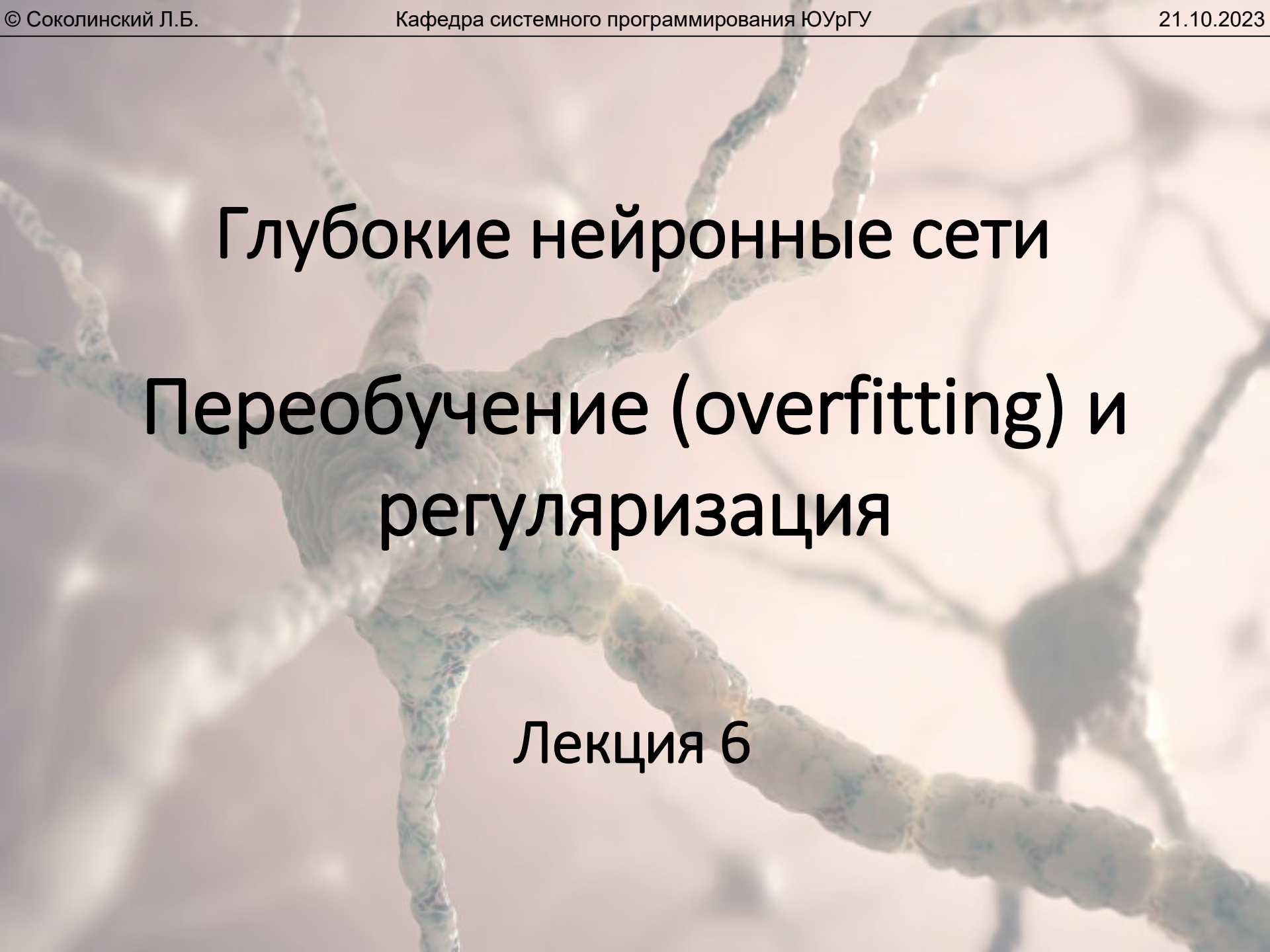


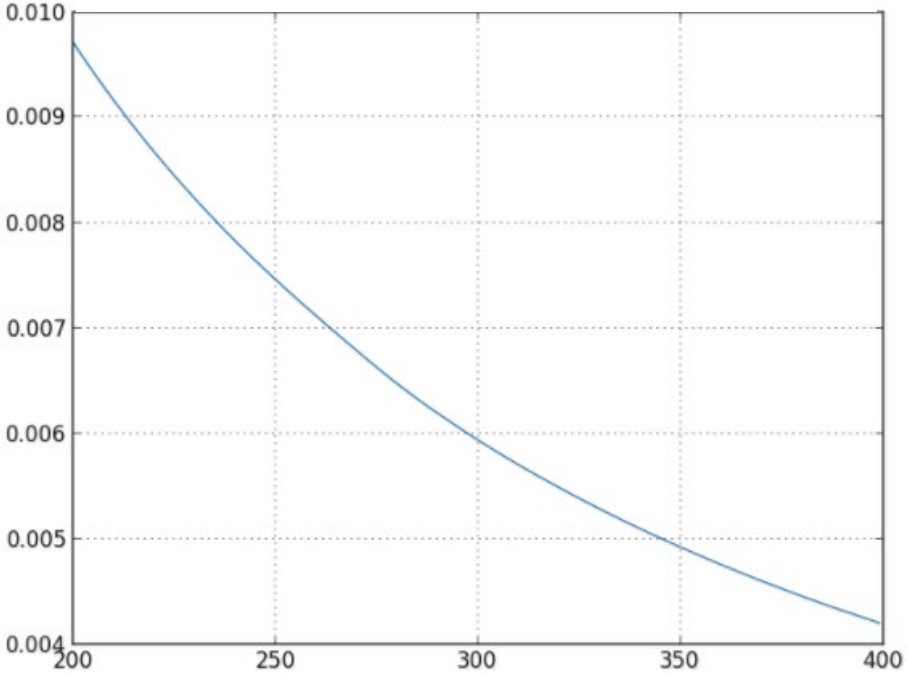
Глубокие нейронные сети

Переобучение (overfitting) и регуляризация

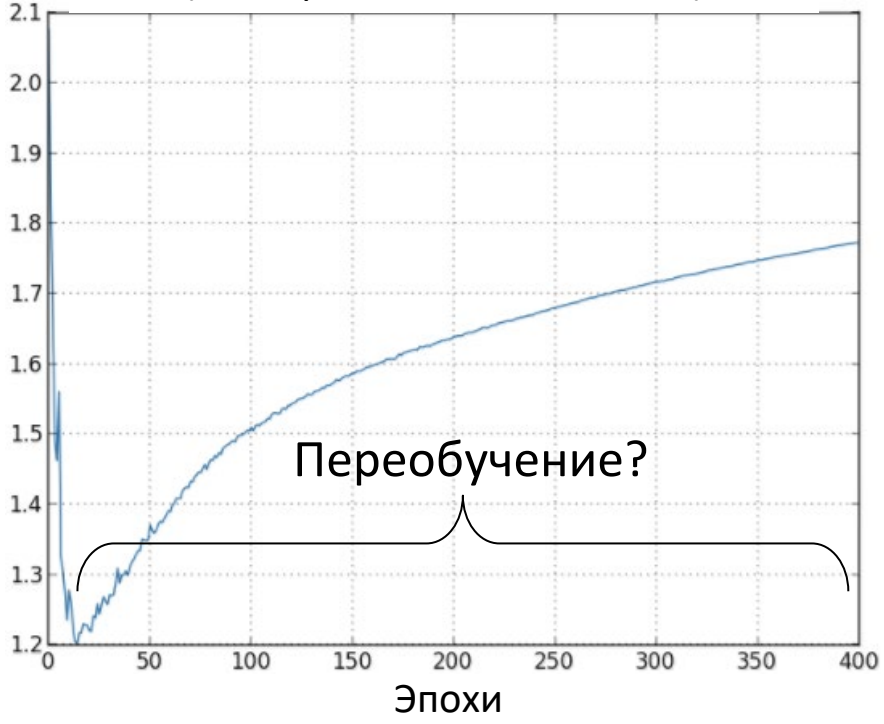
Лекция 6

Величина ошибки

Величина ошибки на обучающей выборке
(70% прецедентов из MNIST)

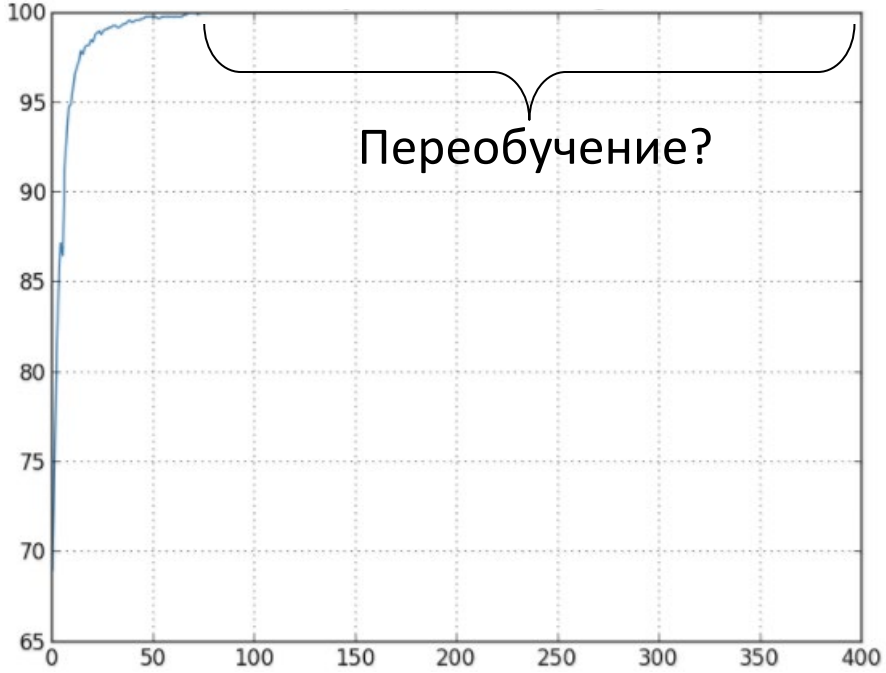


Величина ошибки на тестовой выборке
(20% прецедентов из MNIST)



Точность распознавания

Точность распознавания (%)
на обучающей выборке
(70% прецедентов из MNIST)



Эпохи

Точность распознавания (%)
на тестовой выборке
(20% прецедентов из MNIST)

Эпохи

Борьба с переобучением

- Увеличение обучающей выборки
- Добавление валидационной выборки
- Уменьшение глубины нейронной сети
- Сокращение весов (weight decay): регуляризация L_2 и L_1
- Прореживание (dropout)
- Искусственное увеличение обучающей выборки

Валидационная выборка

- **Обучающая выборка (70% прецедентов)**: подбор весов и смещений
  - **Тестовая выборка (20% прецедентов)**: подбор макропараметров (скорость обучения, количество эпох, количество подвыборок, …)
    - **Валидационная выборка (10% прецедентов)**: интегральная проверка
- **Необходимо останавливать обучение, когда начнет ухудшаться ошибка на валидационном множестве (early stopping)**

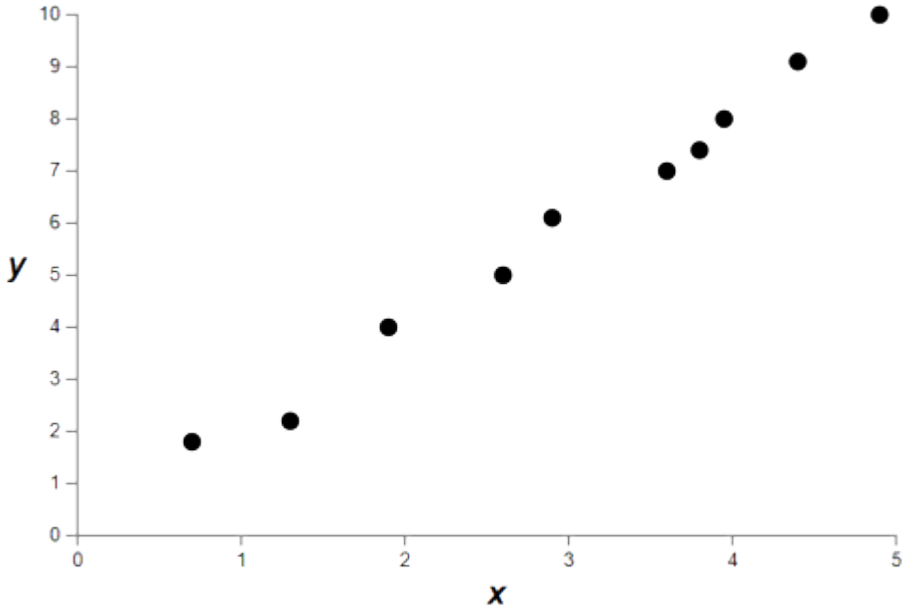
Уменьшение глубины нейронной сети

- Глубина (количество скрытых слоев) нейронной сети должна соответствовать сложности задачи

- Необходимо начинать с сети малой глубины и увеличивать количество слоев, пока количество ошибок уменьшается

Почему бывает горе от ума?

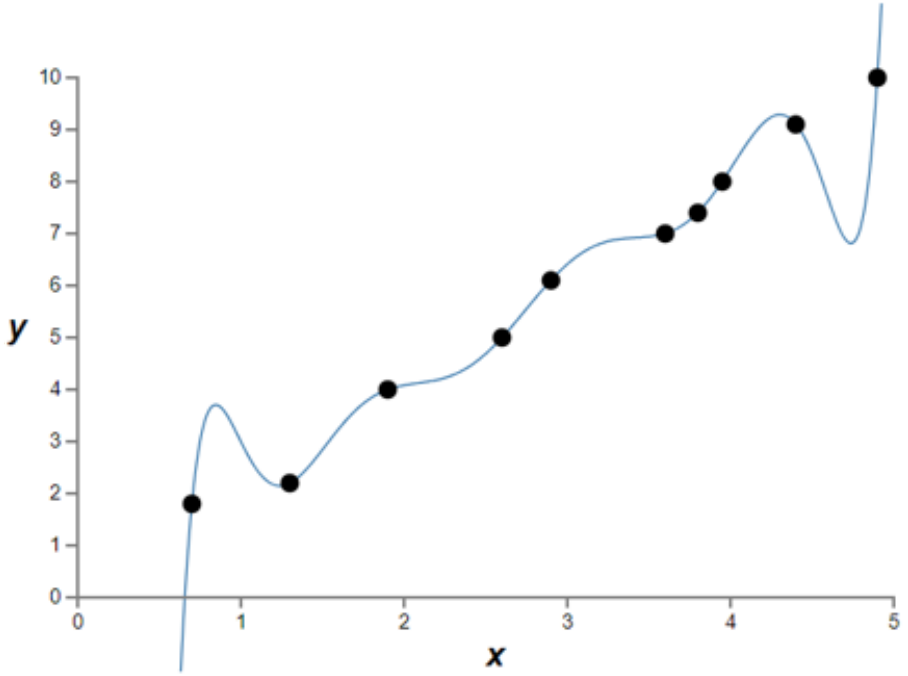
Пример с полиномиальной аппроксимацией



Почему бывает горе от ума?

Пример с полиномиальной аппроксимацией

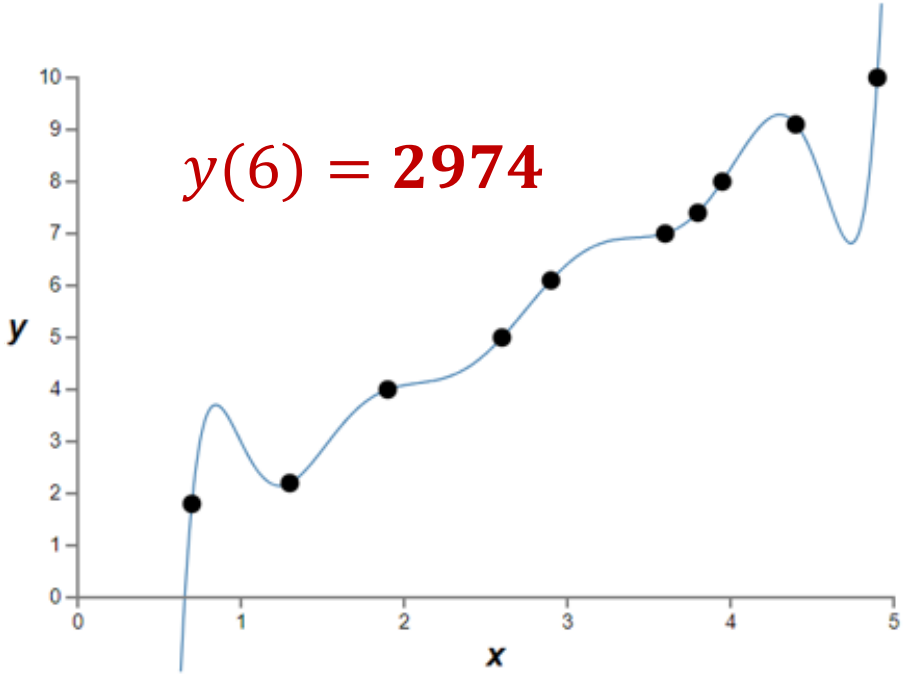
$$y = 0.220539187 * x^9 - 5.49142821 * x^8 + 58.7844045 * x^7 - 353.892824 * x^6 + 1315.49254 * x^5 - 3118.09836 * x^4 + 4690.80366 * x^3 - 4296.12493 * x^2 + 2162.28823 * x - 450.983951$$



Почему бывает горе от ума?

Пример с полиномиальной аппроксимацией

$$y = 0.220539187 * x^9 - 5.49142821 * x^8 + 58.7844045 * x^7 - 353.892824 * x^6 + 1315.49254 * x^5 - 3118.09836 * x^4 + 4690.80366 * x^3 - 4296.12493 * x^2 + 2162.28823 * x - 450.983951$$



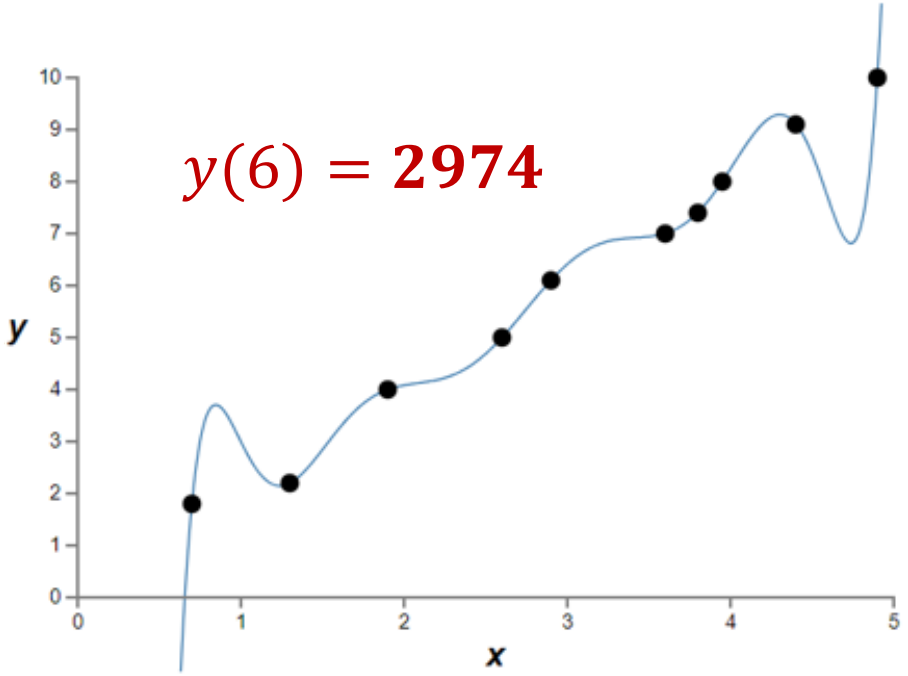
Правильный ответ: $y(6) = \mathbf{11}$

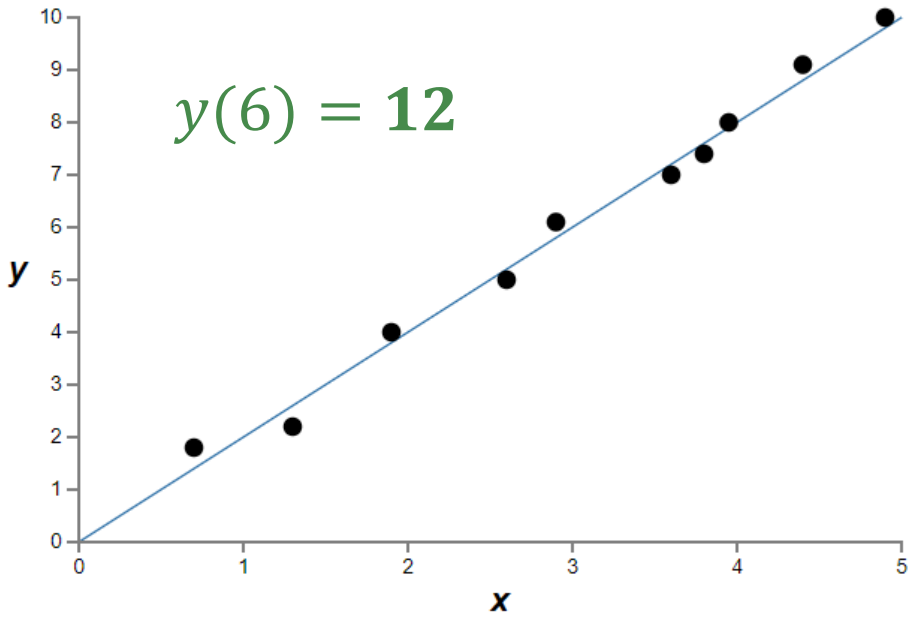
Почему бывает горе от ума?

Пример с полиномиальной аппроксимацией

$$y = 0.220539187 * x^9 - 5.49142821 * x^8 + 58.7844045 * x^7 - 353.892824 * x^6 + 1315.49254 * x^5 - 3118.09836 * x^4 + 4690.80366 * x^3 - 4296.12493 * x^2 + 2162.28823 * x - 450.983951$$

$$y = 2x$$





Правильный ответ: $y(6) = 11$

Регуляризация L_2

Регуляризация L2 (снижение весов - weight decay)

Регуляризационный терм

$$\hat{\mathbb{C}} = -\frac{1}{|V|}\sum_{(\boldsymbol{x},\boldsymbol{y})\in V}\sum_{j}\left(y_j \ln a_j^{(L)} + (1-y_j)\ln\left(1-a_j^{(L)}\right)\right) + \frac{\lambda}{2|V|}\sum_{q=1}^{Q} {w_q}^2$$

Функция перекрестной энтропии

$\lambda > 0$ – регуляризационный терм

- Регуляризационный терм способствует обучению путем подбора малых весов
- Большие веса допускаются, если они существенно уменьшают первое слагаемое

Регуляризация L2 для произвольной функции потерь $\mathbb{C}$

$$\hat{\mathbb{C}} = \mathbb{C} + \frac{\lambda}{2|V|}\sum_{q=1}^{Q} {w_q}^2$$

- $\lambda > 0$ – регулирует компромисс между уменьшением весов и близостью $\hat{\mathbb{C}}$ к исходной функции потерь $\mathbb{C}$:
  - Большие значения λ способствуют уменьшению весов
  - Малые значения λ способствуют близости $\hat{\mathbb{C}}$ к исходной функции потерь $\mathbb{C}$

Как регуляризация L2 работает в градиентном спуске

$$\hat{\mathbb{C}} = \mathbb{C} + \frac{\lambda}{2|V|}\sum_{q=1}^{Q} {w_q}^2$$

$$\frac{\partial \hat{\mathbb{C}}}{\partial w_q} = \frac{\partial \mathbb{C}}{\partial w_q} + \frac{\lambda}{|V|} w_q$$

$$w_q := w_q - \eta \frac{\partial \mathbb{C}}{\partial w_q} - \eta \frac{\lambda}{|V|} w_q = \left(1 - \frac{\eta\lambda}{|V|}\right) w_q - \eta \frac{\partial \mathbb{C}}{\partial w_q}$$

$$\frac{\partial \hat{\mathbb{C}}}{\partial b_p} = \frac{\partial \mathbb{C}}{\partial b_p}$$

$$b_p := b_p - \eta \frac{\partial \mathbb{C}}{\partial b_p}$$

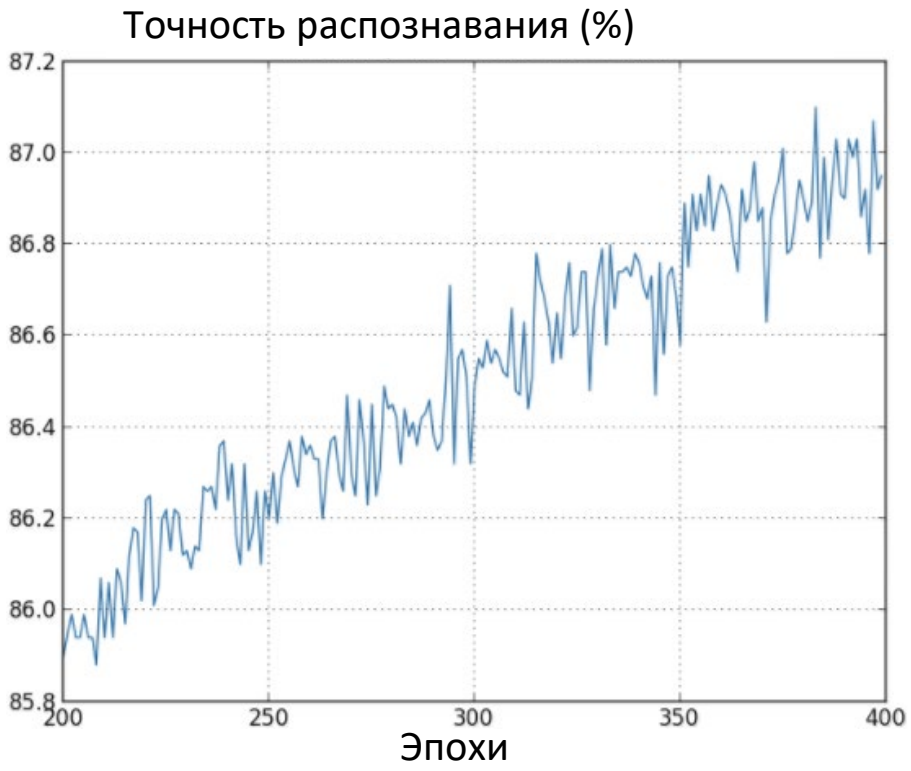
Стохастический градиентный спуск с регуляризацией L2

1. $\boldsymbol{w} \coloneqq \boldsymbol{rnd}; \boldsymbol{b} \coloneqq \boldsymbol{rnd}$ // Присваиваем случайные значения
2. **for** $epoch = 1 \ldots 10$ **do** // Цикл по эпохам
3. $V \to V_1, \ldots, V_M$ // Последовательно разбиваем *V* на подвыборки
4. **for** $i = 1 \ldots M$ **do** // Цикл по подвыборкам
5. $\nabla_{\boldsymbol{w}} \mathbb{C}_{V_i} \coloneqq \frac{1}{|V_i|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in V_i} \nabla_{\boldsymbol{w}} C_{(\boldsymbol{x},\boldsymbol{y})}$
6. $\nabla_{\boldsymbol{b}} \mathbb{C}_{V_i} \coloneqq \frac{1}{|V_i|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in V_i} \nabla_{\boldsymbol{b}} C_{(\boldsymbol{x},\boldsymbol{y})}$
7. $\boldsymbol{w} \coloneqq \left(1 - \frac{\eta\lambda}{|V|}\right) \boldsymbol{w} - \eta \nabla_{\boldsymbol{w}} \mathbb{C}_{V_i}$
8. $\boldsymbol{b} \coloneqq \boldsymbol{b} - \eta \nabla_{\boldsymbol{b}} \mathbb{C}_{V_i}$
9. **end for**
10. *shuffle*(V) // Перемешиваем *V*
11. **end for**

Влияние регуляризации L2

Размер обучающей выборки: 1 000

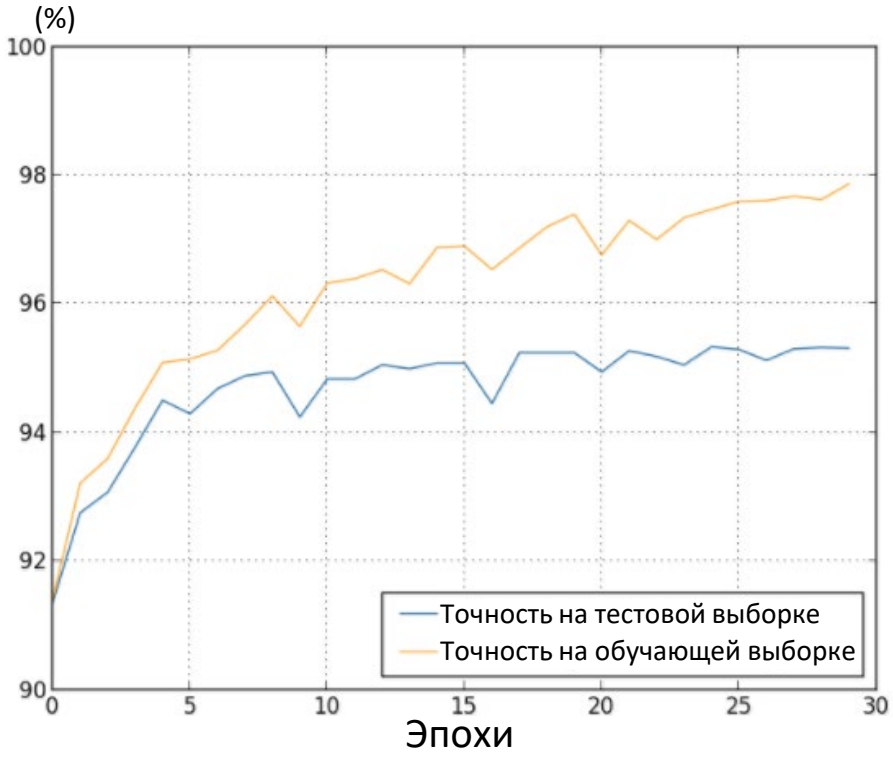
$\lambda = 0$



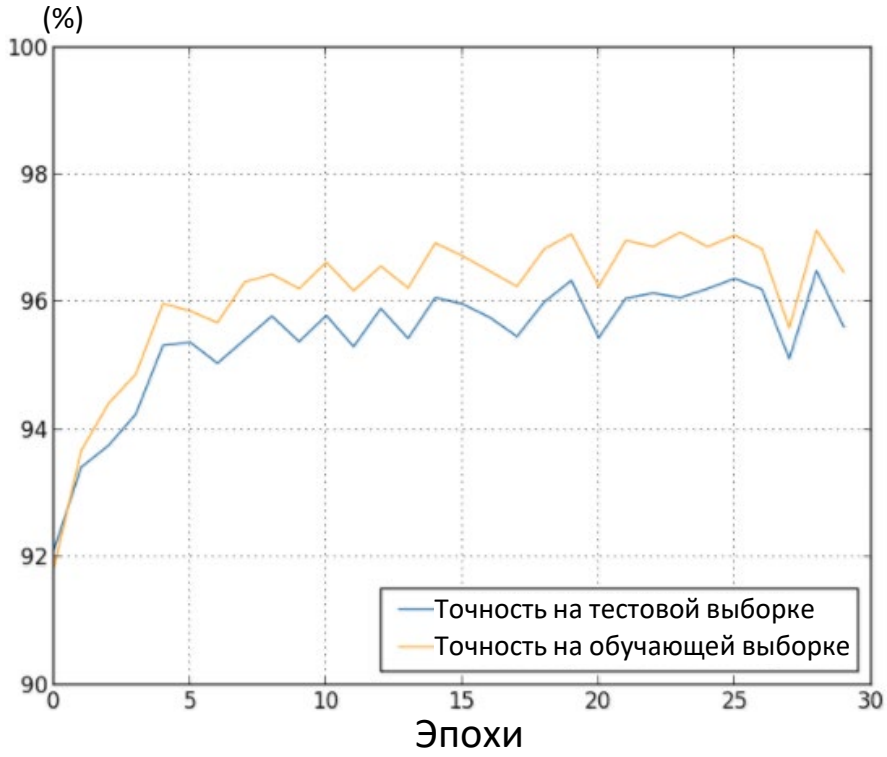
$\lambda = 0.1$

Влияние регуляризации L2

Размер обучающей выборки: 50 000



$\lambda = 0$



$\lambda = 5$

Регуляризация L1

Регуляризация L1

$$\hat{\mathbb{C}} = \mathbb{C} + \frac{\lambda}{|V|}\sum_{q=1}^{Q}|w_q|$$

- $\lambda > 0$ – регулирует компромисс между уменьшением весов и близостью $\hat{\mathbb{C}}$ к исходной функции потерь $\mathbb{C}$:
  - Большие значения λ способствуют уменьшению весов
  - Малые значения λ способствуют близости $\hat{\mathbb{C}}$ к исходной функции потерь $\mathbb{C}$

Как регуляризация L1 работает в градиентном спуске

$$\hat{\mathbb{C}} = \mathbb{C} + \frac{\lambda}{|V|}\sum_{q=1}^{Q}|w_q|$$

$$\frac{\partial \hat{\mathbb{C}}}{\partial w_q} = \frac{\partial \mathbb{C}}{\partial w_q} + \frac{\lambda}{|V|}\operatorname{sgn}(w_q)$$

$$w_q := w_q - \frac{\eta\lambda}{|V|}\operatorname{sgn}(w_q) - \eta\frac{\partial \mathbb{C}}{\partial w_q}$$

$$\frac{\partial \hat{\mathbb{C}}}{\partial b_p} = \frac{\partial \mathbb{C}}{\partial b_p}$$

$$b_p := b_p - \eta\frac{\partial \mathbb{C}}{\partial b_p}$$

Стохастический градиентный спуск с регуляризацией L1

1. $\boldsymbol{w} \coloneqq \boldsymbol{rnd}; \boldsymbol{b} \coloneqq \boldsymbol{rnd}$ // Присваиваем случайные значения
2. **for** $epoch = 1 \ldots 10$ **do** // Цикл по эпохам
3. $V \to V_1, \ldots, V_M$ // Последовательно разбиваем *V* на подвыборки
4. **for** $i = 1 \ldots M$ **do** // Цикл по подвыборкам
5. $\nabla_{\boldsymbol{w}} \mathbb{C}_{V_i} \coloneqq \frac{1}{|V_i|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in V_i} \nabla_{\boldsymbol{w}} C_{(\boldsymbol{x},\boldsymbol{y})}$
6. $\nabla_{\boldsymbol{b}} \mathbb{C}_{V_i} \coloneqq \frac{1}{|V_i|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in V_i} \nabla_{\boldsymbol{b}} C_{(\boldsymbol{x},\boldsymbol{y})}$
7. $\boldsymbol{w} \coloneqq \boldsymbol{w} - \frac{\eta\lambda}{|V|} \mathbf{sgn}(\boldsymbol{w}) - \eta \nabla_{\boldsymbol{w}} \mathbb{C}_{V_i}$
8. $\boldsymbol{b} \coloneqq \boldsymbol{b} - \eta \nabla_{\boldsymbol{b}} \mathbb{C}_{V_i}$
9. **end for**
10. *shuffle*(V) // Перемешиваем *V*
11. **end for**

L2 vs. L1

$$w_q := \left(1 - \frac{\eta\lambda}{|V|}\right) w_q - \eta \frac{\partial \mathbb{C}}{\partial w_q} \quad \Leftrightarrow \quad w_q := w_q - \frac{\eta\lambda}{|V|} \operatorname{sgn}(w_q) - \eta \frac{\partial \mathbb{C}}{\partial w_q}$$

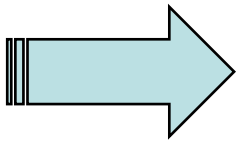
- L2 уменьшает веса пропорционально их значению
- L1 уменьшает все веса, приближая их к нулю на одну и ту же величину $\frac{\eta\lambda}{|V|}$

$\Rightarrow$ L2 уменьшает большие веса намного быстрее, чем L1

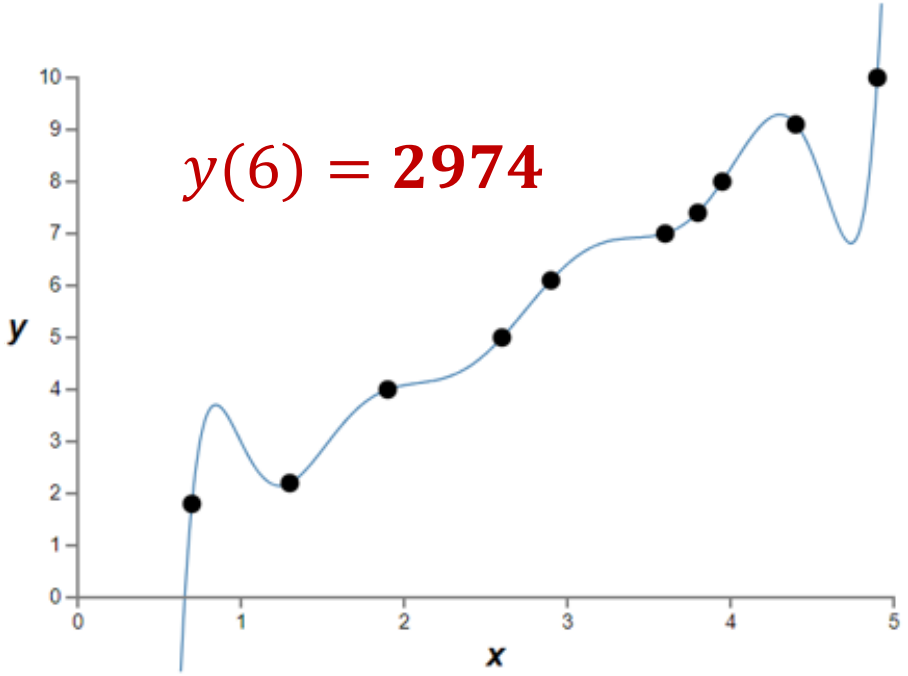
К чему приводит уменьшение коэффициентов многочлена

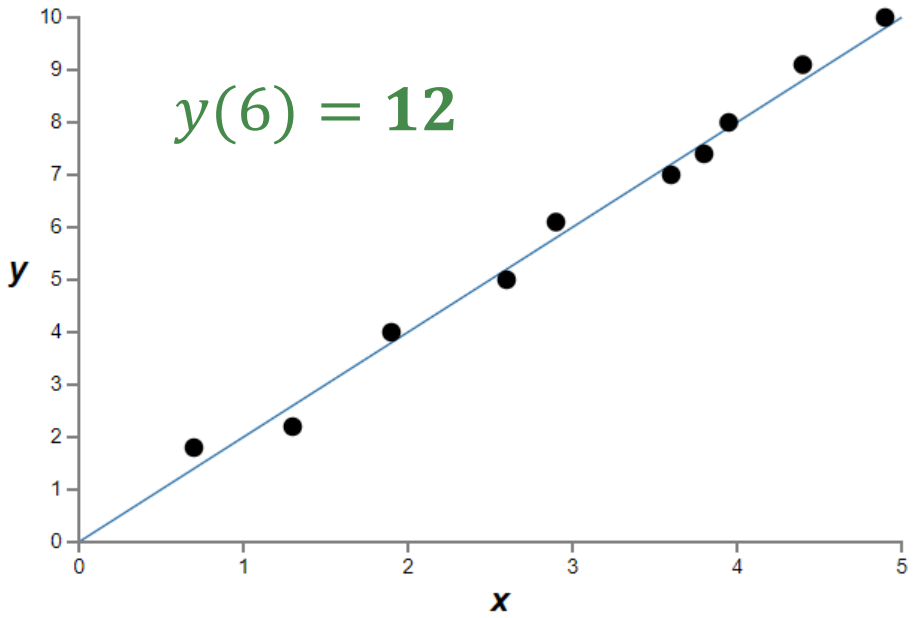
Пример с полиномиальной аппроксимацией

$$y = 0.220539187 * x^9 - 5.49142821 * x^8 + 58.7844045 * x^7 - 353.892824 * x^6 + 1315.49254 * x^5 - 3118.09836 * x^4 + 4690.80366 * x^3 - 4296.12493 * x^2 + 2162.28823 * x - 450.983951$$



$$y = 2x$$





Правильный ответ: $y(6) = 11$

В чем минус регуляризации?

Недостаток регуляризации

$$L_2: \hat{\mathbb{C}} = \mathbb{C} + \frac{\lambda}{2|V|} \sum_{q=1}^{Q} {w_q}^2$$

$$L_1: \hat{\mathbb{C}} = \mathbb{C} + \frac{\lambda}{|V|} \sum_{q=1}^{Q} |w_q|$$

$$\mathbb{C} \geq 0$$

$$\Downarrow$$

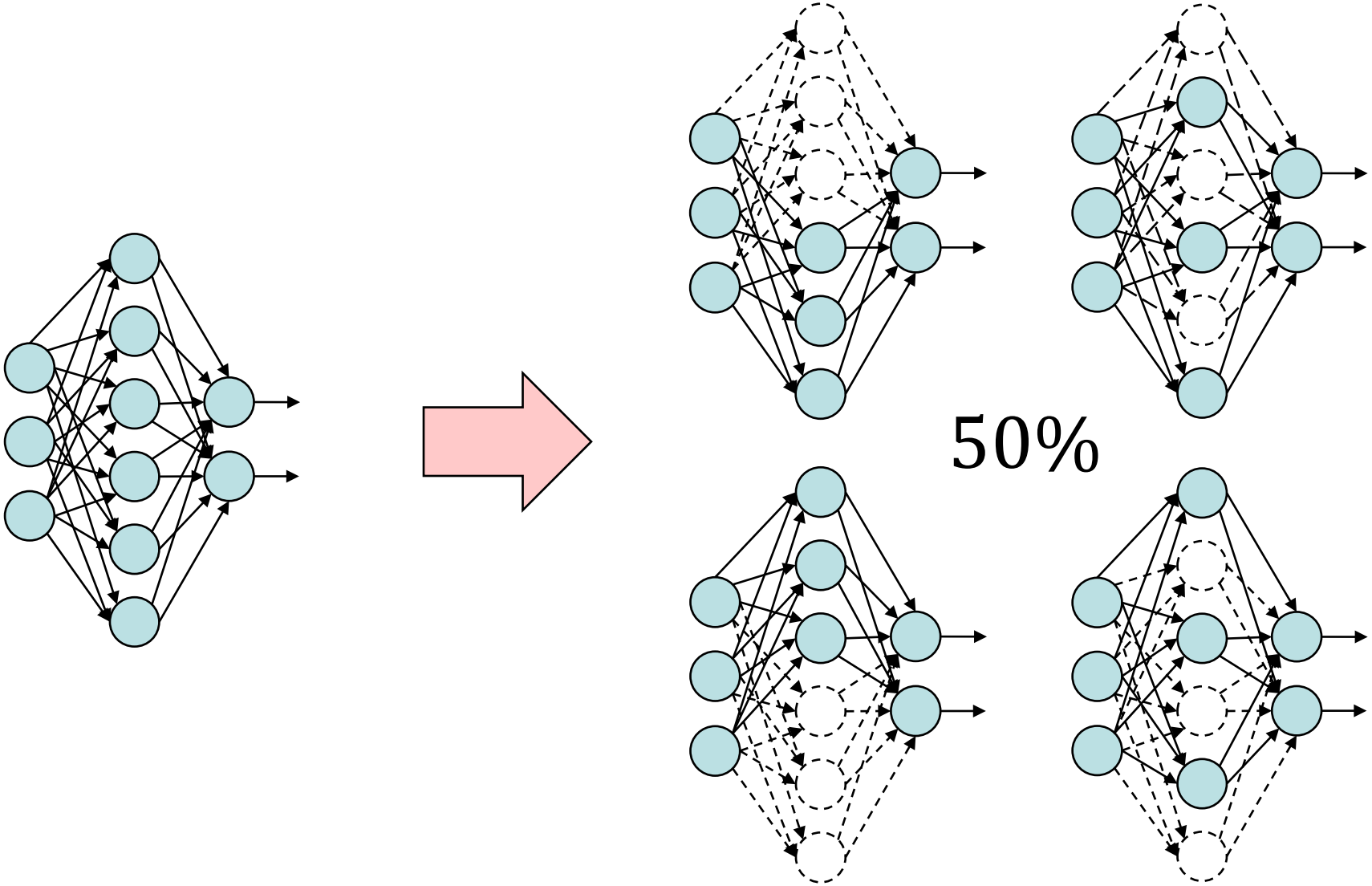
Минимум функции потерь всегда больше нуля

$$\Downarrow$$

Нельзя явно отличить локальный минимум от глобального

Прореживание (dropout)

Прореживание (dropout)



Прореживание (dropout)

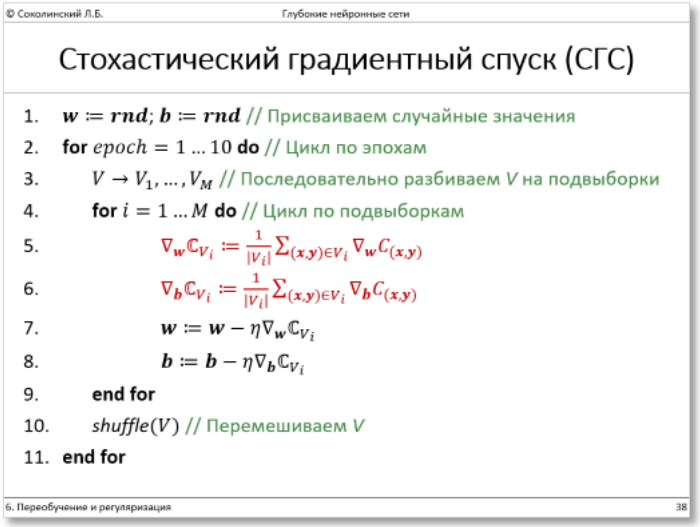
A. Случайным образом временно удаляем половину нейронов внутреннего слоя

B. С помощью алгоритмов прямого распространения сигнала и обратного распространения ошибки вычисляем градиенты функции потерь по весам и смещениям (шаги 5, 6 стохастического градиентного спуска)

C. Обновляем веса и смещения прореженной нейронной сети (шаги 7, 8 стохастического градиентного спуска)

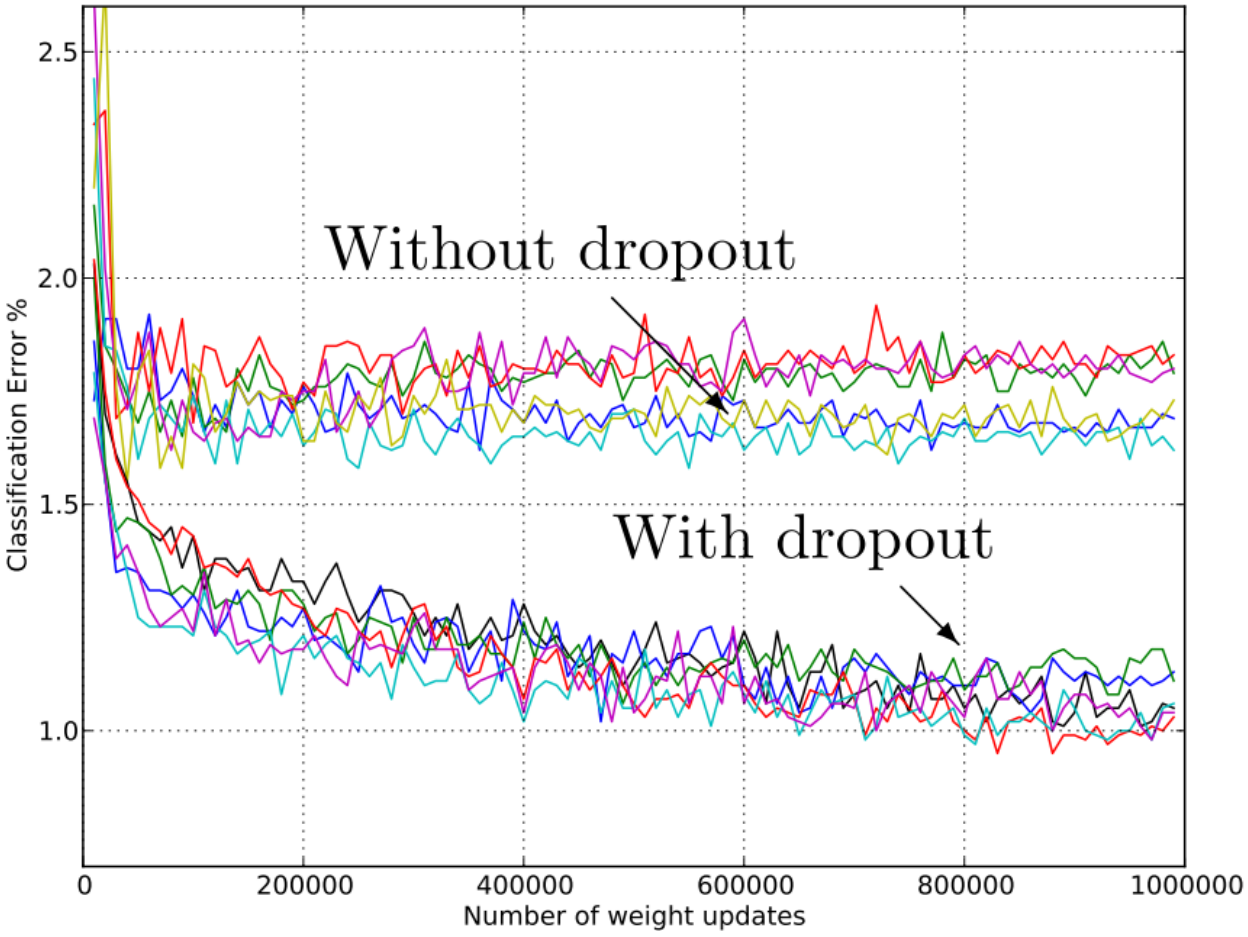
D. Повторяем шаги A-C для каждой подвыборки (в цикле по подвыборкам)

E. После завершения обучения:
в полной сети активационные сигналы внутреннего слоя делим на 2



Эффект от прореживания

Data Set	Domain
MNIST	Vision
SVHN	Vision
CIFAR-10/100	Vision
ImageNet (ILSVRC-2012)	Vision
TIMIT	Speech
Reuters-RCV1	Text
Alternative Splicing	Genetics



Hinton G.E. et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting // Journal of Machine Learning Research. 2012. Vol. 15. P. 1929–1958.

Почему это работает?

- Сети с голосованием
  - Идея:
    - Возьмем пять одинаковых сетей
    - Зададим на них **разные** начальные веса и смещения
    - Обучим их независимо на одной и той же обучающей выборке
    - Для решения реальной задачи используем одновременно все пять сетей
    - При расхождении результатов проводим между ними голосование
  - Сети с голосованием резко повышают процент правильных ответов
  - Обучение сетей с голосованием требует очень большого времени
- Прореживание (dropout) моделирует работу большого количества сетей с голосованием без увеличения времени обучения

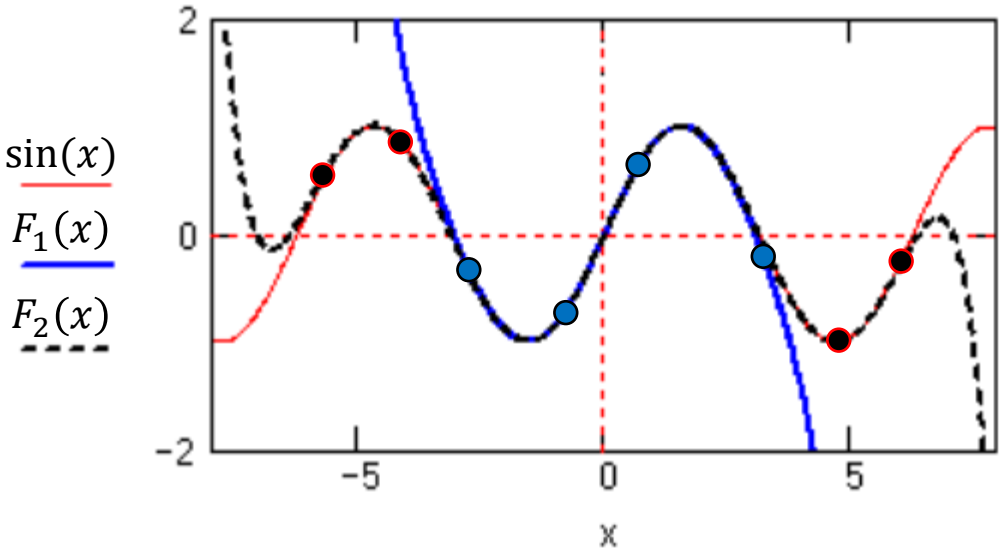
Аугментация (augmentation) данных

Искусственное увеличение обучающей выборки

Пример с аппроксимацией функции

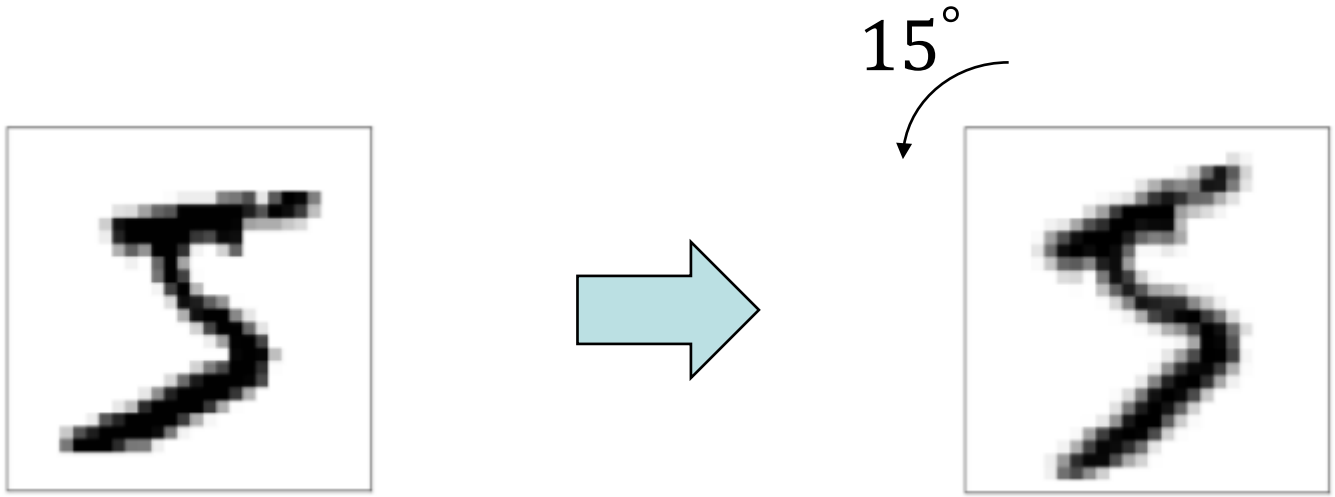
$$F_1(x) = x - \frac{1}{6}x^3 + \frac{1}{120}x^5$$

$$F_2(x) = x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \frac{1}{5040}x^7$$



На малом наборе данных невозможно обучить глубокую нейронную сеть для решения сложной задачи

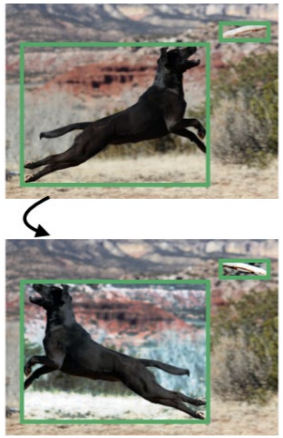
Как искусственно увеличить обучающую выборку?



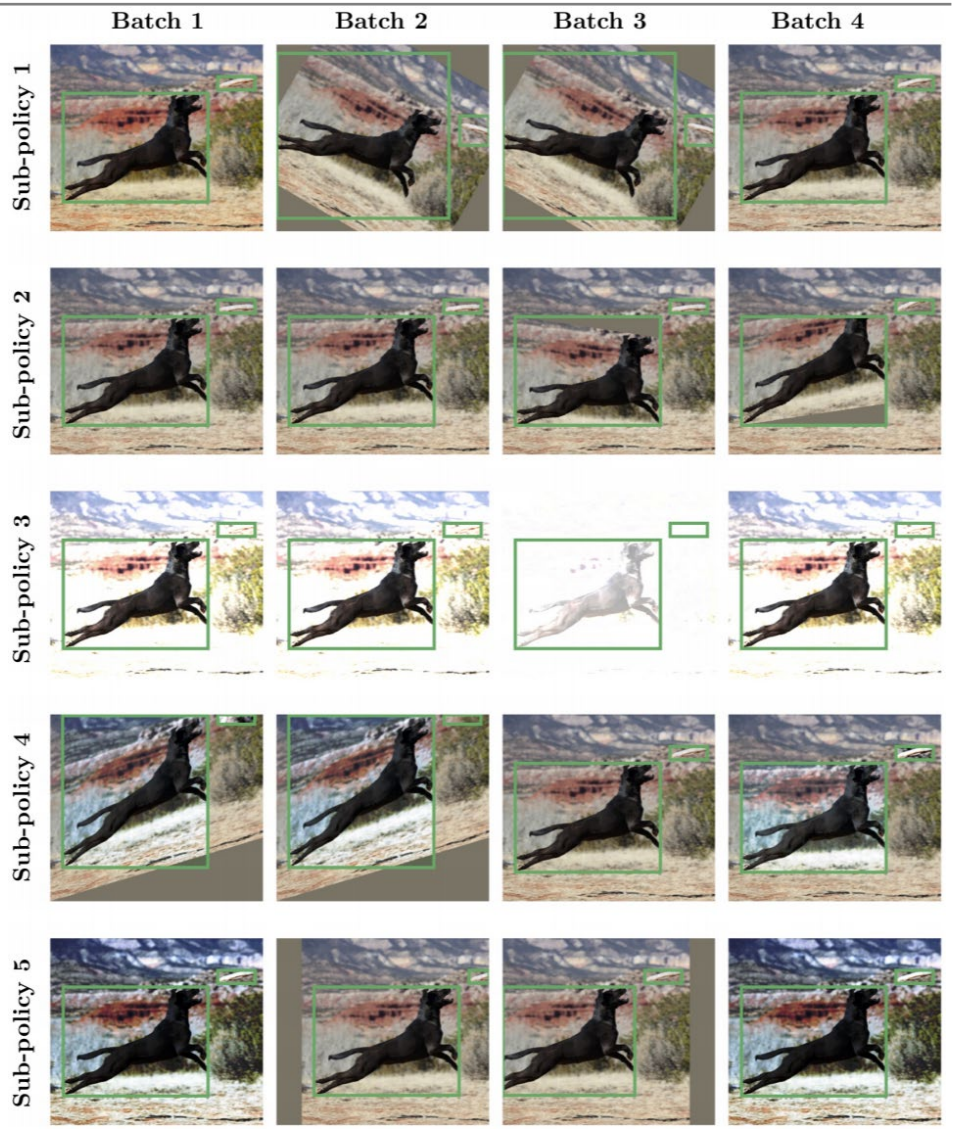
- Поворот
- Сжатие/увеличение
- Размытие
- Изменение фона
- Генеративно-состязательные сети
- …

https://habr.com/ru/company/smartengines/blog/264677/

Аугментация данных для обучения нейронной сети, распознающей собак, ловящих фрисби

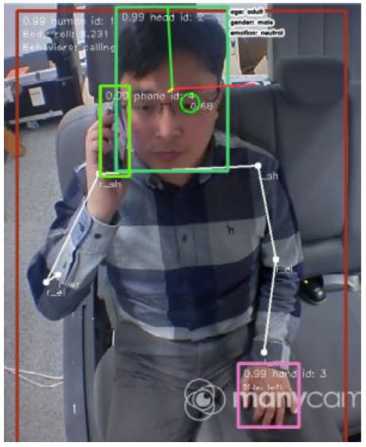


https://arxiv.org/pdf/1906.11172.pdf



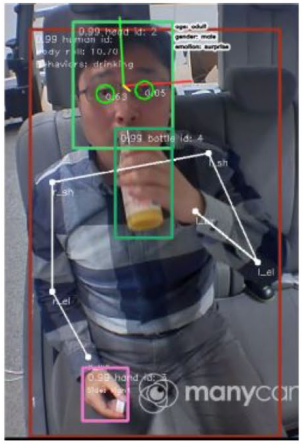
Генерация синтетических данных для видеоаналитики

3DiVi
ООО "ТРИДИВИ"

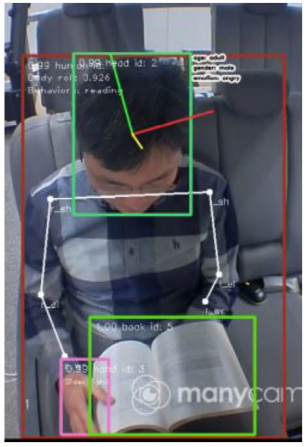


Calling

Smoking



Drinking



Reading

Конец лекции 6

Вспомогательные слайды

Стохастический градиентный спуск (СГС)

1. $\boldsymbol{w} \coloneqq \boldsymbol{rnd}; \boldsymbol{b} \coloneqq \boldsymbol{rnd}$ // Присваиваем случайные значения
2. **for** $epoch = 1 \ldots 10$ **do** // Цикл по эпохам
3. $V \to V_1, \ldots, V_M$ // Последовательно разбиваем *V* на подвыборки
4. **for** $i = 1 \ldots M$ **do** // Цикл по подвыборкам
5. $\nabla_{\boldsymbol{w}} \mathbb{C}_{V_i} \coloneqq \frac{1}{|V_i|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in V_i} \nabla_{\boldsymbol{w}} C_{(\boldsymbol{x},\boldsymbol{y})}$
6. $\nabla_{\boldsymbol{b}} \mathbb{C}_{V_i} \coloneqq \frac{1}{|V_i|} \sum_{(\boldsymbol{x},\boldsymbol{y}) \in V_i} \nabla_{\boldsymbol{b}} C_{(\boldsymbol{x},\boldsymbol{y})}$
7. $\boldsymbol{w} \coloneqq \boldsymbol{w} - \eta \nabla_{\boldsymbol{w}} \mathbb{C}_{V_i}$
8. $\boldsymbol{b} \coloneqq \boldsymbol{b} - \eta \nabla_{\boldsymbol{b}} \mathbb{C}_{V_i}$
9. **end for**
10. *shuffle*(V) // Перемешиваем *V*
11. **end for**

Функция сигнум (знак) $\mathrm{sgn}(x)$

$$\mathrm{sgn}(x) = \begin{cases} 1 \text{ для } x > 0 \\ 0 \text{ для } x = 0 \\ -1 \text{ для } x < 0 \end{cases}$$

Глобальный и локальные минимумы функции потерь $\mathbb{C}$