Основы глубинного обучения

Лекция 5

Свёрточные сети. Оптимизация в глубинном обучении.

Евгений Соколов

esokolov@hse.ru

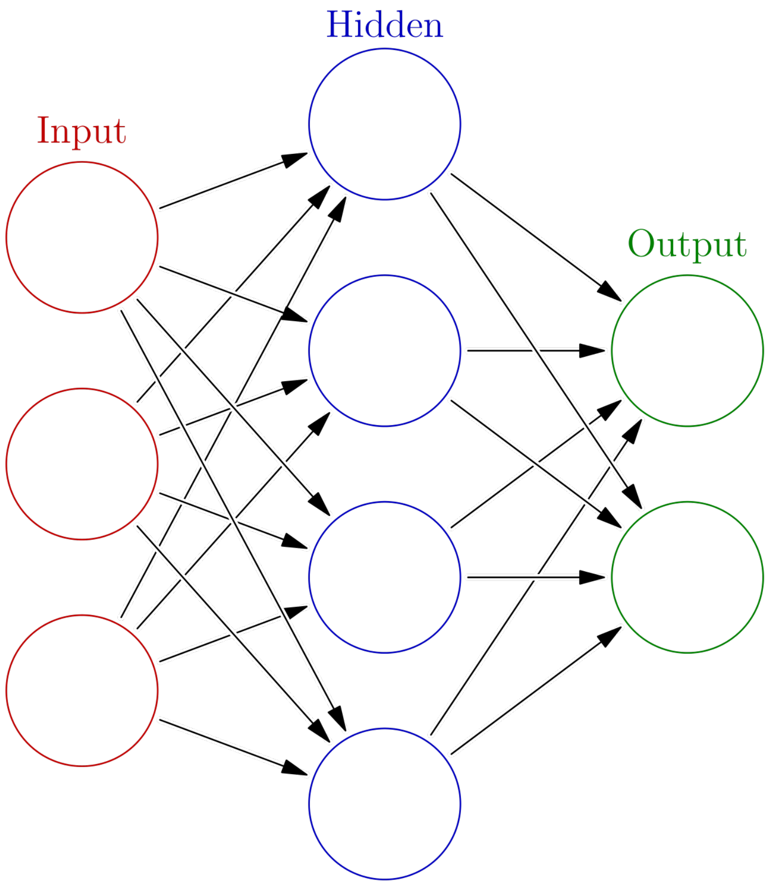
НИУ ВШЭ, 2021

Dropout

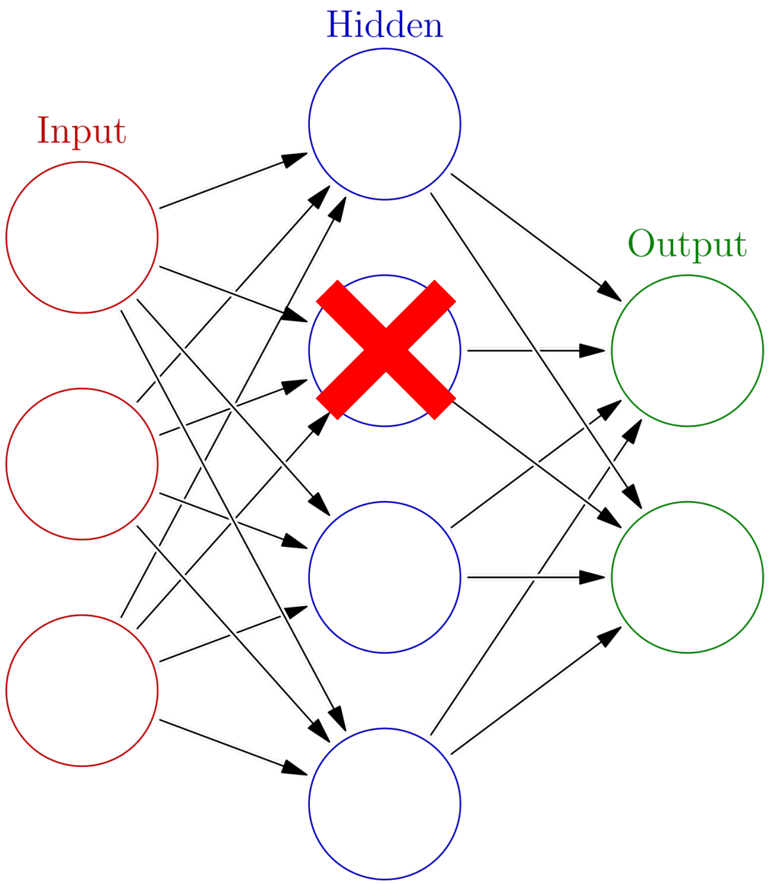
Борьба с переобучением

- Сокращение числа параметров (свёрточные слои помогают с этим)
- Регуляризация
- Можно как-то ещё мешать модели подгоняться под обучающую выборку

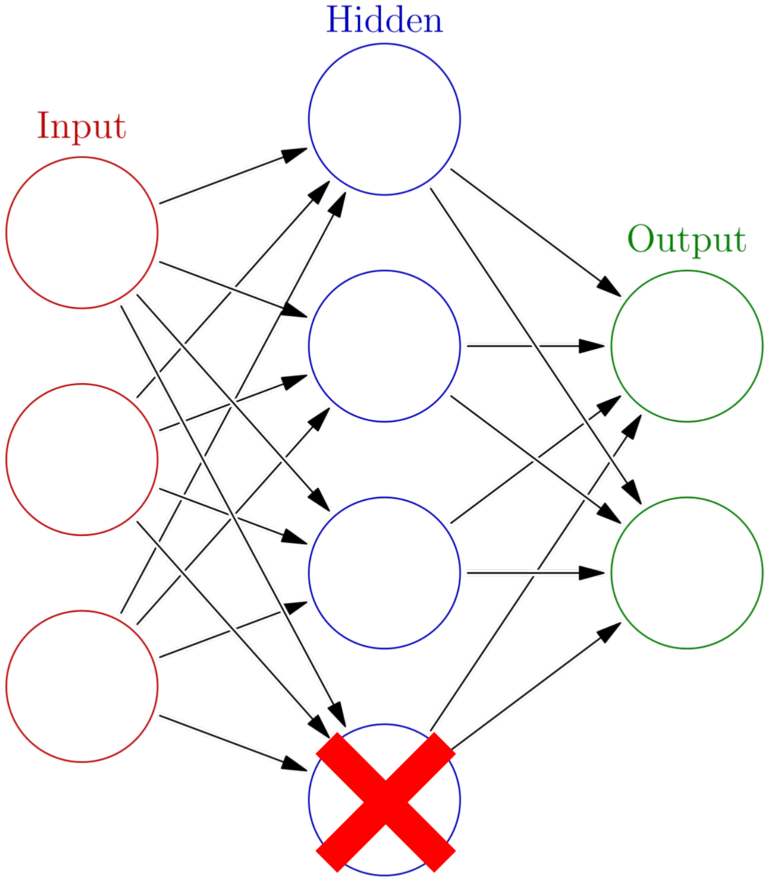
Dropout



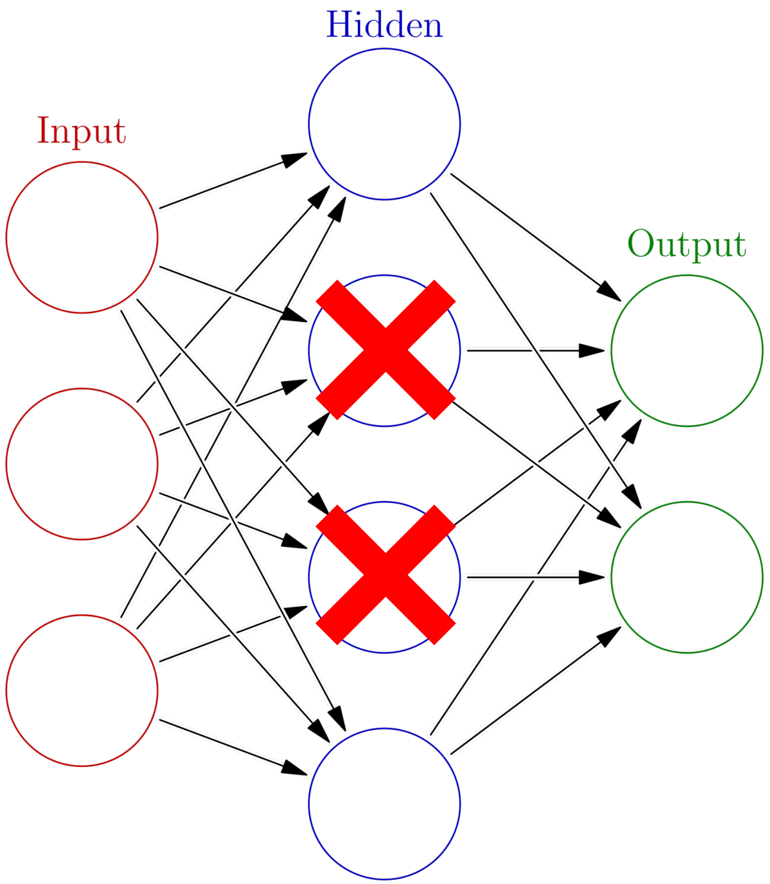
Dropout



Dropout



Dropout



Dropout

- Можно определить как слой $d(x)$
- Параметров нет, единственный гиперпараметр — p (вероятность удаления нейрона)
- На этапе обучения:

$$d(x) = \frac{1}{p} m \circ x$$

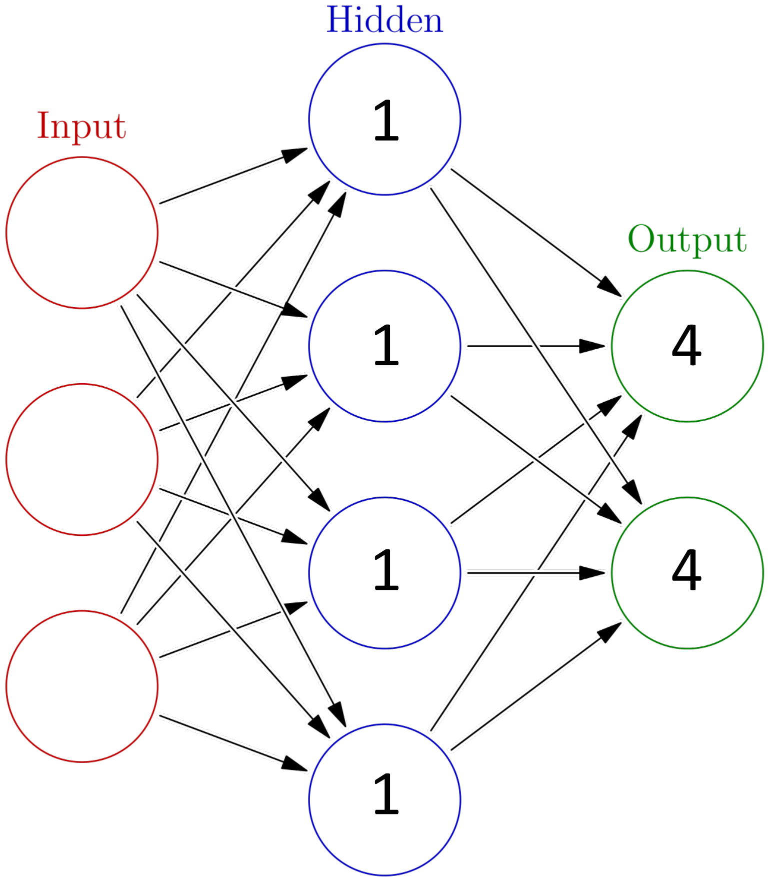
(m — вектор того же размера, что и x, элемент берутся из распределения $\mathrm{Ber}(p)$)

- Деление на p — для сохранения суммарного масштаба выходов

https://jmlr.org/papers/v15/srivastava14a.html

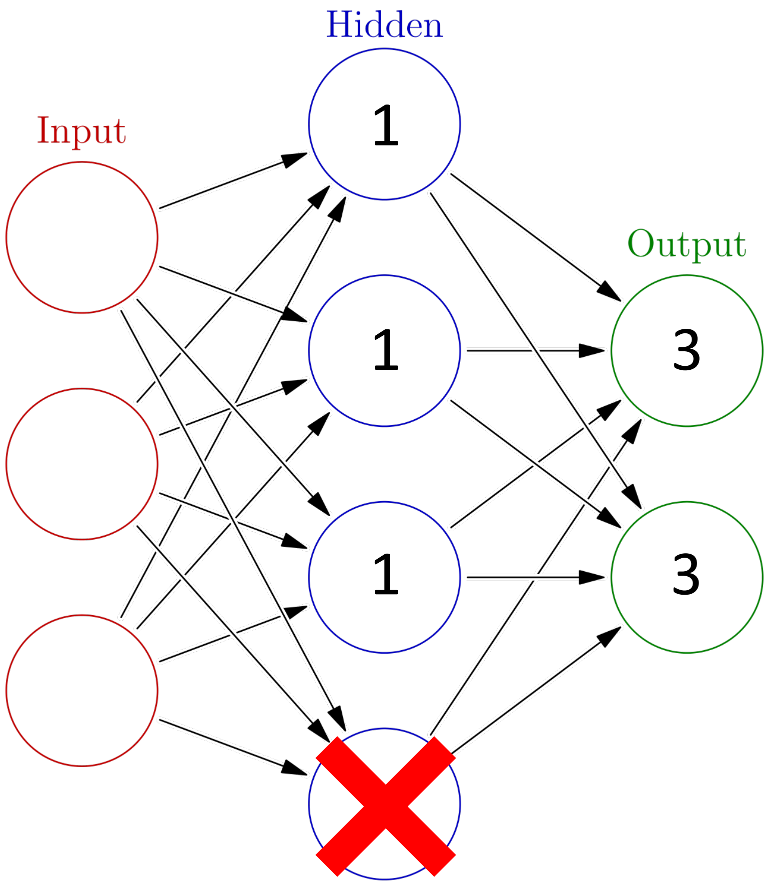
Dropout

Пусть все веса единичные



Dropout

Пусть все веса единичные



Надо компенсировать снижение масштаба суммы выходов!

Dropout

- На этапе обучения:

$$d(x) = \frac{1}{p} m \circ x$$

- На этапе применения:

$$d(x) = x$$

В оригинальной статье нет нормировки на этапе обучения, но есть домножение на p на этапе применения

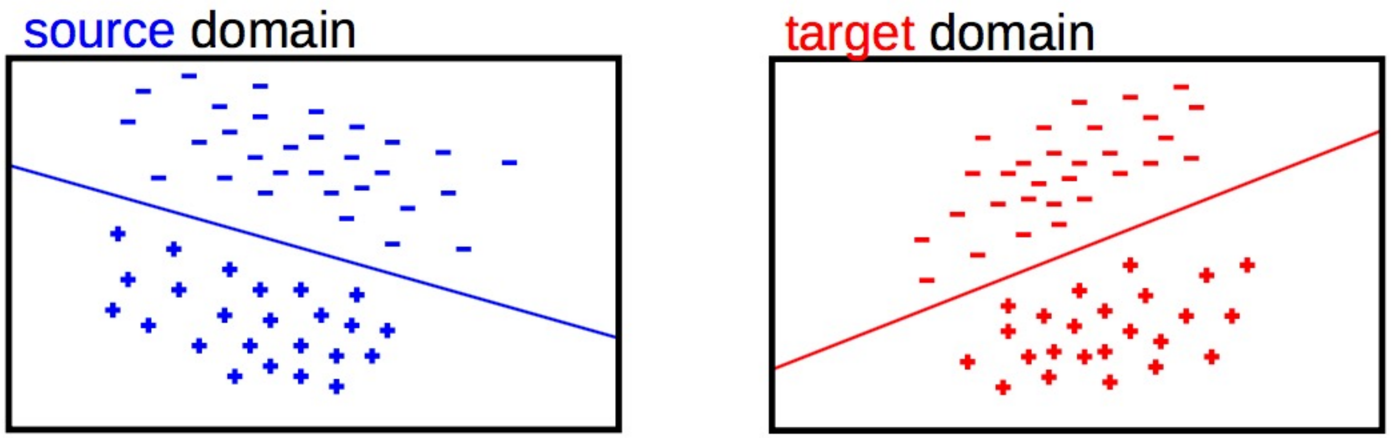
Вариант на слайде — inverted dropout (чуть меньше операций во время применения сети)

Dropout

- Интерпретация: мы обучаем все возможные архитектуры нейросетей, которые получаются из исходной выбрасыванием отдельных нейронов
- У всех этих архитектур общие веса
- На этапе применения (почти) усредняем прогнозы всех этих архитектур

Нормализации

Covariate shift



Covariate shift

- В классическом машинном обучении — изменение распределения данных
- Много методов решения

Domain adaptation

- Объекты по-разному распределены на обучении и на контроле
- Идея: взвешивать объекты при обучении

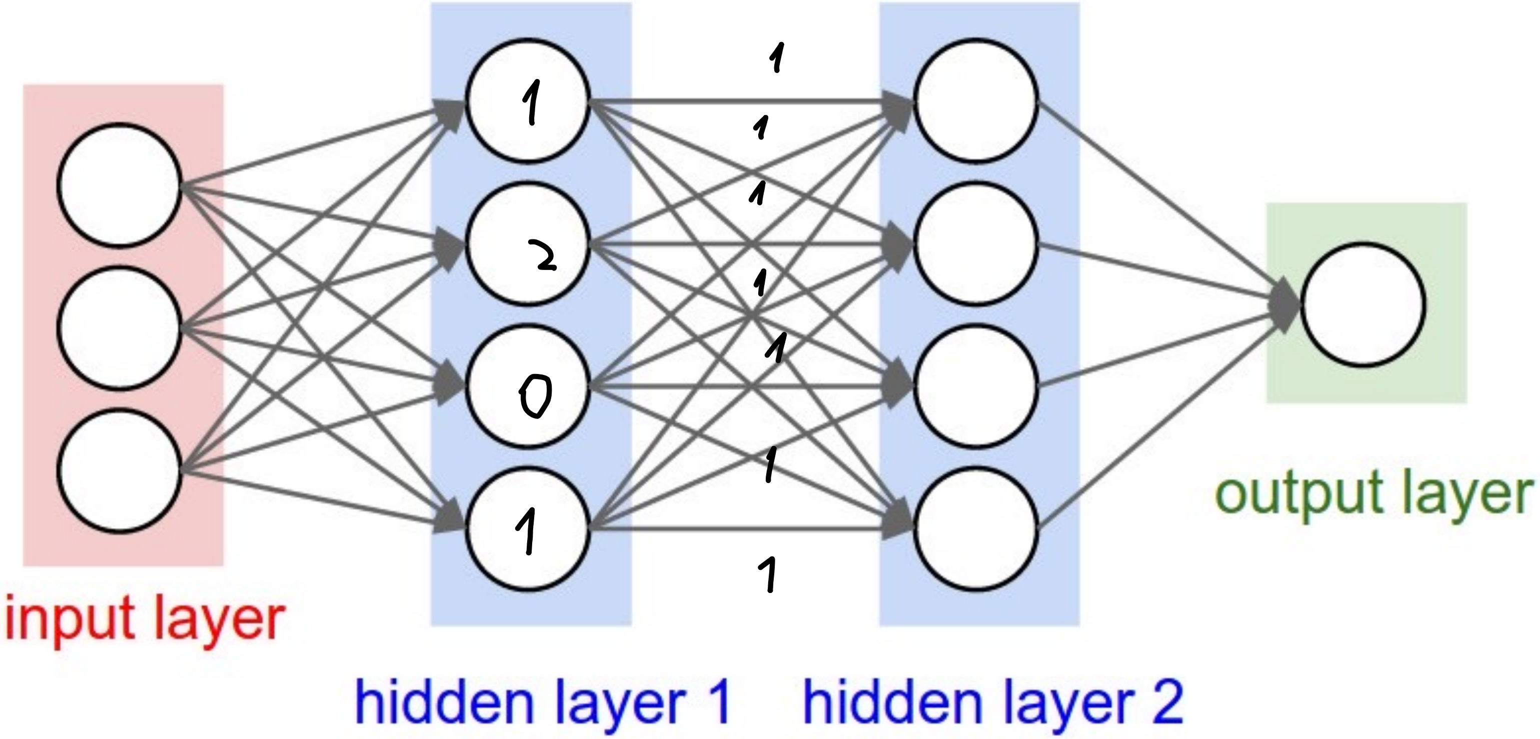
$$\sum_{i=1}^{\ell} s_i (a(x_i) - y_i)^2 \to \min$$

- Большие веса будем ставить объектам, которые похожи на объекты из тестовой выборки

Internal covariate shift

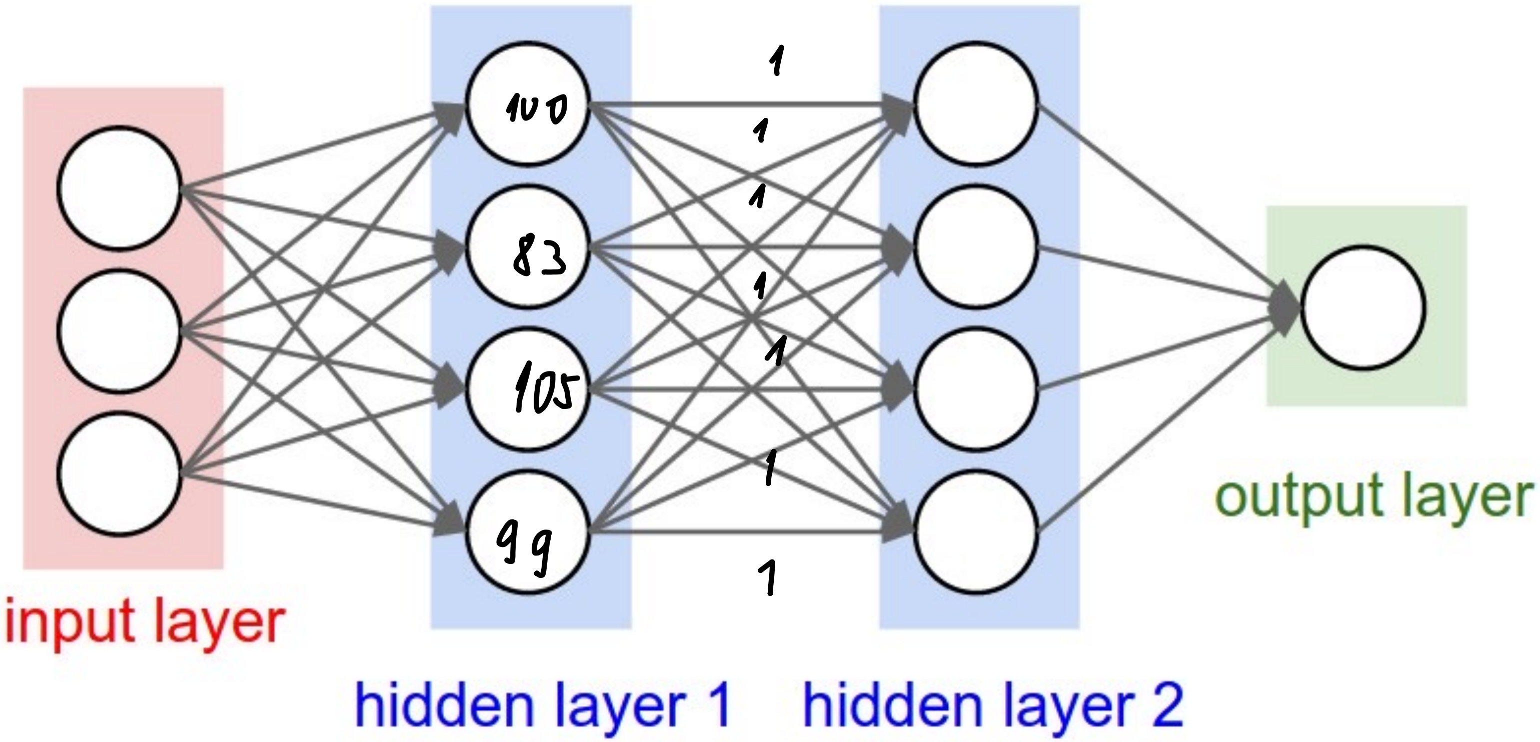
- В нейронной сети каждый слой обучается на выходах предыдущих слоёв
- Если слой в начале сильно меняется, то все следующие слои надо переделывать

Internal covariate shift



Internal covariate shift

Допустим, веса первого слоя сильно поменялись после градиентного шага



Internal covariate shift

- Идея: преобразовывать выходы слоёв так, чтобы они гарантированно имели фиксированное распределение

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

Batch Normalization

- Реализуется как отдельный слой
- Вычисляется для текущего батча
- Оценим среднее и дисперсию каждой компоненты входного вектора:

$$\mu_B = \frac{1}{n}\sum_{j=1}^{n} x_{B,j}$$

покоординатно

$$\sigma_B^2 = \frac{1}{n}\sum_{j=1}^{n} \left(x_{B,j} - \mu_B\right)^2$$

$x_{B,j}$ — j-й объект в батче B

Batch Normalization

- Отмасштабируем все выходы:

$$\tilde{x}_{B,j} = \frac{x_{B,j} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

- Зададим нужные нам среднее и дисперсию:

$$z_{B,j} = \gamma \circ \tilde{x}_{B,j} + \beta$$

обучаемые параметры (векторы, размерность равна размерности входных векторов)

Batch Normalization

$$\begin{pmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{pmatrix}$$

Приводим среднее и дисперсию к β_1 и γ_1

Приводим среднее и дисперсию к β_d и γ_d

- n — размер батча
- d — размерность входного вектора

Batch Normalization

Важно: после BatchNorm среднее и дисперсия каждого выхода зависят только от параметров нормализации, но не от параметров прошлых слоёв!

Batch Normalization

Во время применения нейронной сети:

- Те же самые формулы, но вместо μ_B и σ_B^2 используем их средние значения по всем батчам

Batch Normalization

- Обычно вставляется между полносвязным/свёрточным слоём и нелинейностью
- Позволяет увеличить длину шага в градиентном спуске

- Не факт, что действительно устраняет covariance shift

В чём польза от BatchNorm?

How Does Batch Normalization Help Optimization?

Shibani Santurkar* MIT shibani@mit.edu

Dimitris Tsipras* MIT tsipras@mit.edu

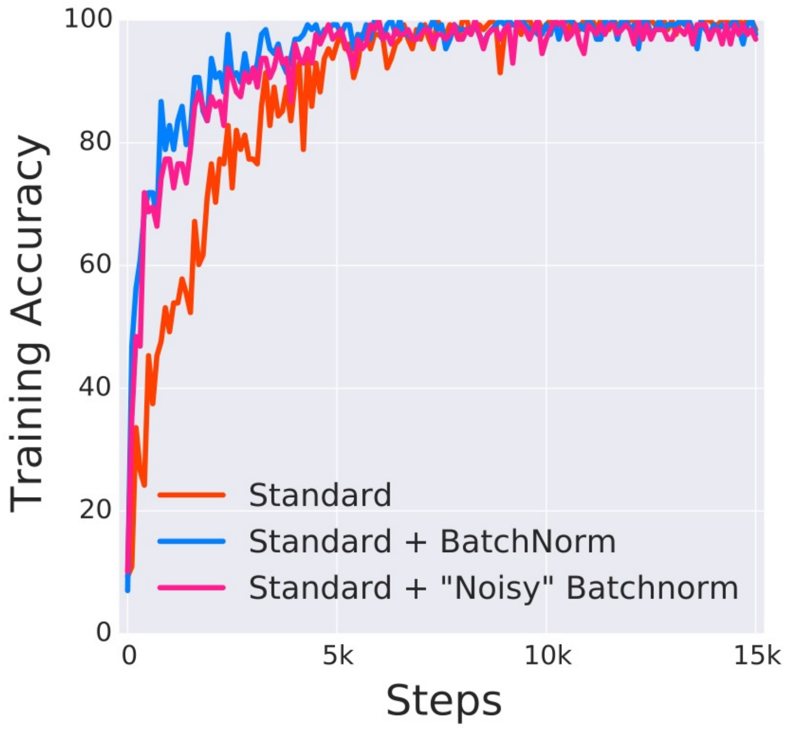
Andrew Ilyas* MIT ailyas@mit.edu

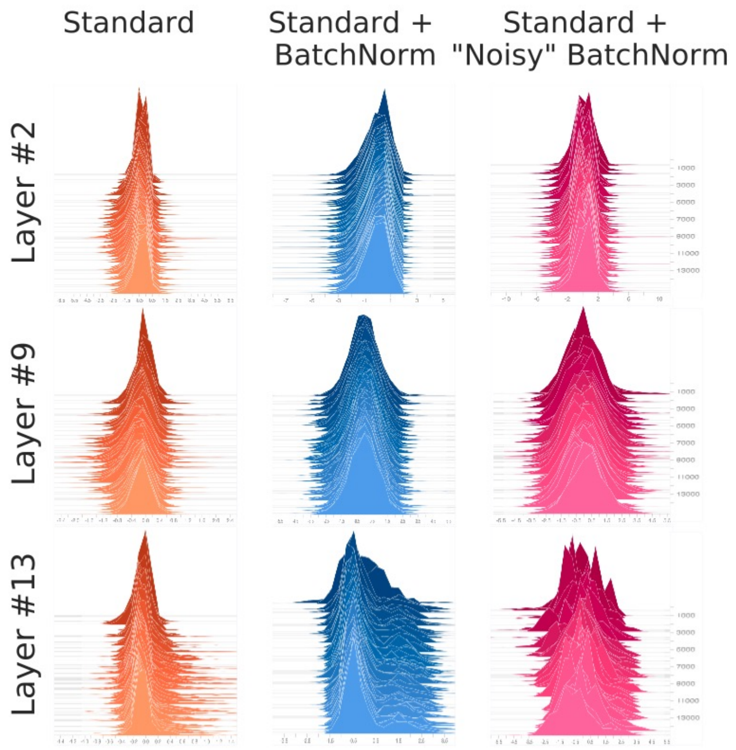
Aleksander Mądry MIT madry@mit.edu

https://arxiv.org/pdf/1805.11604.pdf

В чём польза от BatchNorm?

• Добавим шум после нормализации — хуже не становится!

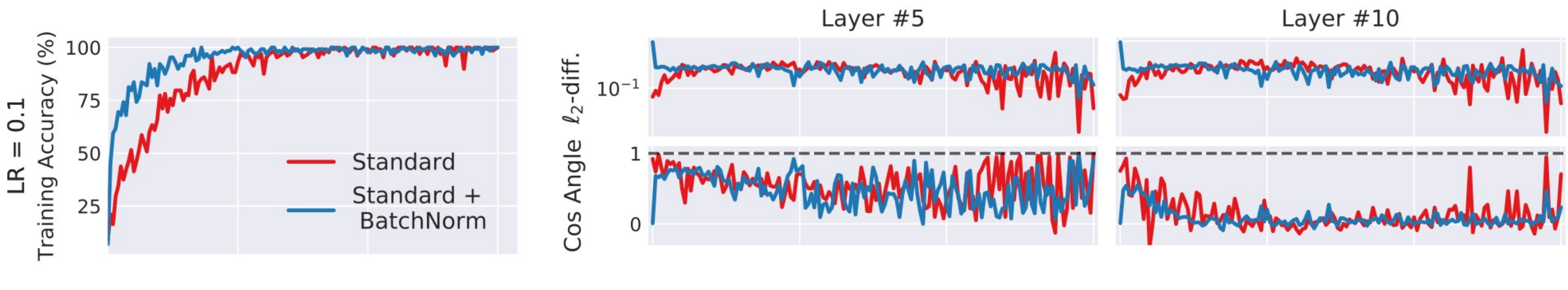




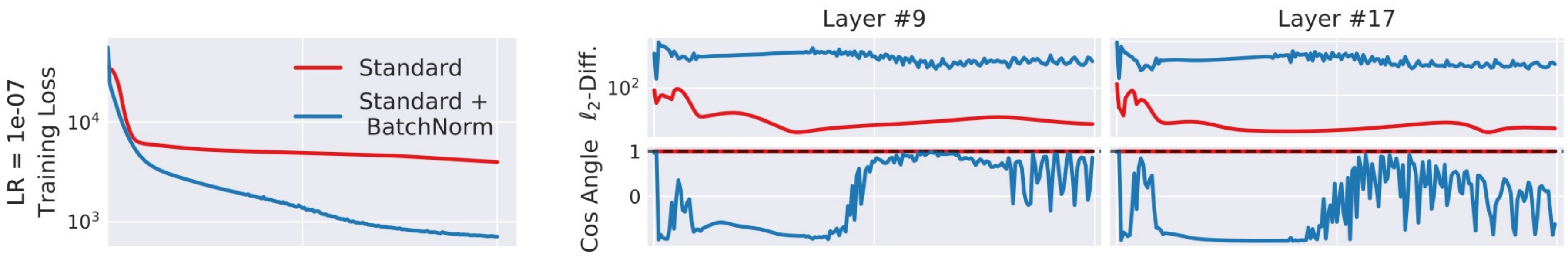
https://arxiv.org/pdf/1805.11604.pdf

В чём польза от BatchNorm?

- Как связаны градиенты на соседних итерациях?



(a) VGG

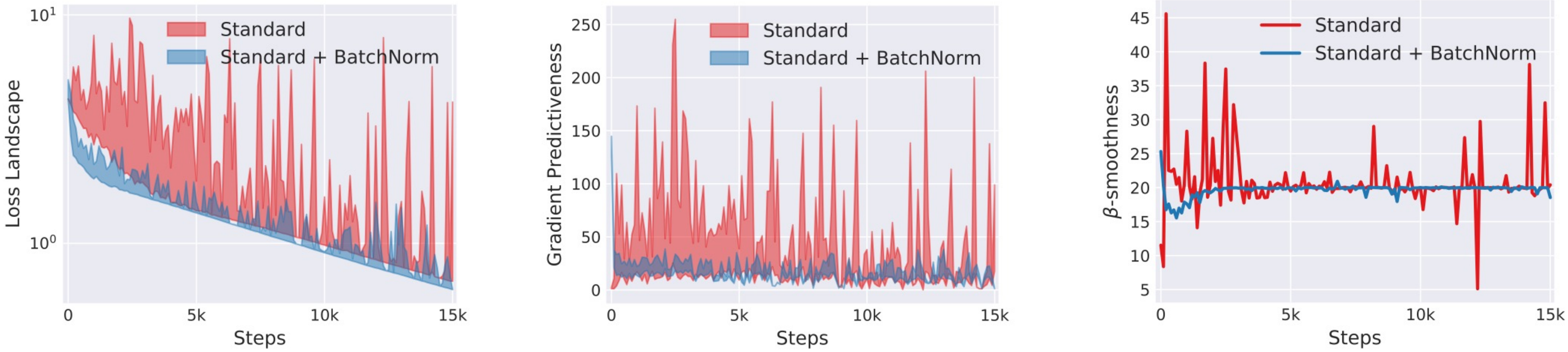


(b) DLN

https://arxiv.org/pdf/1805.11604.pdf

В чём польза от BatchNorm?

- Функционал ошибки становится более «гладким»!



(a) loss landscape (b) gradient predictiveness (c) “effective” β-smoothness

https://arxiv.org/pdf/1805.11604.pdf

Batch Normalization

$$\begin{pmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{pmatrix}$$

Приводим среднее и дисперсию к β_1 и γ_1

Приводим среднее и дисперсию к β_d и γ_d

- n — размер батча
- d — размерность входного вектора

Layer Normalization

$$\begin{pmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{pmatrix}$$

Приводим среднее и дисперсию к β и γ

Приводим среднее и дисперсию к β и γ

- n — размер батча
- d — размерность входного вектора

https://arxiv.org/pdf/1607.06450.pdf

Layer Normalization

- Нормализуем распределение «признаков» одного объекта

$$\mu_i = \frac{1}{d}\sum_{j=1}^{d} x_{ij}$$

$$\sigma_i^2 = \frac{1}{d}\sum_{j=1}^{d} \left(x_{ij} - \mu\right)^2$$

x_{ij} — j-й признак i-го объекта

https://arxiv.org/pdf/1607.06450.pdf

Layer Normalization

- Отмасштабируем все выходы:

$$\tilde{x}_{ij} = \frac{x_{ij} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}}$$

- Зададим нужные нам среднее и дисперсию:

$$z_{ij} = \gamma \circ \tilde{x}_{ij} + \beta$$

обучаемые параметры (скаляры)

https://arxiv.org/pdf/1607.06450.pdf

Инициализации

Инициализация весов

- Не должно быть симметрий (плохо инициализировать всё одним числом)
- Хороший вариант:

$$w_j \sim \frac{2}{\sqrt{n}} \mathcal{N}(0, 1)$$

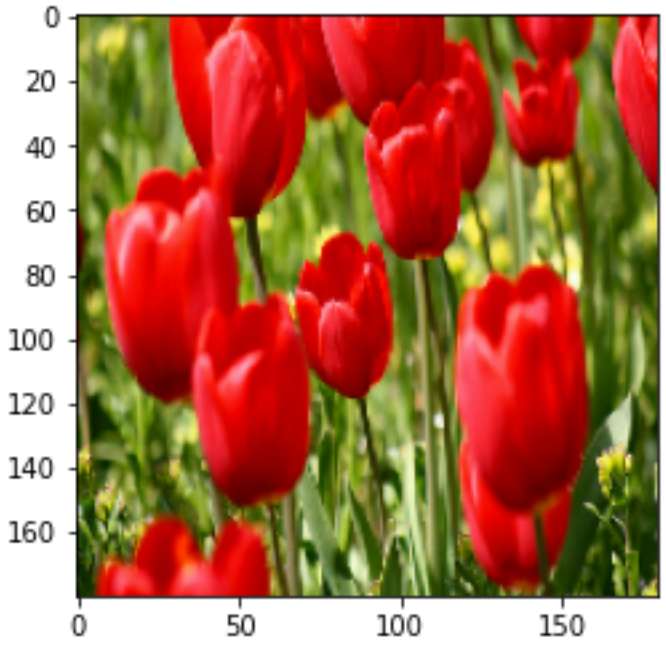
n — число входов

- Пытаемся сделать так, чтобы масштаб всех выходов был примерно одинаковым

https://pouannes.github.io/blog/initialization/

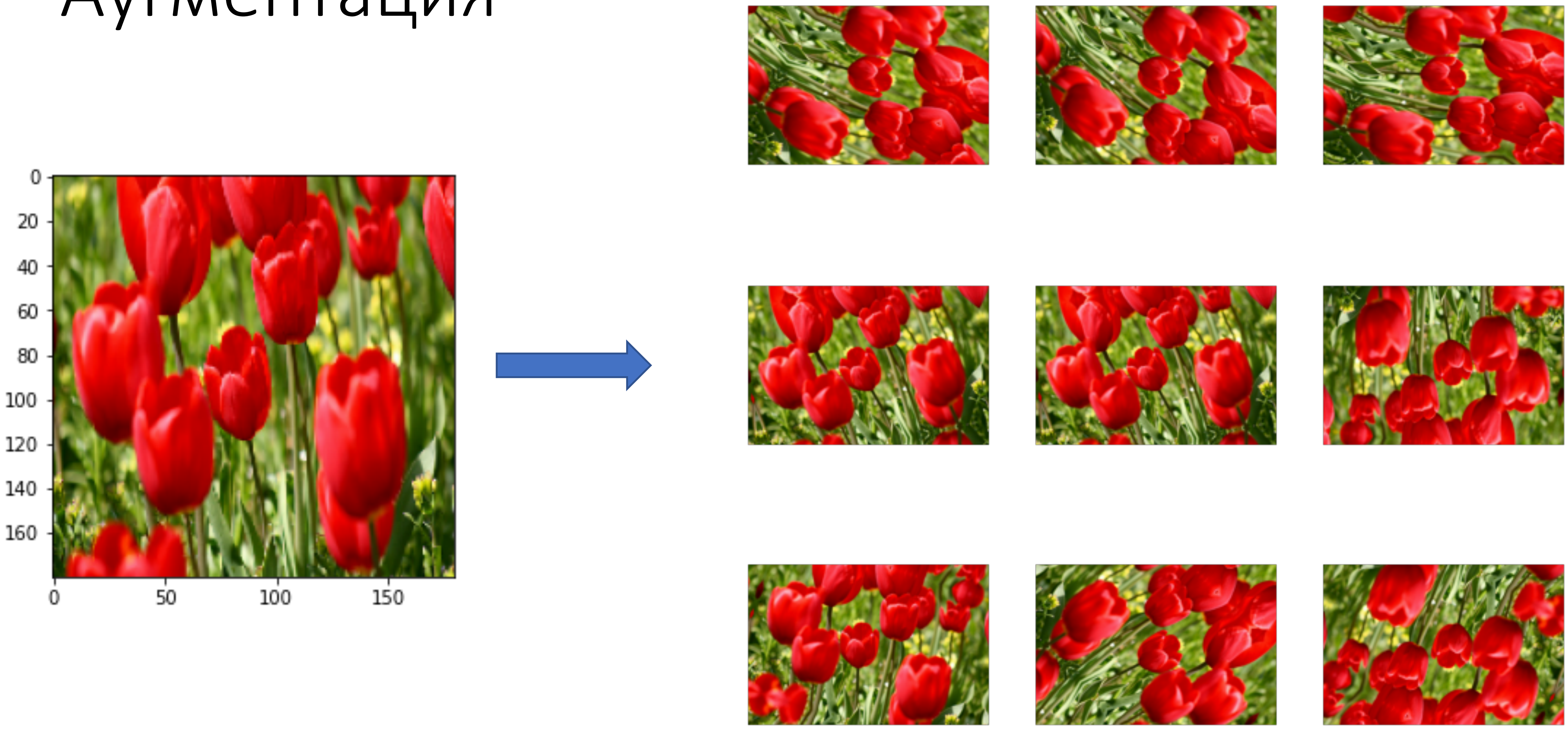
Аугментация

Аугментация



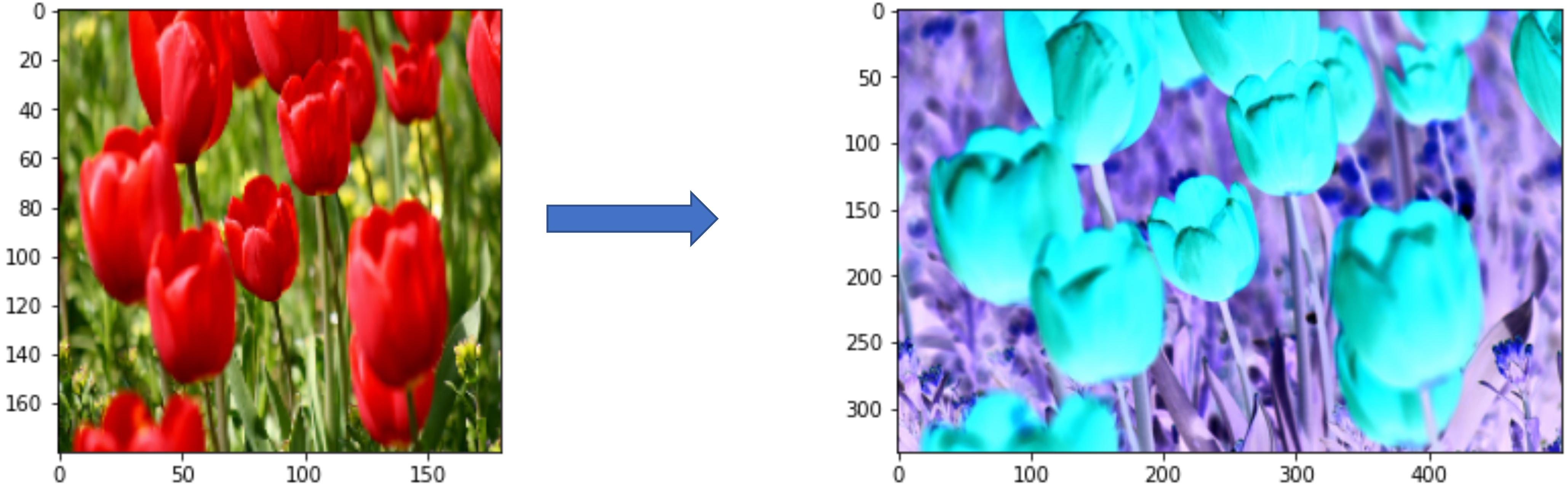
https://www.tensorflow.org/tutorials/images/data_augmentation

Аугментация



https://www.tensorflow.org/tutorials/images/data_augmentation

Аугментация



https://www.tensorflow.org/tutorials/images/data_augmentation

https://github.com/albumentations-team/albumentations

Аугментация

- Много разных вариантов
- «Бесплатное» расширение обучающей выборки
- В некотором смысле регуляризация модели

- Обычно аугментации случайно применяют к картинкам из текущего батча
- На этапе применения можно сделать несколько аугментаций картинки, применить сеть к каждой, усреднить предсказания