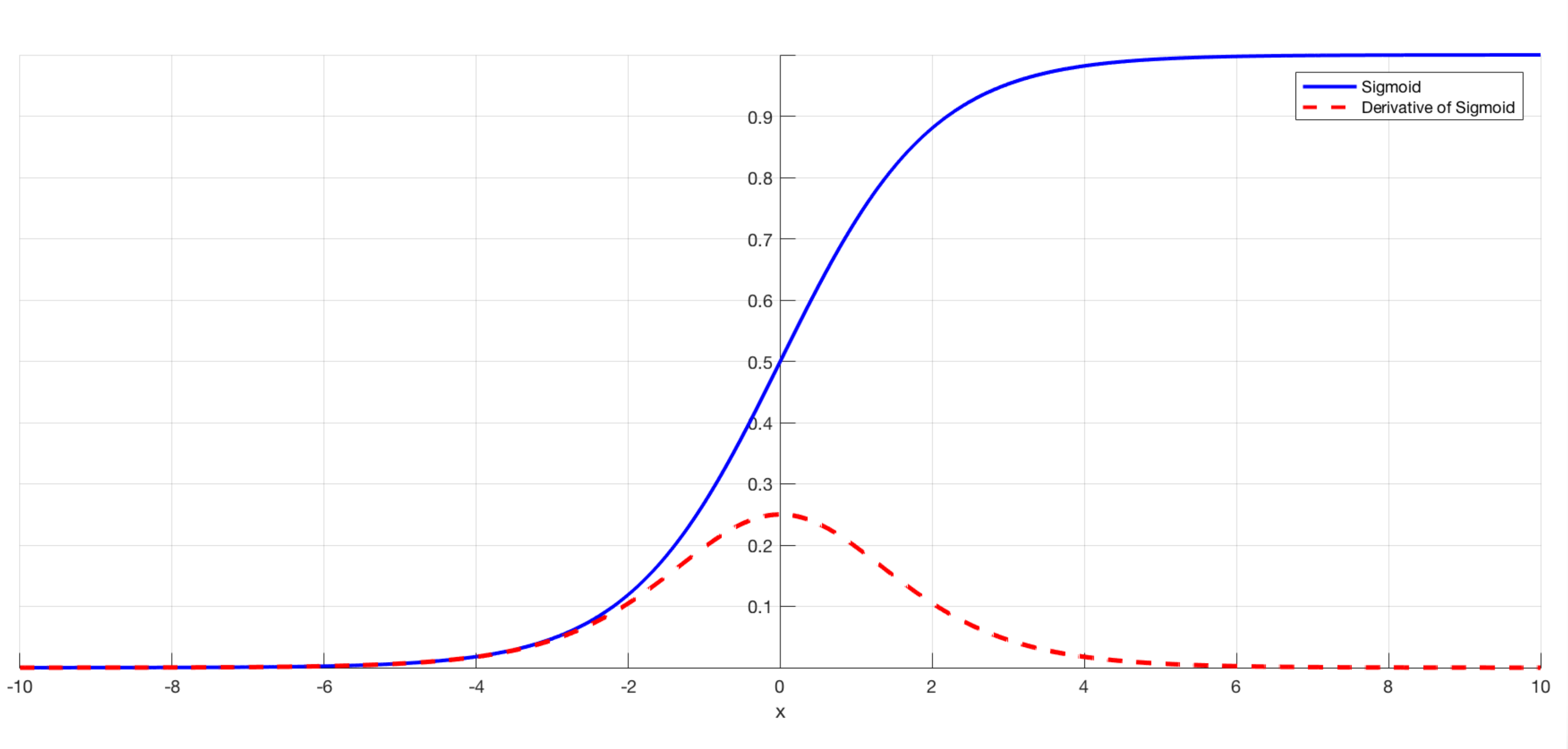
$w_{(k+1)} = w_{(k)} - \mu\mathcal{L}'\big(\langle w_{(k)}, x_{(k)}\rangle y_{(k)}\big) x_{(k)} y_{(k)}$,

$\mathcal{L}_{(k+1)} = (1-\alpha)\mathcal{L}_{(k)} + \alpha\mathcal{L}\big(\langle w_{(k)}, x_{(k)}\rangle y_{(k)}\big)$.

Критерий останова: когда значения $\mathcal{L}$ и/или w почти не меняются

Пример

- Рассмотрим глубокую сеть с d слоями
- Функция активации $\sigma(x) = \frac{1}{1+e^{-x}}$



Пример

- Рассмотрим глубокую сеть с d слоями
- Функция активации $\sigma(x) = \frac{1}{1+e^{-x}}$
- Каждая производная на каждом уровне

$$\frac{\partial L(w)}{\partial u_d} = \frac{\partial L(w)}{\partial a} \cdot \frac{\partial a}{\partial u_d} = (y - a)\sigma'(w_d u_d) w_d \leq 2 \cdot \frac{1}{4} w_d$$

$$\frac{\partial L(w)}{\partial u_{d-1}} = \frac{\partial L(w)}{\partial u_d} \cdot \frac{\partial u_d}{\partial u_{d-1}} \leq 2 \cdot \left(\frac{1}{4}\right)^2 w_d w_{d-1}$$

Градиент либо исчезает, либо «взрывается»

План лекции

- Проблема исчезающих градиентов
- **Функции активации**
- Методы второго порядка
- Улучшения градиентного спуска
- Предобработка данных и инициализация весов
- Батчевая нормализация

Tanh

- Функция активации $a = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- Градиент относительно входа

$$\frac{\partial a}{\partial x} = 1 - \tanh^2(x)$$

- Аналогичен сигмоиде, но с другим диапазоном выходных значений $[-1, +1]$
- Более «сильные» градиенты, потому что данные сосредоточены вокруг 0 (а не 0.5)
- Меньше предвзятости к нейронам скрытого слоя, поскольку теперь выходные данные могут быть как положительными, так и отрицательными (с большей вероятностью в конце будет нулевое среднее значение)

ReLU

- Функция активации $a = h(x) = \max(0, x)$
- Градиент относительно входа

$$\frac{\partial a}{\partial x} = \begin{cases} 1, \text{если } x > 0, \\ 0, \text{в противном случае.} \end{cases}$$

- Очень популярен в компьютерном зрении и распознавании речи

Анализ ReLU

- Гораздо более быстрые вычисления градиентов
- Никаких исчезающих или взрывающихся проблем, только сравнение, сложение, умножение
- Люди заявляют о биологической достоверности
- Редкие активации
- Без насыщенности

- Несимметричная функция
- Не дифференцируется в 0
- Большой градиент во время тренировки может привести к «смерти» нейрона. Более высокая скорость обучения смягчает проблему

10

Softplus

Гладкая аппроксимация (softplus):

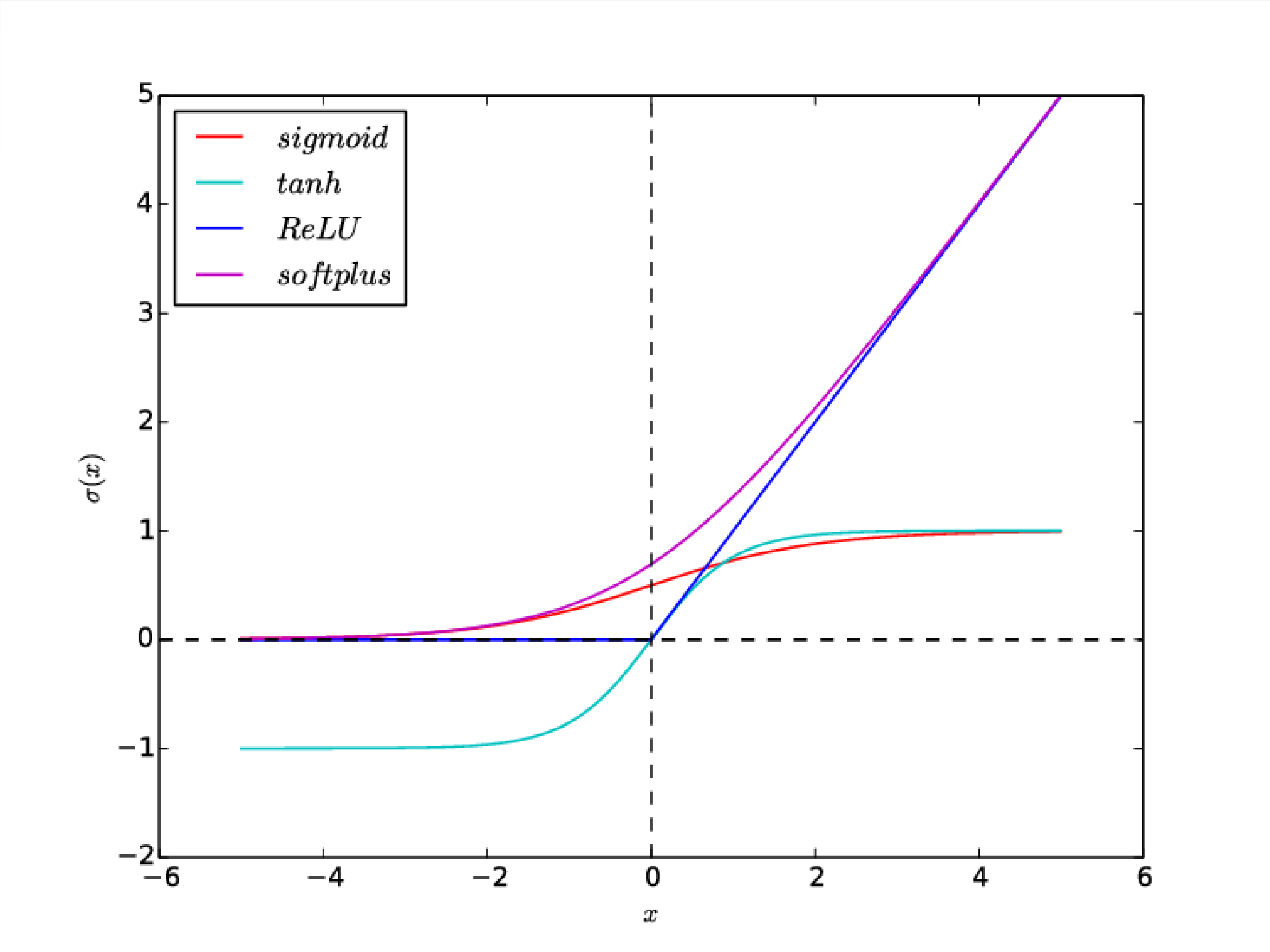
$$a = h(x) = \ln(1 + e^x)$$

- Дифференцируема в 0

- Медленная
- Эмпирически было выявлено, что она не превосходит ReLU

11

Графики различных функций активации



12

Другие виды ReLU

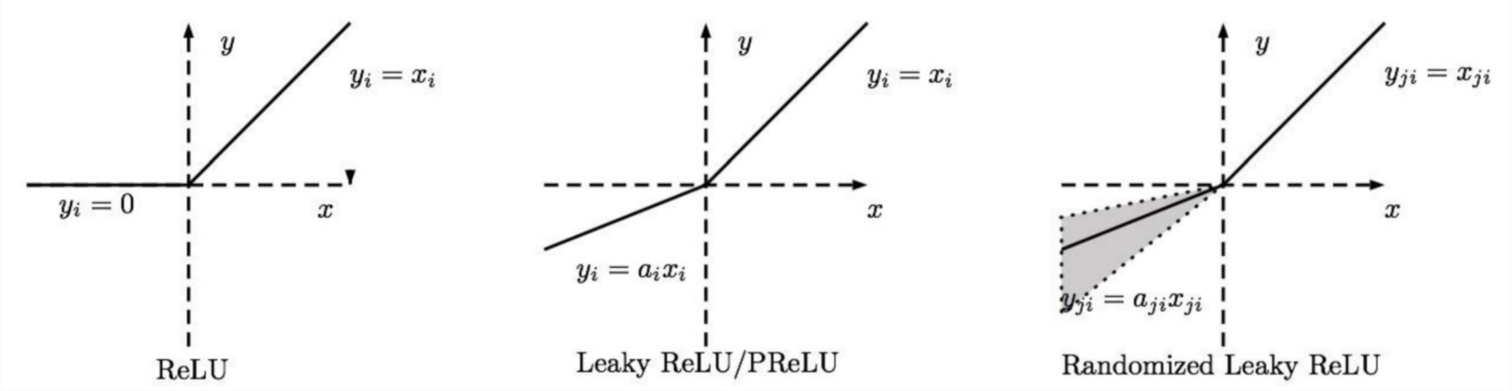
Шумный (noisy) ReLU: $h(x) = \max(0, x + \varepsilon), \varepsilon \sim N(0, \sigma(x))$

ReLU с утечкой (leaky ReLU):

$$h(x) = \begin{cases} x, \text{if } x > 0, \\ 0.01x, \text{в противном случае.} \end{cases}$$

Параметрический ReLu: $h(x) = \begin{cases} x, \text{if } x > 0 \\ \beta x, \text{в противном случае} \end{cases}$

(параметр β настраиваемый)



13

План лекции

- Проблема исчезающих градиентов
- Функции активации
- **Методы второго порядка**
- Улучшения градиентного спуска
- Предобработка данных и инициализация весов
- Батчевая нормализация

Метод Ньютона-Рафсона

$$\mathcal{L}(a, \mathcal{D}) = \sum_{i=1}^{|\mathcal{D}|} (f(x_i, w) - y_i)^2 \to \min_{\theta \in \mathbb{R}^p}.$$

1. Выбор начальных значений $w^{(0)} = \left(w_1^{(0)}, \dots, w_p^{(0)}\right)$.
2. Шаг итерации:

$$w^{(t+1)} = w^{(t)} - \eta_t \left(\mathcal{L}''\left(w^{(t)}\right)\right)^{-1} \mathcal{L}'\left(w^{(t)}\right),$$

где $\mathcal{L}'\left(w^{(t)}\right)$ – градиент $\mathcal{L}$ в $w^{(t)}$;

$\mathcal{L}''\left(w^{(t)}\right)$ – гессиан $\mathcal{L}$ в $w^{(t)}$;

η_t – шаг (обычно $\eta_t = 1$).

15

Градиент и гессиан

j-й элемент градиента:

$$\frac{\partial \mathcal{L}(w)}{\partial w_j} = 2\sum_{i=1}^{|\mathcal{D}|}\left(f(x_i, w) - y_i\right)\frac{\partial f(x_i, w)}{\partial w_j}.$$

(j, k)-й элемент гессиана:

$$\frac{\partial^2 \mathcal{L}(w)}{\partial w_j \delta w_k} = 2\sum_{i=1}^{|\mathcal{D}|}\frac{\partial f(x_i, w)}{\partial w_j}\frac{\partial f(x_i, w)}{\partial \mathrm{w}_k} +$$

$$+2\sum_{i=1}^{|\mathcal{D}|}\left(f(x_i, w) - y_i\right)\frac{\partial^2 f(x_i, w)}{\partial w_j \partial w_k}.$$

Проблема

Очень неудобно вычислять гессиан каждый раз в каждой точке (кубическая сложность).

Чтобы избежать этого, используются **квазиньютоновские** методы, использующие приближенную оценку гессиана.

17

Метод Ньютона-Гаусса

Основная идея — **линеаризация**:

$$f(x_i, w) \approx f\left(x_i, w^{(t)}\right) + \sum_{j=1}^{p} \left(w_j - w_j^{(t)}\right) \frac{\partial f\left(x_i,\ w^{(t)}\right)}{\partial w_j} + $$

$$+ o\left(w_j - w_j^{(t)}\right).$$

$$F_t = \left(\frac{\partial f_i}{\partial w_j}\left(x_i, w^{(t)}\right)\right)_{i=1..|\mathcal{D}|}^{j=1..p}$$ — матрица частных производных

$$f_t = \left(f\left(x_i, w^{(t)}\right)\right)_{i=1..|\mathcal{D}|}$$ — вектор значений f.

18

Связь метода Ньютона-Гаусса с многомерной линейной регрессией

$$w^{(t+1)} = w^{(t)} - \eta_t \left(F_t^\top F_t\right)^{-1} F_t\left(f^{(t)} - y\right),$$

$\beta = \left(F_t^\top F_t\right)^{-1} F_t\left(f^{(t)} - y\right)$ – решение задачи:

$$\left\|F_t\beta - \left(f^{(t)} - y\right)\right\|^2 \to \min_\beta.$$

Он сходится с той же скоростью, что и метод Ньютона-Рафсона.

План лекции

- Проблема исчезающих градиентов
- Функции активации
- Методы второго порядка
- **Улучшения градиентного спуска**
- Предобработка данных и инициализация весов
- Батчевая нормализация

Стохастический градиентный спуск (второе напоминание)

Стохастический градиентный спуск:

$w^{(0)}$ – начальные значения

$$w^{(k+1)} = w^{(k)} - \mu \frac{\partial \mathcal{L}(w^{(k)})}{\partial w}$$

Модификация Momentum

Momentum:

$w^{(0)}$ – начальные значения;

ν – вектор изменений:

$w^{(k+1)} = w^{(k)} - \nu^{(k)}$

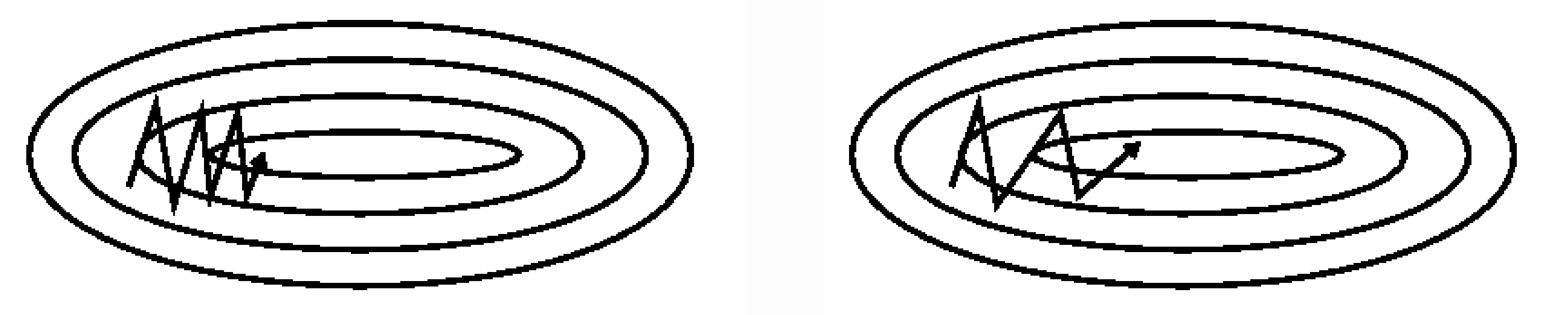
$\nu^{(k+1)} = \gamma\nu^{(k)} + \mu\frac{\partial\mathcal{L}(w^{(k)})}{\partial w}$,

γ – момент, обычно устанавливается 0.9

Анализ метода Momentum

Преимущества:

- в целом быстрее на сложном «рельефе» при движении в правильном направлении



Недостатки:

- может пропускать минимумы

Метод Нестерова (Nesterov accelerated gradient, NAG)

NAG:

$w^{(0)}$ — начальные значения;

ν вектор изменений:

$$w^{(k+1)} = w^{(k)} - \nu^{(k)}$$

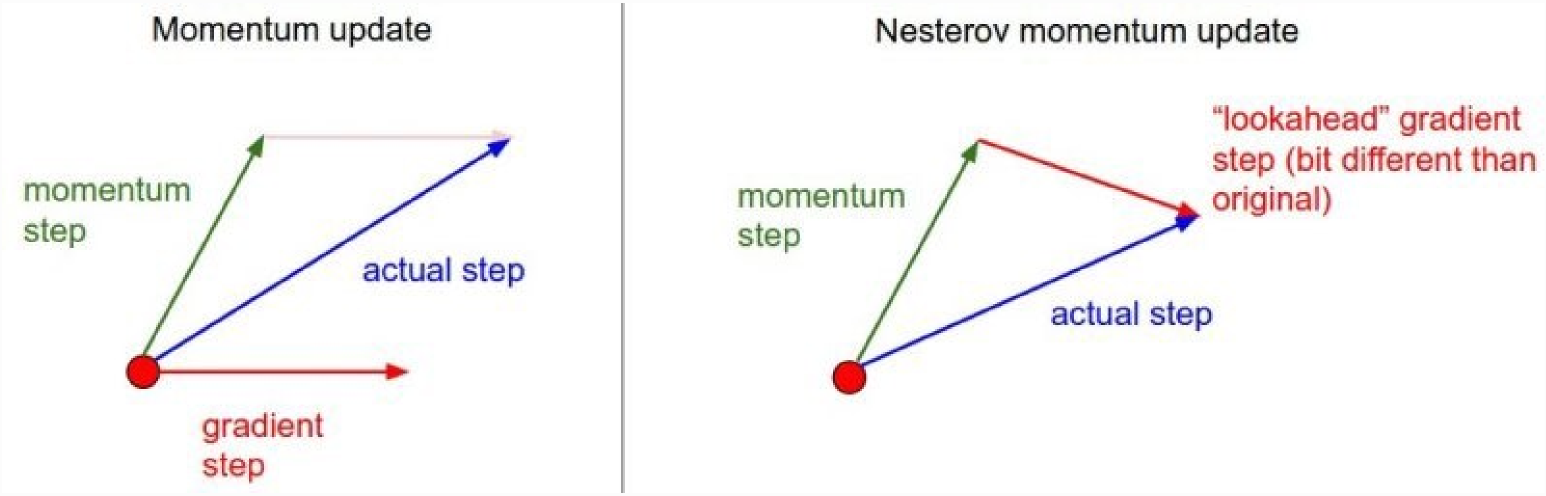
$$\nu^{(k+1)} = \gamma\nu^{(k)} + \mu\frac{\partial\mathcal{L}(w^{(k)} - \nu^{(k)})}{\partial w},$$

γ — момент, обычно устанавливается 0.9

Анализ метода Нестерова

Преимущества:

- В целом, работает лучше
- Сходимость доказана при определенных условиях



- Как выбирать момент?

Метод Adagrad

$$g_{i,(k)} = \frac{\partial L\left(w^{(k)}\right)}{\partial w_i}.$$

Adagrad:

$w^{(0)}$ – начальные значения;

Для каждого i

$$w_i^{(k+1)} = w_i^{(k)} - \frac{\mu}{\sqrt{G_{i,i}^{(k)} + \varepsilon}} g_{i,(k)},$$

где G – диагональная матрица, где каждый диагональный элемент i, i – сумма квадратов градиентов $g_{i,(k)}$ до шага k, и ε – сглаживающая переменная, избегающая деления на ноль.

Анализ метода Adagrad

Преимущества:

- Устраняет необходимость вручную настраивать скорость обучения. Большинство реализаций используют значение по умолчанию 0,01 и оставляют его как есть.

Недостатки:

- Накопление квадратов градиентов в знаменателе приводит к тому, что в процессе обучения сумма продолжает расти. В конце концов алгоритм перестает чему-либо учиться.

Метод RMSProp

$$E^{(k)}[g_i^2] = \gamma E^{(k-1)}[g_i^2] + (1-\gamma) g_{i,(k)}^2$$

RMSProp:

$w^{(0)}$ — начальные значения;

Для каждого i

$$w_i^{(k+1)} = w_i^{(k)} - \frac{\mu}{\sqrt{E^{(k)}[g_i^2] + \varepsilon}} g_{i,(k)},$$

где ε — сглаживающая переменная, избегающая деления на ноль.

γ устанавливается равным 0.9

Не растёт с ростом k

Метод Adadelta (1/3)

$$w^{(k+1)} = w^{(k)} - \mu\left(\mathcal{L}''\left(w^{(k)}\right)\right)^{-1} Q'\left(w^{(k)}\right)$$

$$w^{(k+1)} = w^{(k)} + \Delta w^{(k)}$$

$\left(\mathcal{L}''\left(w^{(k)}\right)\right)^{-1}$ сложно оценить, поэтому предполагаем, что это диагональная матрица

$$\left(\mathcal{L}''\left(w^{(k)}\right)\right) \approx \mathrm{diag}\left(\frac{\partial \mathcal{L}^2\left(w^{(k)}\right)}{\partial w_i^2}\right)$$

$$\Delta w_i^{(k)} \approx \left(\frac{\partial \mathcal{L}^2\left(w^{(k)}\right)}{\partial w_i^2}\right)^{-1} \left(\frac{\partial \mathcal{L}\left(w^{(k)}\right)}{\partial w_i}\right)$$

$$\frac{\partial \mathcal{L}^2\left(w^{(k)}\right)}{\partial w_i^2} \approx \frac{\left(\frac{\partial \mathcal{L}\left(w^{(k)}\right)}{\partial w_i}\right)}{\Delta w_i^{(k)}}$$

Метод Adadelta (2/3)

$$E^{(k)}[g_i^2] = \gamma E^{(k-1)}[g_i^2] + (1-\gamma) g_{i,(k)}^2$$

$$RMS^{(k)}[g_i] = \sqrt{E^{(k)}[g_i^2] + \varepsilon}$$

$$RMS^{(k)}[\Delta w_i] = \sqrt{E^{(k)}\left[\Delta w_i^2\right] + \varepsilon}$$

$$\frac{\partial \mathcal{L}^2\left(w^{(k)}\right)}{\partial w_i^2} \approx \frac{\left(\frac{\partial \mathcal{L}\left(w^{(k)}\right)}{\partial w_i}\right)}{\Delta w_i^{(k)}} \approx \frac{g_i^{(k)}}{\Delta w_i^{(k-1)}} = \frac{RMS^{(k)}[g_i]}{RMS^{(k-1)}[\Delta w_i]}.$$

Метод Adadelta (3/3)

Adadelta:

$w^{(0)}$ – начальные значения;

Для каждого i

$$w_i^{(k+1)} = w_i^{(k)} - \frac{RMS^{(k-1)}[\Delta w_i]}{RMS^{(k)}[g_i]} g_i^{(k)}$$

Устанавливать скорость обучения не требуется!

На практике скорость обучения все еще добавляется для повышения производительности.

Метод Adam (Adaptive Moment Estimation)

$$m_{(k)} = E^{(k)}[g_i] = \gamma_1 E^{(k-1)}[g_i] + (1-\gamma_1) g_{i,(k)}$$
$$b_{(k)} = E^{(k)}[g_i^2] = \gamma_2 E^{(k-1)}[g_i^2] + (1-\gamma_2) g_{i,(k)}^2$$

Хочется, чтобы они были несмещённные:

$$\mathrm{E}\left(m_{(k)}\right) = \mathrm{E}\left(g_{(k)}\right), \mathrm{E}\left(b_{(k)}\right) = \mathrm{E}\left(g_{(k)}^2\right)$$

Чтобы удовлетворить данное требование понадобится следующая поправка:

$$\begin{cases} \widehat{m}_{(k)} = \dfrac{m_{(k)}}{1-\gamma_1^k} \\ \hat{b}_{(k)} = \dfrac{b_{(k)}}{1-\gamma_2^k} \end{cases}$$

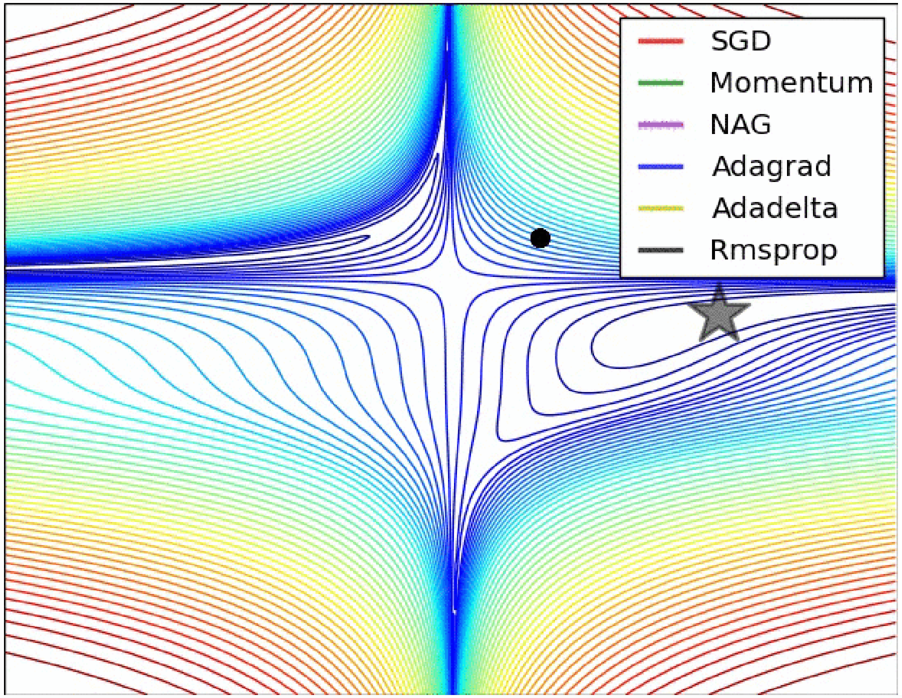
Метод Adam (Adaptive Moment Estimation)

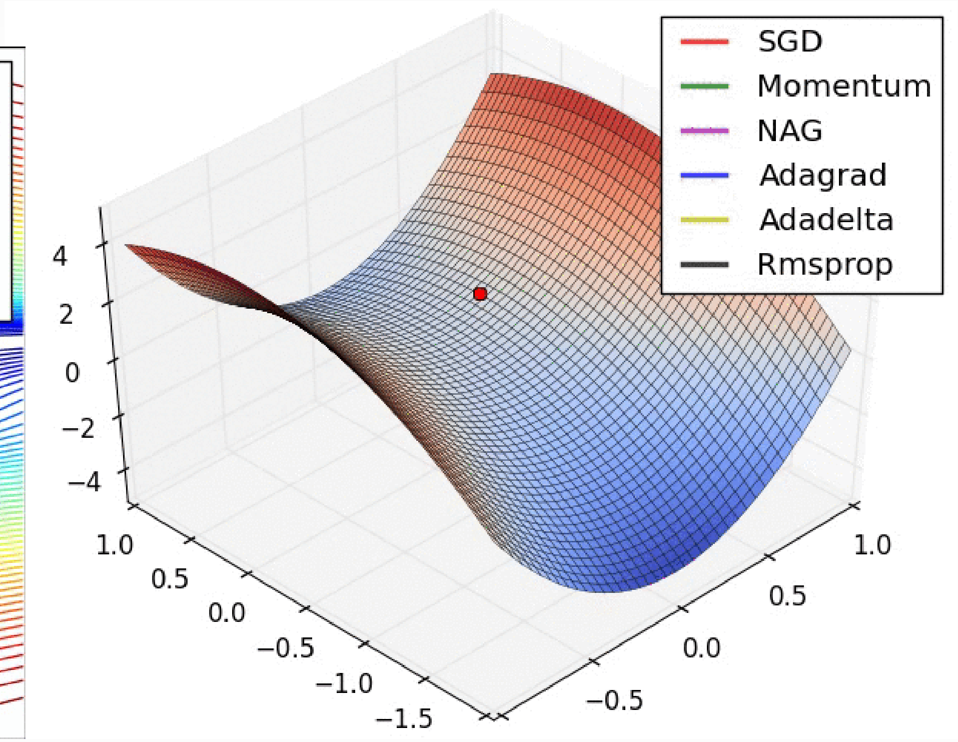
Adam:

$w^{(0)}$ – начальные значения;

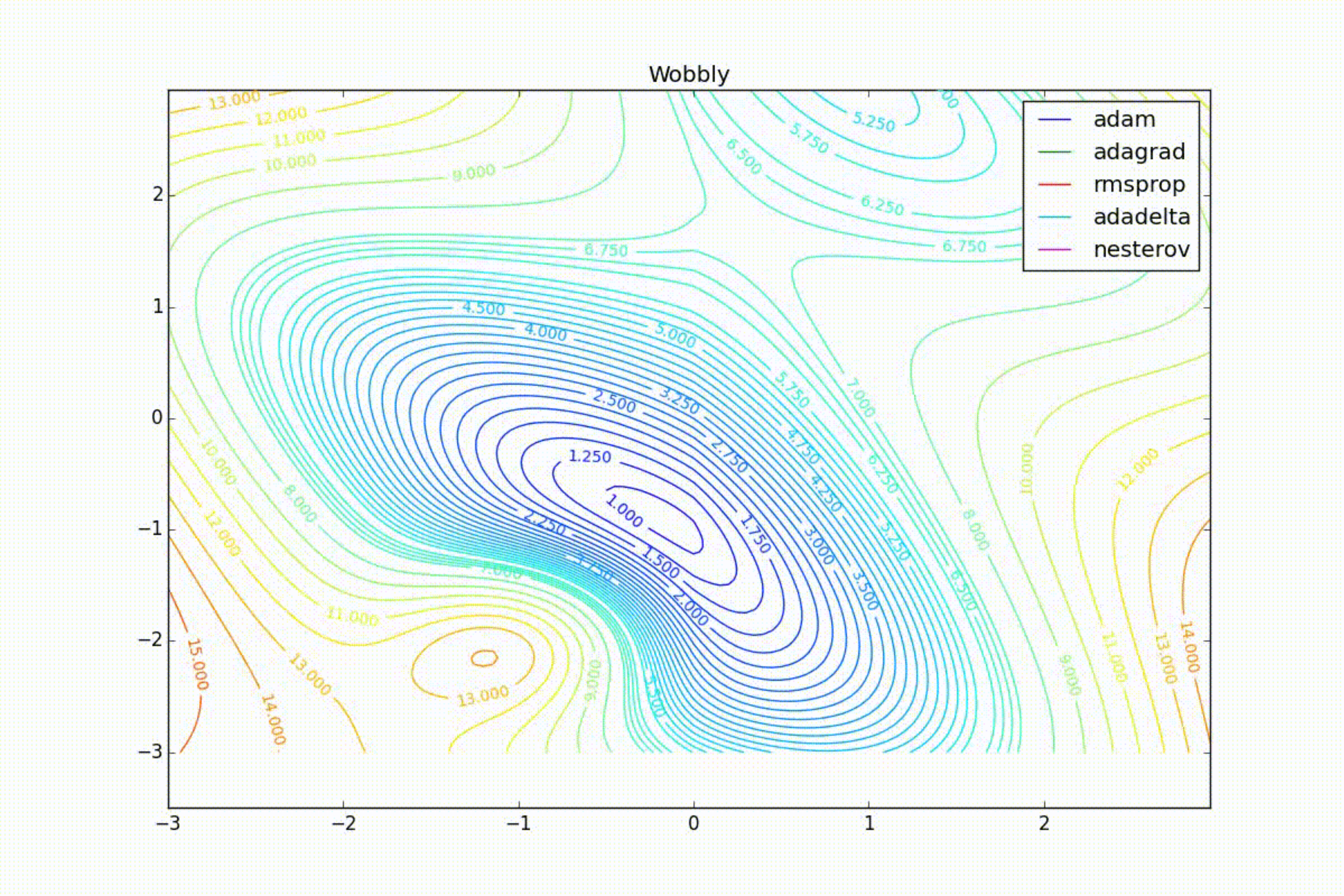
$$w^{(k+1)} = w^{(k)} - \frac{\mu}{\sqrt{\hat{b}_{(k)}^2 + \varepsilon}} \hat{m}_{(k)}$$

Сравнение

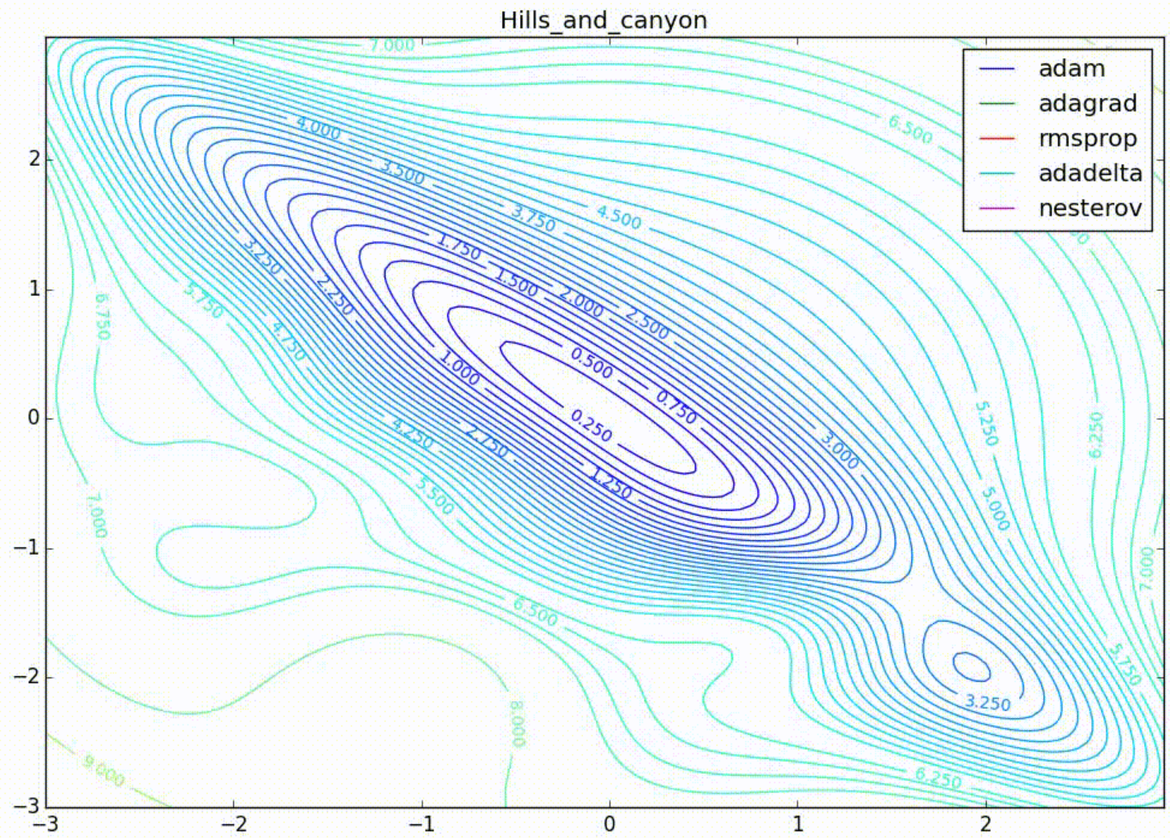

SGD
Momentum
NAG
Adagrad
Adadelta
Rmsprop


SGD
Momentum
NAG
Adagrad
Adadelta
Rmsprop
4
2
0
−2
−4
1.0
0.5
0.0
−0.5
−1.0
−1.5
1.0
0.5
0.0
−0.5

Сравнение



Сравнение



План лекции

- Проблема исчезающих градиентов
- Функции активации
- Методы второго порядка
- Улучшения градиентного спуска
- **Предобработка данных и инициализация весов**
- Батчевая нормализация

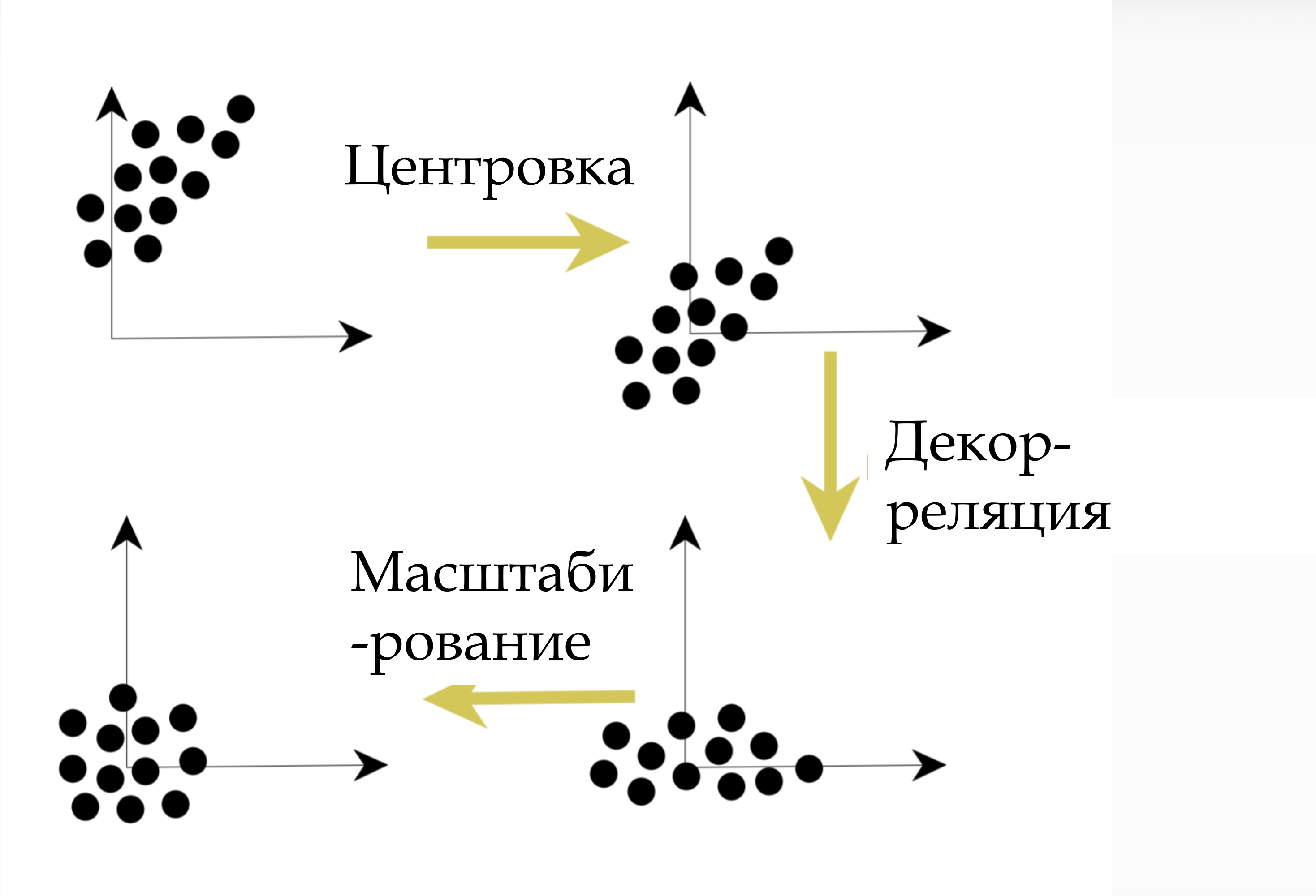
Предварительная обработка данных

Предобработка данных является полезной и очень важна в контексте оптимизации нейронных сетей. Три основных приёма:

- Центровка
- Декорреляция [Расширение Карунена – Лоэва]
- Масштабирование

38

Предварительная обработка данных



39

Декорреляция

Ковариационная матрица: $\mathrm{Cov}(X) = \frac{1}{N} X X^{\mathrm{T}}$

Декорреляция: $\hat{X} = \mathrm{Cov}^{-1/2}(X) \cdot X$

$\mathrm{Cov}\left(\hat{X}\right) = I$

Выбор начальных значений весов

- Подбор весов важен для качества решения и даже сходимости спуска.
- Типичный сценарий — инициализировать веса чем-то маленьким случайным.

Метод Xavier. Мотивация

- Предположим, у нас есть функция активации f, линейная вблизи 0:
$$f(x) = x$$
- tanh — пример такой функции

Основная идея состоит в том, чтобы поставить веса в такой линейной области и поддерживать постоянство дисперсии внутри «области линейности».

Оценка дисперсии (1/2)

$$u_{d+1} = f(u_d w_d) \approx u_d w_d$$

$$\mathrm{D}(u_{d+1,k}) = \mathrm{D}\left(\sum_{i=1}^{n_d} u_{d,i} w_{d,i,k}\right) = \sum_{i=1}^{n_d} \mathrm{D}(u_{d,i} w_{d,i,k})$$

n_d — количество нейронов слоя d

Можно считать, что они независимы

$$\mathrm{D}(u_{d+1,k}) = n_d \mathrm{D}(u_{d,i} w_{d,i,k}) =$$

$$= n_d \left(\mathrm{E}(u_{d,i}^2)\mathrm{E}(w_{d,i,k}^2) - \mathrm{E}^2(u_{d,i})\mathrm{E}^2(w_{d,i,k})\right) =$$

$$n_d \mathrm{D}(u_{d,i})\mathrm{D}(w_{d,i,k})$$

Оценка дисперсии (2/2)

$$\mathrm{D}(u_{d+1}) = \mathrm{D}(x)\prod_{j=1}^{d} n_j\,\mathrm{D}(w_j)$$

$$\mathrm{D}\left(\frac{\partial L}{\partial u_d}\right) = \mathrm{D}\left(\frac{\partial L}{\partial u_N}\right)\prod_{j=d}^{N} n_{j+1}\,\mathrm{D}(w_j)$$

При требованиях $\forall d, h \le N$:

$$\mathrm{D}(u_d) = \mathrm{D}(u_h)$$

$$\mathrm{D}\left(\frac{\partial L}{\partial u_d}\right) = \mathrm{D}\left(\frac{\partial L}{\partial u_h}\right)$$

Xavier

$$\mathrm{D}(u_d) = \mathrm{D}(u_h), \mathrm{D}\left(\frac{\partial L}{\partial u_d}\right) = \mathrm{D}\left(\frac{\partial L}{\partial u_h}\right)$$ – эквивалентно

$$\forall d \begin{cases} n_d D(w_d) = 1 \\ n_{d+1} D(w_d) = 1 \end{cases}$$

Компромисс: $D(w_d) = \frac{2}{n_d + n_{d+1}}$

$$w_d \sim U\left[\frac{-\sqrt{6}}{n_d + n_{d+1}}, \frac{\sqrt{6}}{n_d + n_{d+1}}\right].$$

Что делать ReLU? Не

$$\mathrm{D}(u_{d+1}) = \mathrm{D}(x)\prod_{j=1}^{d}\frac{1}{2}n_j\,\mathrm{D}(w_j)$$

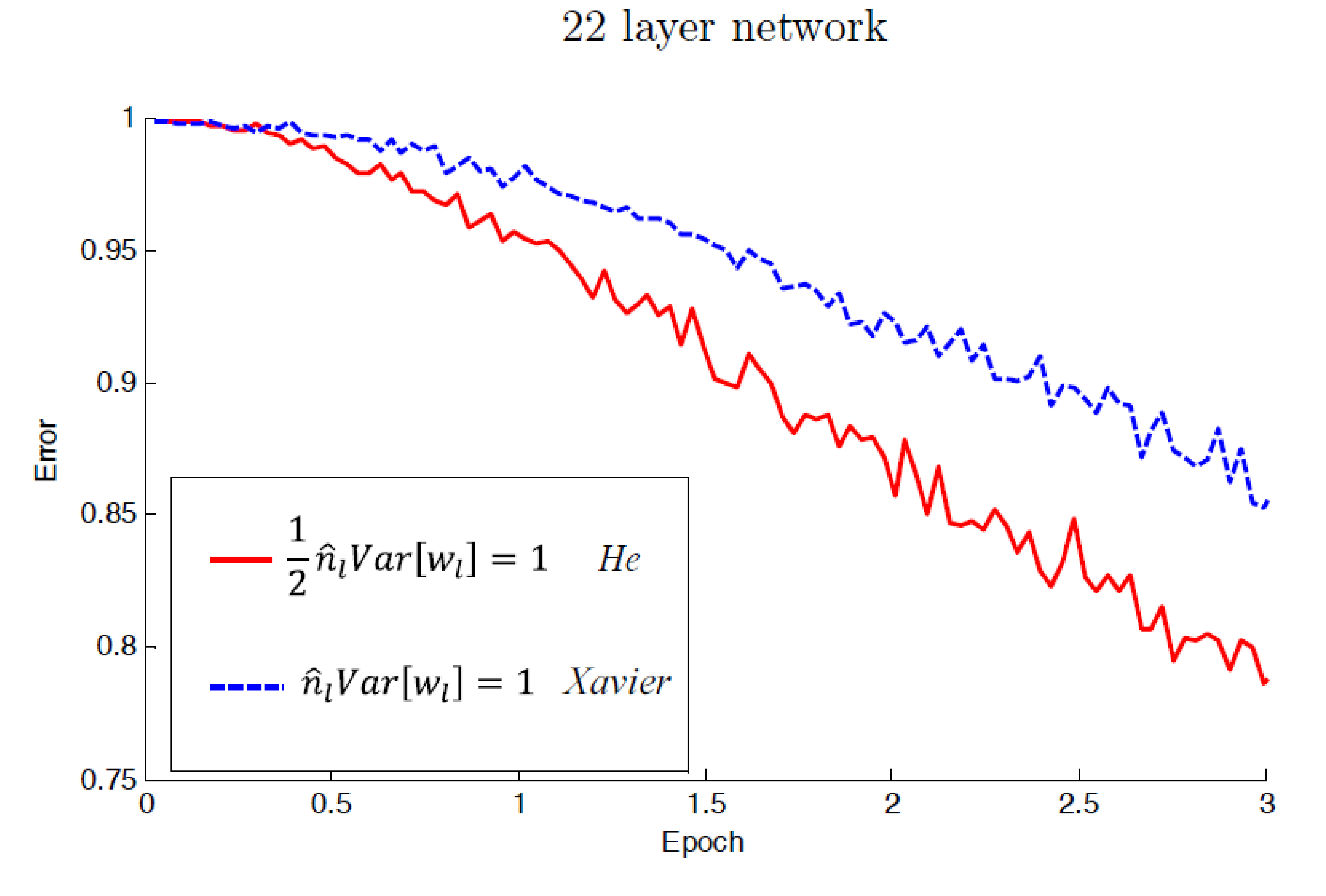
$$\mathrm{D}\left(\frac{\partial L}{\partial u_d}\right) = \mathrm{D}\left(\frac{\partial L}{\partial u_N}\right)\prod_{j=d}^{N}\frac{1}{2}n_{j+1}\,\mathrm{D}(w_j)$$

$$\forall d\ \begin{cases} n_d D(w_d) = 2 \\ n_{d+1} D(w_d) = 2 \end{cases}$$

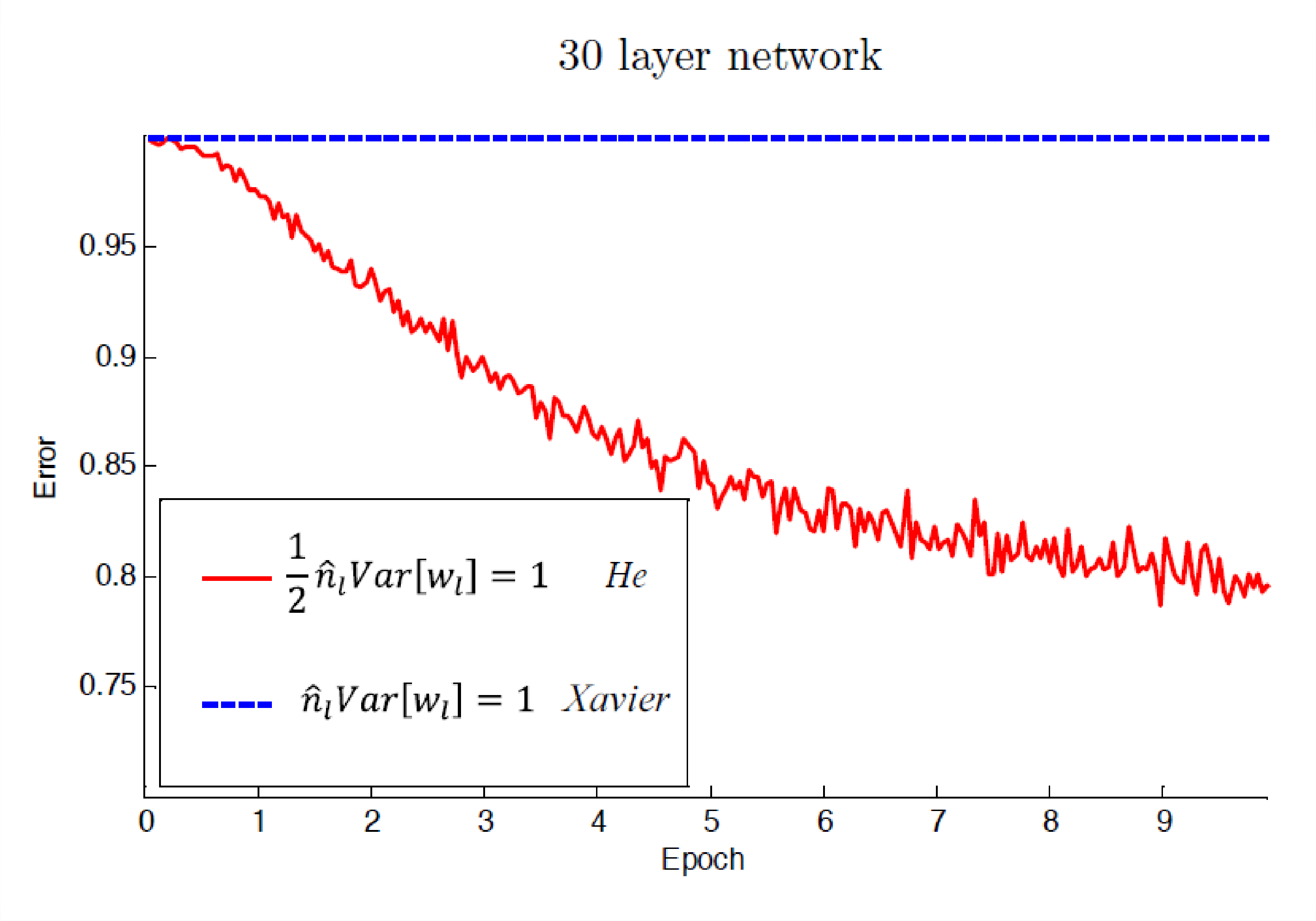
Используется распределение по Гауссу:

$w_d \sim N\left[0, \frac{2}{n_d}\right]$ или $w_d \sim N\left[0, \frac{2}{n_{d+1}}\right]$

Xavier vs He (1/2)



Xavier vs He (2/2)



План лекции

- Проблема исчезающих градиентов
- Функции активации
- Методы второго порядка
- Улучшения градиентного спуска
- Предобработка данных и инициализация весов
- **Батчевая нормализация**

Идея

Когда меняются веса слоя, также меняются распределение его выходов

Основная идея: поддерживать константу ковариации для каждого входного слоя:

$$\hat{x}_d = \frac{x_d - \mathrm{E}(x_i)}{\sqrt{\mathrm{D}(x_d) + \varepsilon}}$$

E и D следует оценивать на каждом мини-батче

Параметрический слой для пакетной нормализации

Добавим параметрический слой с масштабированием:

$$\hat{y}_d = \gamma_d \hat{x}_d + \beta_d$$

γ и β настраиваются

Алгоритм батчевой нормализации

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

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

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

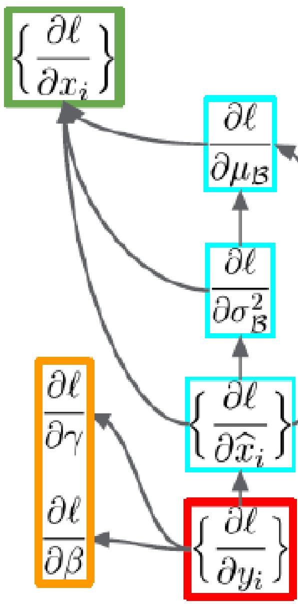
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

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

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

Градиентный спуск для батчевой нормализации

Будем обучать слой батчевой нормализации как обычный слой



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

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

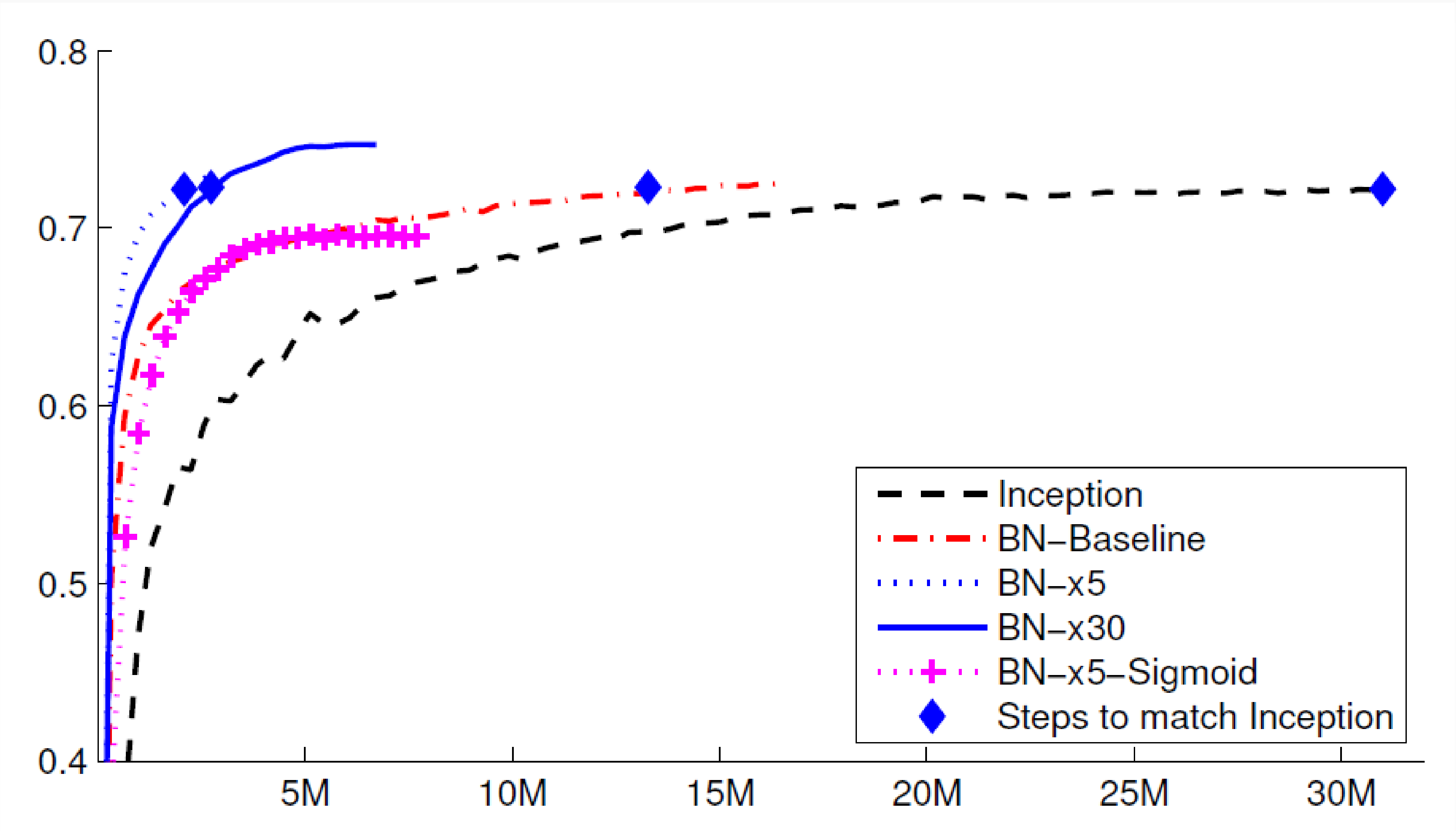
$$\frac{\partial \ell}{\partial \mu_{\mathcal{B}}} = \left(\sum_{i=1}^{m} \frac{\partial \ell}{\partial \widehat{x}_i} \cdot \frac{-1}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}}\right) + \frac{\partial \ell}{\partial \sigma^2_{\mathcal{B}}} \cdot \frac{\sum_{i=1}^{m} -2(x_i - \mu_{\mathcal{B}})}{m-1}$$

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

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

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

Сравнение



0.8
0.7
0.6
0.5
0.4
5M
10M
15M
20M
25M
30M
Inception
BN–Baseline
BN–x5
BN–x30
BN–x5–Sigmoid
Steps to match Inception

Анализ батчевой нормализации

- Работает быстро
- Сходится быстро
- Делает сторонние регуляризаторы не такими полезными