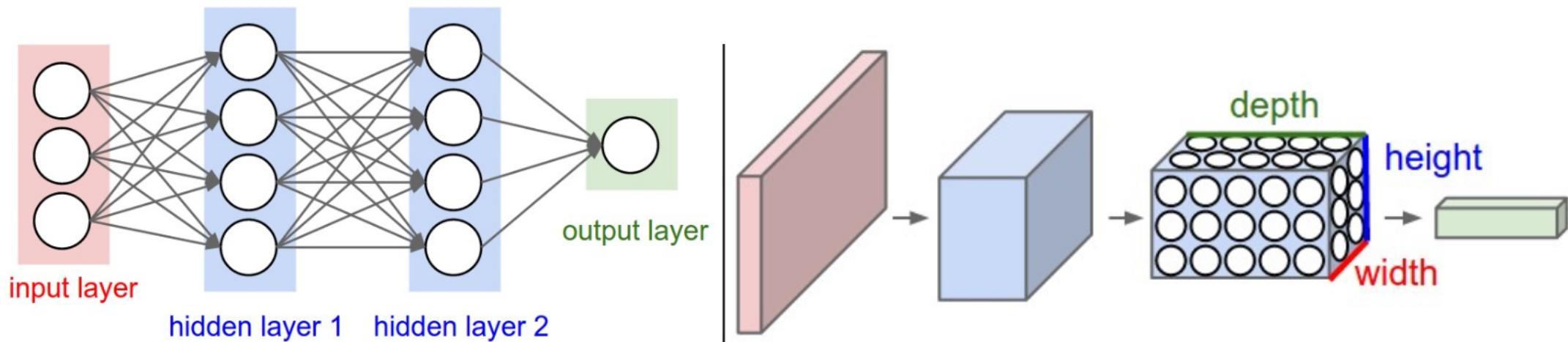


# LSML #6

Сверточные сети

# Convolutional NN (сверточные сети)

- Простые нейросети содержат только полно связные слои (очень много параметров)
- Сверточные сети преобразуют объем ( $W \times H \times D$ ) входов в другой объем и используют меньшее число параметров



# Как работает свертка

- Свертка – это скалярное произведение фильтра и области изображения
- Картина (5 x 5, зеленая), фиксированный фильтр (3 x 3, оранжевый)
- Задается шаг фильтра (1)

1 <small><math>\times 1</math></small>	1 <small><math>\times 0</math></small>	1 <small><math>\times 1</math></small>	0	0
0 <small><math>\times 0</math></small>	1 <small><math>\times 1</math></small>	1 <small><math>\times 0</math></small>	1	0
0 <small><math>\times 1</math></small>	0 <small><math>\times 0</math></small>	1 <small><math>\times 1</math></small>	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved  
Feature

Через 2 шага

1	1	1 <small><math>\times 1</math></small>	0 <small><math>\times 0</math></small>	0 <small><math>\times 1</math></small>
0	1	1 <small><math>\times 0</math></small>	1 <small><math>\times 1</math></small>	0 <small><math>\times 0</math></small>
0	0	1 <small><math>\times 1</math></small>	1 <small><math>\times 0</math></small>	1 <small><math>\times 1</math></small>
0	0	1	1	0
0	1	1	0	0

Image

4	3	4

Convolved  
Feature

# Свертки давно использовали для изображений



Детектор границ

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



Повышение четкости

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$



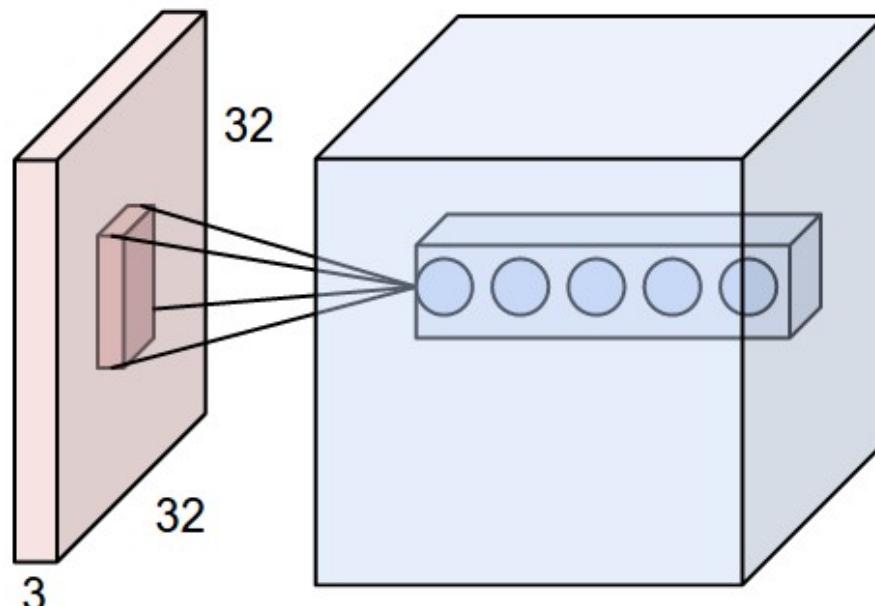
Размытие

$$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



# Откуда берется объем

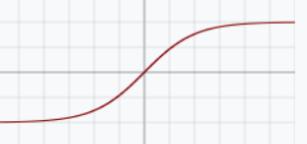
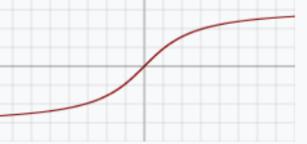
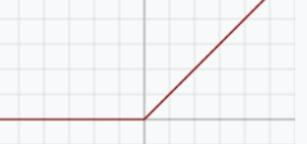
- Картинка цветная, 3 ( $C_{in}$ ) канала RGB, а значит размера  $W \times H \times C_{in}$
- Можно обучить  $C_{out}$  фильтров размера  $W_f \times H_f \times C_{in}$
- На выходе объем  $W_{out} \times H_{out} \times C_{out}$ , реализуется через матричное умножение (каждая область изображения преобразуется в вектор)



# Свертка $1 \times 1$

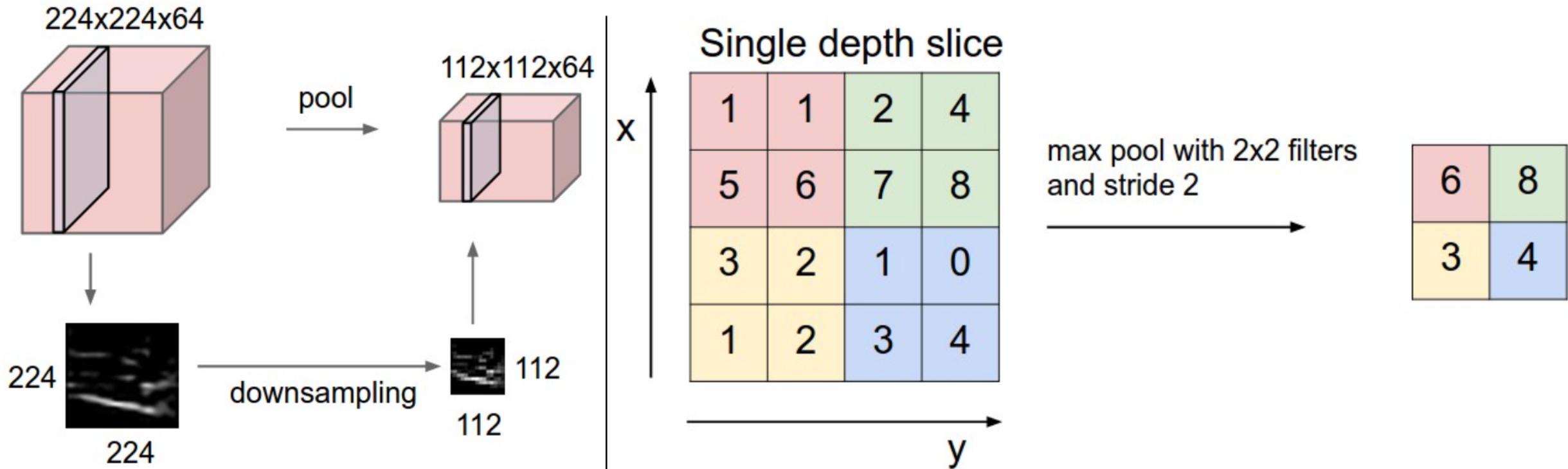
- Учитывает взаимосвязи разных входных каналов в одном пикселе
- Можно использовать для сжатия представления (фильтров может быть избыточное число) как в РСА

# Не забываем про функцию активации

Logistic (a.k.a. Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Softsign [7][8]		$f(x) = \frac{x}{1 +  x }$
Rectified linear unit (ReLU) <sup>[9]</sup>		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$
Leaky rectified linear unit (Leaky ReLU) <sup>[10]</sup>		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

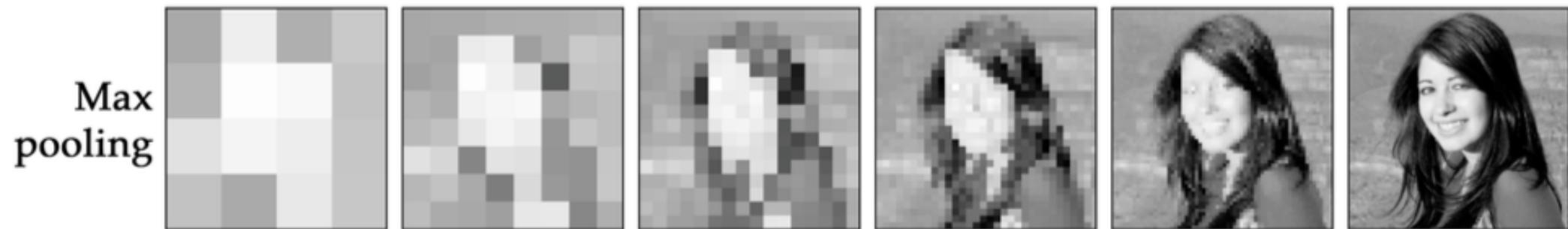
# Pooling слой

- Уменьшает высоту и ширину объемного слоя нейронов
- Увеличивает поле зрения при последующих свертках

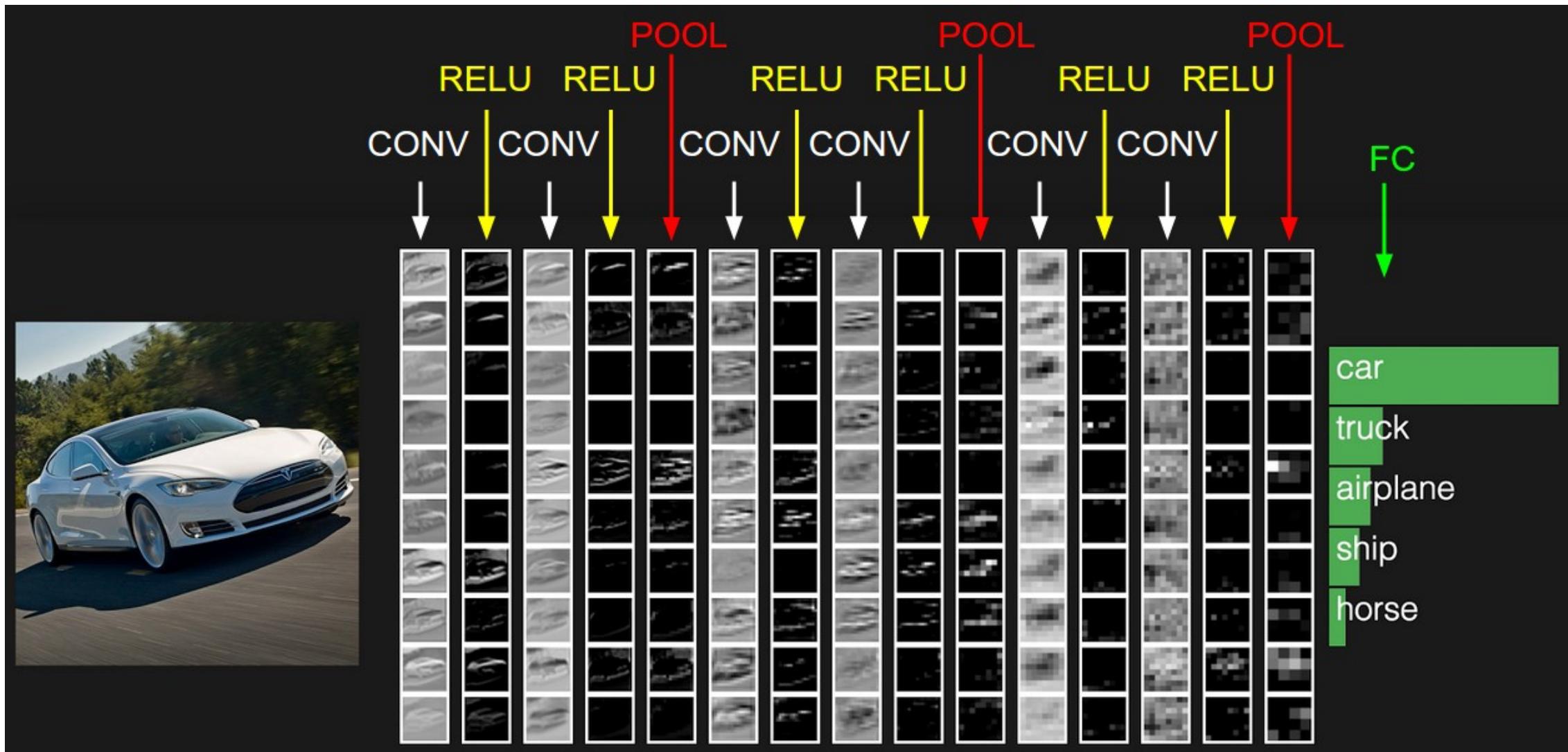


# Можно и без pooling

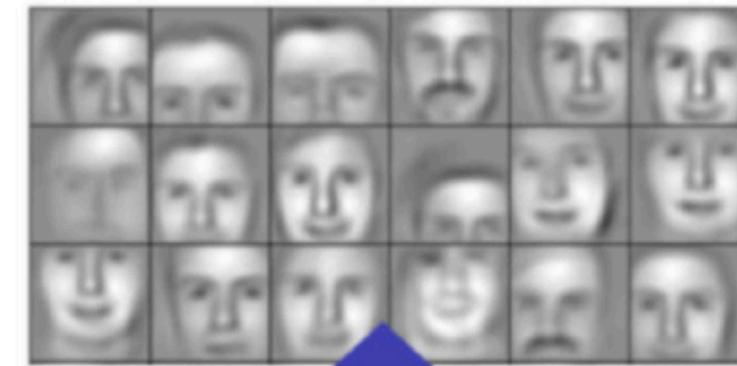
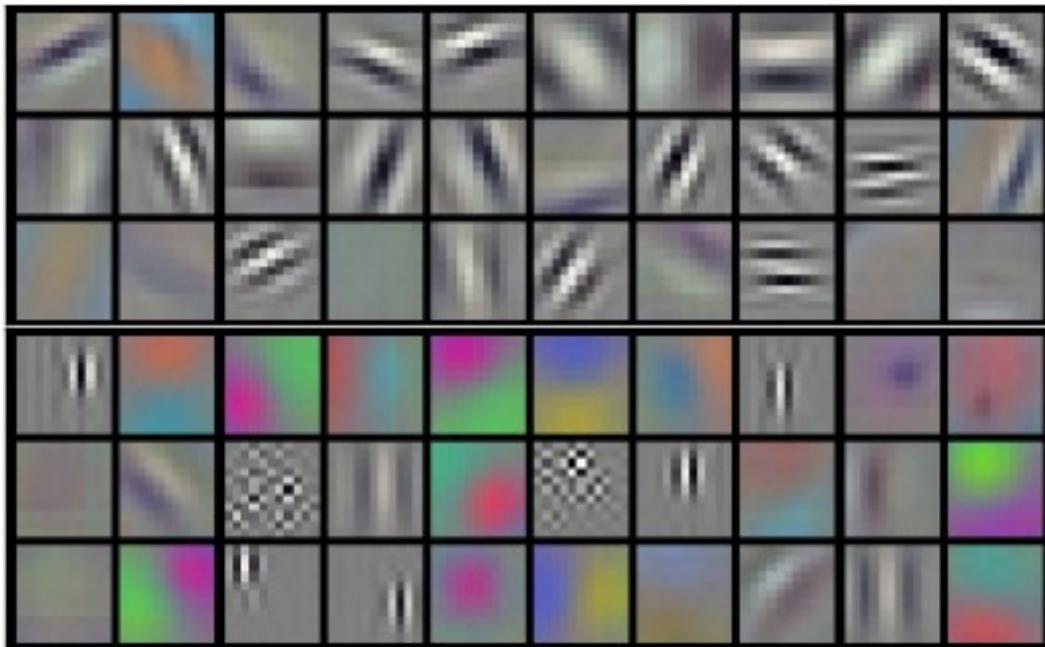
- Можно увеличить шаг фильтра вместо pooling
- Тогда можно обойтись только свертками



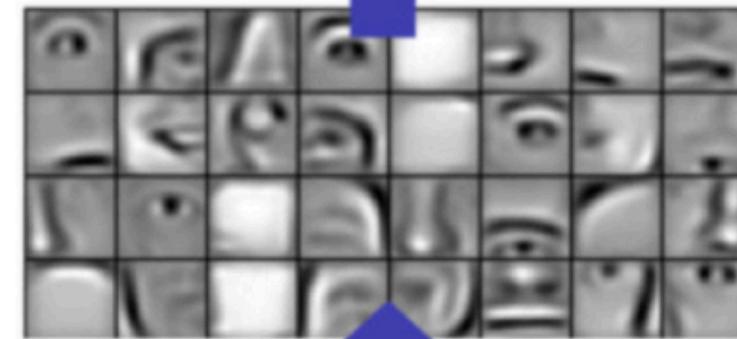
# Как устроены сверточные сети



# Какие обучаются фильтры



Layer 3

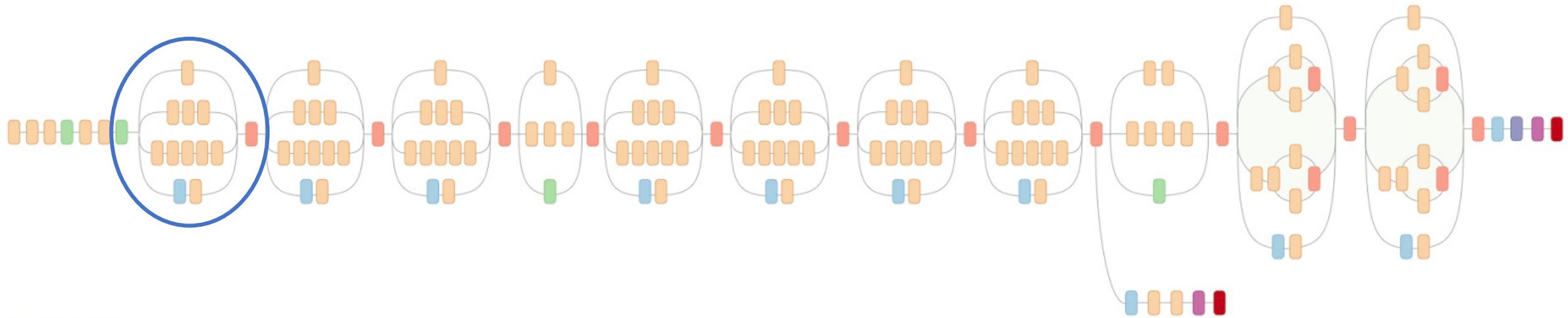


Layer 2



Layer 1

# Inception V3 сложнее

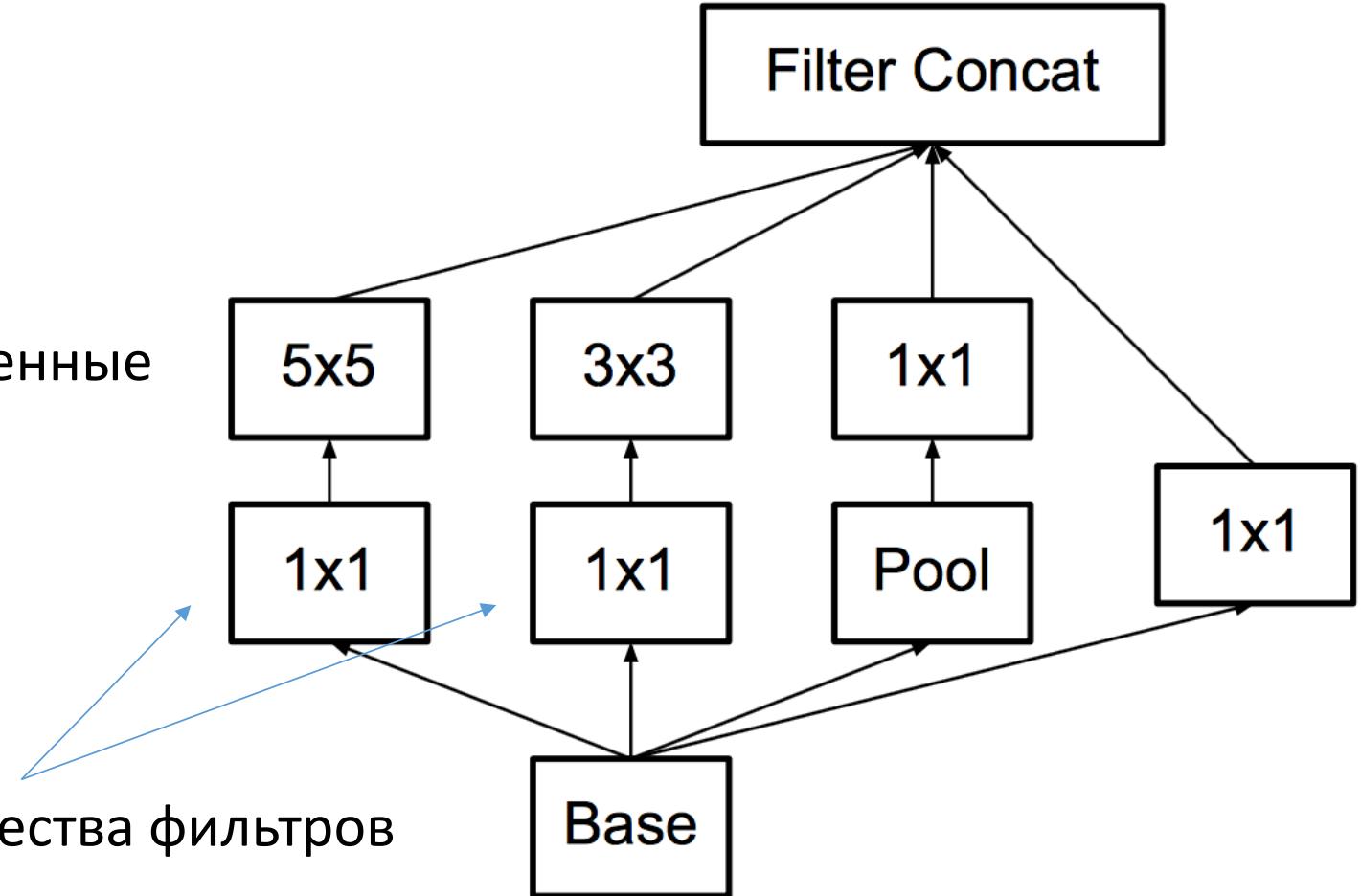


- Convolution
- AvgPool
- MaxPool
- Concat
- Dropout
- Fully connected
- Softmax

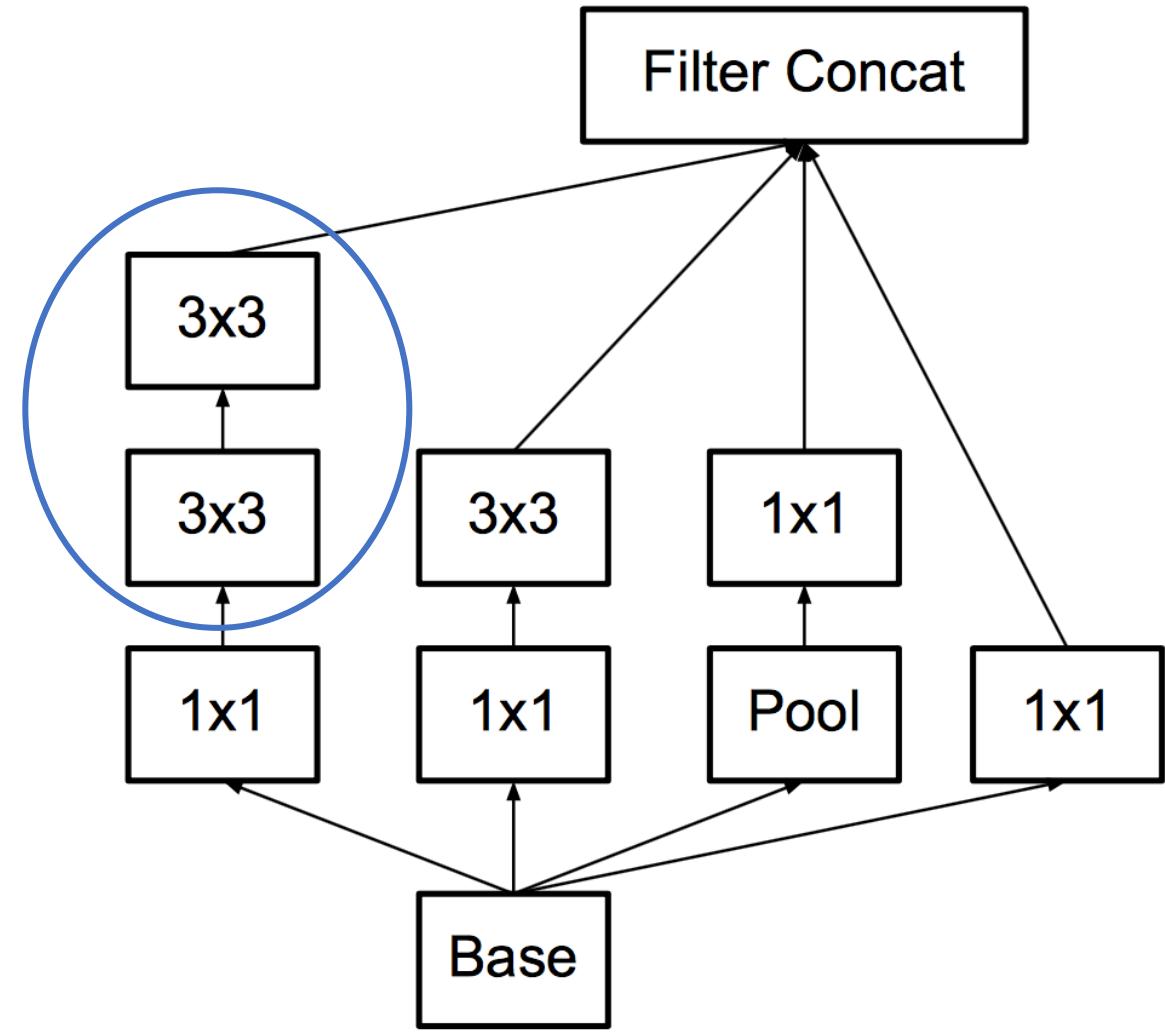
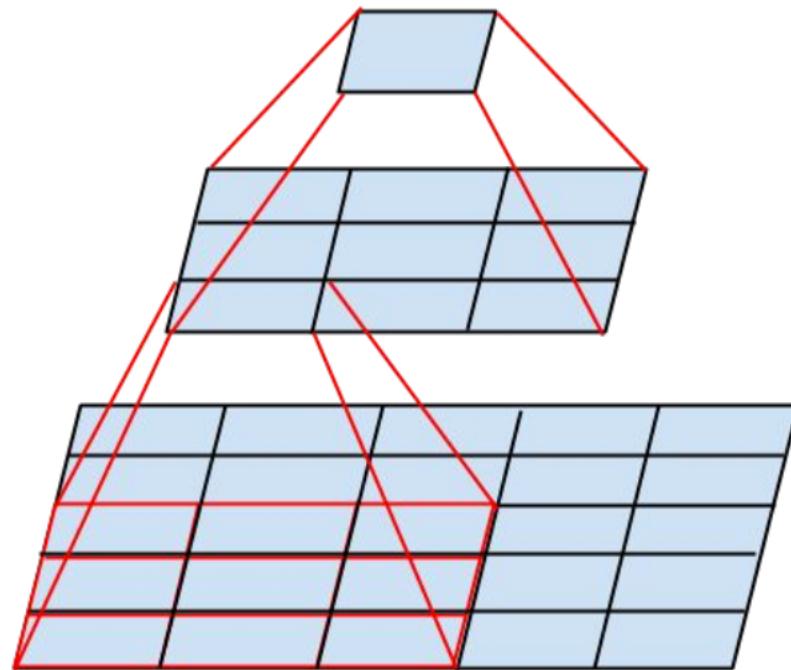
# Inception block

Свертки 5x5 медленные

Сжатие количества фильтров

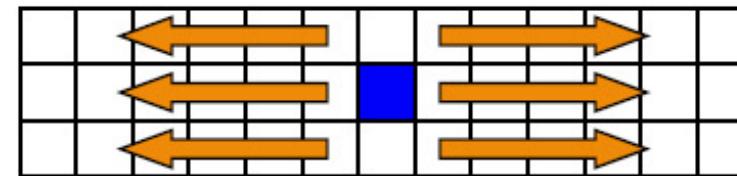


# Свертки 5x5 дорогие, факторизуем их

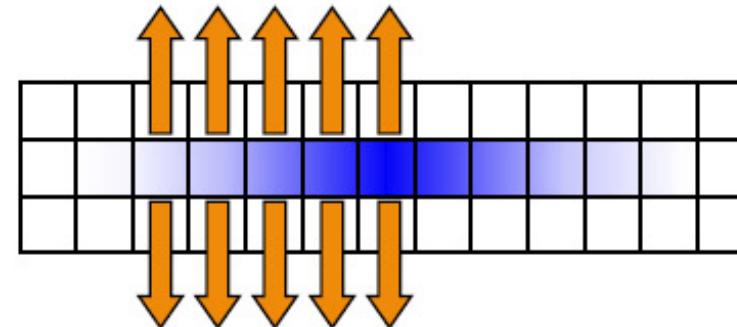


# Размытие по Гауссу это сепарабельный фильтр

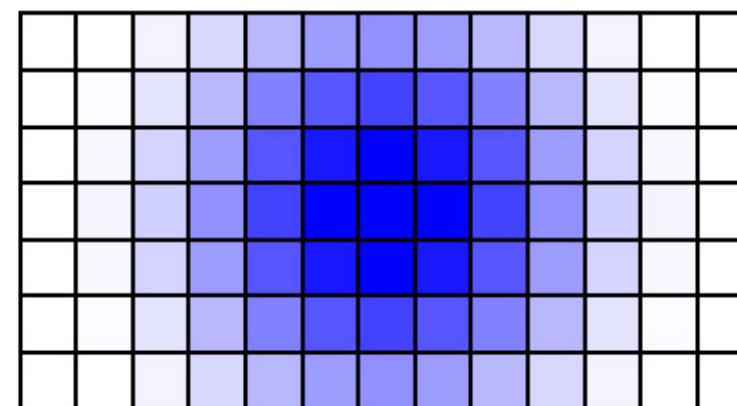
- Размытие можно сделать сначала по горизонтали
- Затем размыть результат по вертикали
- Это эквивалентно двумерной свертке, но сильно быстрее
- Вдруг сработает для других фильтров?



Blur the source horizontally

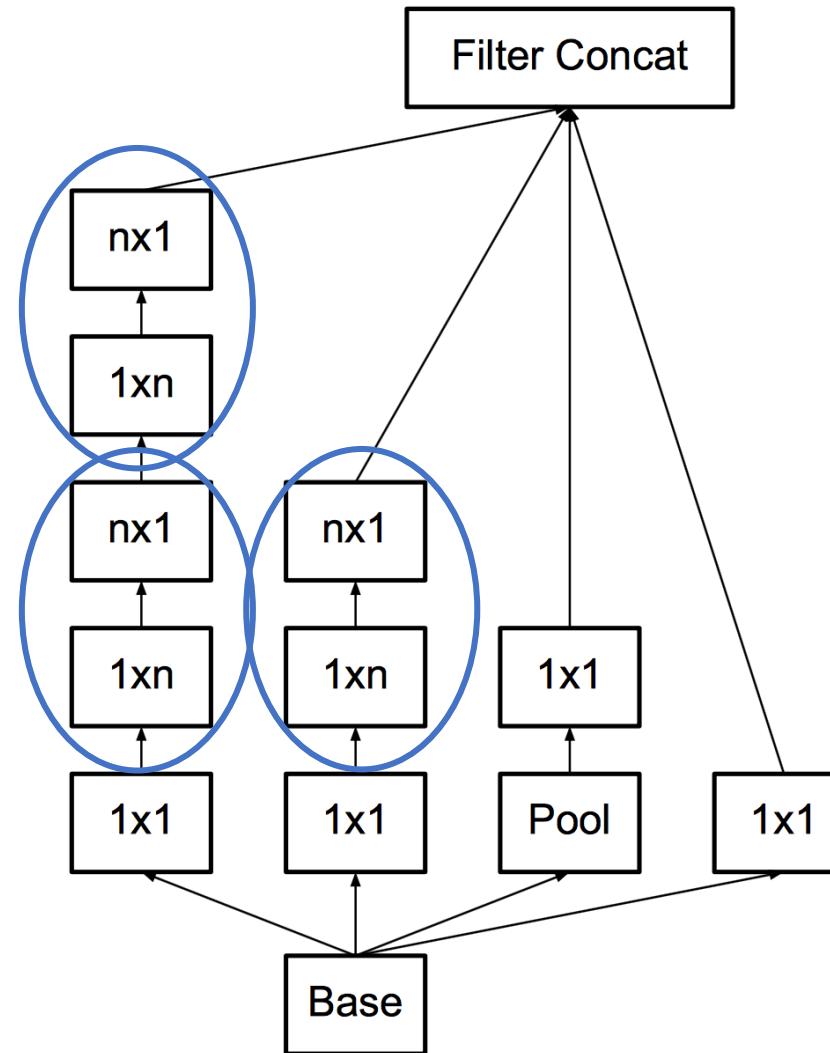


Blur the blur vertically



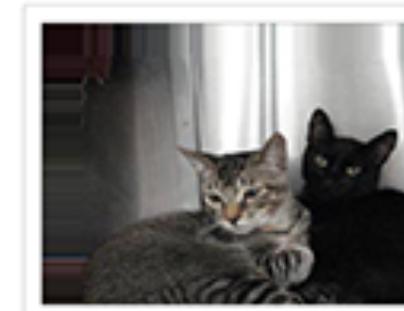
Result

# Продолжим факторизовать свертки



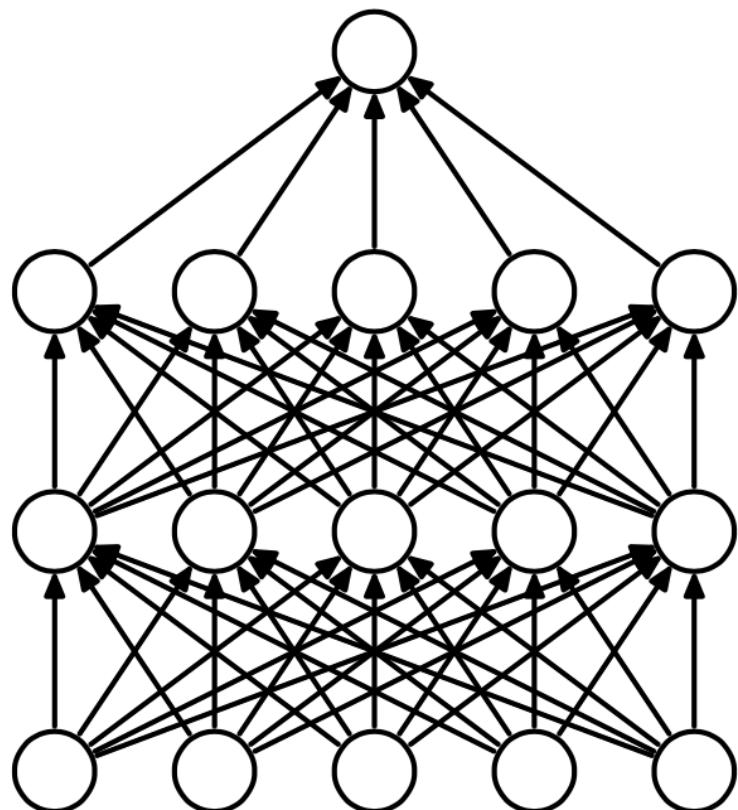
# Data augmentation

- В нейросетях много миллионов параметров, а выборки ограниченные
- Хочется получить больше данных из имеющихся – добавим повороты, растяжения, отражения, и т.д. как новые картинки

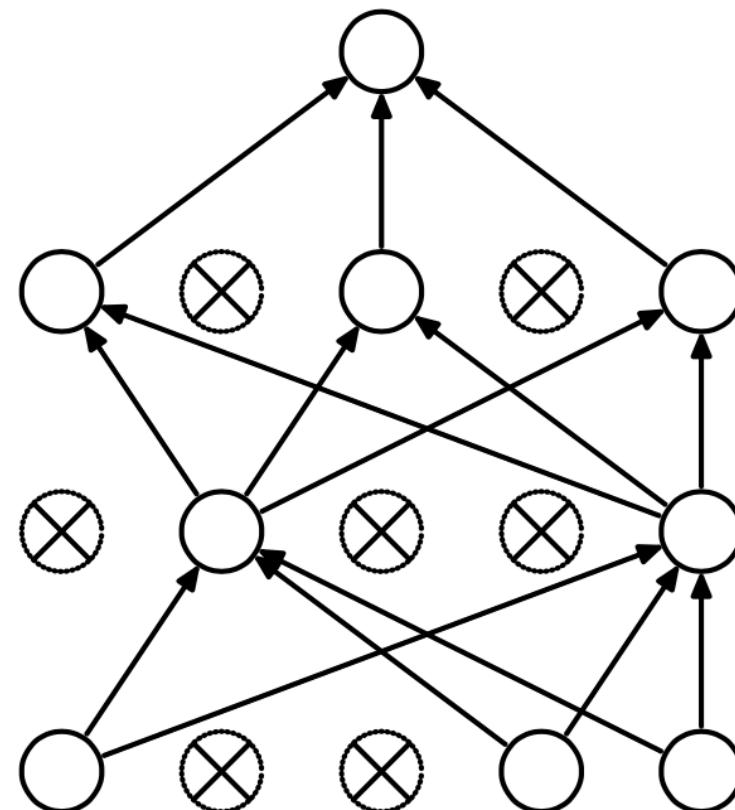


# Dropout

- С вероятностью  $p$  выкидываем нейрон из сети



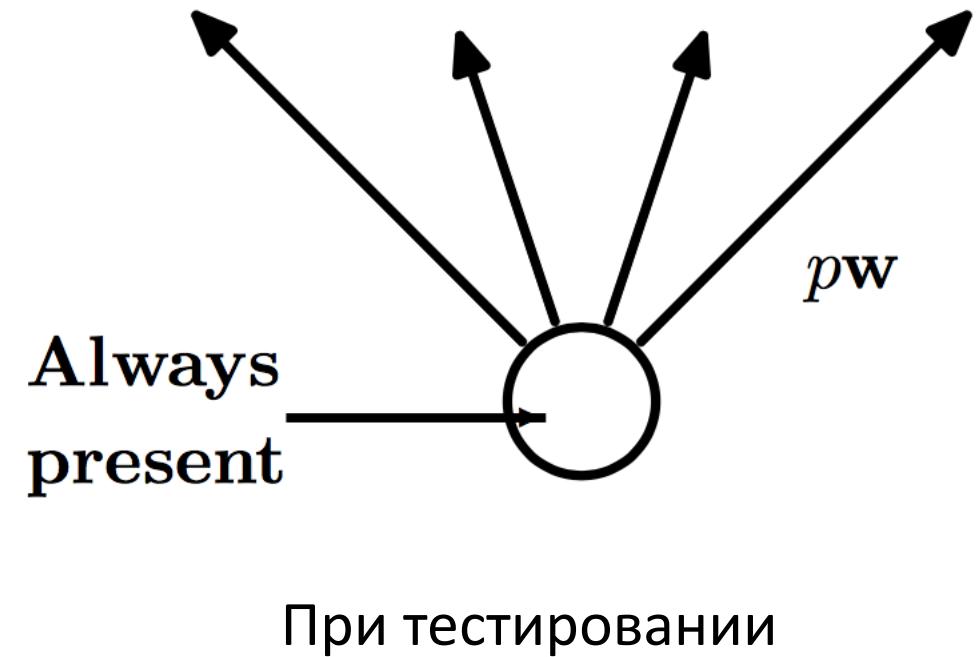
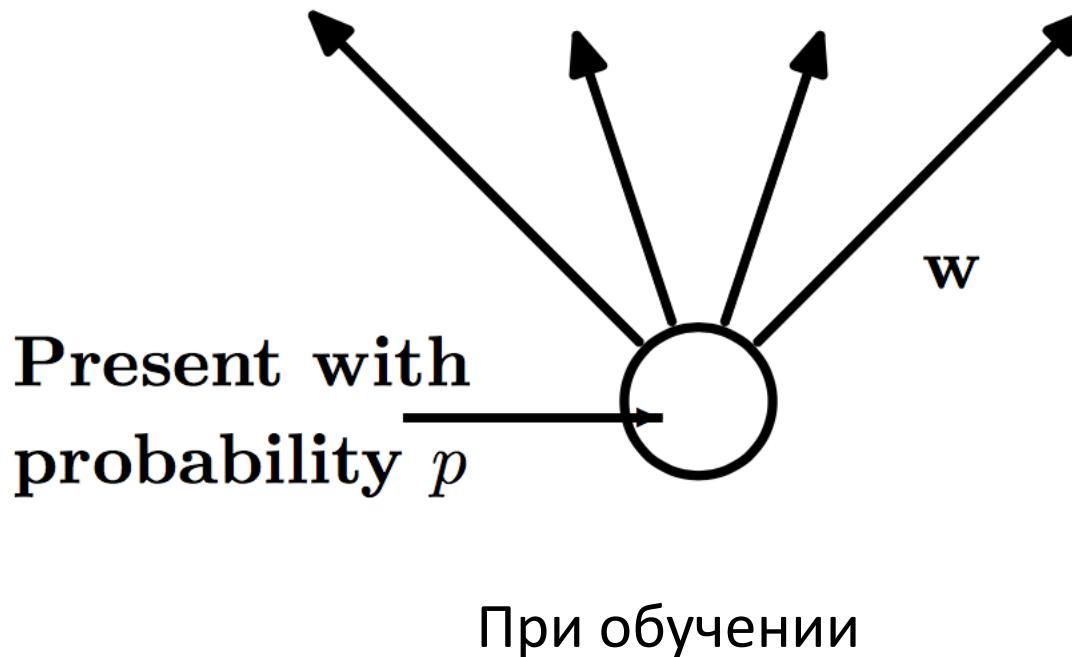
(a) Standard Neural Net



(b) After applying dropout.

# Dropout

- Можно рассматривать как ансамбль из большого числа более простых сетей (байесовские методы объясняют лучше)



# Batch normalization

- Всем известно, что линейные модели лучше сходятся, если признаки нормированы
- В нейросетях у нас много линейных преобразований + активаций
- Выходы нейронов будут использоваться на следующем слое
- Было бы здорово научиться как-то нормировать выходы каждого нейрона

# Batch normalization

- Нормализуем выход каждого нейрона:

$$h_i = \frac{h_i - \mu_i}{\sqrt{\sigma_i^2}}$$

- Откуда брать  $\mu_i$  и  $\sigma_i^2$ ?

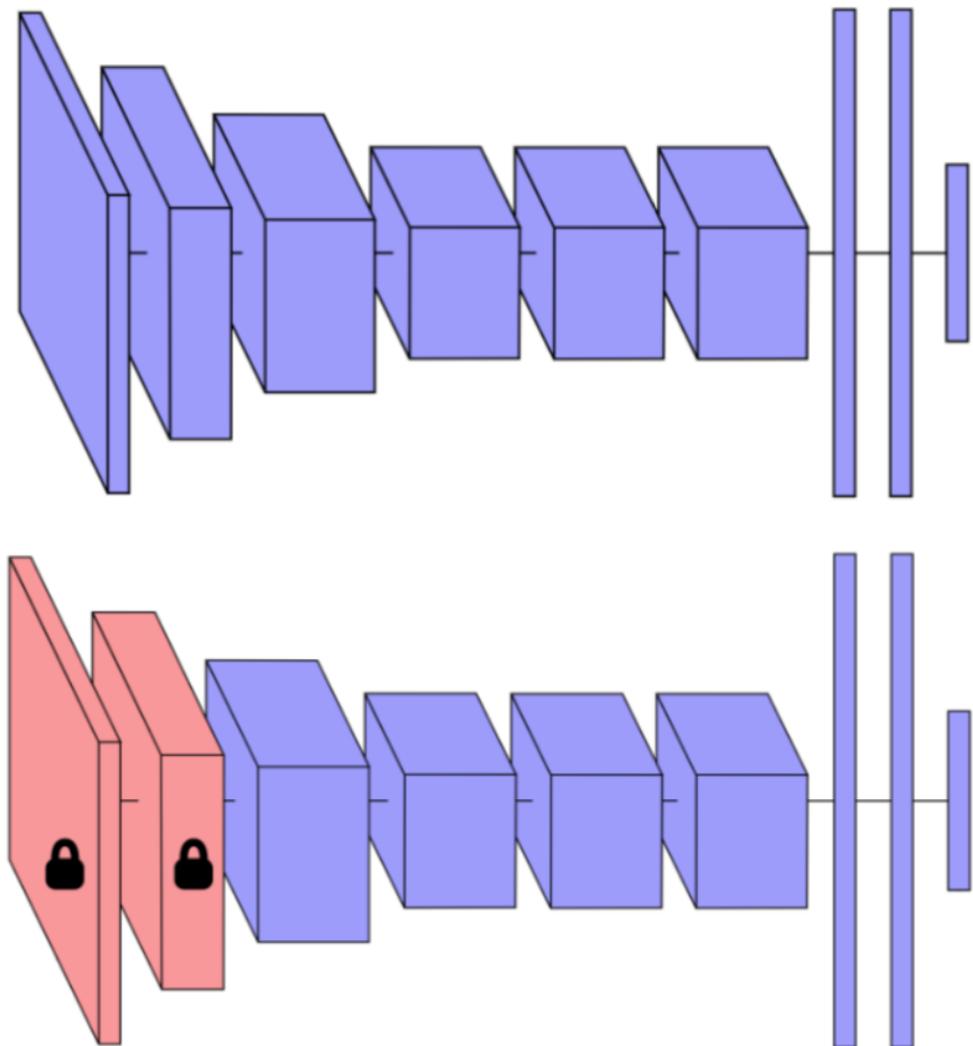
Экспоненциальное сглаживание оценок по batch!

$$\mu_i := \alpha \cdot \text{mean}_{batch} + (1 - \alpha) \cdot \mu_i$$

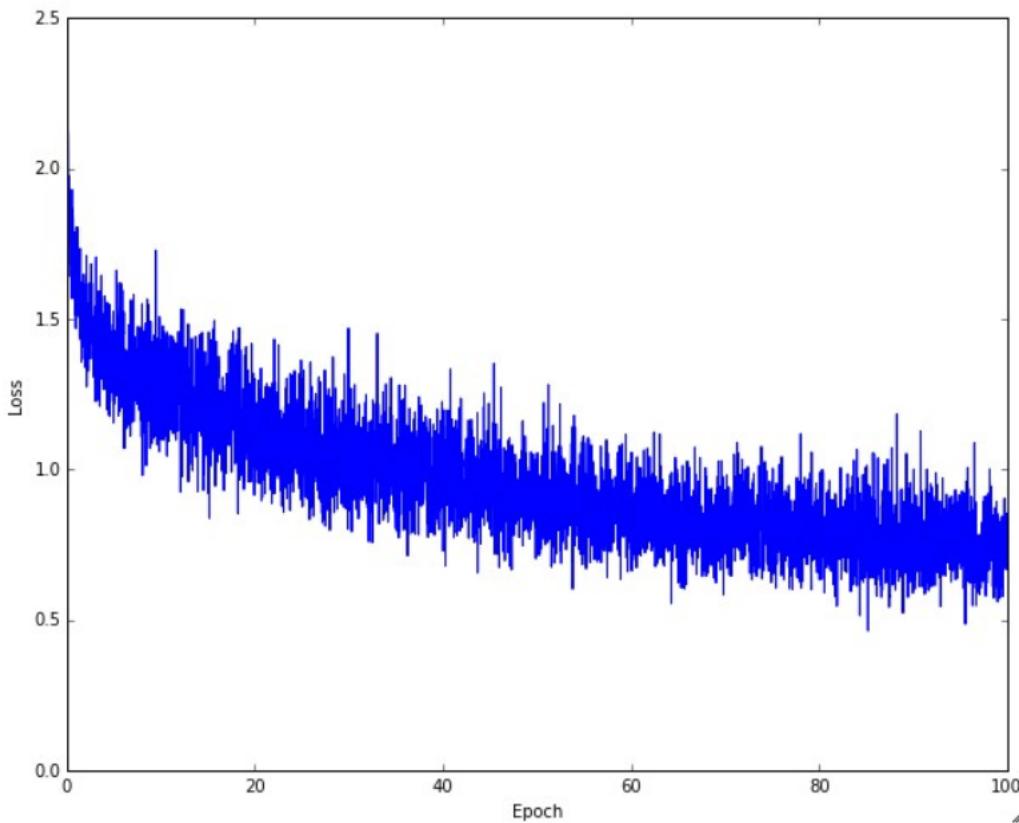
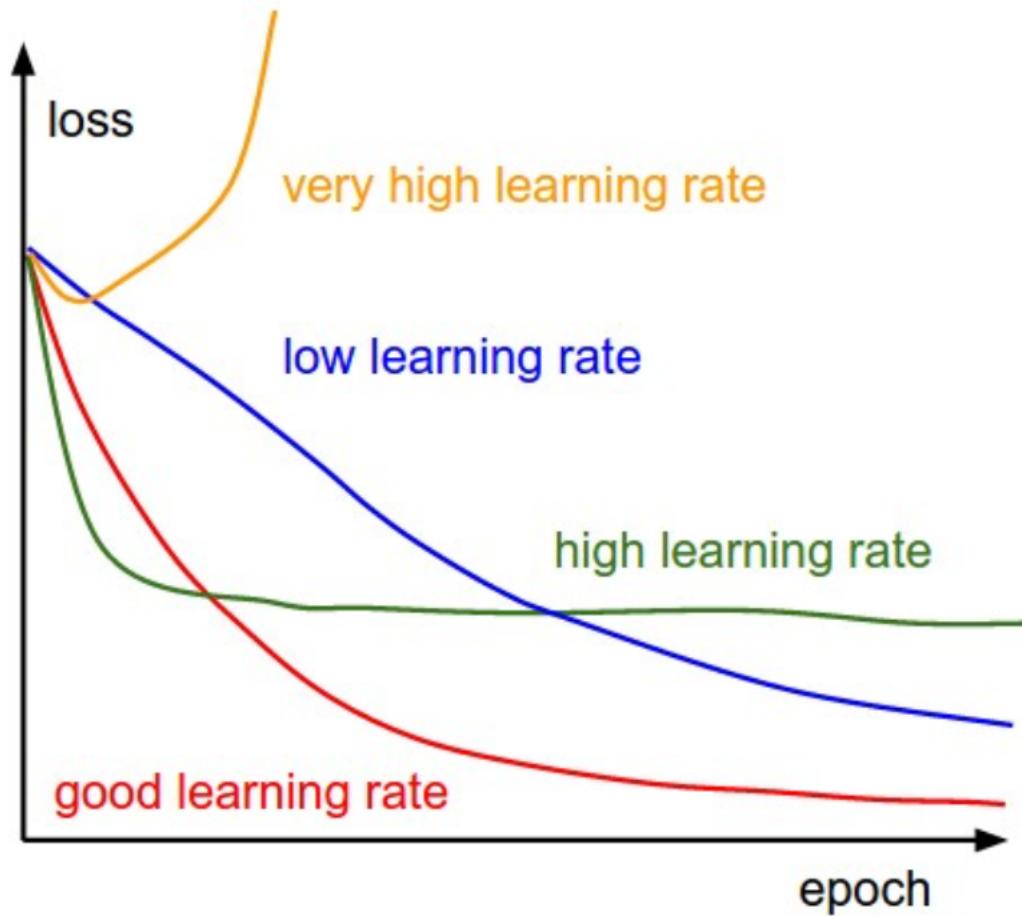
$$\sigma_i^2 := \alpha \cdot \text{variance}_{batch} + (1 - \alpha) \cdot \sigma_i^2$$

# Transfer learning и fine-tuning

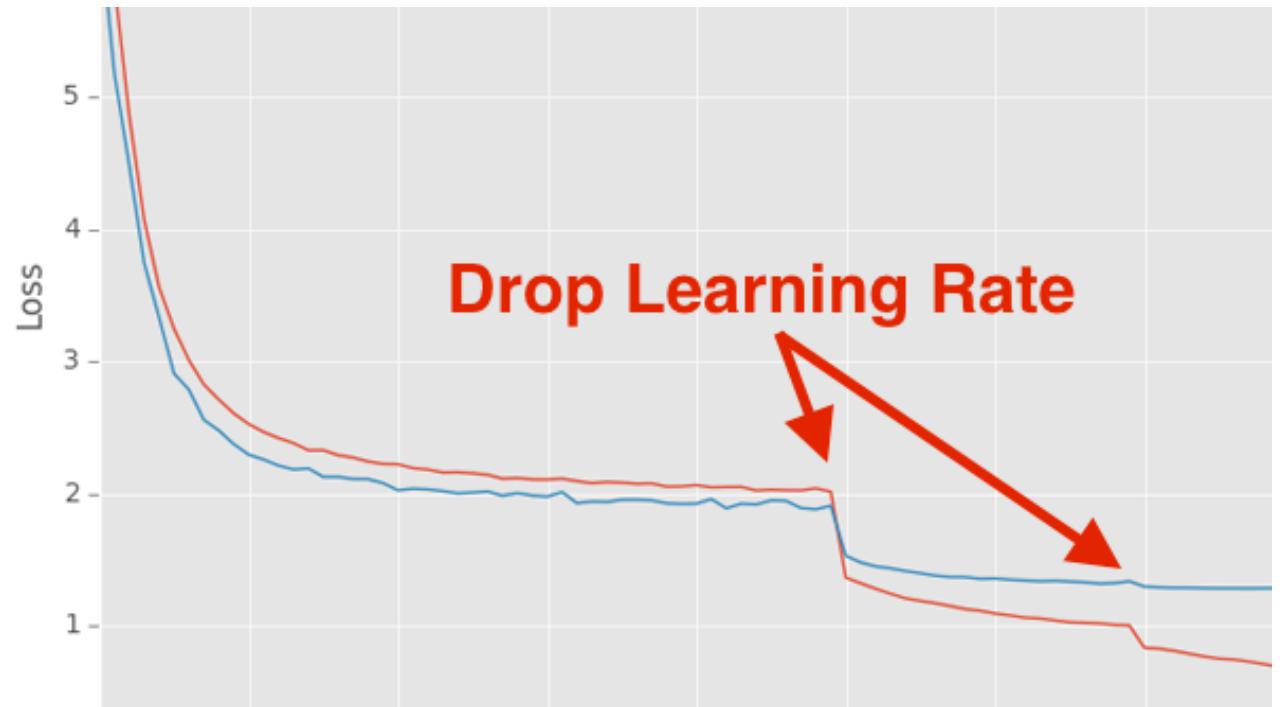
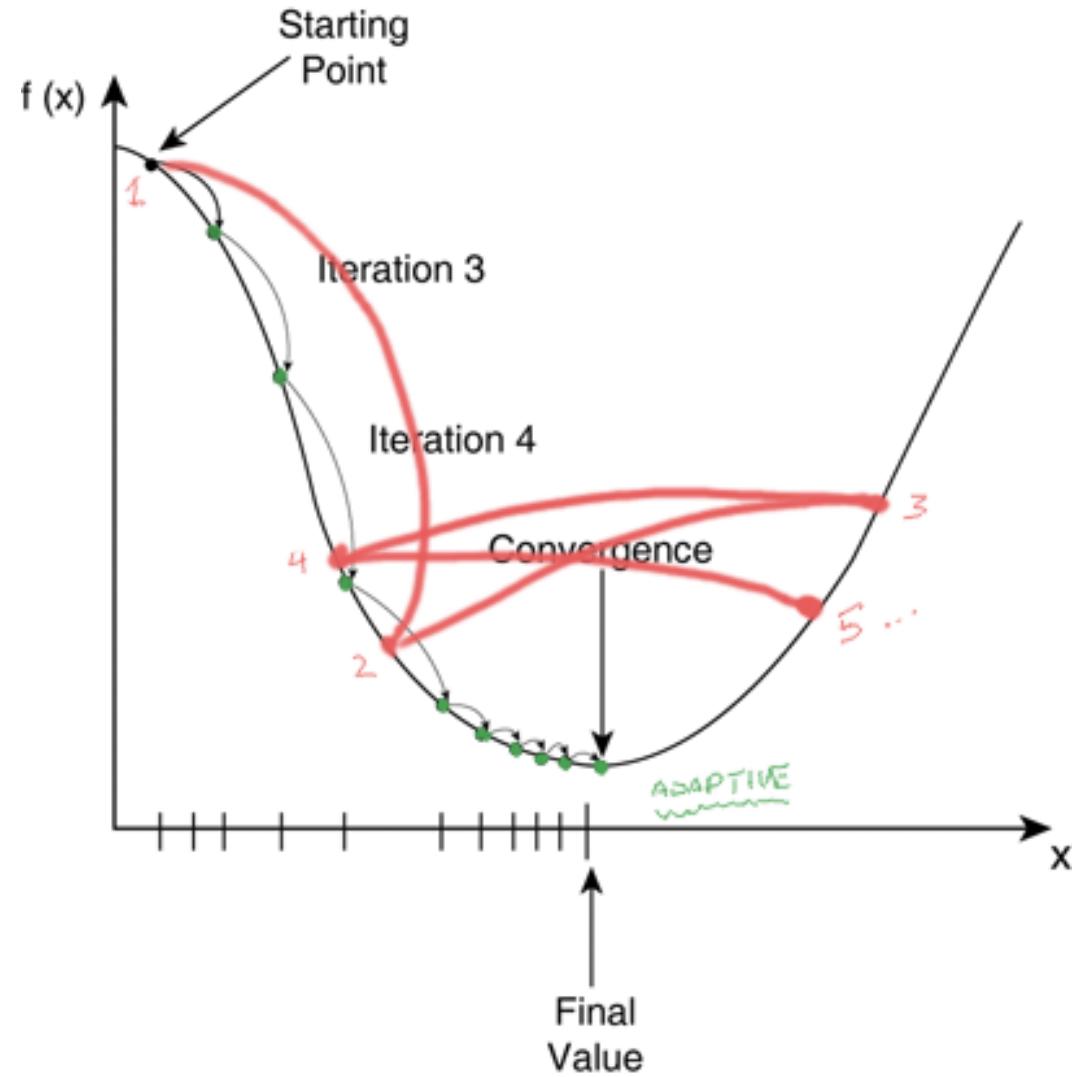
- Берем сеть, обученную на другую задачу
- Фильтры могут быть полезны и для нашей задачи
- Настраиваем только более глубокие слои
- Затем настраиваем с маленьким шагом всю сеть



# Learning rate



# Learning rate decay



# Разновидности SGD

- Обычный SGD

$$w^{k+1} = w^k - \alpha \nabla f(w^k)$$

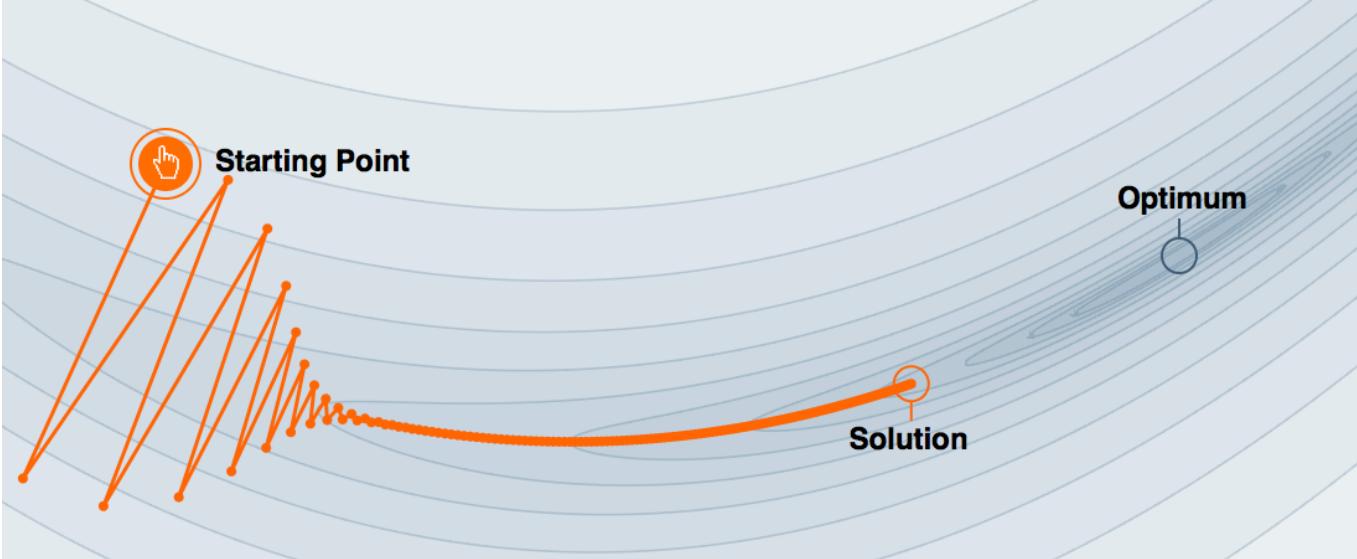
- SGD с моментами

Почти экспоненциальное  
сглаживание градиента

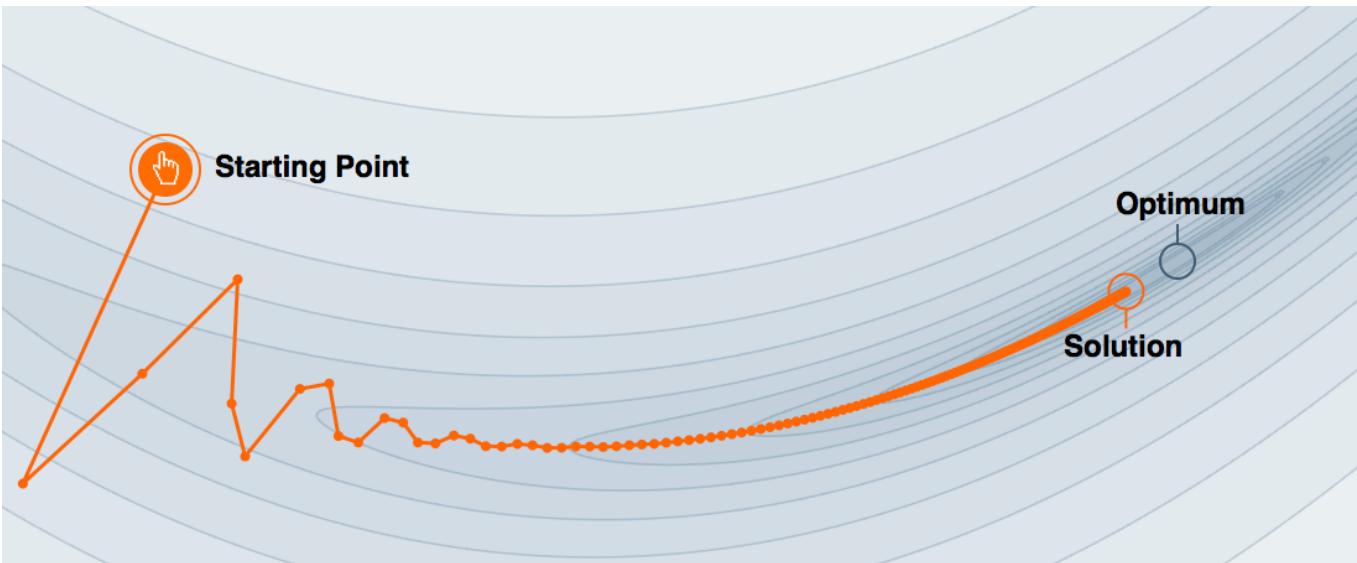

$$\begin{aligned} z^{k+1} &= \beta z^k + \nabla f(w^k) \\ w^{k+1} &= w^k - \alpha z^{k+1} \end{aligned}$$

# SGD с моментами

Обычный SGD



SGD с моментами



# RMSprop

Экспоненциальное  
сглаживание квадрата  
градиента

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2.$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t.$$

# Ссылки

- <http://cs231n.github.io/convolutional-networks/>
- <https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/>