

Сверточные нейронные сети:
практическое применение

ЦИФРОВАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ

ПЛАН ЗАНЯТИЯ

1

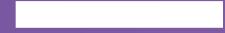
Данные
по задачам
компьютерного
зрения

2

Архитектуры
сверточных
сетей

3

Обучение
сверточной
сети
на практике



IMAGENET

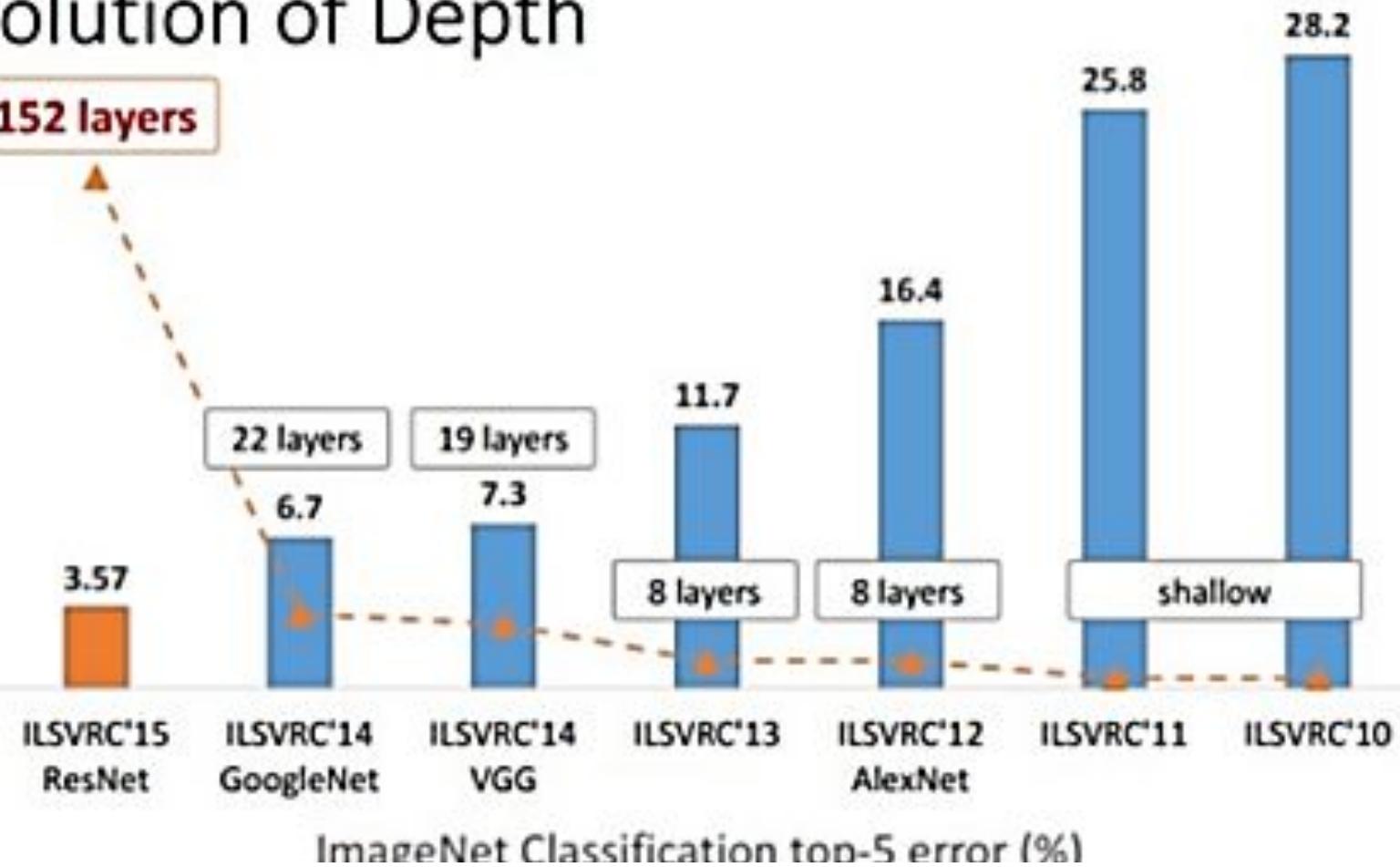
IMAGENET

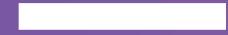
- 1,000 object classes (categories).
- Images:
 - 1.2 M train
 - 100k test.

<http://image-net.org>



Revolution of Depth





Pascal VOC



Visual Object Classes Challenge 2009 (VOC2009)



[click on an image to see the annotation]

PASCAL VOC

Bicycle



Bus



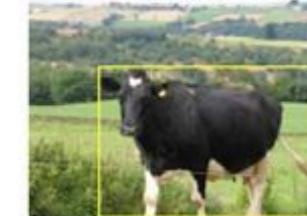
Car



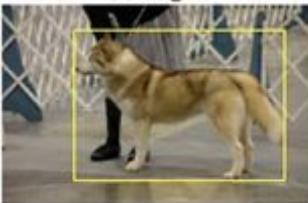
Cat



Cow



Dog



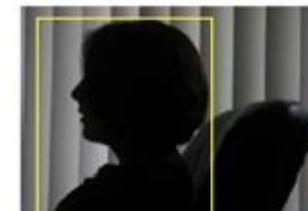
Horse



Motorbike



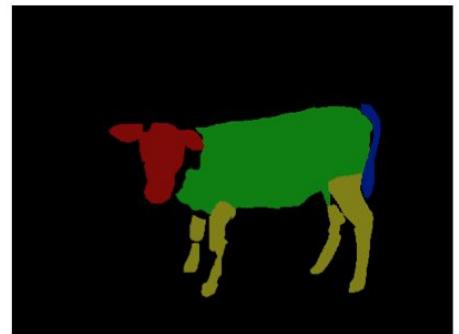
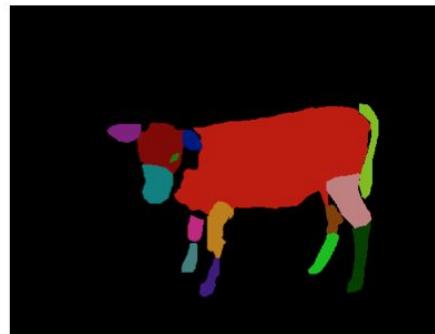
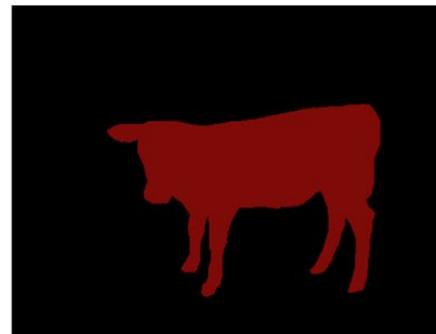
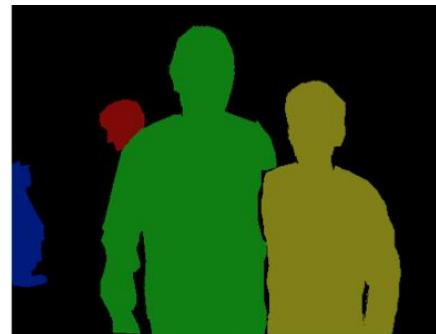
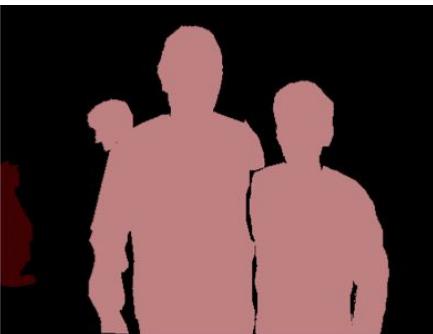
Person



Sheep



PASCAL VOC



Image

Class map

Instance map

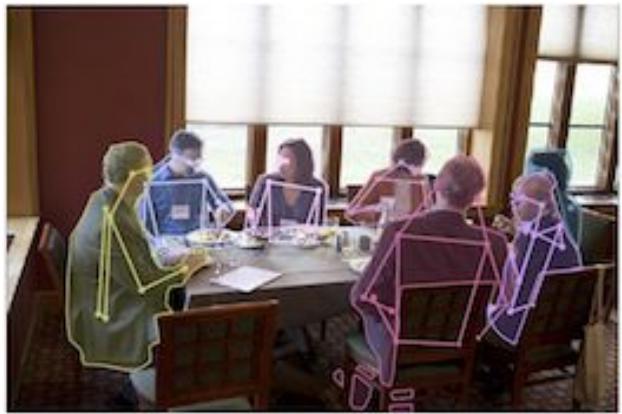
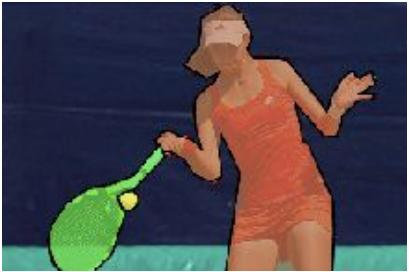
Part map

Part map (high level)



COCO COMMON OBJECT IN CONTEXT

COCO Dataset

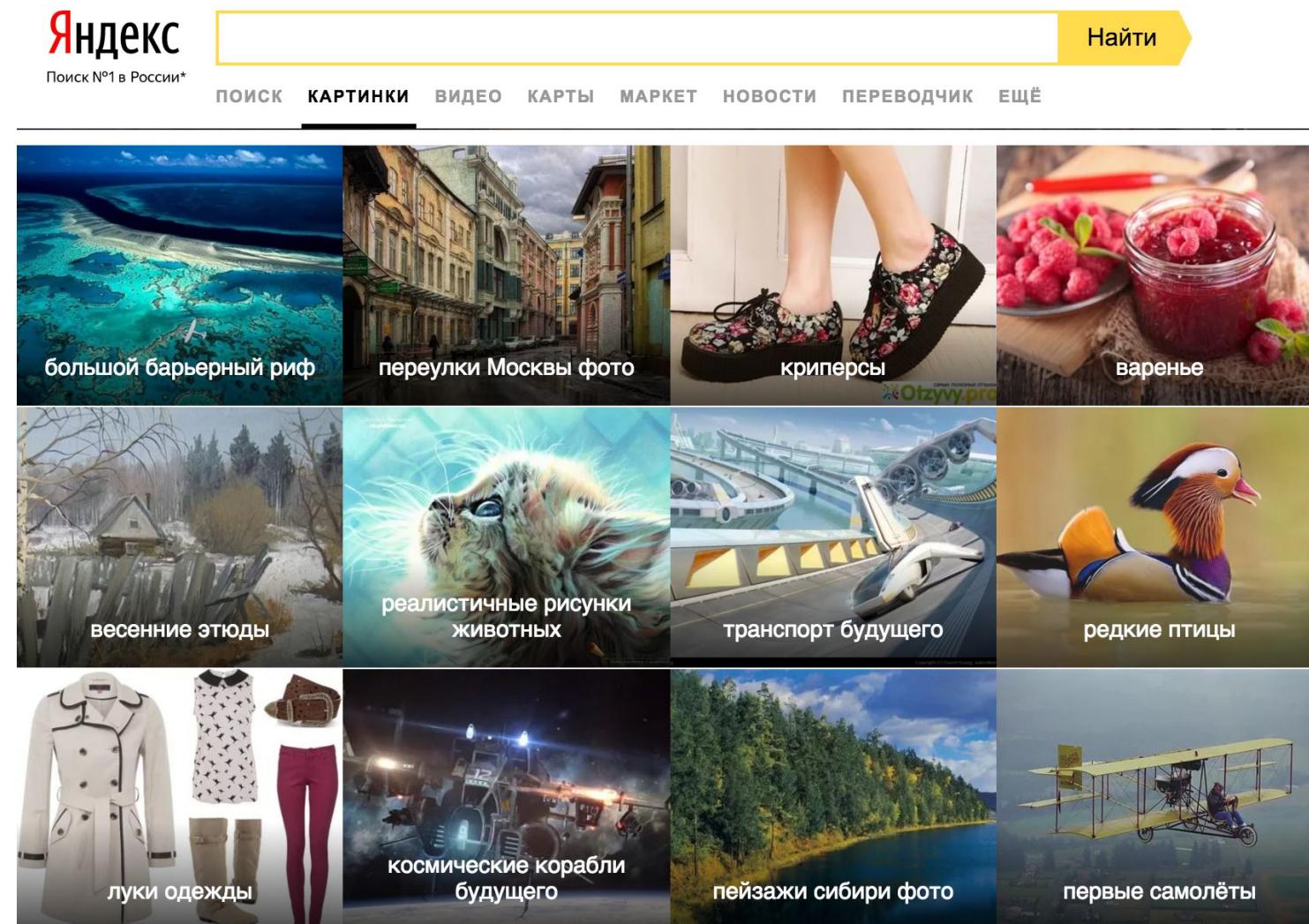


Списки открытых датасетов

- [Are we there yet?](#)
- [Computer Vision Dataset on the web](#)
- [Yet Another Computer Vision Index To Datasets](#)
- [Computer Vision Online Datasets](#)
- [CVOnline Dataset](#)
- [CV datasets](#)
- [Visionbib](#)

COCO COMMON OBJECT IN CONTEXT

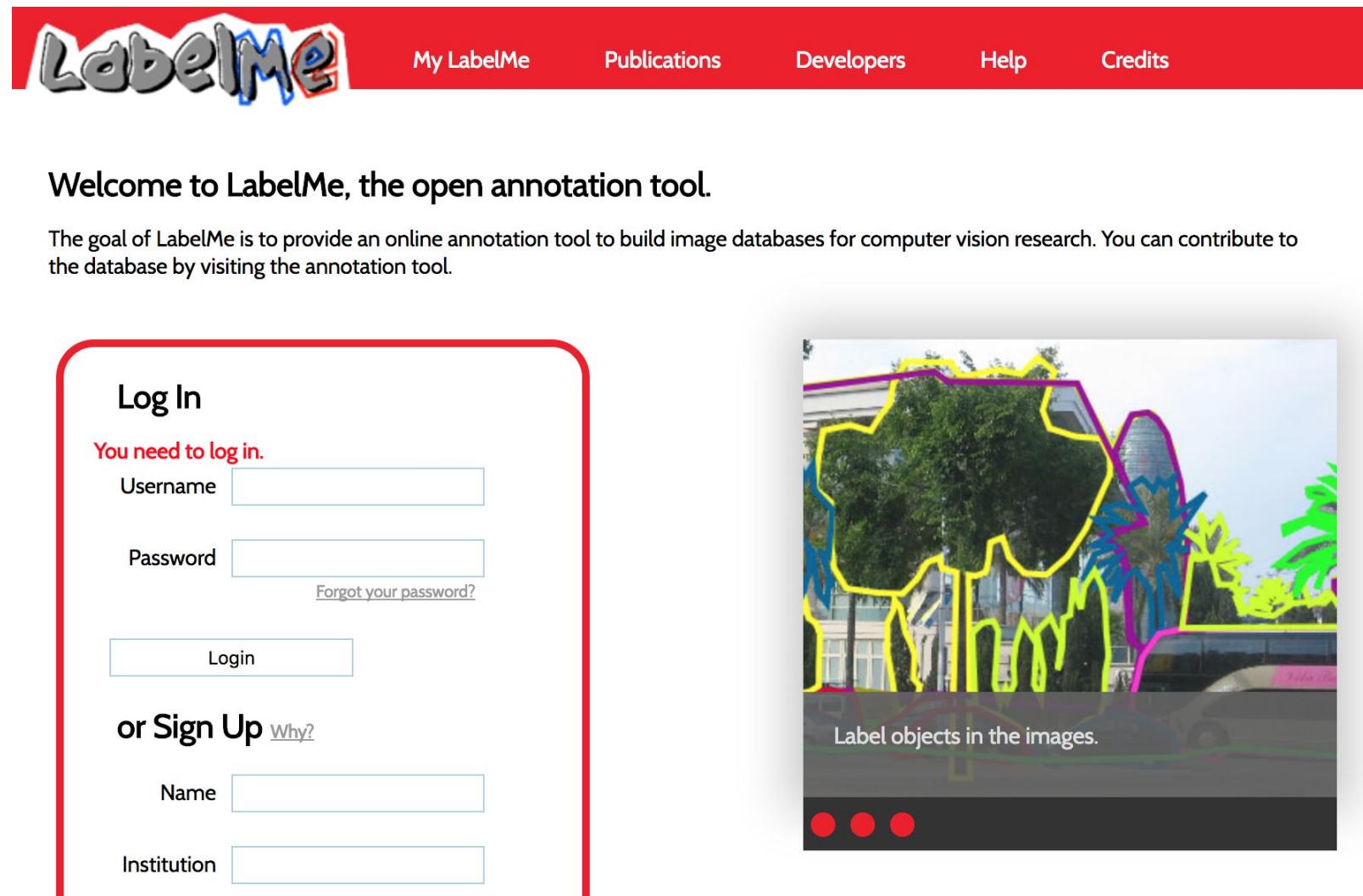
Данные из поисковых систем



COCO COMMON OBJECT IN CONTEXT

Инструмент
разметки —
LabelMe

<http://labelme2.csail.mit.edu>

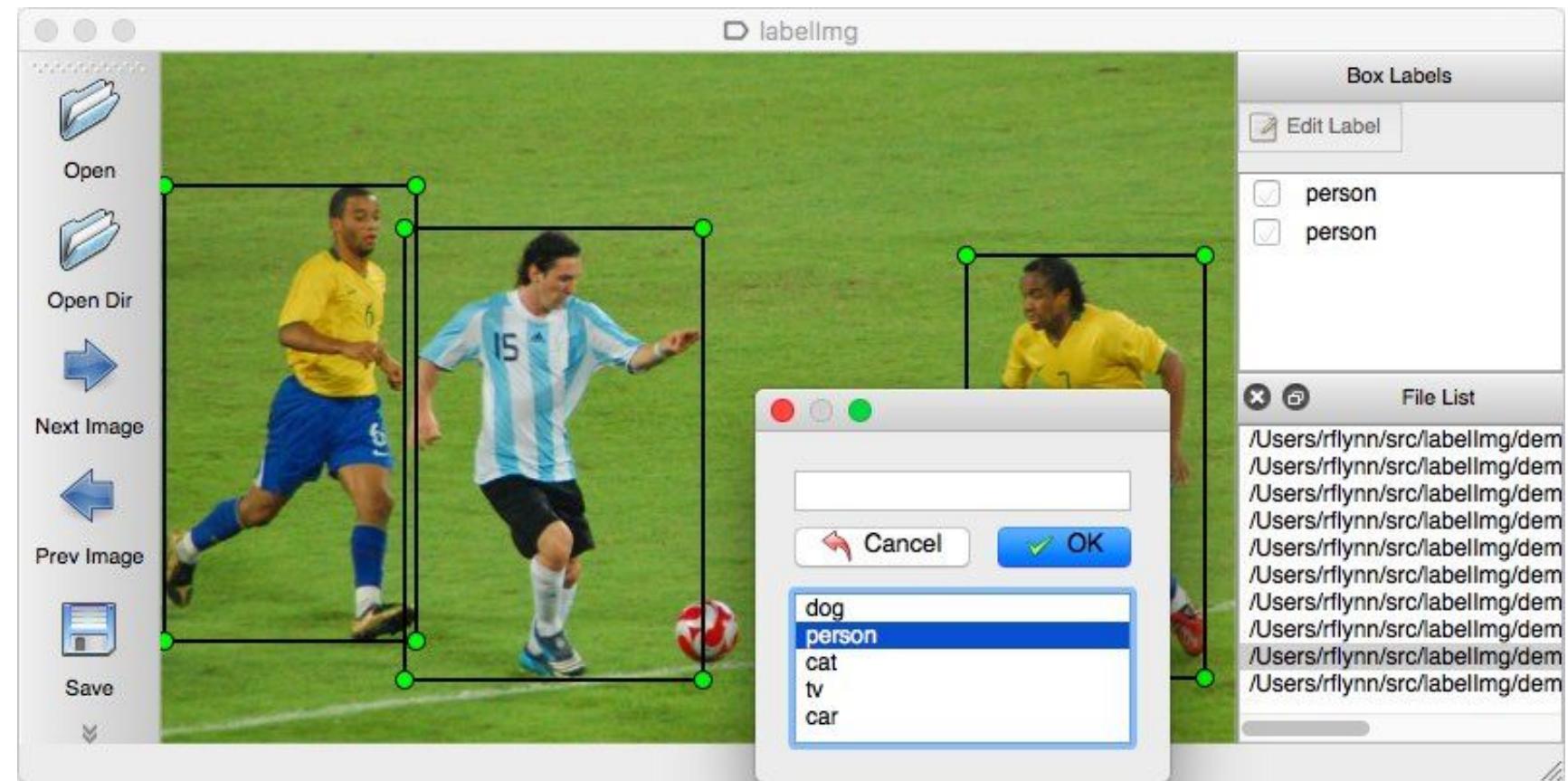


The screenshot shows the LabelMe website interface. At the top, there is a red navigation bar with the LabelMe logo on the left and links for "My LabelMe", "Publications", "Developers", "Help", and "Credits". Below the navigation bar, the main content area has a white background. On the left, a red-bordered box contains the "Log In" form. It includes fields for "Username" and "Password", a "Forgot your password?" link, and a "Login" button. Below the login form is a "or Sign Up" section with fields for "Name" and "Institution". To the right of the login form is a sample image of a city street scene. The image features several objects outlined with yellow and green polygons, representing annotated regions. Below the image, the text "Label objects in the images." is displayed.

COCO COMMON OBJECT IN CONTEXT

Инструмент
разметки —
LabelImg

[https://github.com/
tzutalin/labelImg](https://github.com/tzutalin/labelImg)

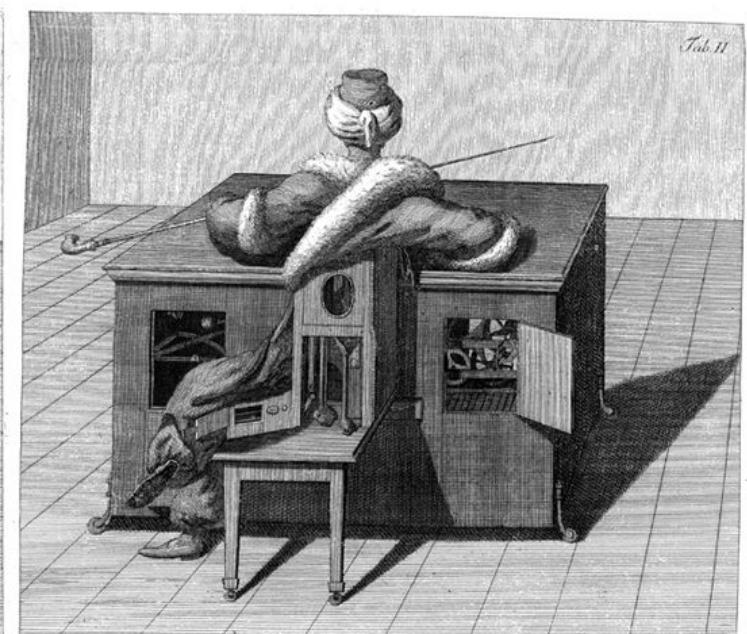
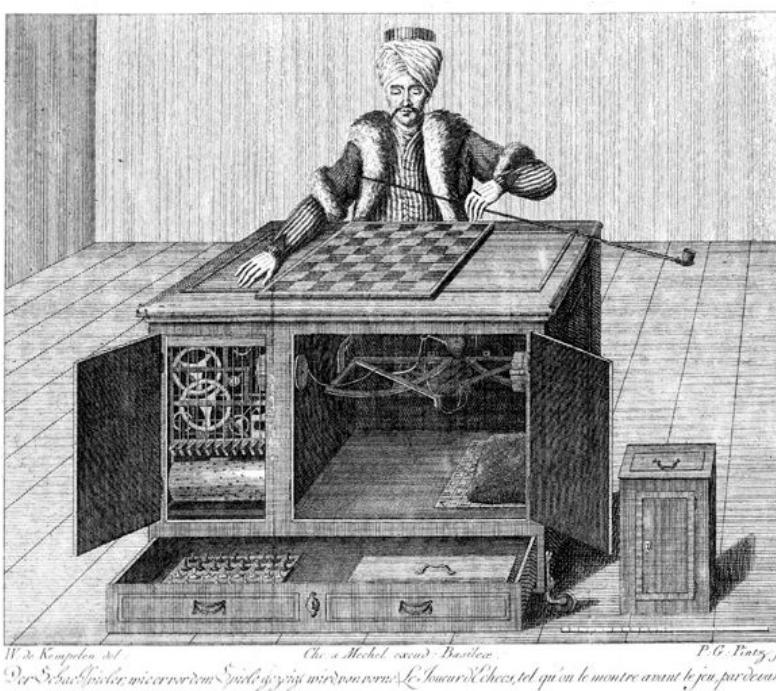
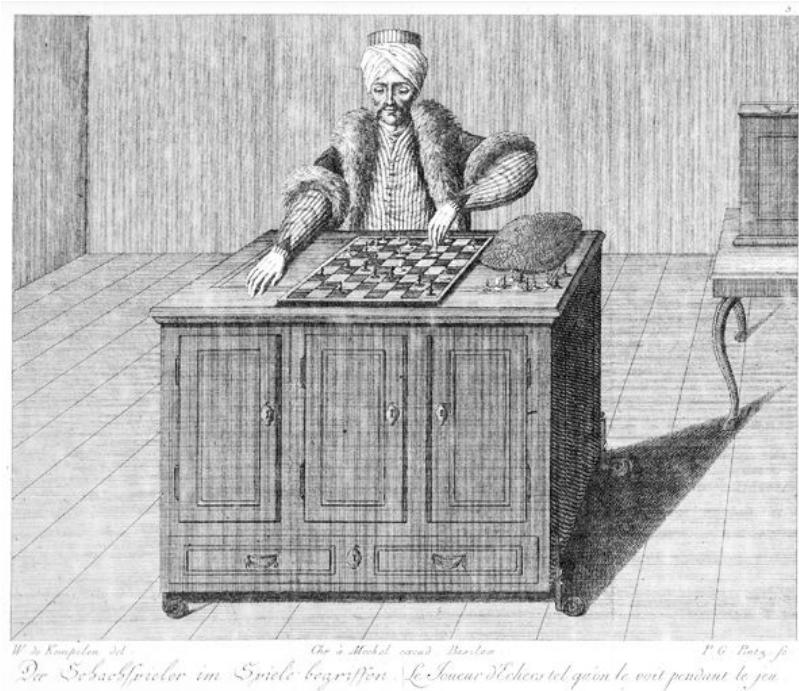


Сервис
разметки —
**Mechanical
Turk**

<https://www.mturk.com>



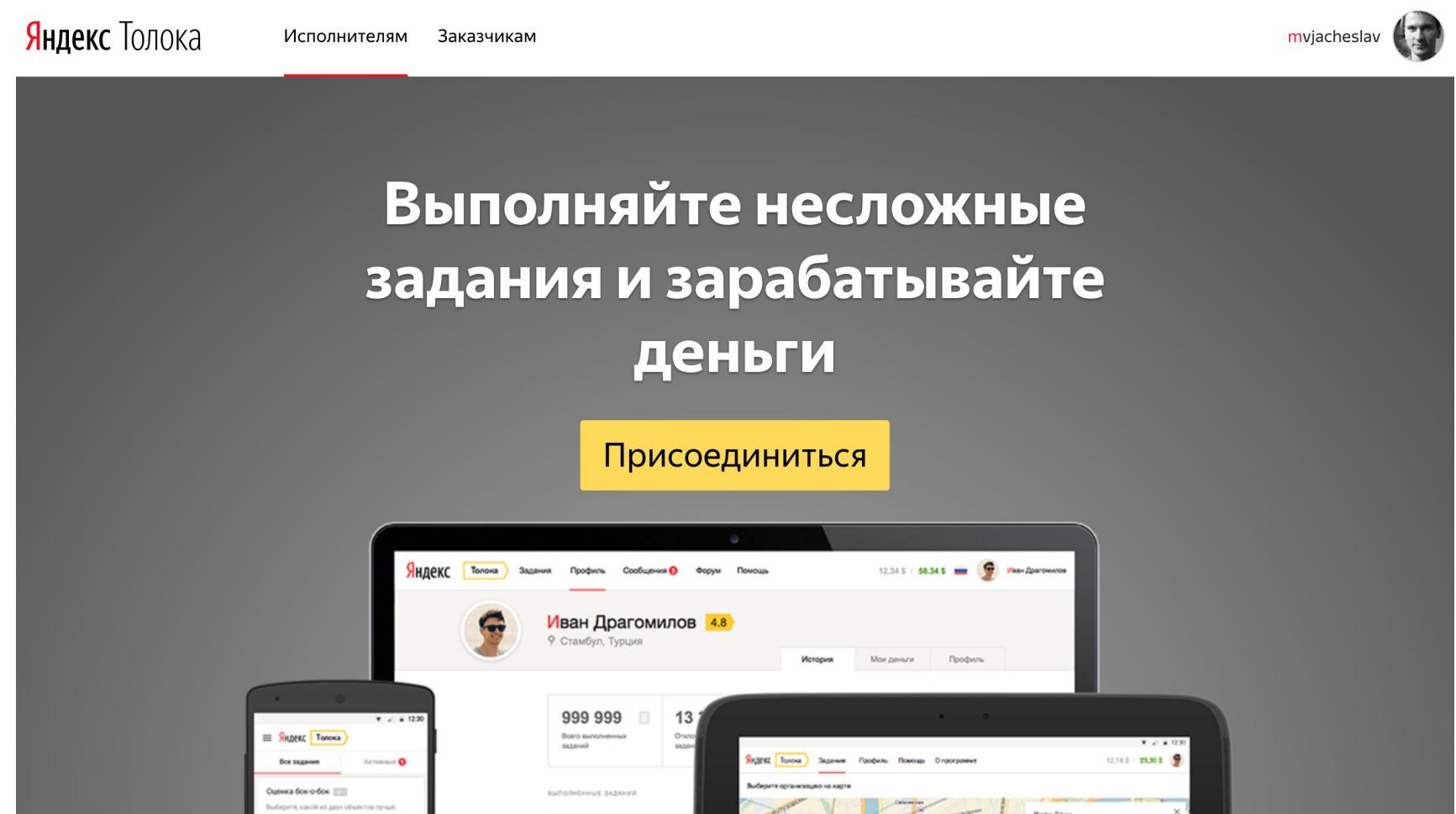
Сервис разметки — **Mechanical Turk**



COCO COMMON OBJECT IN CONTEXT

Сервис
разметки —
**Yandex.
Toloka**

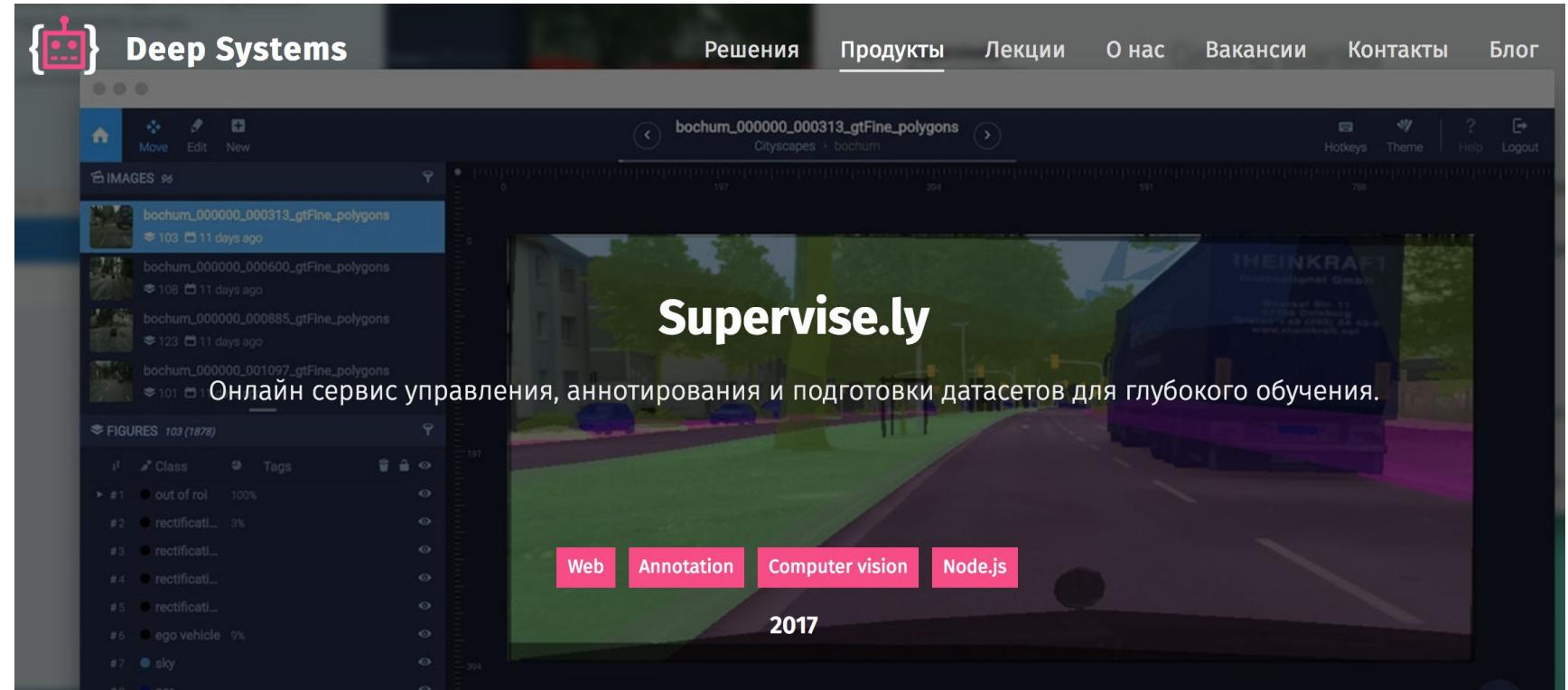
<https://toloka.yandex.ru/>



COCO COMMON OBJECT IN CONTEXT

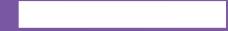
Сервис
разметки —
Supervise.ly

<https://supervise.ly>



Перейти на
[Supervise.ly](https://supervise.ly)

Читать на
[Medium](#)

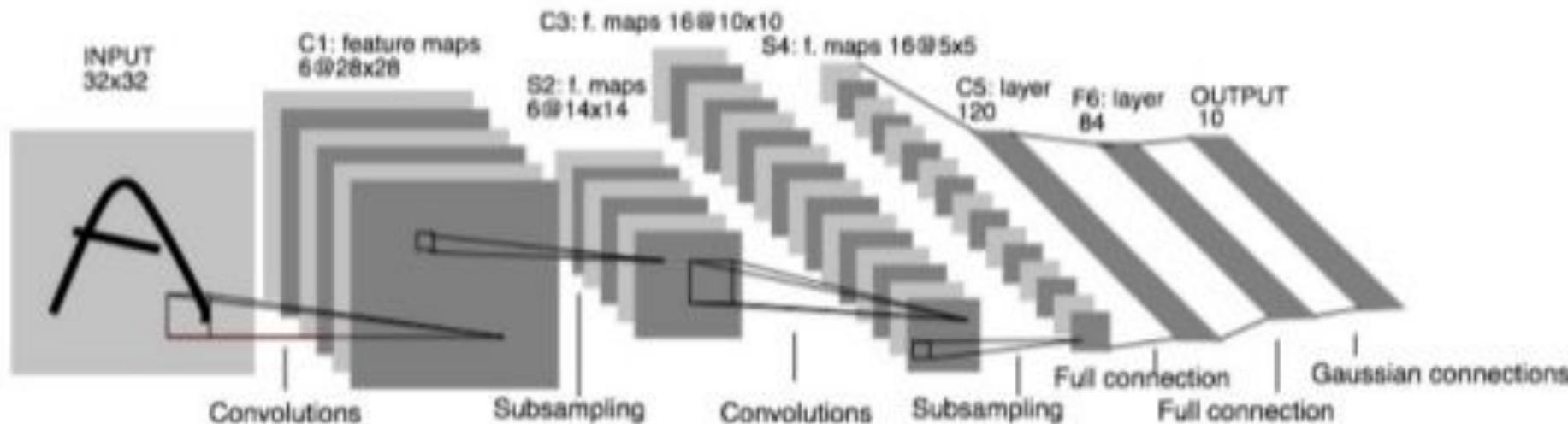


АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ

LeNet

1989

Convolutional Neural Nets (CNNs): 1989



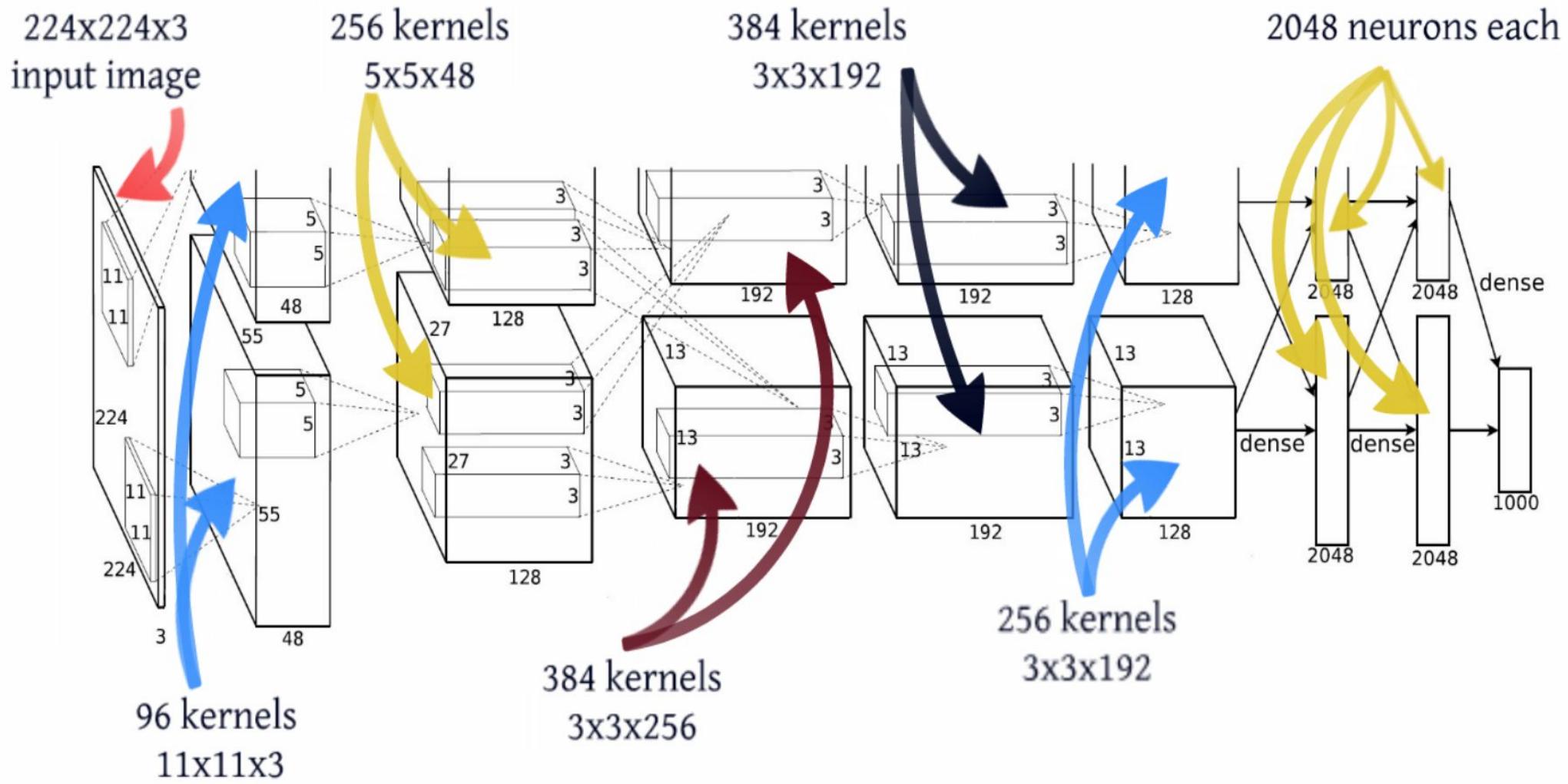
LeNet: a layered model composed of convolution and subsampling operations followed by a holistic representation and ultimately a classifier for handwritten digits. [LeNet]

AlexNet

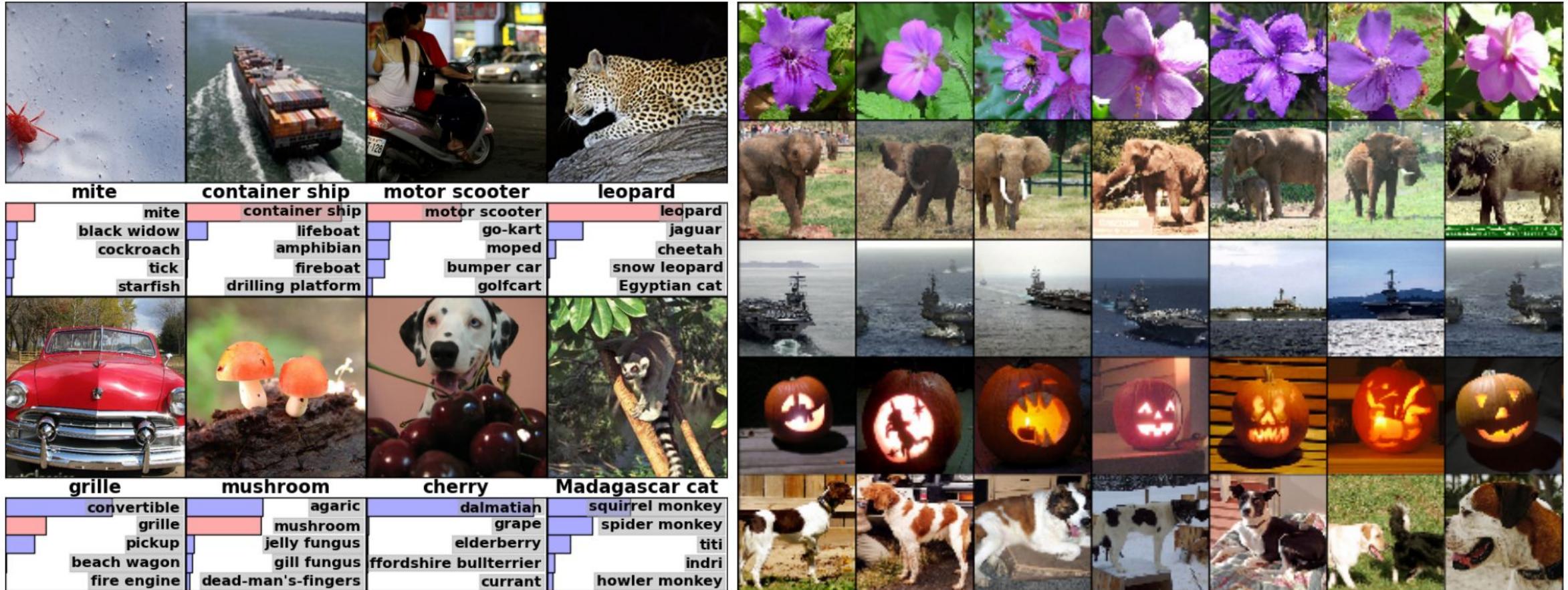
2012

АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ

ImageNet Classification with Deep Convolutional Neural Networks



АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ



AlexNet, 2012

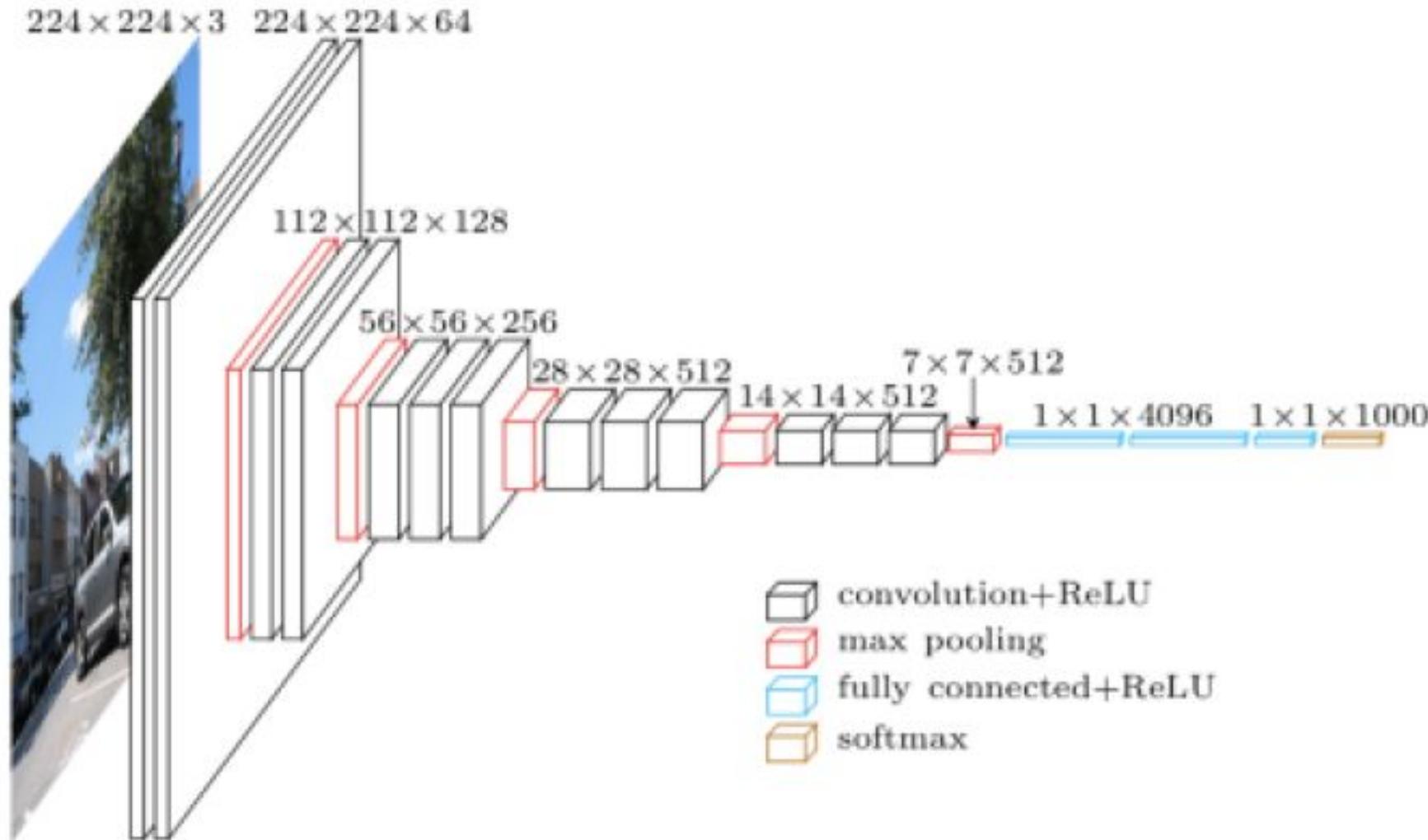
- Параллельная архитектура
- Использование *relu* в качестве функций активации
- Для регуляризации использование *dropout* перед полносвязными слоями
- Число параметров ~60М

АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ

VGG

2014

АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ



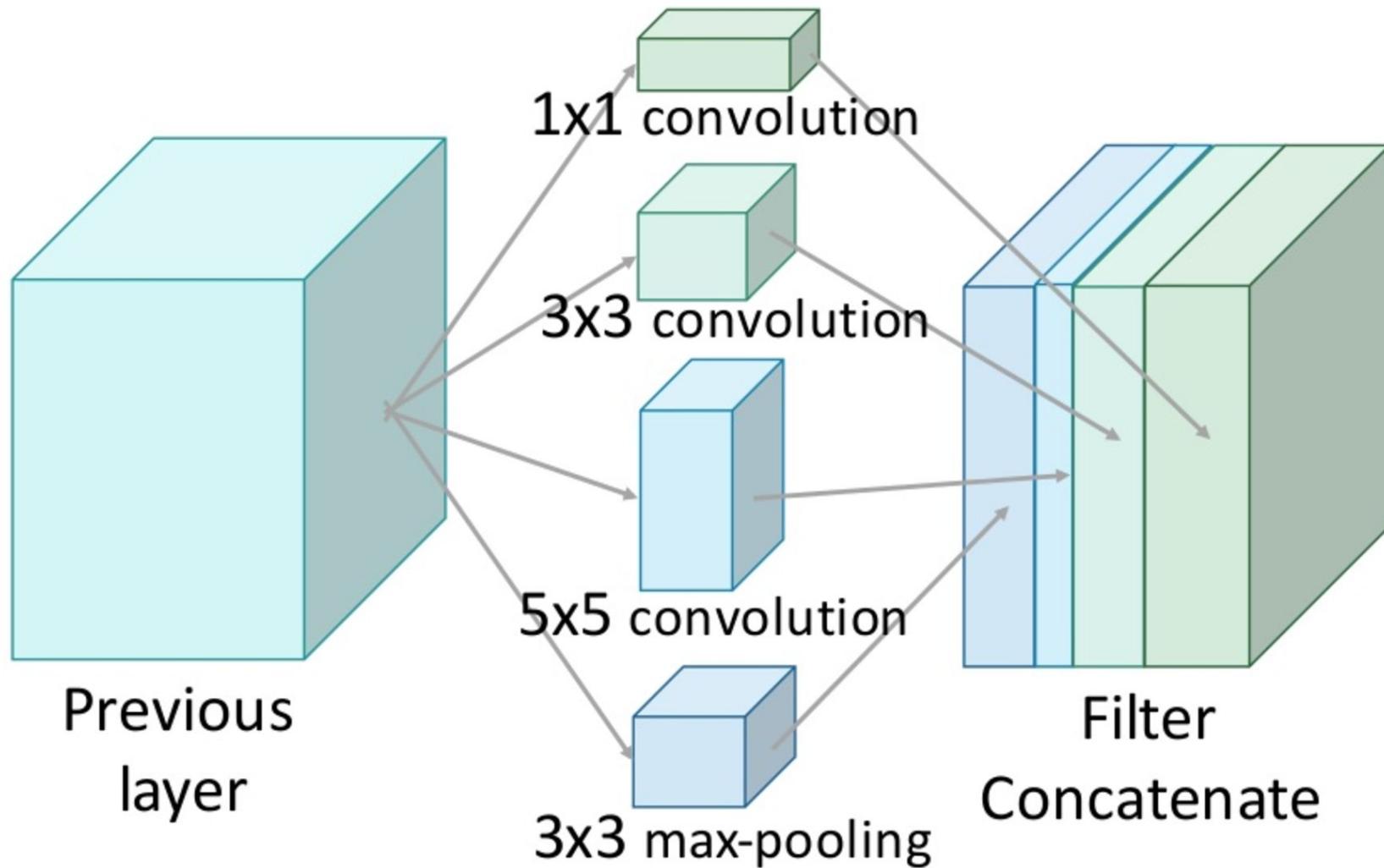
VGG, 2014

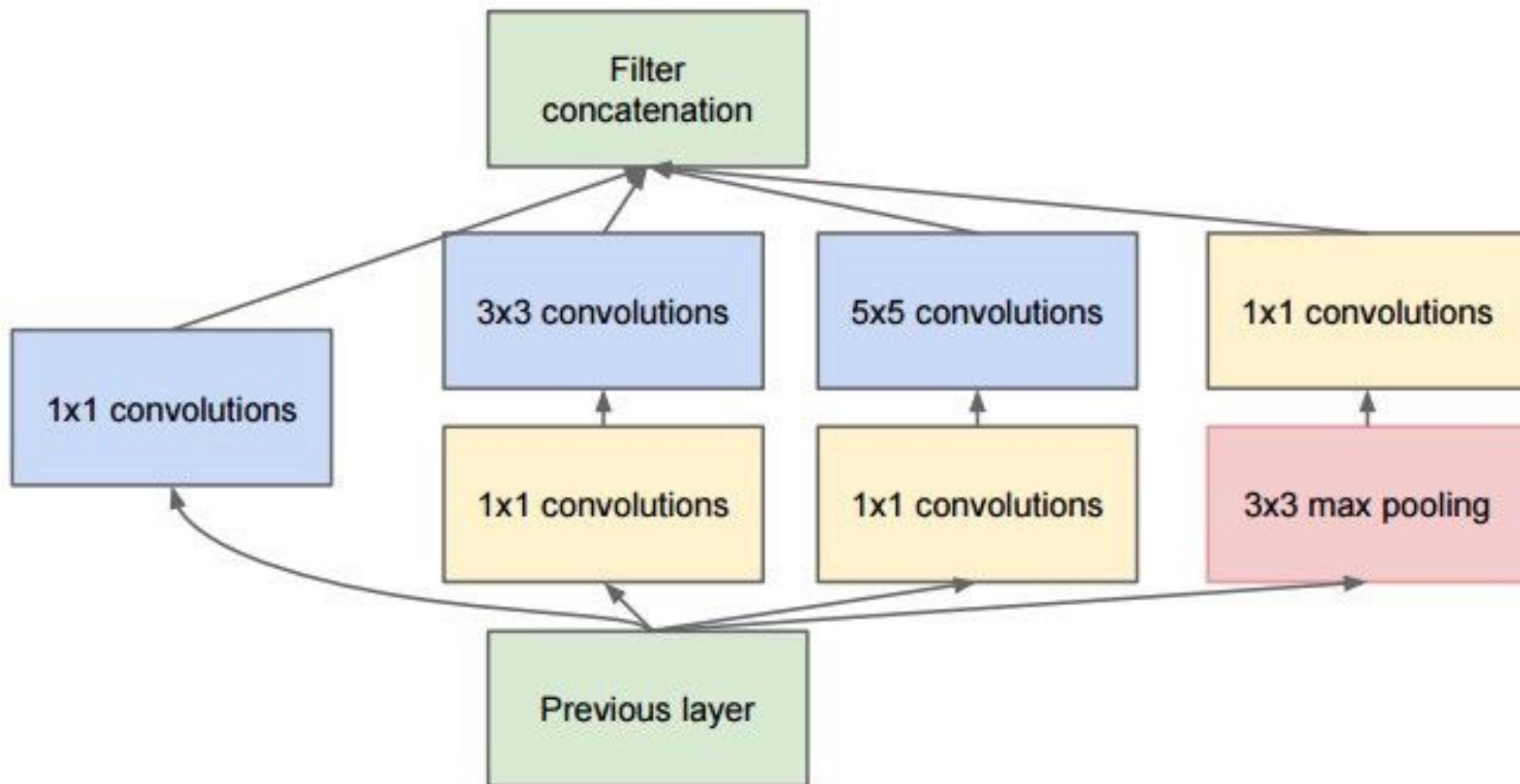
- Последовательно применяются свертки фильтров с небольшим размером ядра
- Большой объем данных на выходе каждого слоя требует большого количества памяти
- Число параметров ~130М

GoogleNet *Inception*

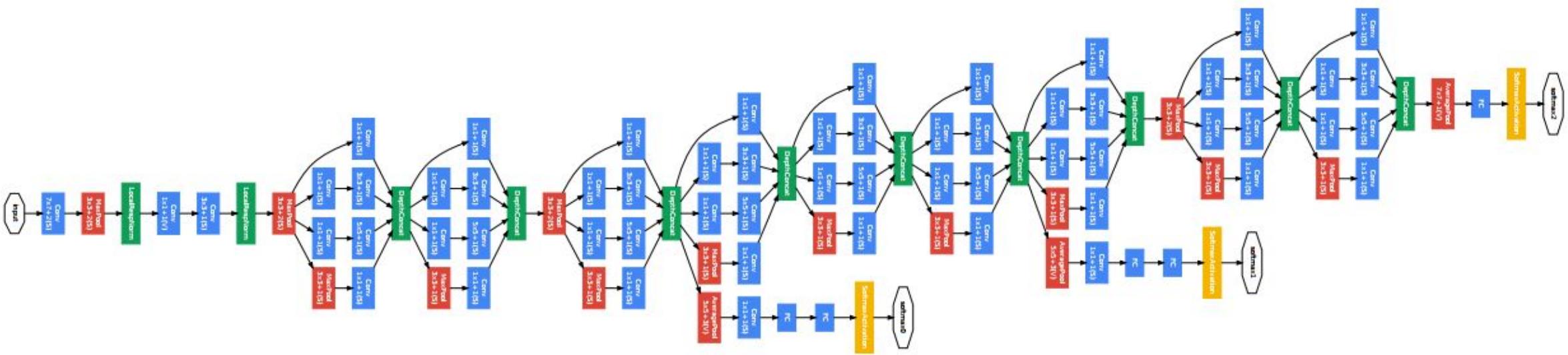
2014

АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ





АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ

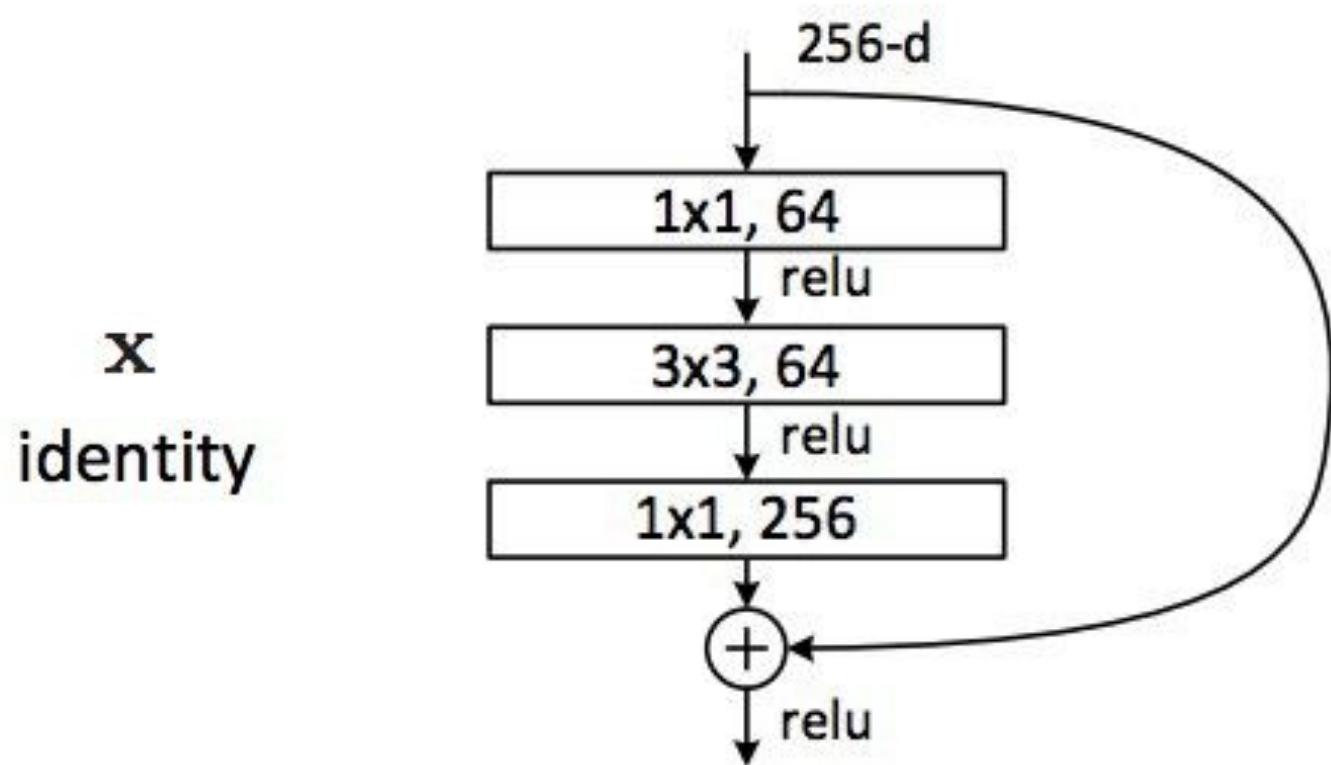
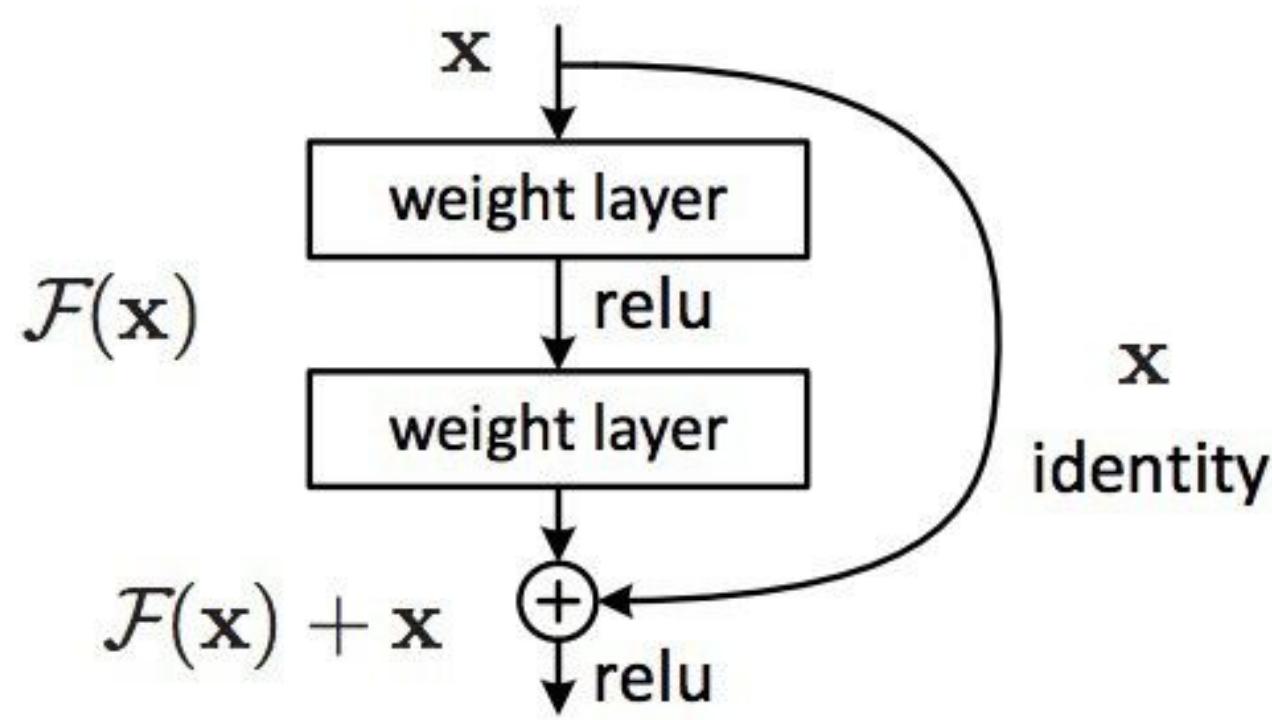


GoogleNet *Inception*, 2014

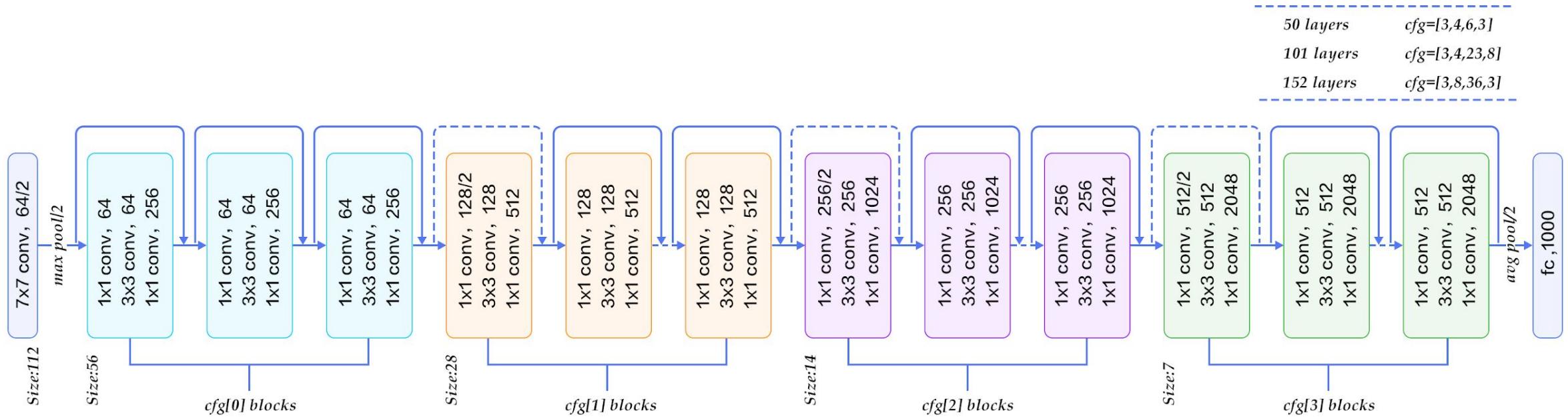
- Применяются фильтры разного размера к одним и тем же данным
- Уменьшение размерности выхода *глубины* за счет свертки 1×1
- В результате уменьшения глубины получаем ускорение сверток 3×3 и 5×5
- Число параметров ~7М

ResNet

2015

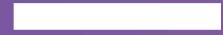


АРХИТЕКТУРЫ СВЕРТОЧНЫХ СЕТЕЙ



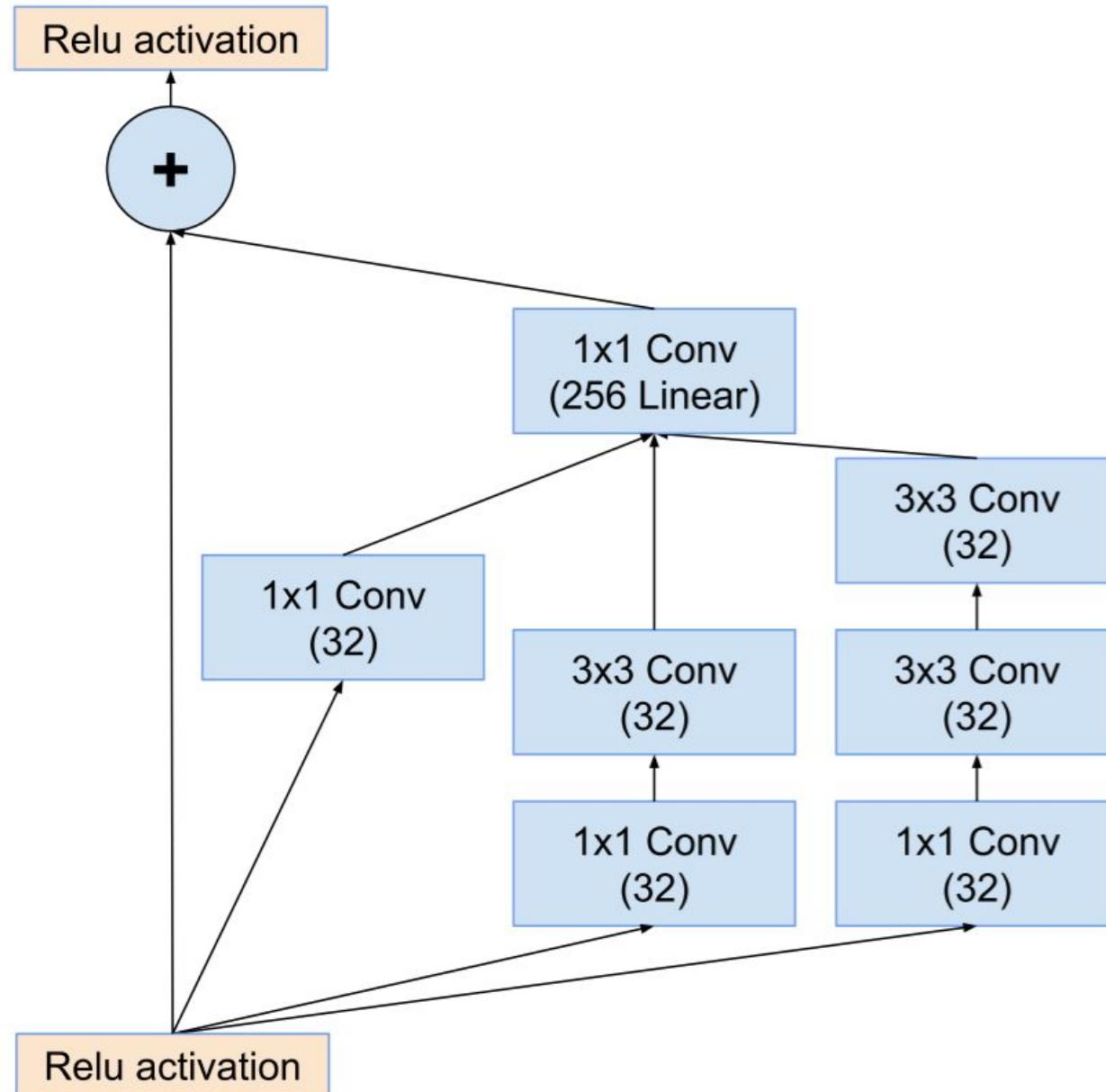
ResNet, 2015

- Добавлена параллельная ветка с исходными данными
- Первая архитектура, у которой более 100 слоев
- Число параметров resnet 50 ~25M

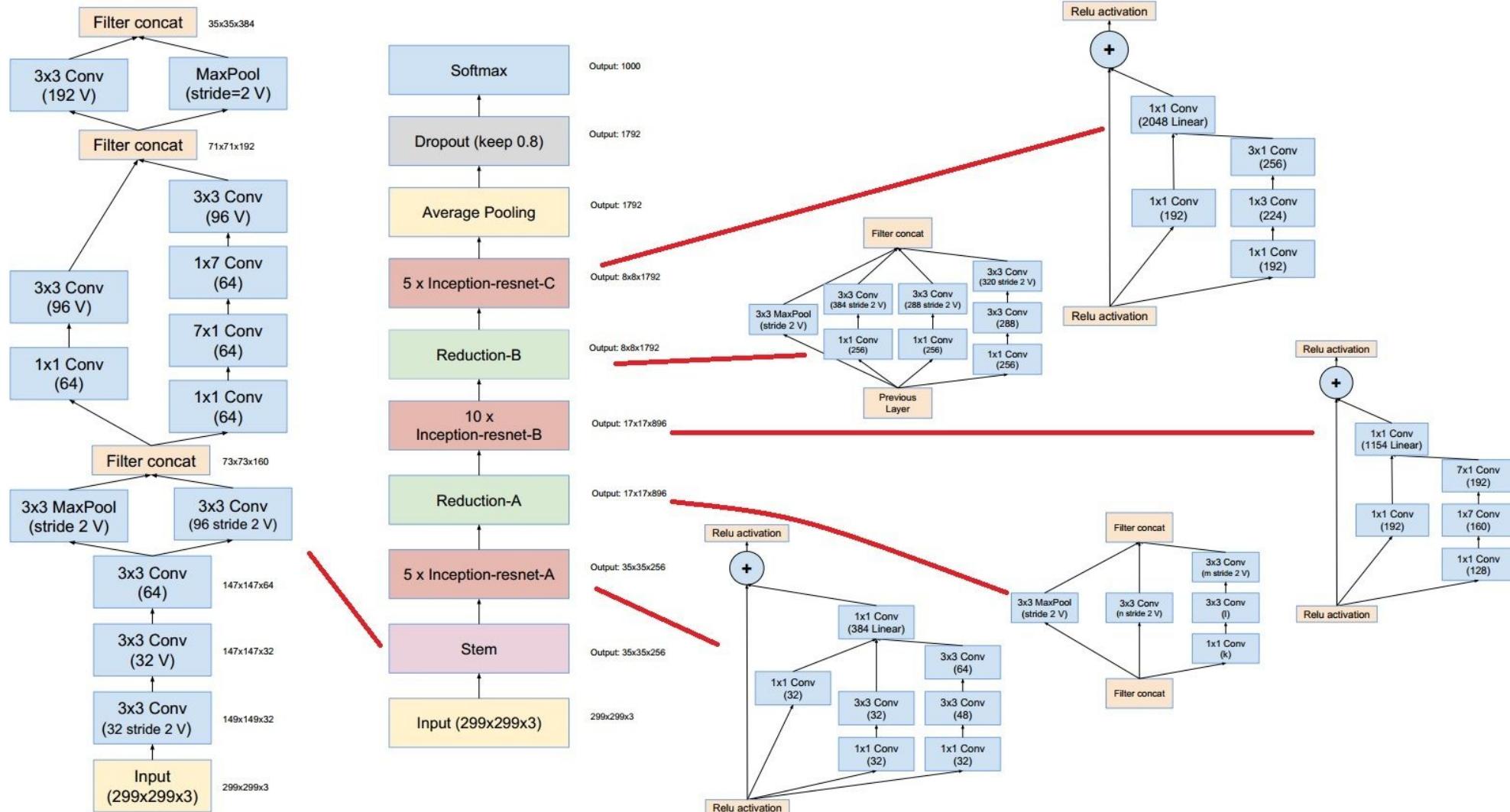


INCEPTION RESNET

INCEPTION RESNET



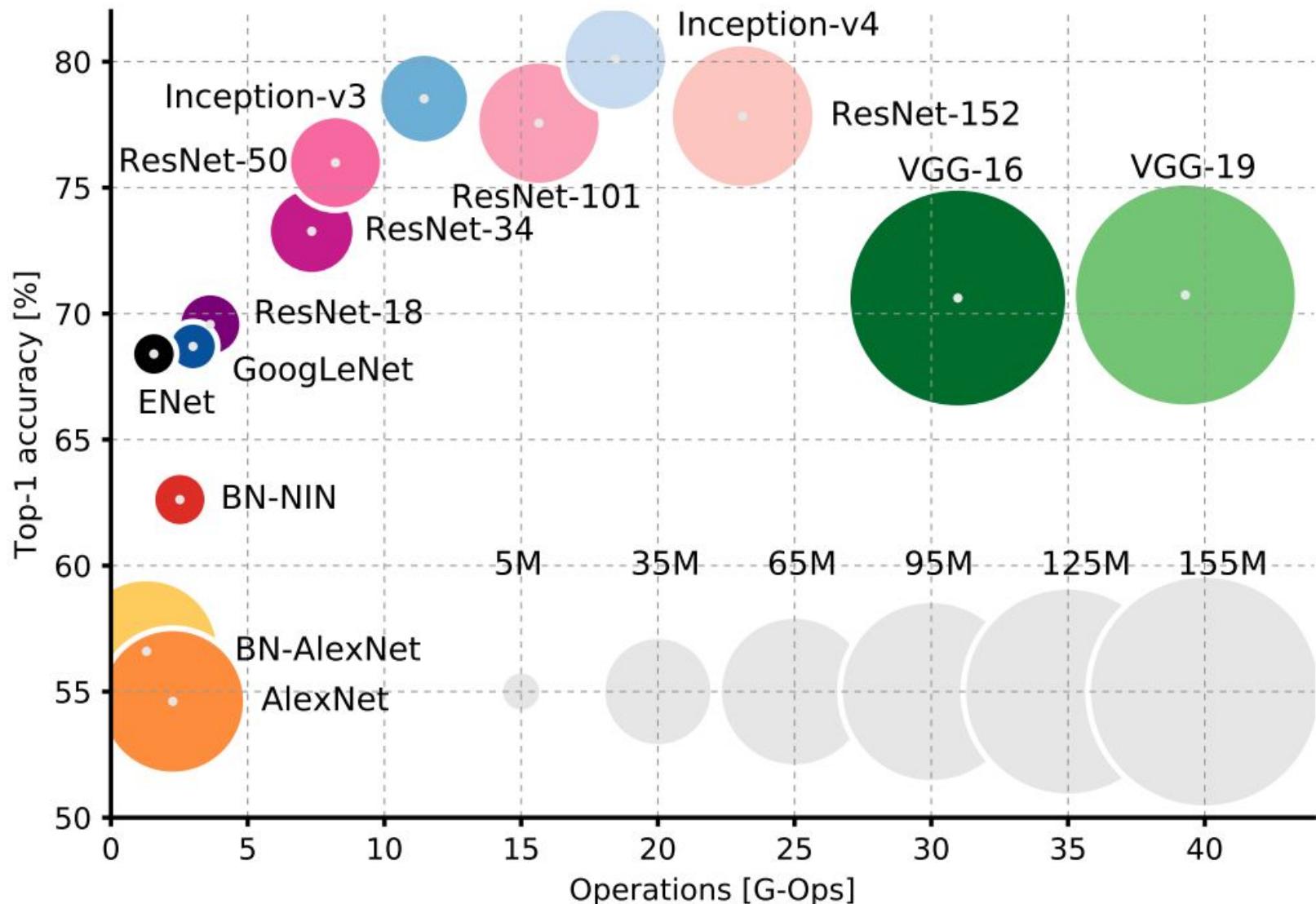
INCEPTION RESNET



INCEPTION RESNET

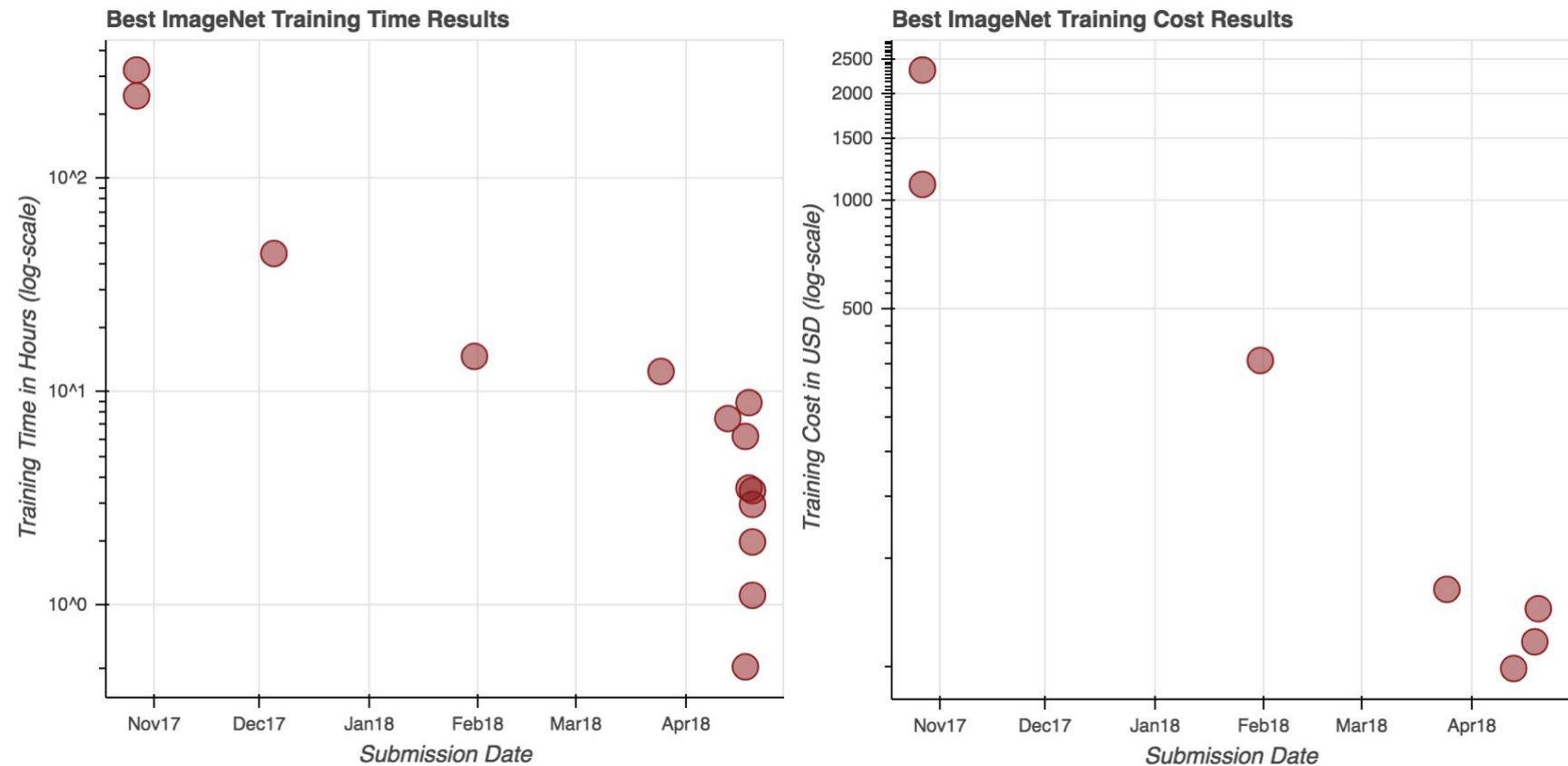
Сравнение:
качество
vs
скорость

AN ANALYSIS OF DEEP
NEURAL NETWORK
MODELS FOR
PRACTICAL
APPLICATIONS



INCEPTION RESNET

Сравнение:
время
обучения
vs
стоимость



<https://dawn.cs.stanford.edu/2018/04/30/dawnbench-v1-results/>

Предобученные модели

- [CAFFE Model-Zoo](#)
- [Keras Applications](#)
- [Tensorflow/Research/Slim](#)
- [Deep Learning Model Convertors](#)

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. TRANSFER LEARNING



Transfer Learning

На практике

ОЧЕНЬ РЕДКО ОБУЧАЮТ СВЕРТОЧНЫЕ СЕТИ С НУЛЯ

Это связано,
как правило:

- с ограниченным объемом доступных данных,
- с ограниченными вычислительными ресурсами.



Как правило,
**существующие
предобученные сети
адаптируют
под конкретную
задачу**

Transfer Learning

Как правило выходной слой предобученной сети

ТРЕБУЕТ ИЗМЕНЕНИЯ ДЛЯ КАЖДОЙ ЗАДАЧИ

разное число классов



Копируем
архитектуру и веса
предобученной сети
и заменяем
последний слой



Дообучаем
новый выходной слой
на данных задачи

Transfer Learning

	Задача похожа	Задача сильно отличается
Данных мало	Обучаем линейный классификатор, на признаках последнего внутреннего слоя	<i>У нас проблемы :)</i> Обучаем классификатор на признаках с разных слоев
Данных много	Дообучаем несколько последних внутренних слоев	Фиксируем первые слои, остальные слои дообучаем

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. БАЛАНС КЛАССОВ

Баланс классов

Часто на практике

**ВЫБОРКА
НЕ СБАЛАНСИРОВАНА**



Это приводит к тому, что

МОДЕЛЬ

переобучается на классы
с большим числом примеров

и **дает смещённое предсказание**

Баланс классов

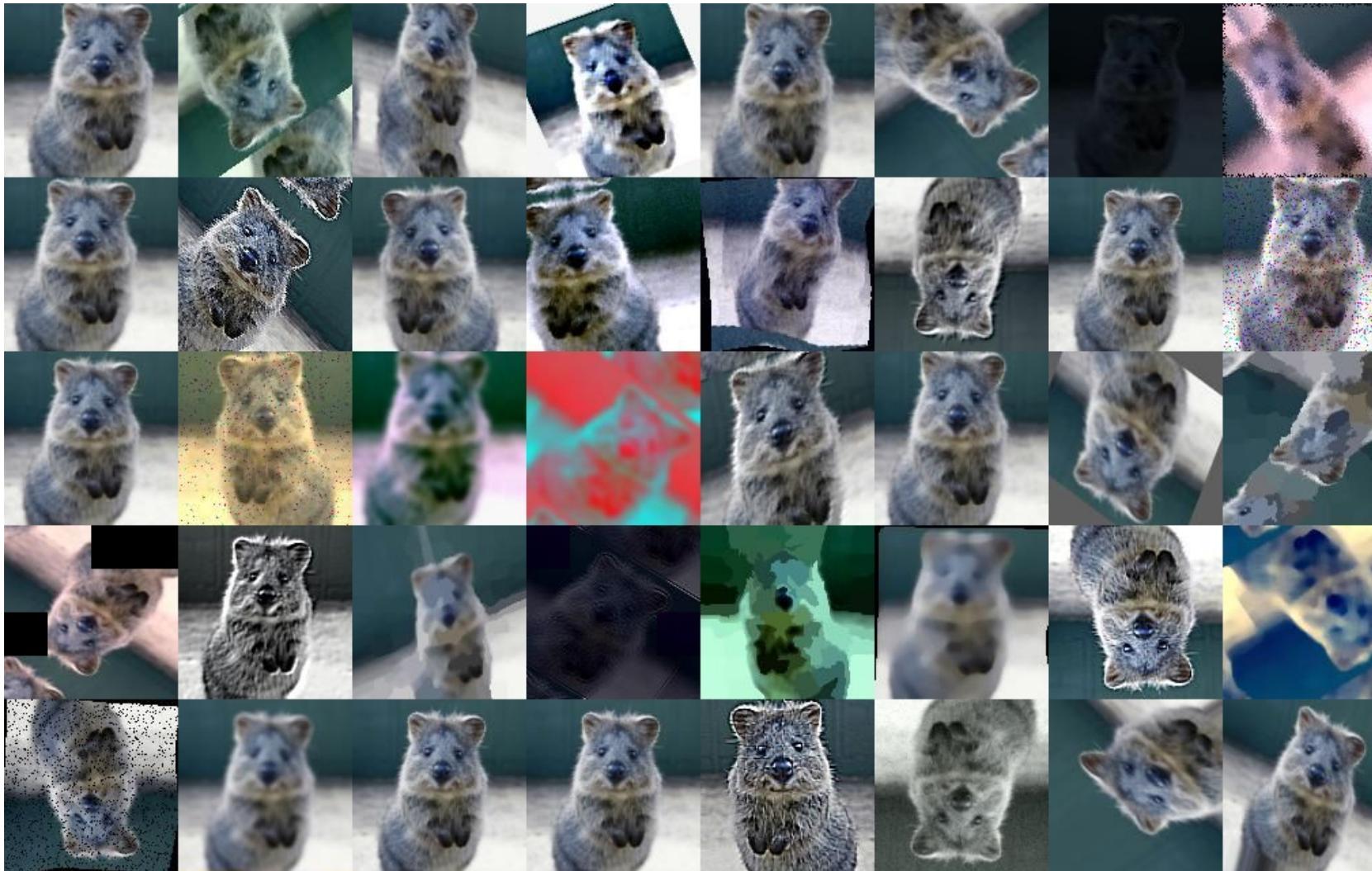
Выравниваем градиенты, взвешивая ошибку обратно пропорционально числу классов

Выравниваем баланс классов на уровне батча

Добавляем число примеров **редких классов** за счет ументации

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. АУГМЕНТАЦИЯ ДАННЫХ

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. АУГМЕНТАЦИЯ ДАННЫХ



[https://github.com/
aleju/imgaug](https://github.com/aleju/imgaug)

Аугментация данных

- Зеркальное отражение по горизонтали
- Вырезаем случайную часть из изображения *crop* и масштабируем до исходного
- Аугментация освещенности
в пространстве HSV
- Аугментация цвета
случайный шум по каналам

Реализация

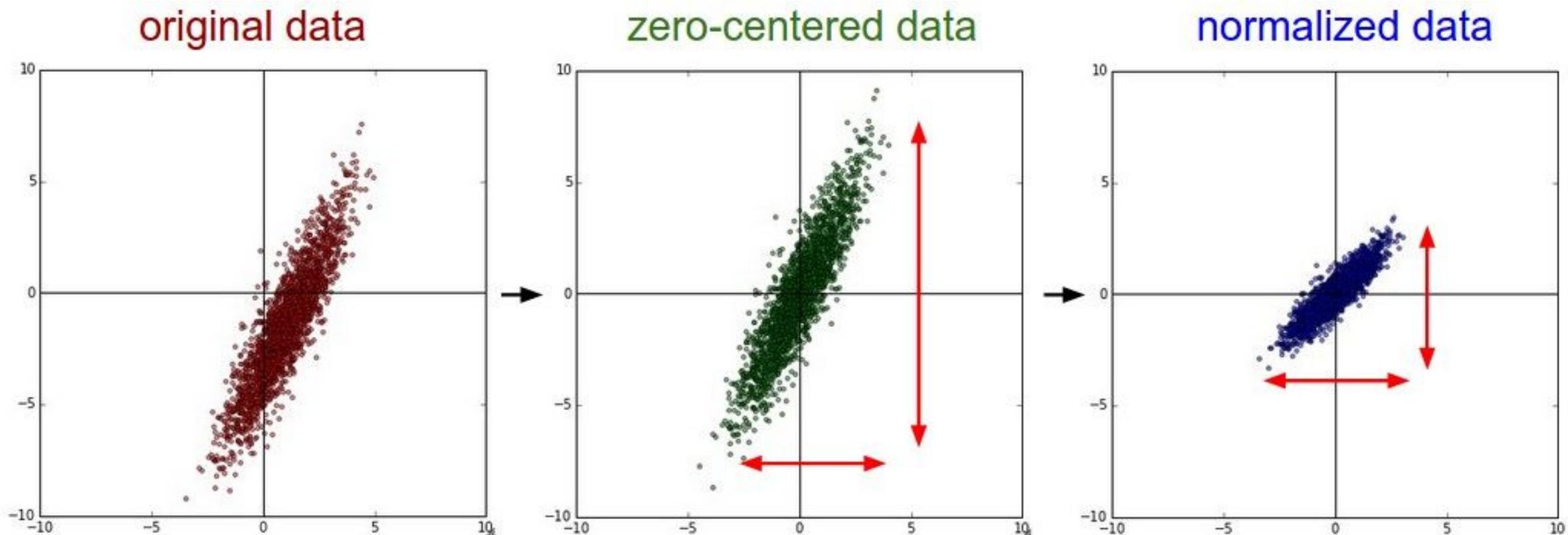
- На практике аугментированные изображения сильно увеличивают размер выборки
- Хранить на диске аугментированные копии нецелесообразно
- Процесс аугментации запускают на лету параллельно с обучением
- Аугментация выполняется на CPU
и существенно не влияет на скорость при обучении на GPU

Библиотеки аугментации изображений

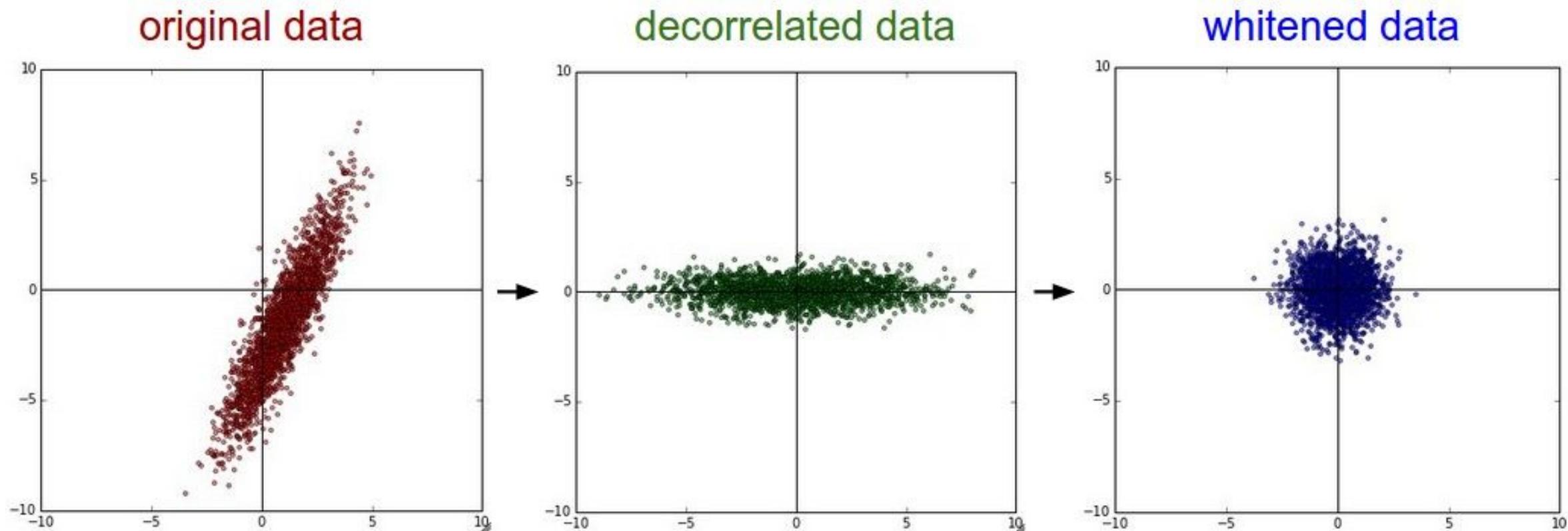
- [aleju/imgaug](#)
- [Augmentor](#)
- [keras.preprocessing.image.ImageDataGenerator](#)
- [tf.image](#)

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ПРЕДОБРАБОТКА ДАННЫХ

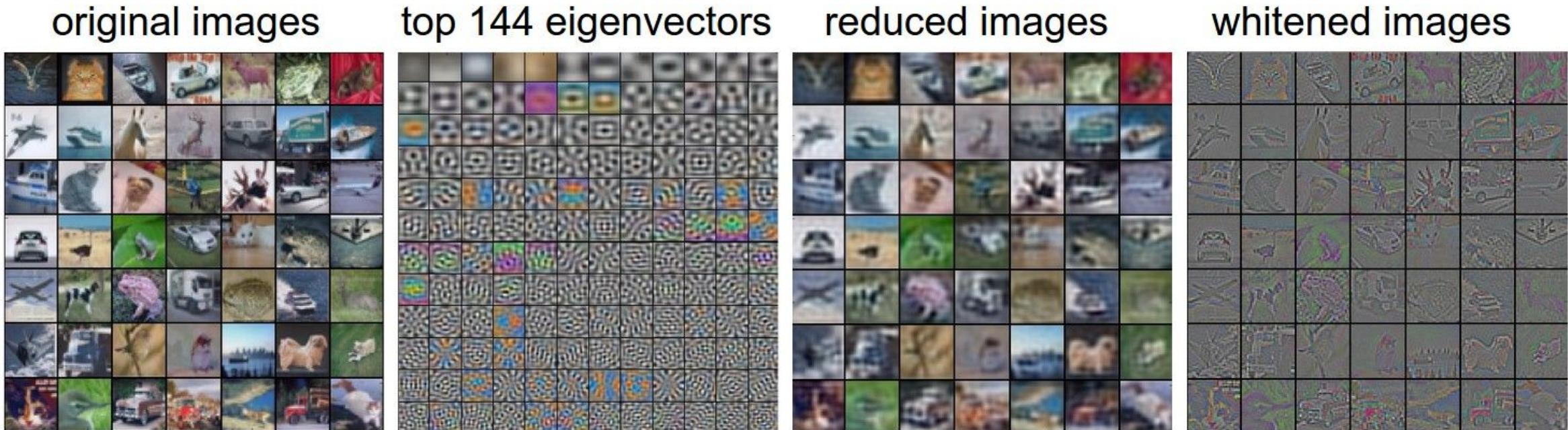
Центрирование и нормализация



Обеление PCA Whitening



Обеление PCA Whitening



ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ИНИЦИАЛИЗАЦИЯ

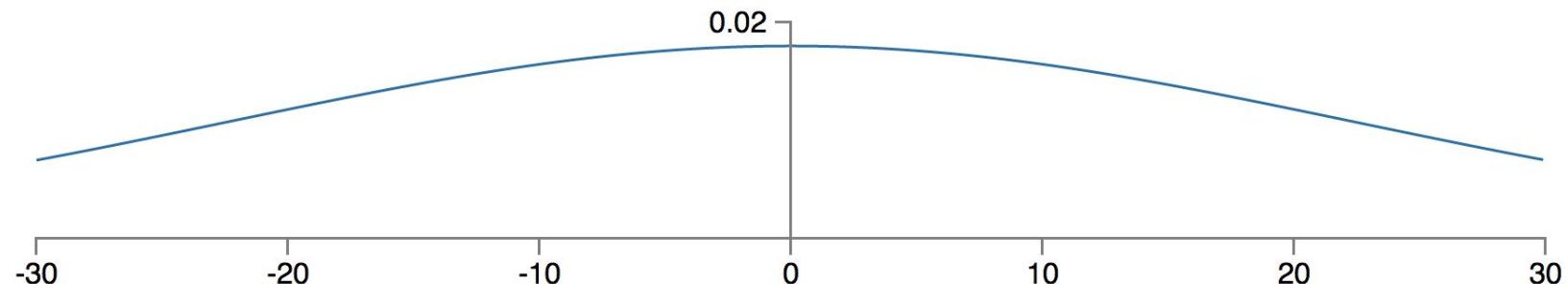
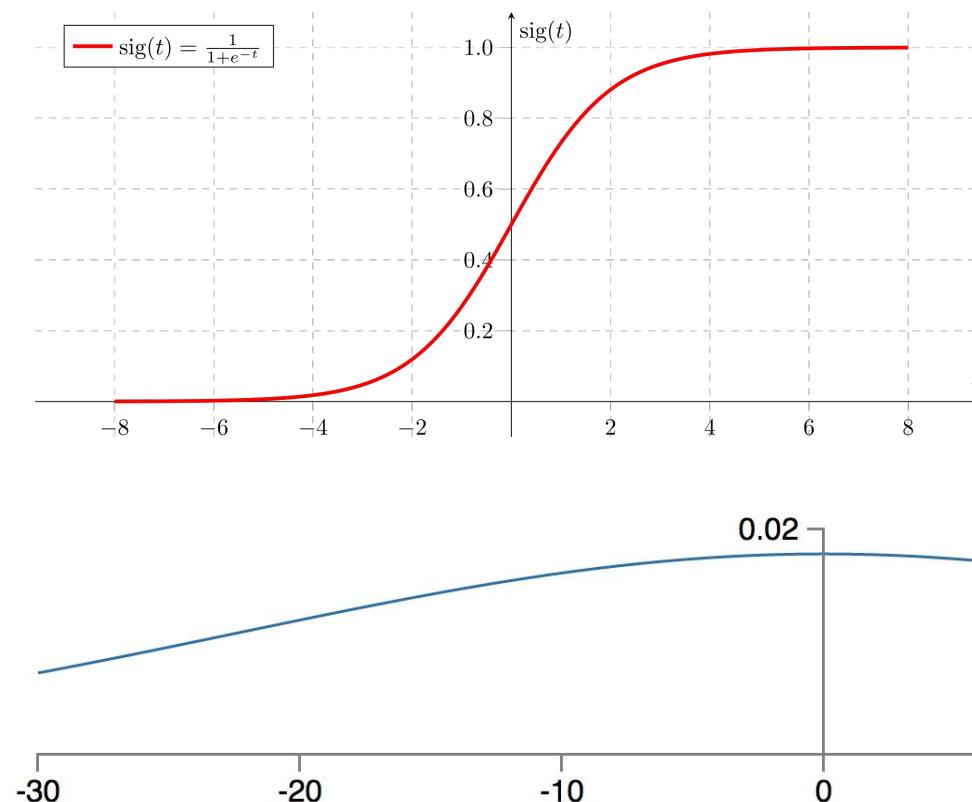
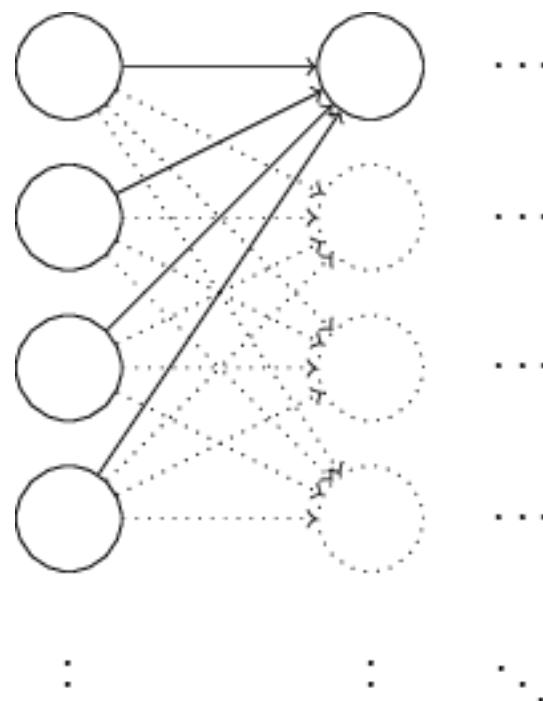
—

Инициализация весов

- Инициализация различными весами
*если нейроны возвращают одинаковые значения,
значит и градиенты для них будут одинаковыми*
- Инициализация случайными значениями с небольшой дисперсией
гаусс или нормальное — не существенно важно
- Дисперсия растет с увеличением числа входов —
нормируем дисперсию на число входов *lécun_normal*

<https://keras.io/initializers/>

Инициализация весов



$$z = \sum_j w_j x_j + b$$

$$\sigma_{X+Y}^2 = \sigma_X^2 + \sigma_Y^2$$

$$\sigma_{X-Y}^2 = \sigma_X^2 + \sigma_Y^2$$

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ГРАДИЕНТНЫЙ СПУСК

Оптимизация Градиентный спуск

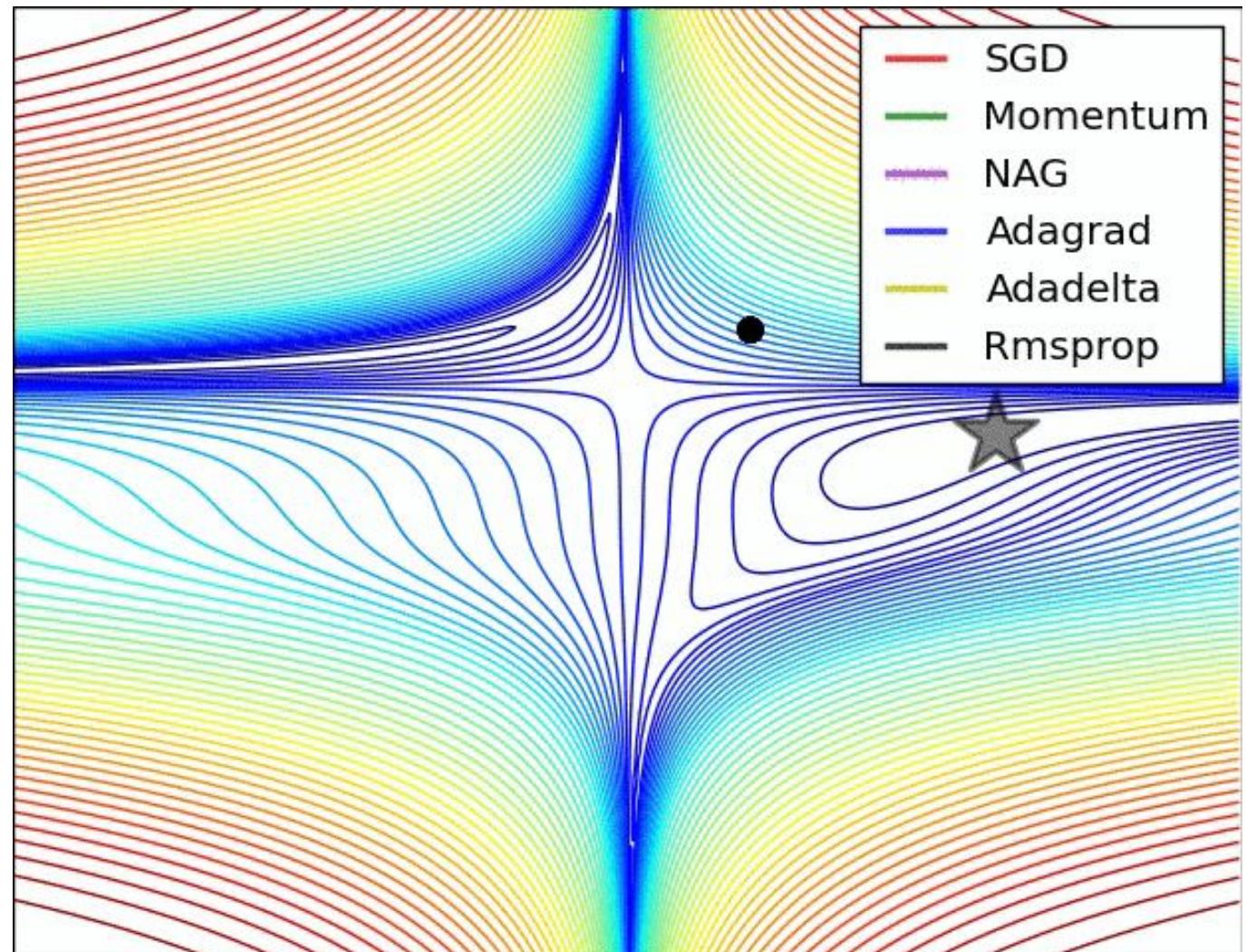
- При увеличении размера батча оценка градиента получается стабильнее, что позволяет увеличить learning rate
- В случае, если не происходит изменение метрики на валидации, стоит уменьшить значение learning rate [reduceLronplateau](#)
- Наиболее популярные оптимизаторы: [sgd](#), [adam](#)

<https://keras.io/optimizers/>

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ГРАДИЕНТНЫЙ СПУСК

Оптимизация Градиентный спуск

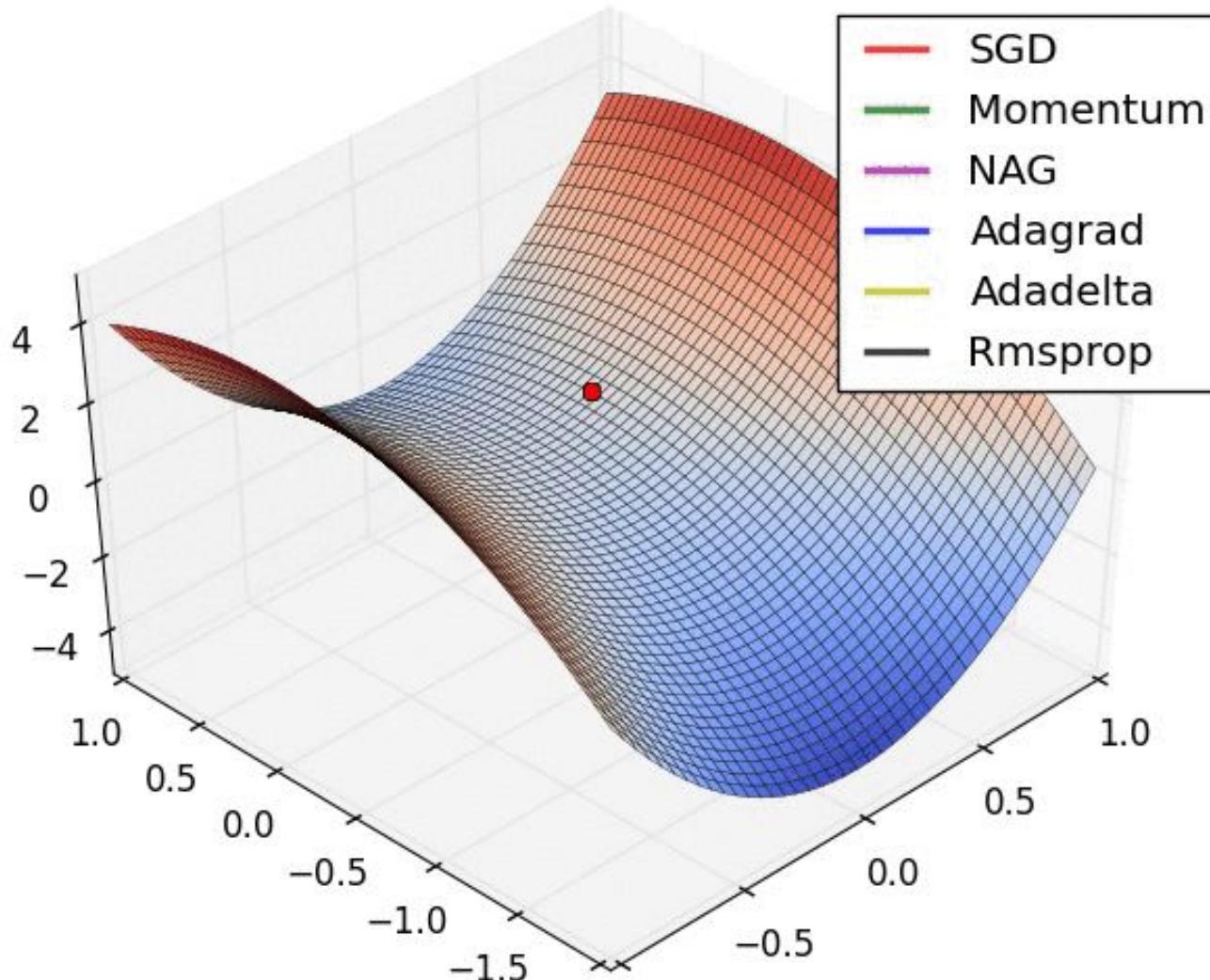
<http://ruder.io/optimizing-gradient-descent/>



ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ГРАДИЕНТНЫЙ СПУСК

Оптимизация —
седловая точка

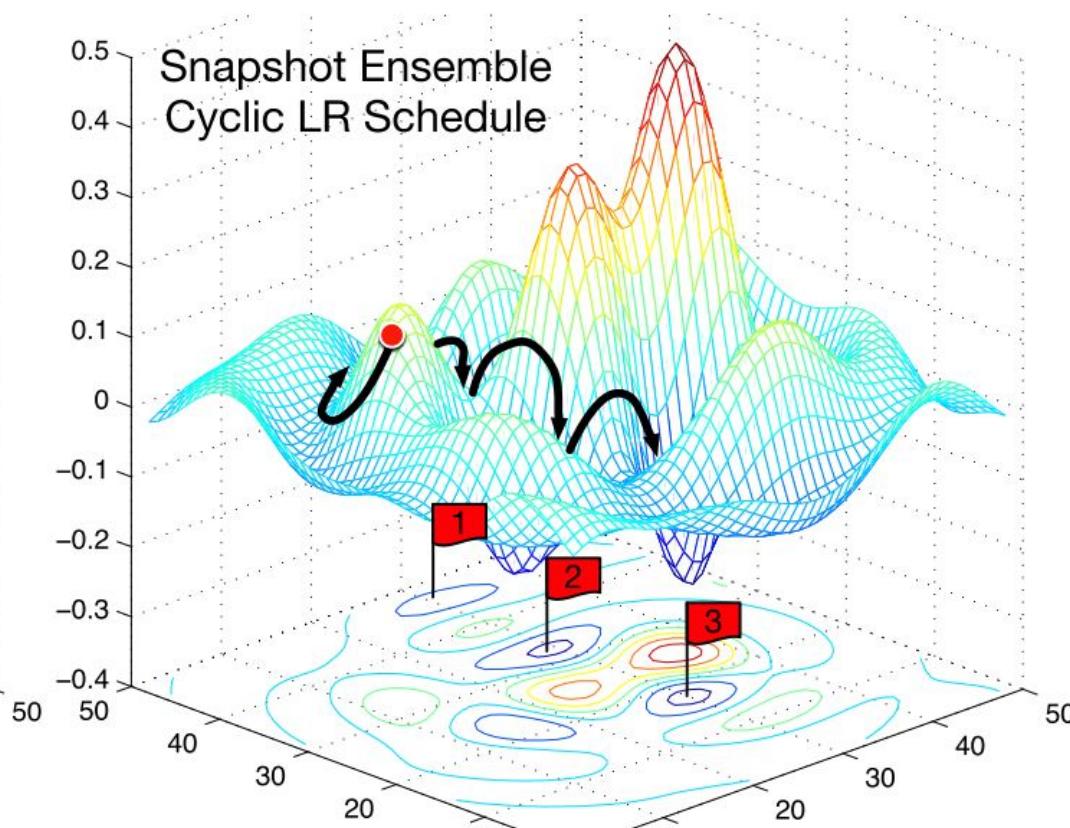
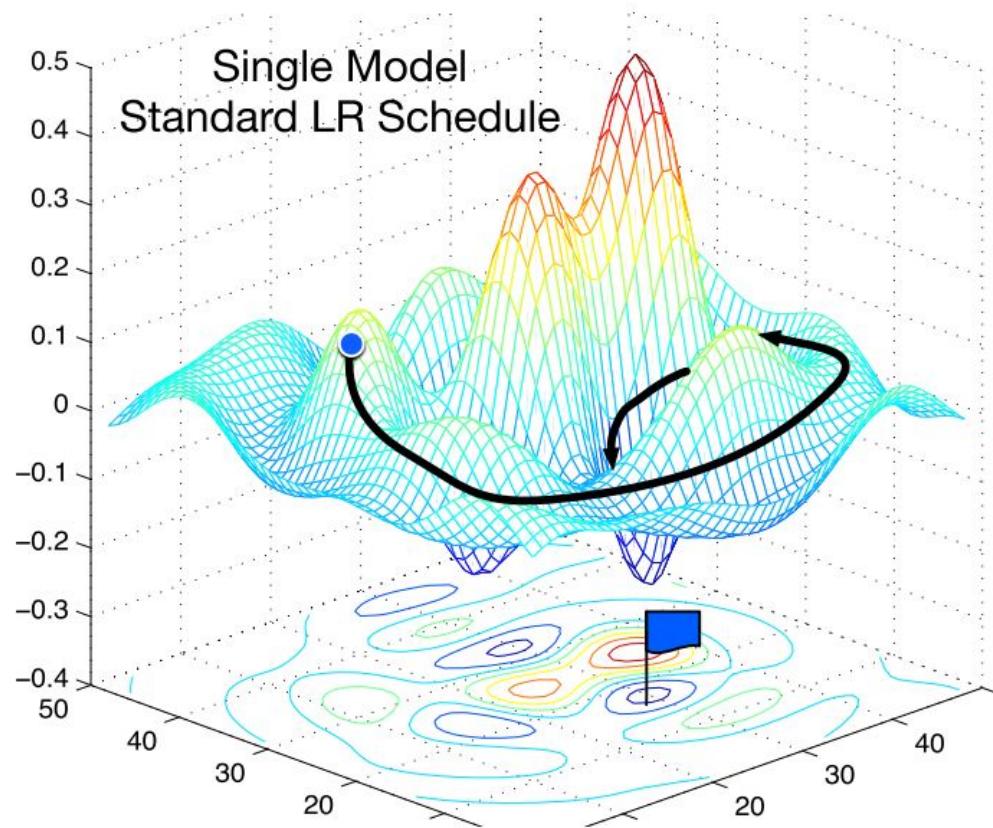
[http://ruder.io/optimizing
-gradient-descent/](http://ruder.io/optimizing-gradient-descent/)



ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ГРАДИЕНТНЫЙ СПУСК

Изменение LR по расписанию

<http://ruder.io/deep-learning-optimization-2017/>

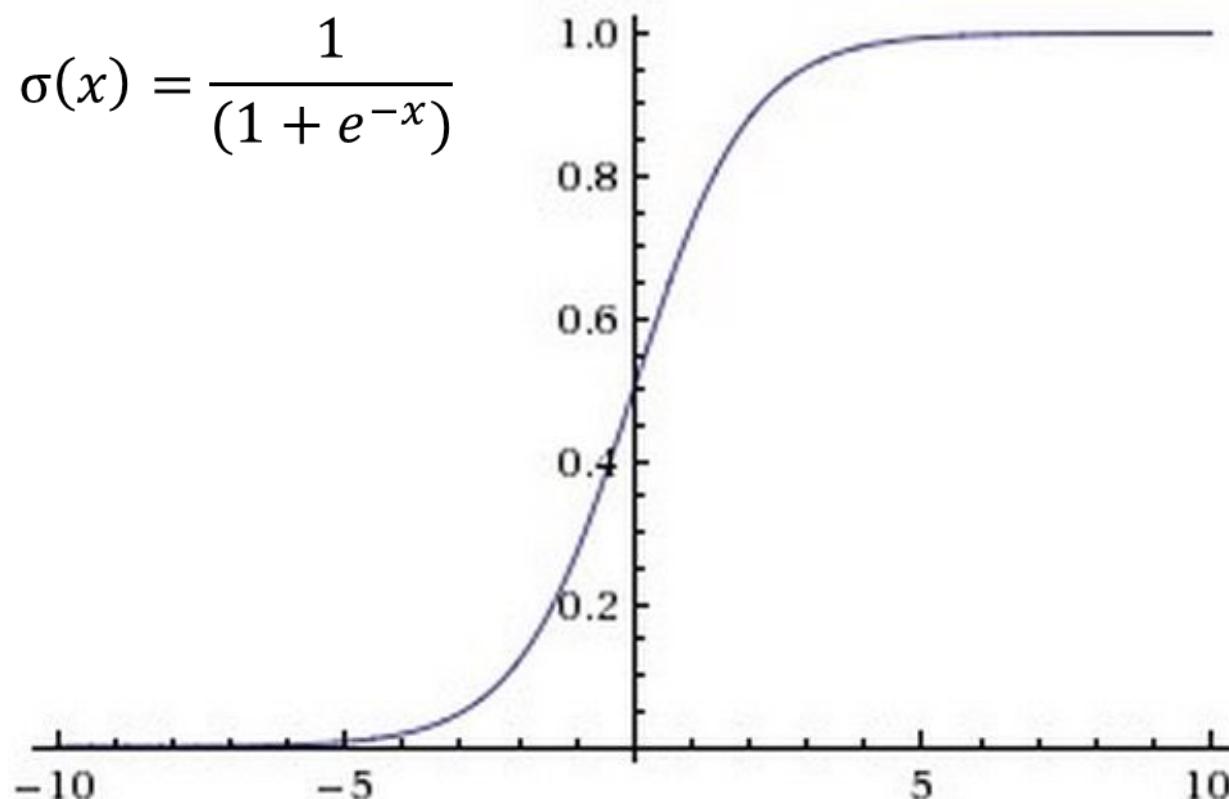


ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ФУНКЦИИ АКТИВАЦИИ

<https://keras.io/layers/advanced-activations/>

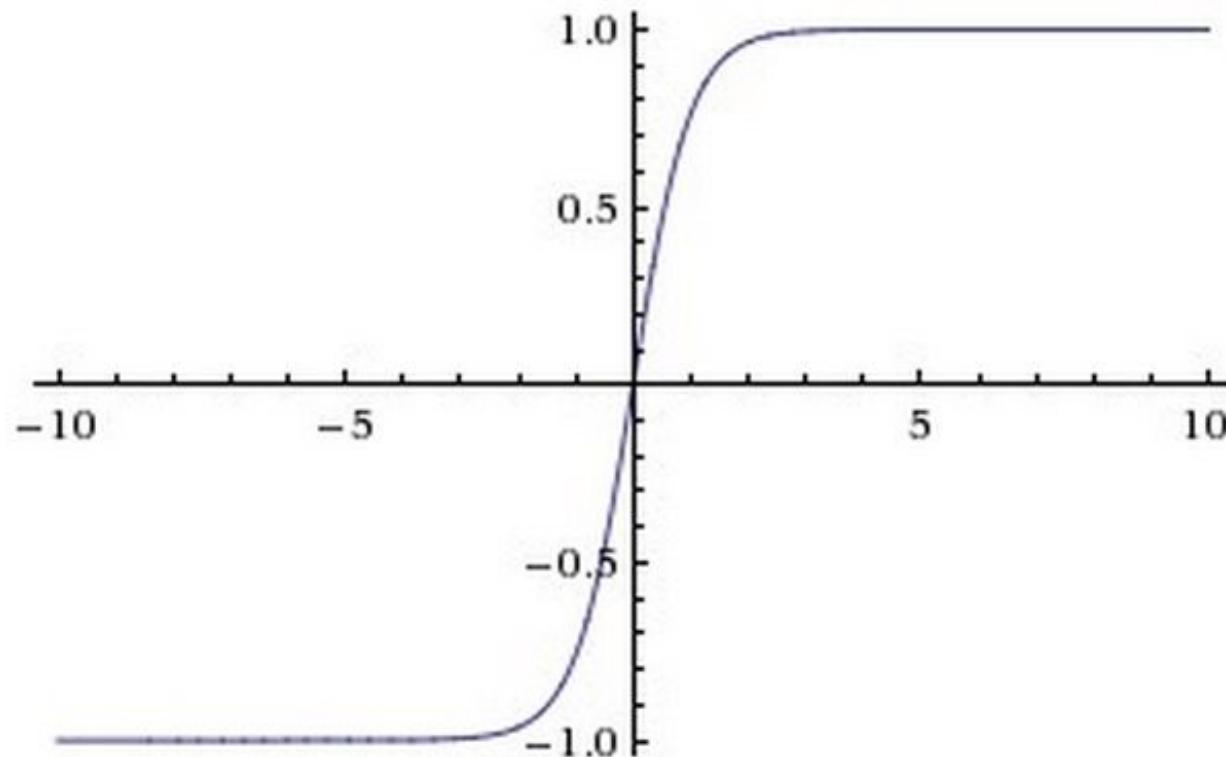
Функции активации — Sigmoid

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



- Убивает градиент
- Смещена относительно нуля — проблема для обучения
- Используется на выходном слое

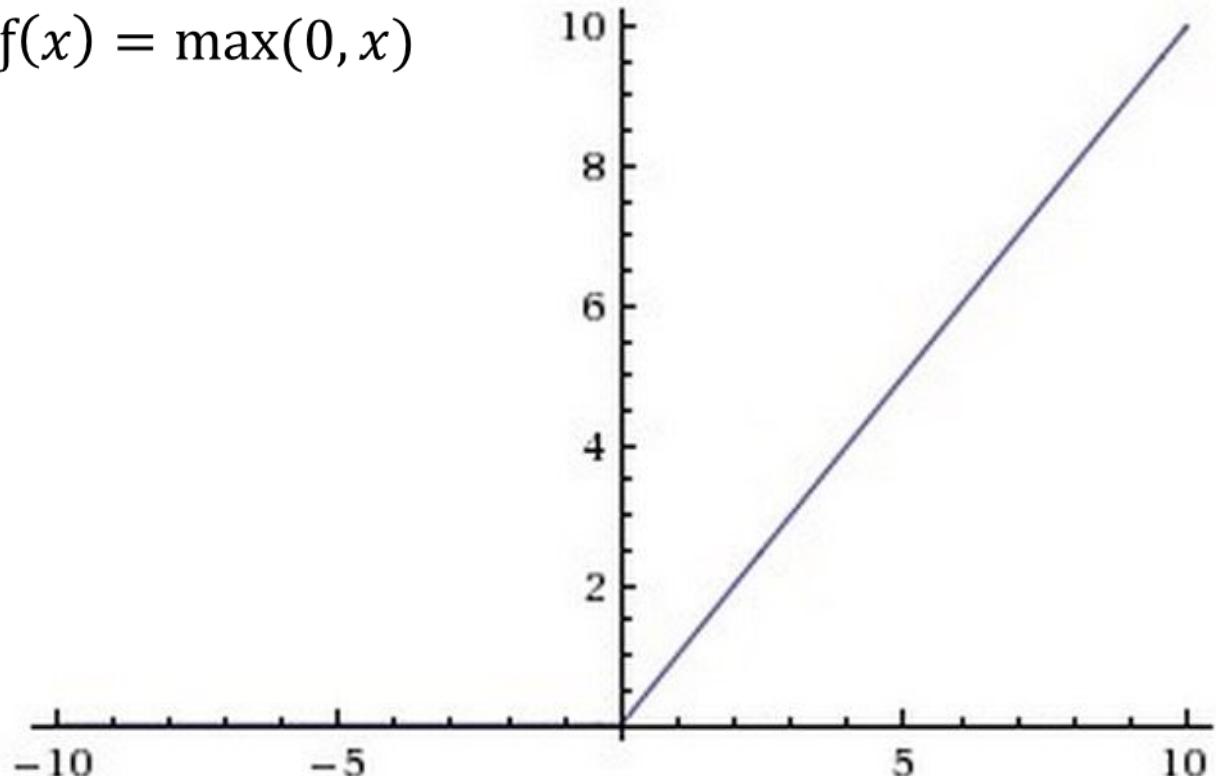
Функции активации — Tanh



- Остается проблема с градиентами
- Решена проблема с центрированием

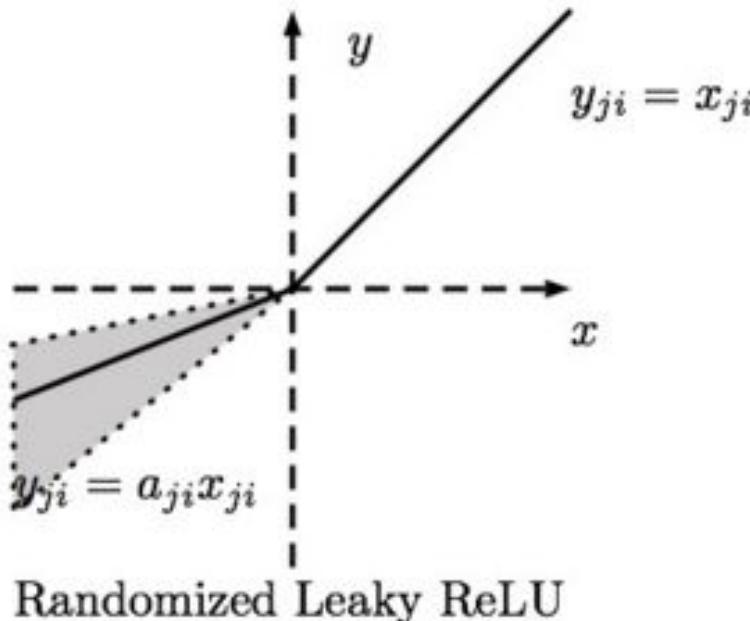
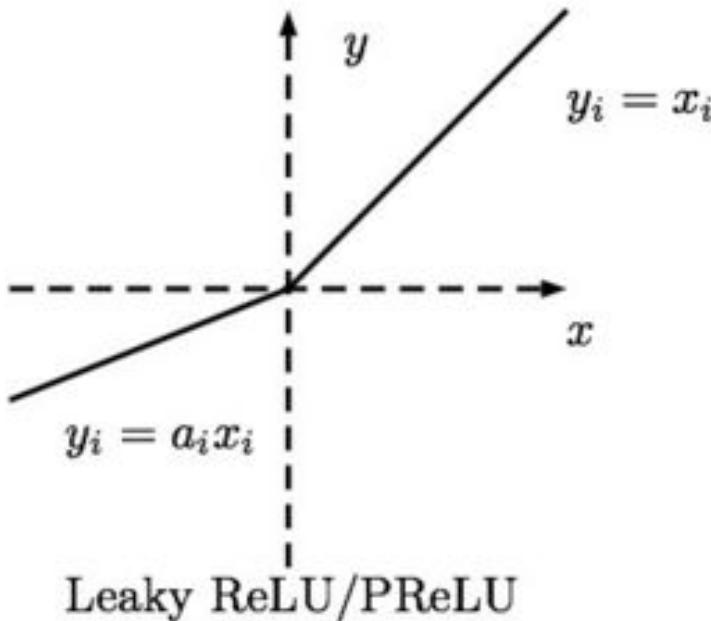
Функции активации — ReLU

$$f(x) = \max(0, x)$$



- Простая в реализации
- Отсутствие насыщения ускоряет процесс сходимости
- При большом значении градиента значение может уйти в минус и не вернуться
dying relu

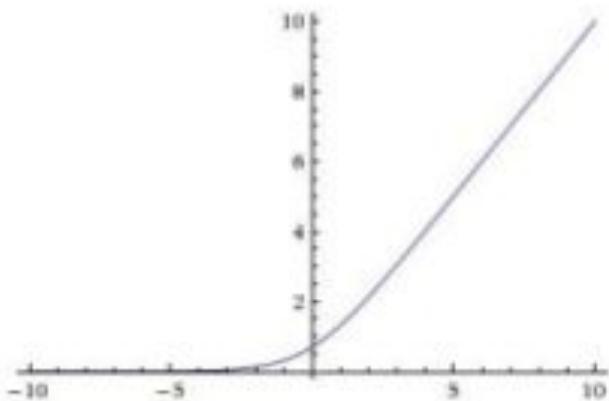
Функции активации — Parametric LU



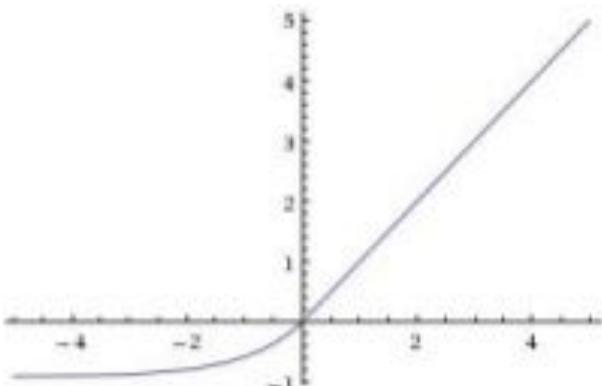
- Угол наклона является обучаемым параметром
- Угол наклона изменяется случайным образом

Функции активации — Exponential LU

Softplus and Exponential Linear Unit (ELU)



$$\text{softplus}(x) = \log(1 + e^x)$$



$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha (\exp(x) - 1) & \text{if } x \leq 0 \end{cases}$$

- Решена неоднозначность с градиентом в области нуля

Функции активации ReLU

Activation	Training Error	Test Error
ReLU	0.1356	0.429
Leaky ReLU, $a = 100$	0.11552	0.4205
Leaky ReLU, $a = 5.5$	0.08536	0.4042
PReLU	0.0633	0.4163
RReLU	0.1141	0.4025

Table 4. Error rate of CIFAR-100 Network in Network with different activation function

ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. РЕГУЛЯРИЗАЦИЯ

—

Регуляризация

L1, L2 —

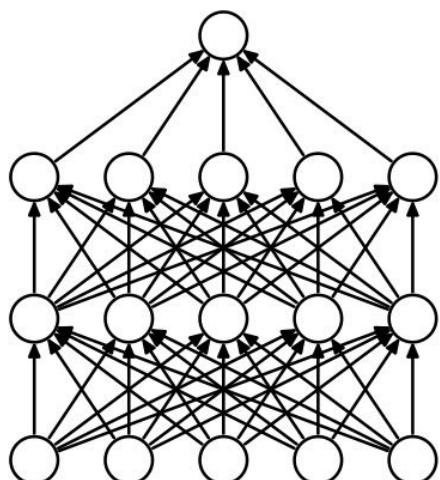
добавка в функцию потерь
модуль, квадрат,
при добавлении обоих —
называется
ELASTIC NET

Регуляризация

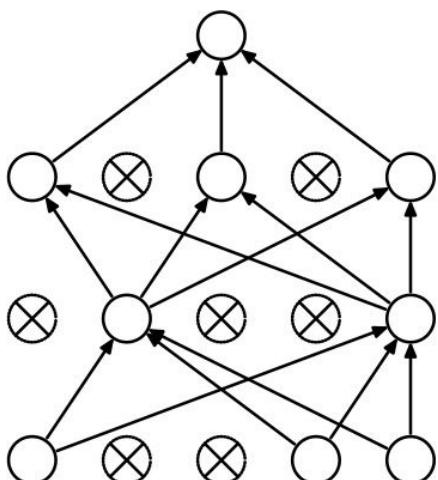
заключается
в добавлении
соответствующего
слагаемого
в функцию потерь

<https://keras.io/regularizers/>

Регуляризация — *Dropout*

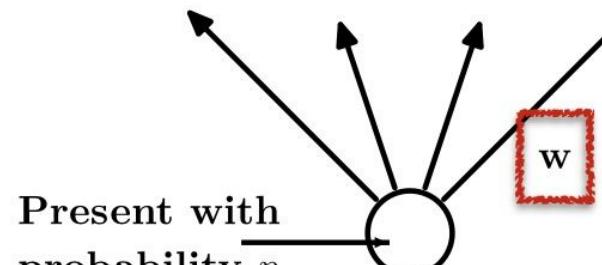


(a) Standard Neural Net

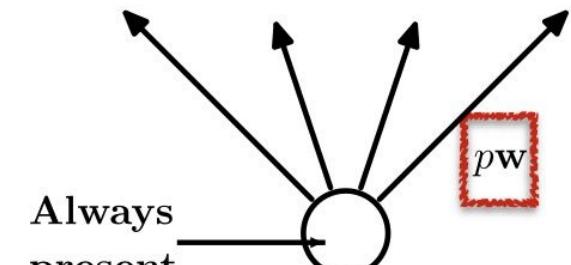


(b) After applying dropout.

Present with
probability p



(c) At training time



(d) At test time

Dropout

```
keras.layers.core.Dropout(rate, noise_shape=None, seed=None)
```

rate: диапазон 0..1

noise_shape: позволяет задать одинаковую маску по каналам

seed: инициализация генератора случайных чисел

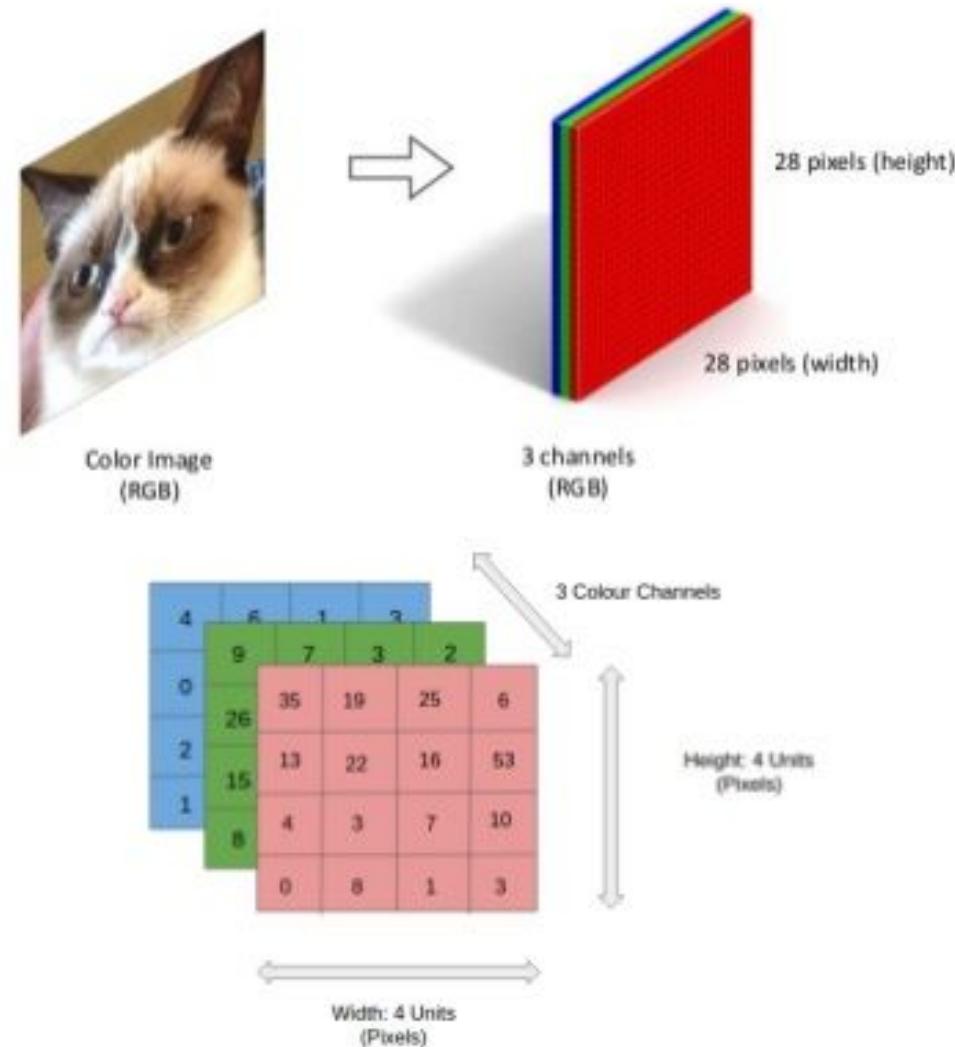
ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ОПТИМИЗАЦИЯ ВЫЧИСЛЕНИЙ

Параметры фильтров

- Размеры изображения кратны числу семплирующих слоев *pooling*
- Чем меньше размер фильтра, тем меньше операций необходимо выполнить для вычисления свертки
- Отступы *padding* необходимо добавлять при свертке для сохранения информации на краях картинки
актуально с увеличением глубины

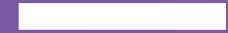
ОБУЧЕНИЕ СВЕРТОЧНОЙ СЕТИ НА ПРАКТИКЕ. ОПТИМИЗАЦИЯ ВЫЧИСЛЕНИЙ

Формат данных и производительность



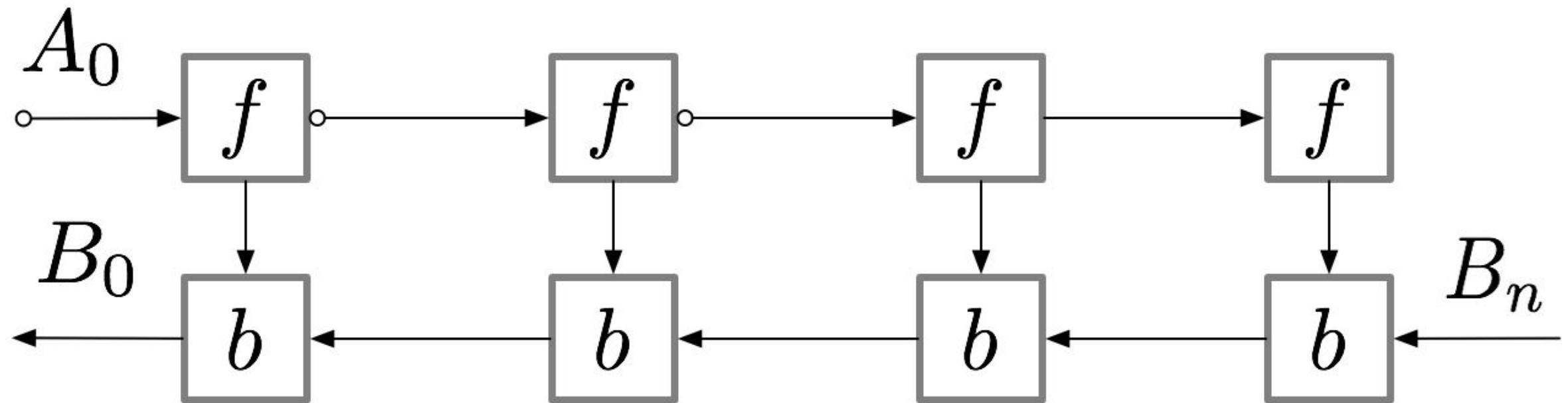
Формат данных и производительность

- Производительность зависит от порядка измерений
- Изменение порядка с NHWC *channel_last* на NCHW *channel_first* дает прирост в скорости вычислений на GPU до 10%
- Это связано с особенностью библиотеки cudnn, которая поддерживает только NCHW формат
- Если вы используете NHWC tensorflow автоматически приводит данные в формат NCHW перед обработкой на GPU
- При этом, в формате NCHW некоторые операции tensorflow могут не поддерживаться на CPU

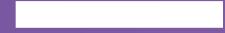


GRADIENT CHECKPOINTING

Gradient checkpointing

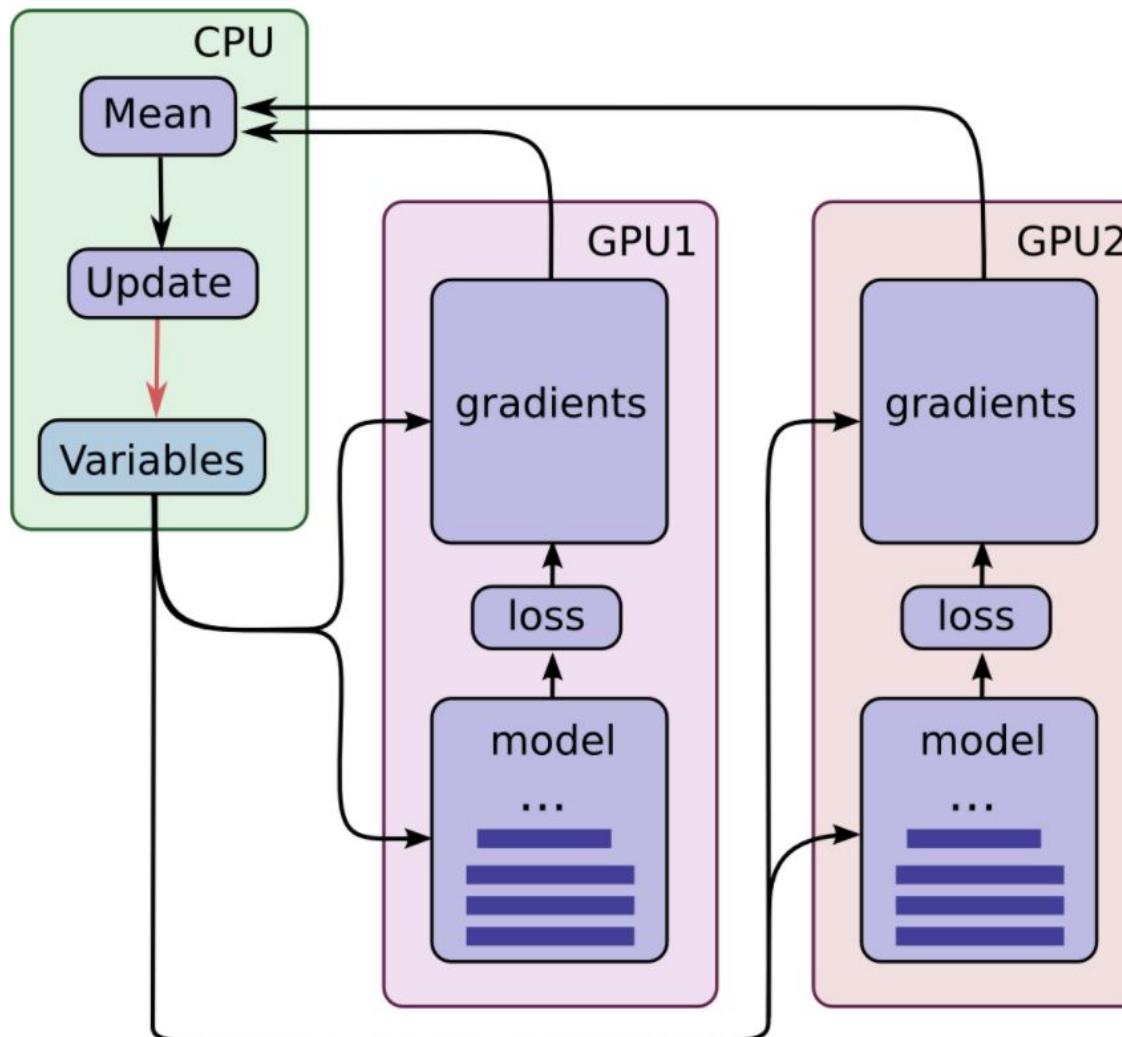


<https://github.com/openai/gradient-checkpointing>

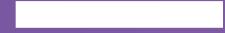


РАСПРЕДЕЛЕННОЕ ОБУЧЕНИЕ

РАСПРЕДЕЛЕННОЕ ОБУЧЕНИЕ



[https://www.tensorflow.org/
api_docs/python/tf/keras/
utils/multi_gpu_model](https://www.tensorflow.org/api_docs/python/tf/keras/utils/multi_gpu_model)



РЕЗЮМЕ

Резюме

- Качество различных нейросетевых архитектур
сравнивают на открытых датасетах
- Предобученные на открытых датасетах
модели доступны для скачивания
- Для решения практической задачи как правило **адаптируют готовую архитектуру**, предобученную на открытом датасете
- **Существует набор подходов**, позволяющий повысить качество модели:
 - аугментация, предобработка и нормализация входных данных,
 - регуляризация модели,
 - использование relu в качестве функции активации

Полезные материалы

- [AN ANALYSIS OF DEEP NEURAL NETWORK MODELS FOR PRACTICAL APPLICATIONS](#)
- [The 9 Deep Learning Papers You Need To Know About](#)
- [Must Know Tips/Tricks in Deep Neural Networks](#)
- [Practical Recommendations for Gradient-Based Training of Deep Architectures](#)
- [Building powerful image classification models using very little data](#)
- [An overview of gradient descent optimization algorithms](#)