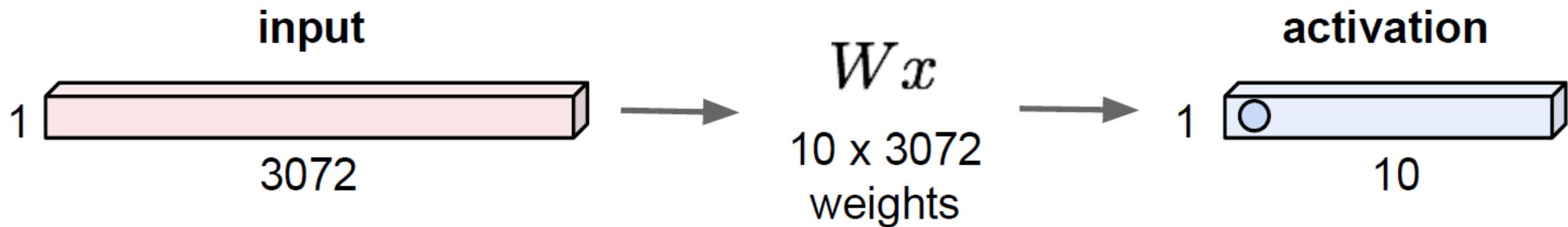


# Свёрточные Нейронные Сети

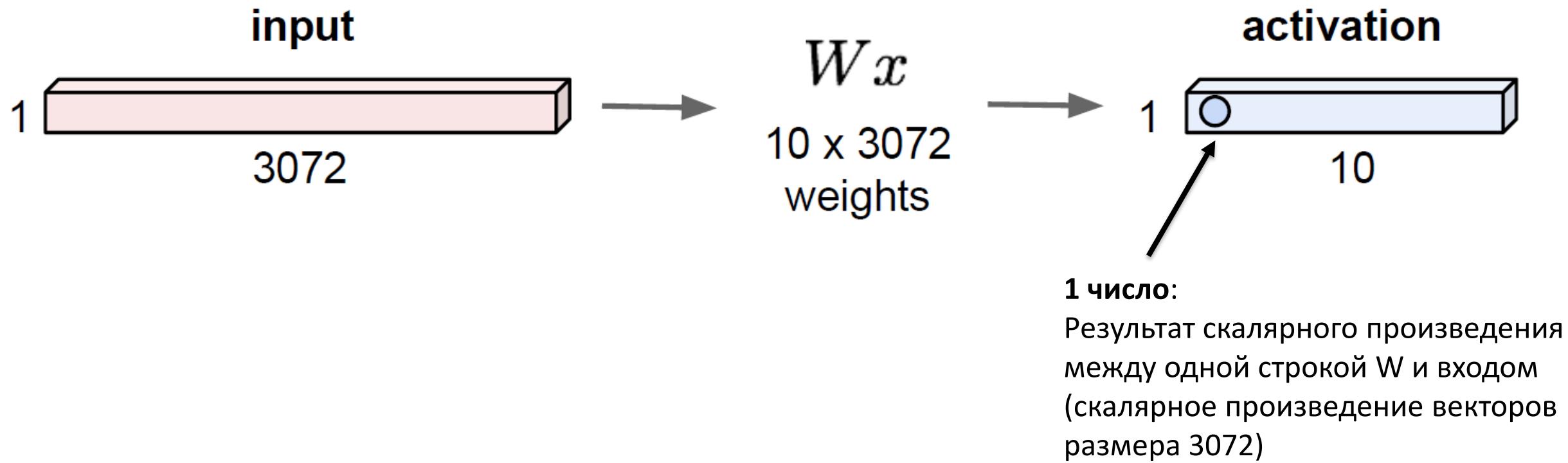
## Полносвязный (Fully-Connected) слой

Картинка 32x32x3 -> вытягиваем в 3072x1



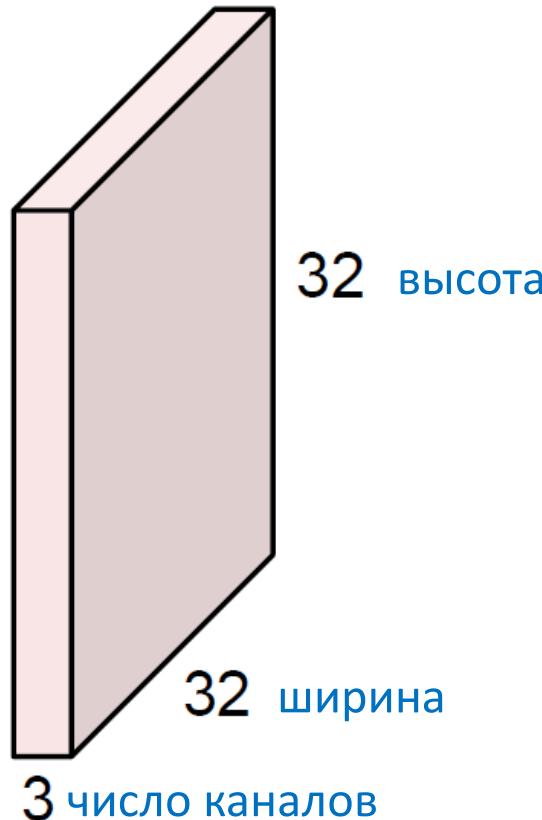
# Полносвязный (Fully-Connected) слой

Картинка 32x32x3 -> вытягиваем в 3072x1



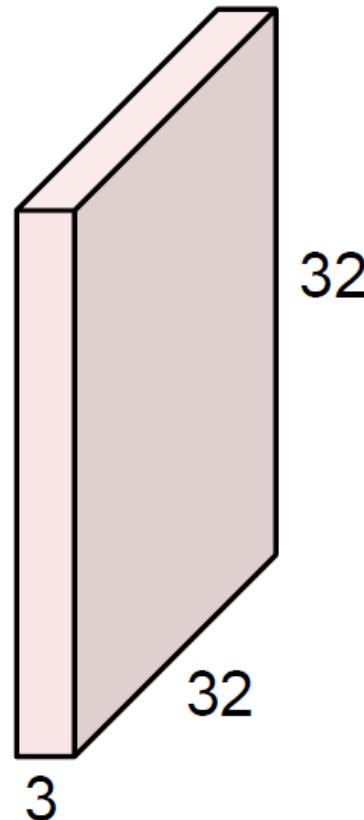
# Свёрточный слой

Картина 32x32x3 -> сохраняем пространственную структуру



# Свёрточный слой

Картина  $32 \times 32 \times 3$



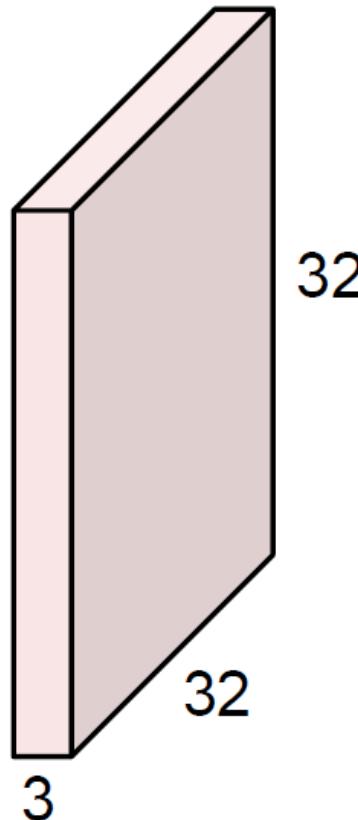
Фильтр  $5 \times 5 \times 3$



**Сворачиваем фильтр с изображением, т.е. перемещаем его вдоль всего изображения, вычисляя для каждого положения скалярное произведение**

# Свёрточный слой

Картина  $32 \times 32 \times 3$



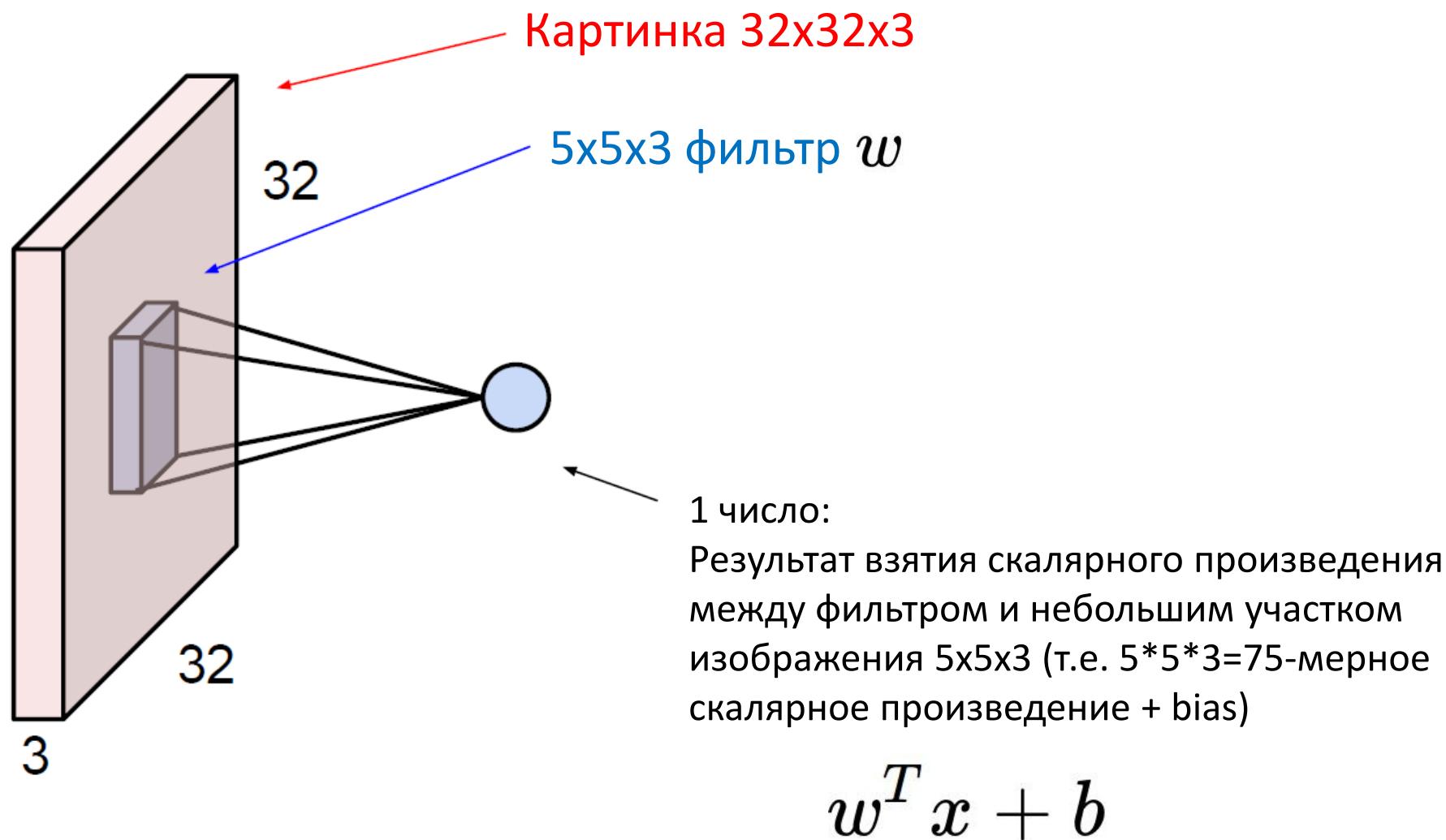
Фильтр  $5 \times 5 \times 3$



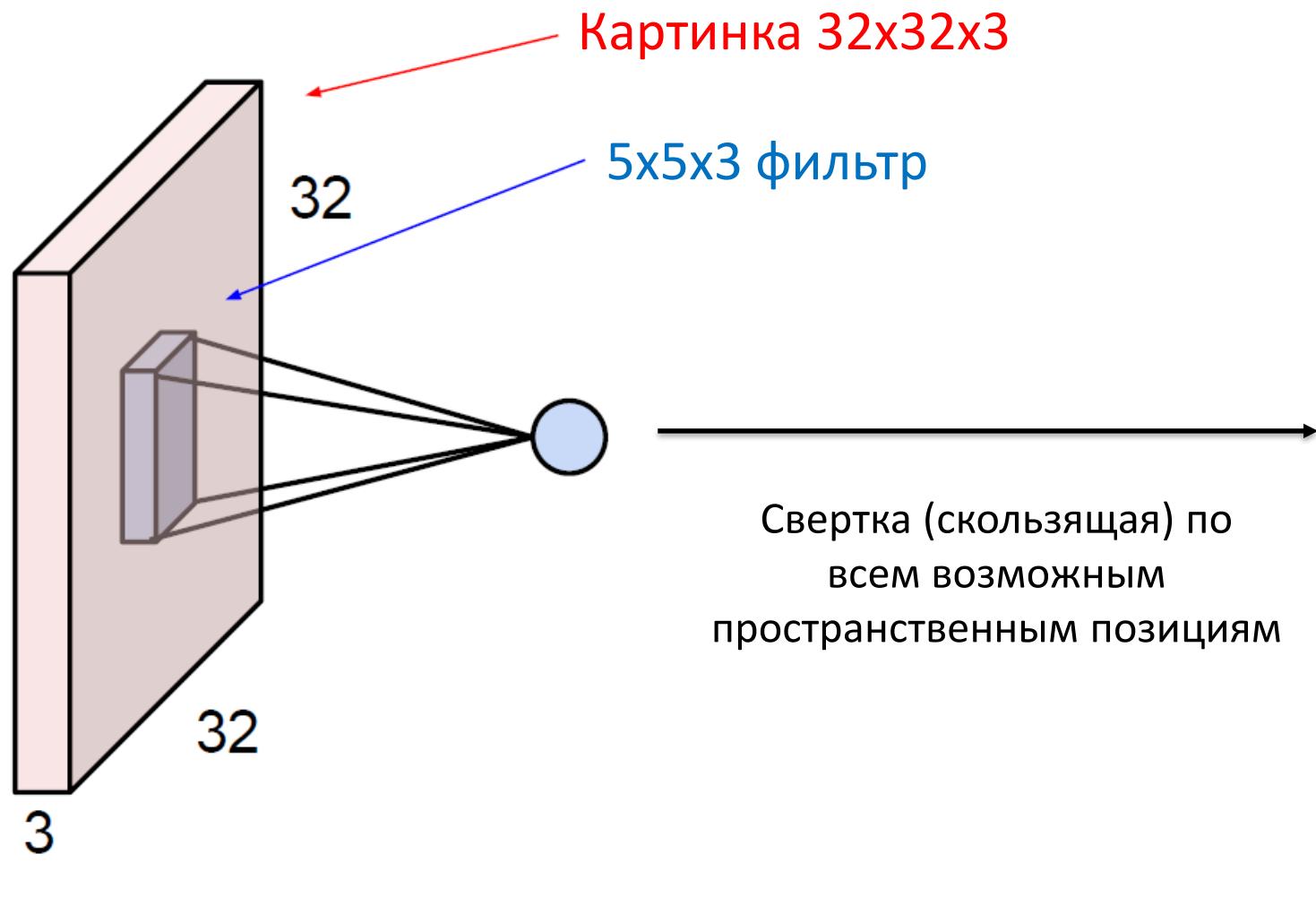
Фильтры всегда проходят через  
все каналы входного тензора

**Сворачиваем** фильтр с  
изображением, т.е. перемещаем  
его вдоль всего изображения,  
вычисляя для каждого  
положения скалярное  
произведение

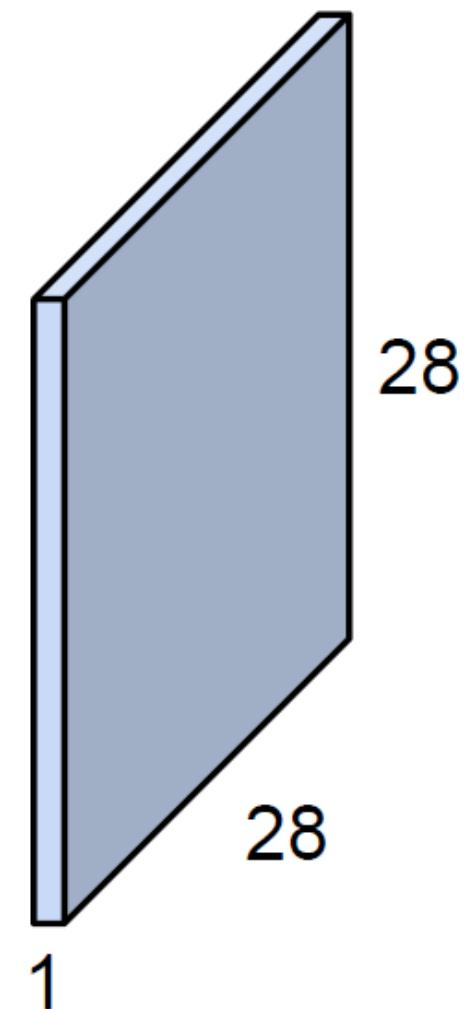
# Свёрточный слой



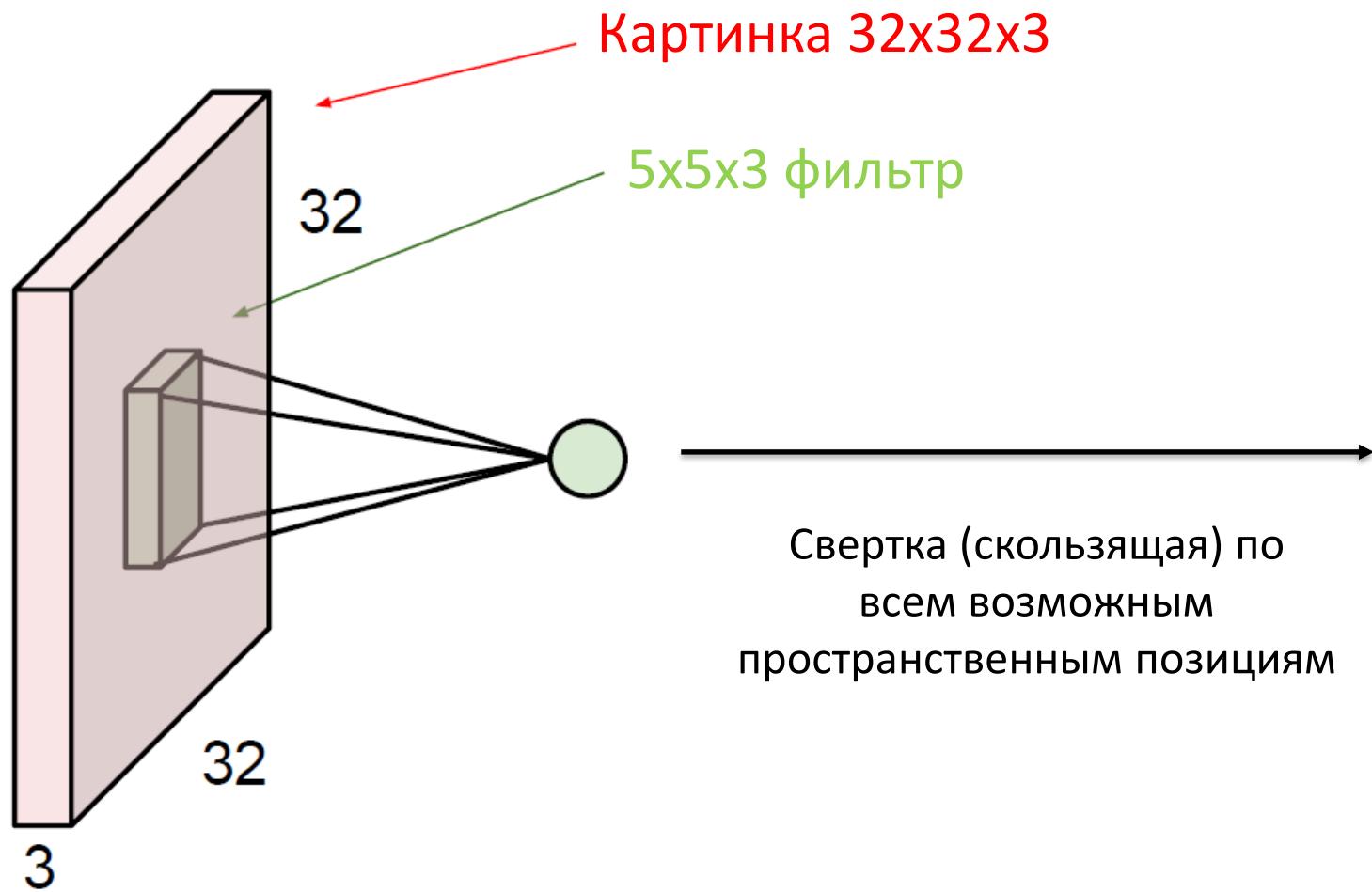
# Свёрточный слой



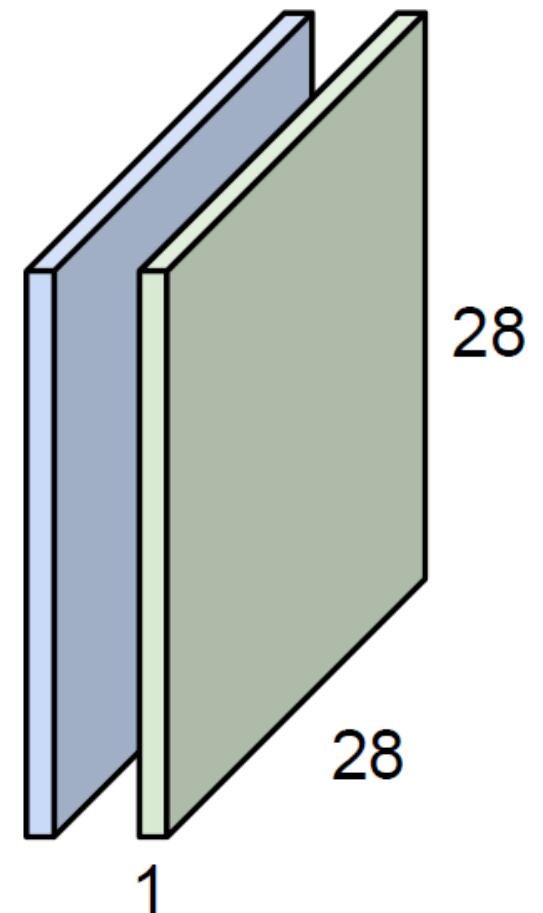
Карта активации



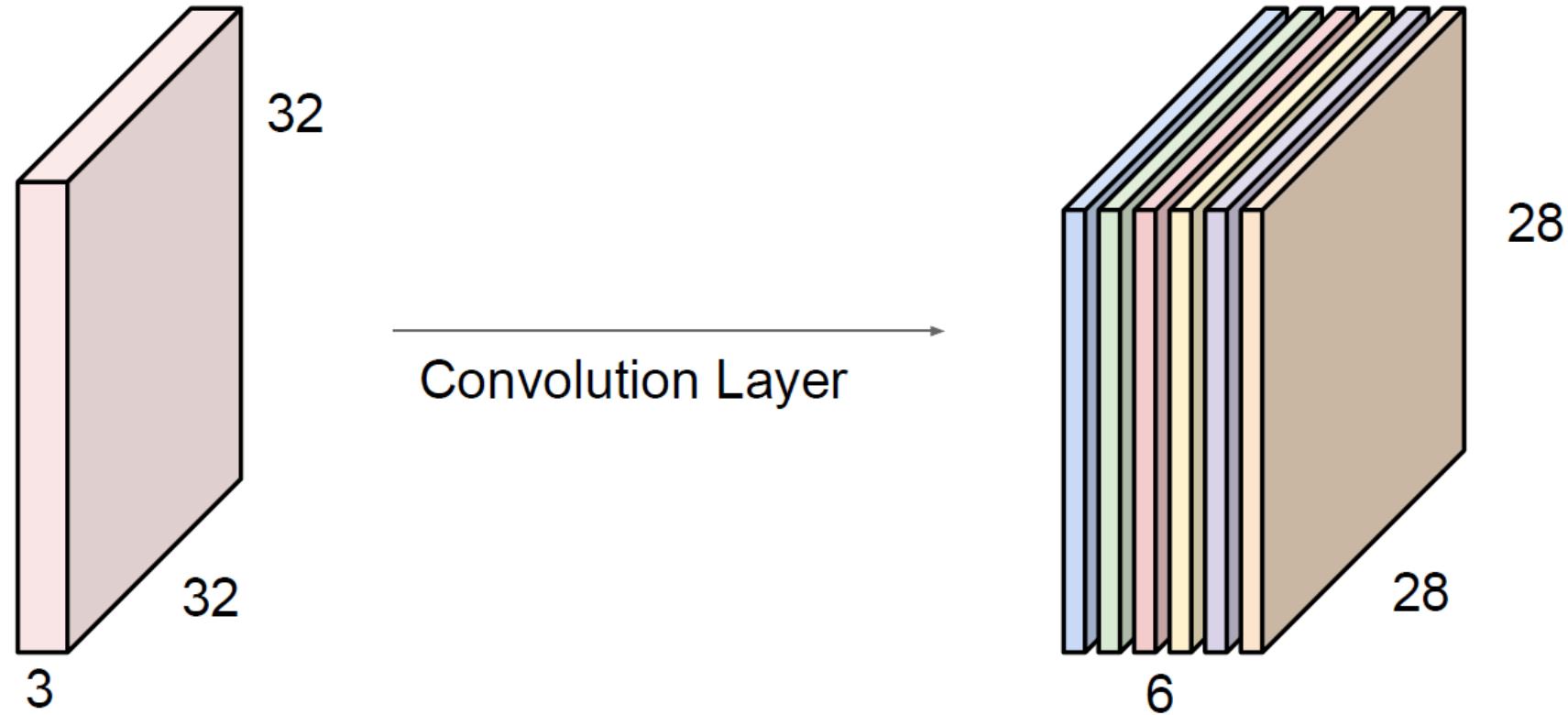
# Свёрточный слой



Карты активации

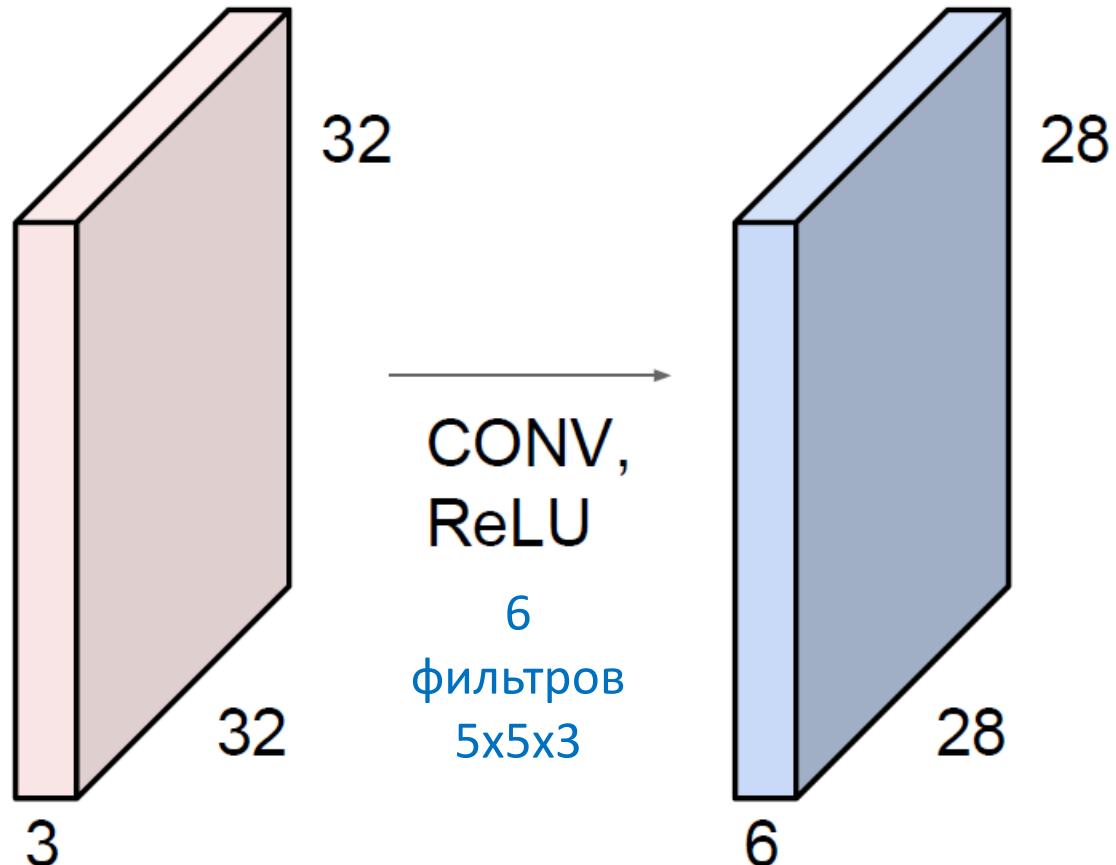


Например, если у нас 6 фильтров  $5 \times 5$ , мы получим 6 различных карт активации:

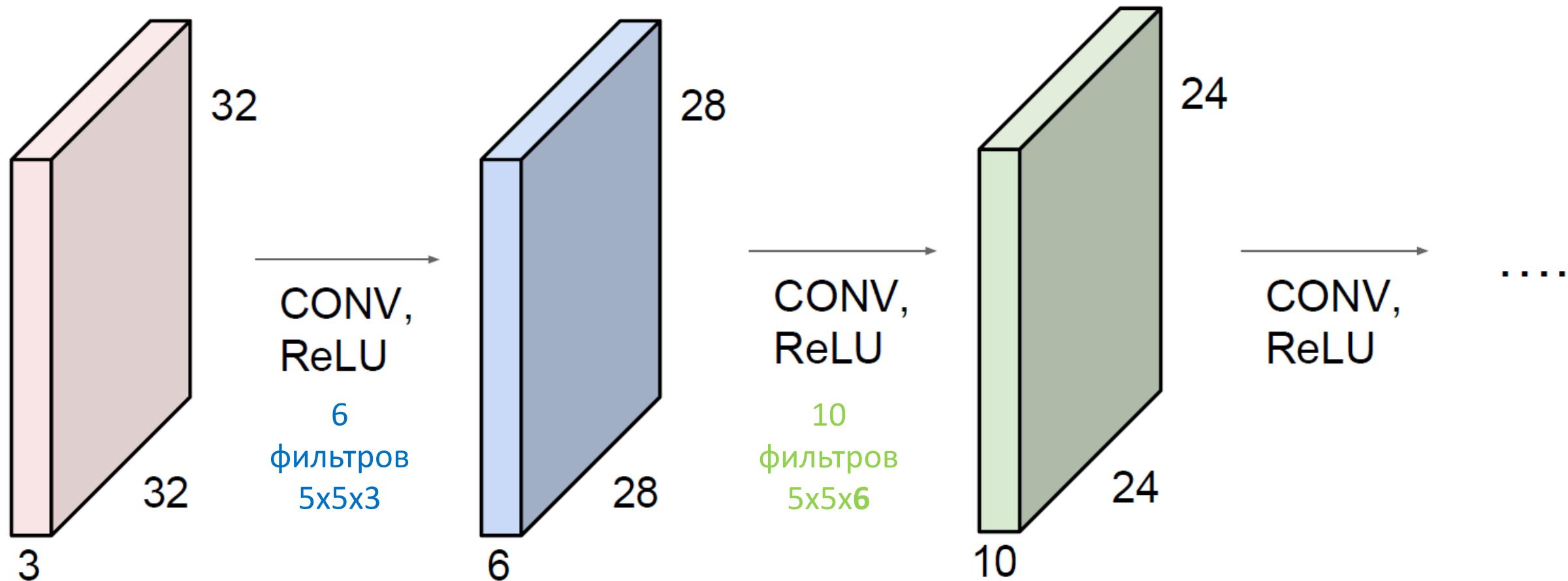


Мы объединяем их вместе, чтобы получить новую «картинку»  $28 \times 28 \times 6$ !

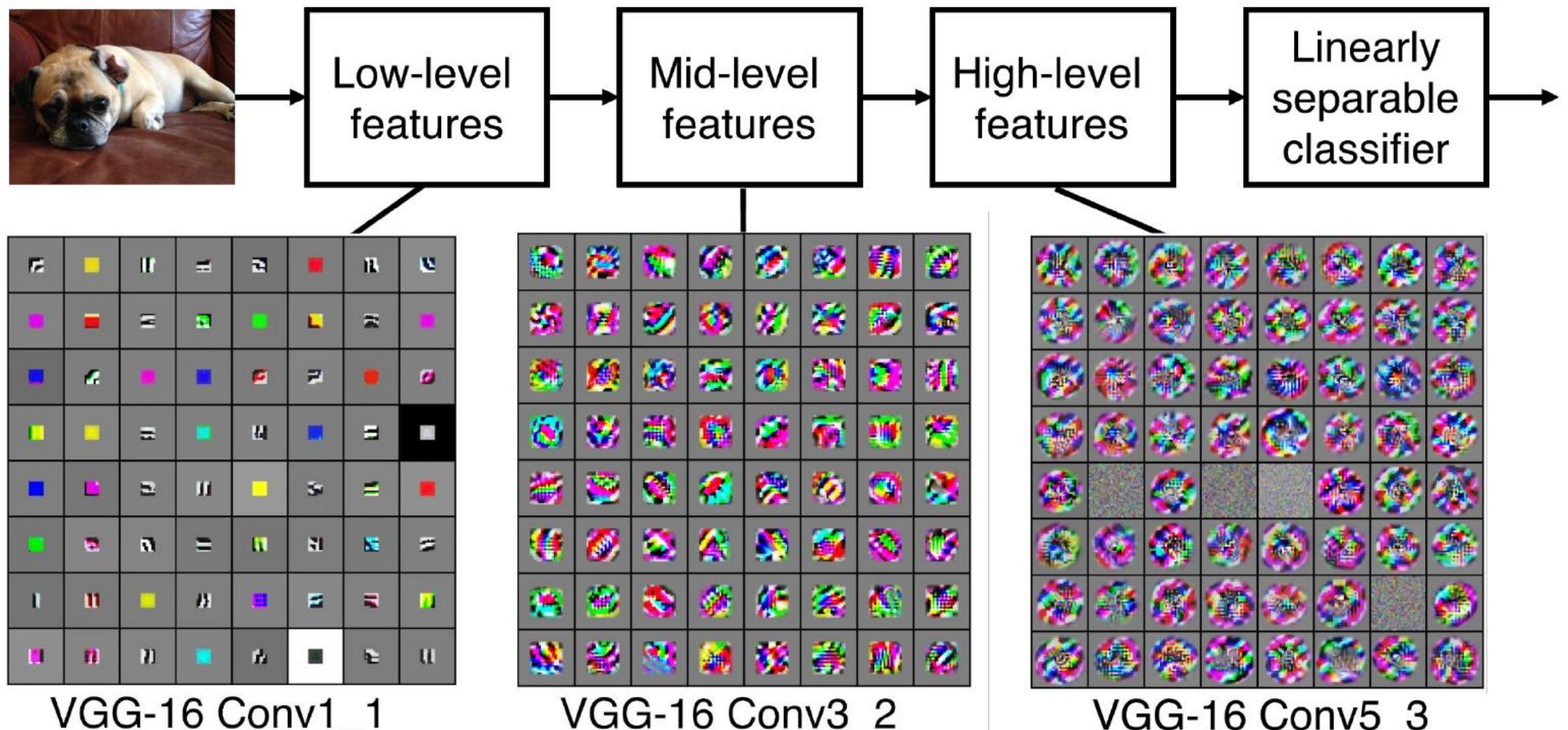
**Спойлер:** Свёрточная сеть – это последовательность свёрточных слоёв, чередующихся со слоями активации

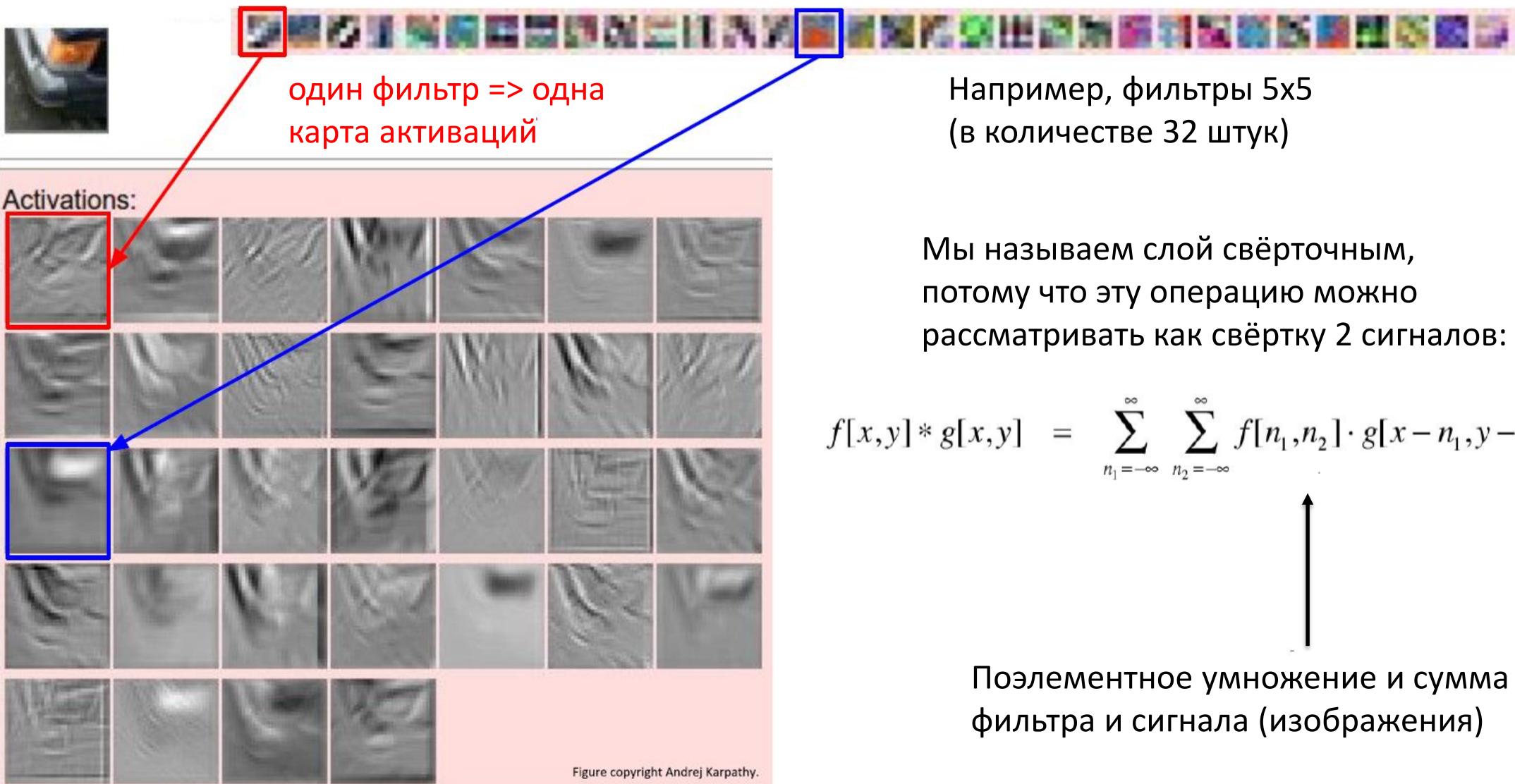


**Спойлер:** Свёрточная сеть – это последовательность свёрточных слоёв, чередующихся со слоями активации

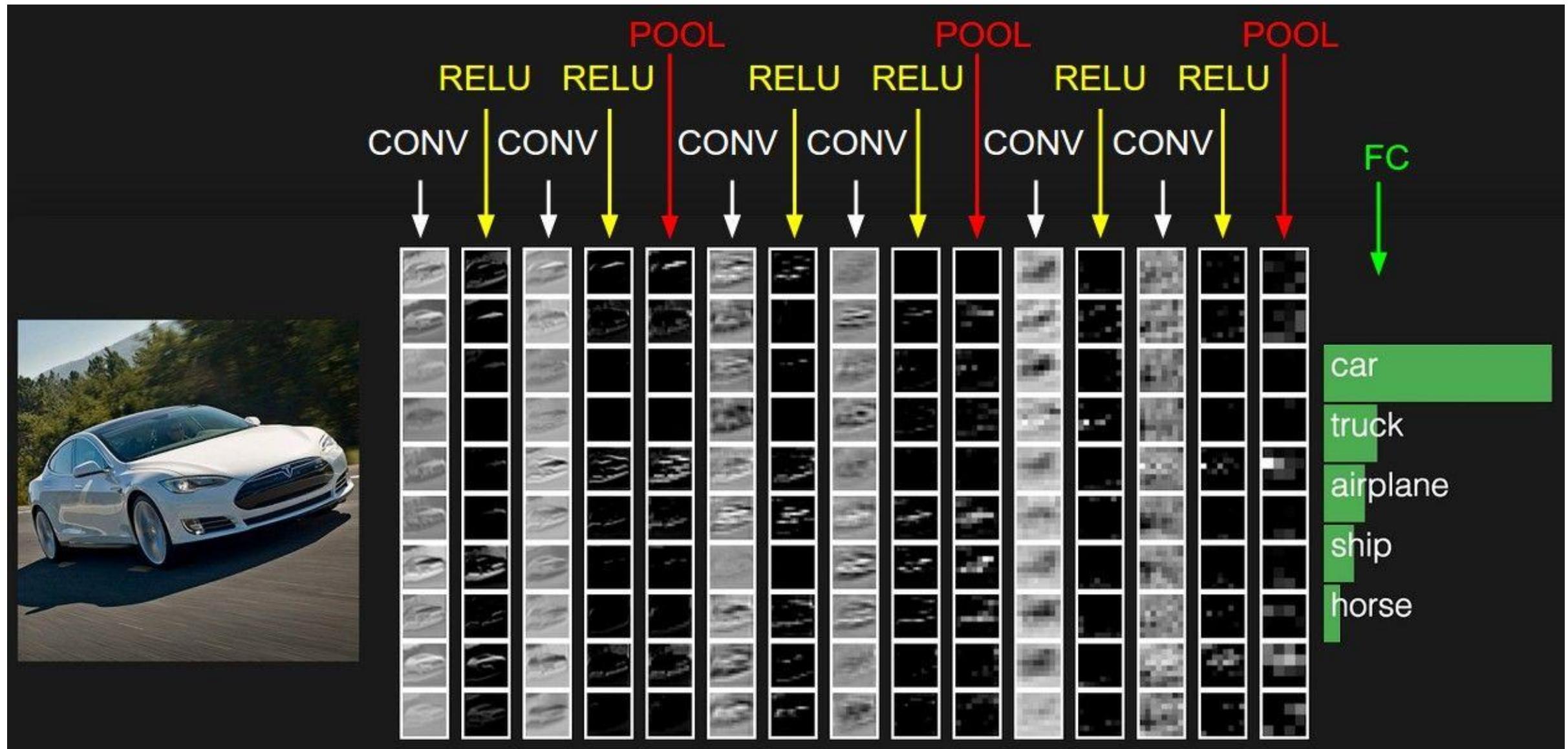


[Zeiler, Fergus, 2013]



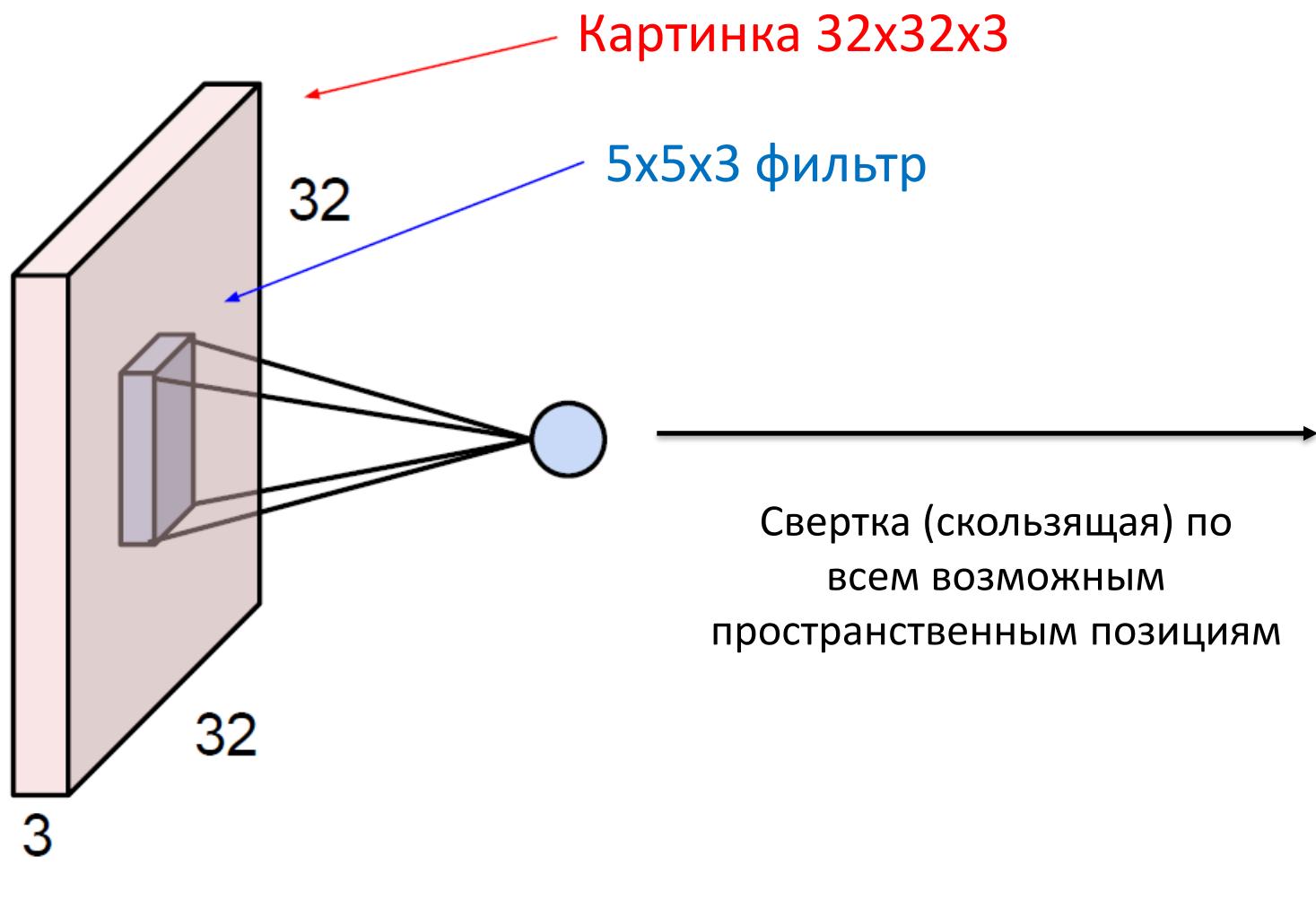


Спойлер:



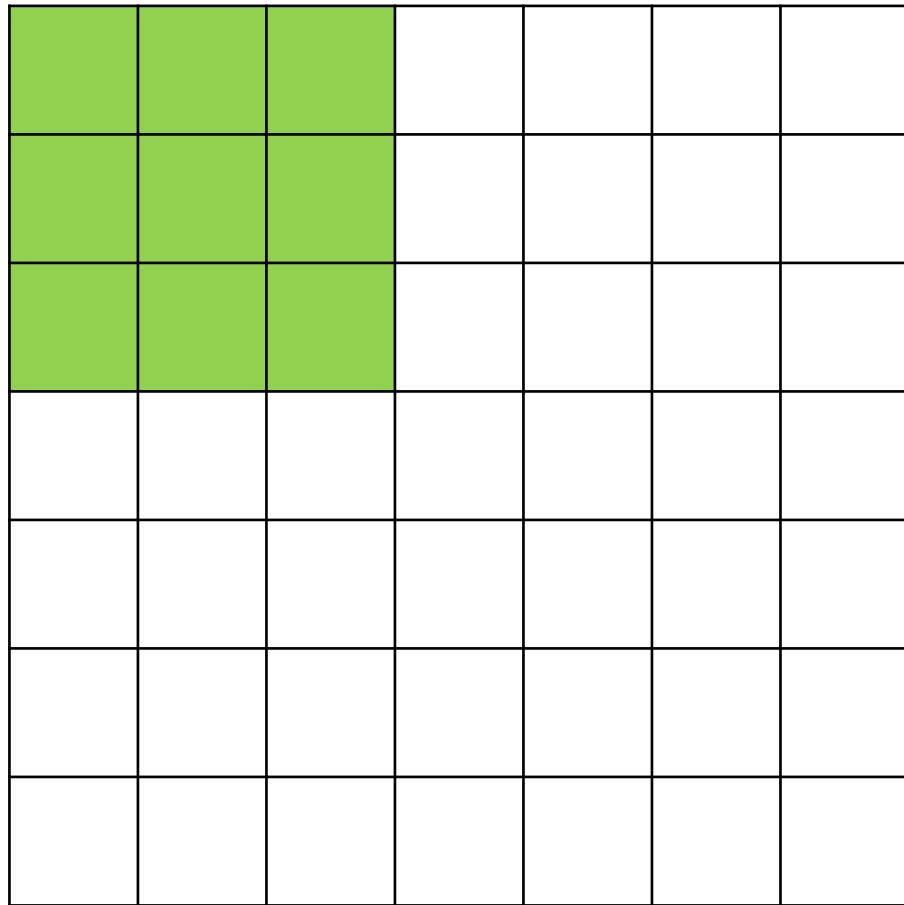
# Посмотрим ближе на пространственные размерности

Карта активации



# Посмотрим ближе на пространственные размерности

7



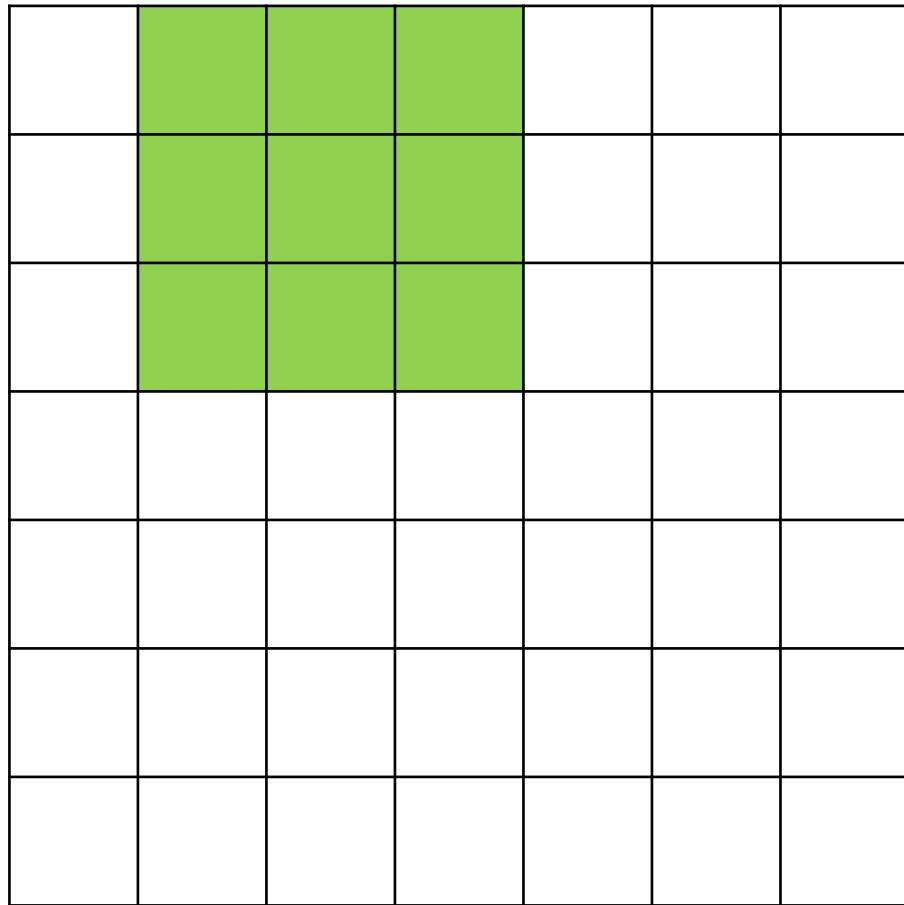
7

Вход  $7 \times 7$  (по  
пространственным  
координатам)

Фильтр  $3 \times 3$

# Посмотрим ближе на пространственные размерности

7



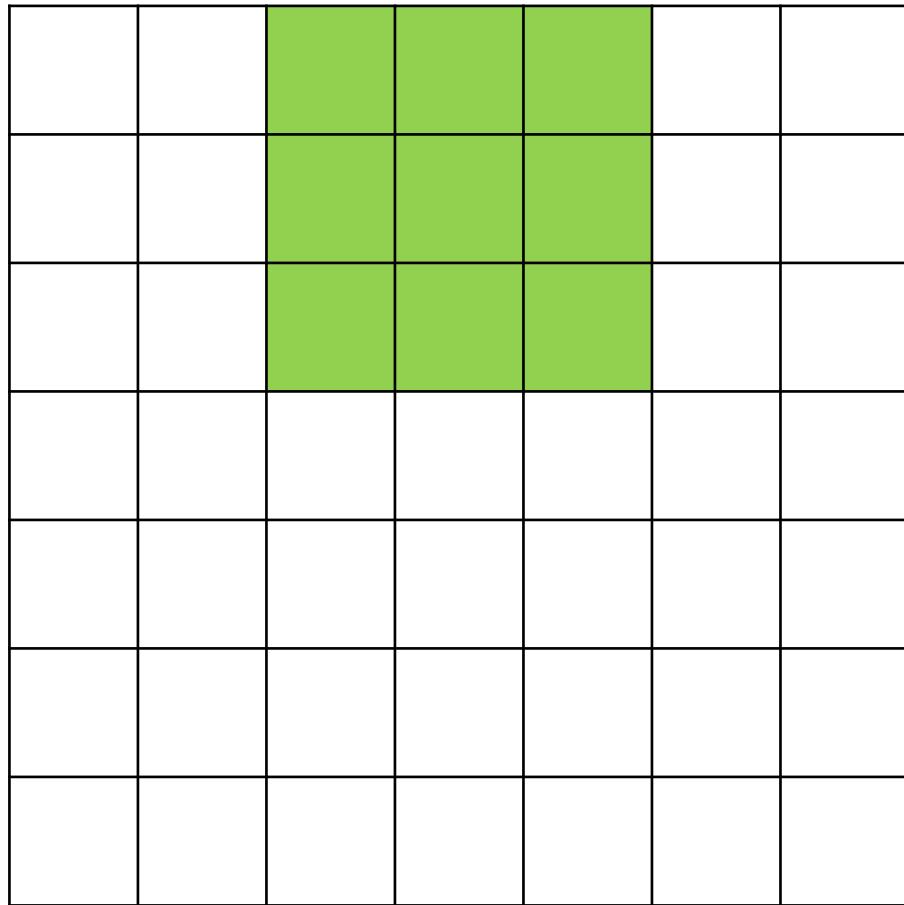
7

Вход  $7 \times 7$  (по  
пространственным  
координатам)

Фильтр  $3 \times 3$

# Посмотрим ближе на пространственные размерности

7



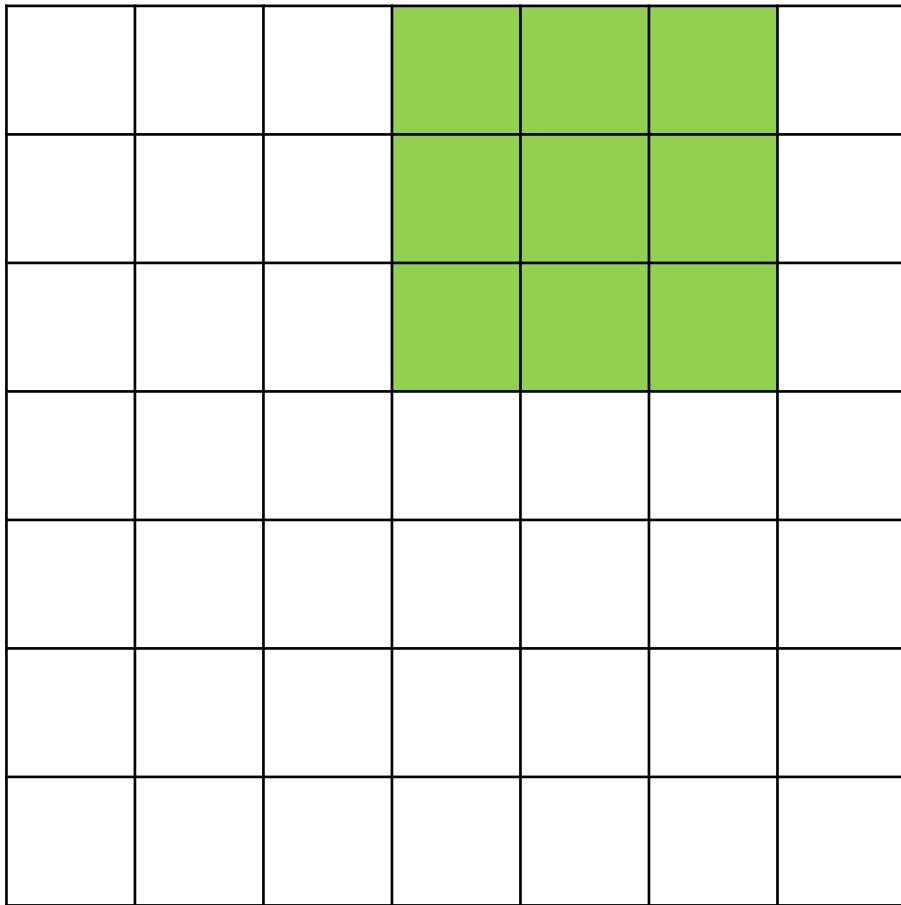
7

Вход  $7 \times 7$  (по  
пространственным  
координатам)

Фильтр  $3 \times 3$

# Посмотрим ближе на пространственные размерности

7



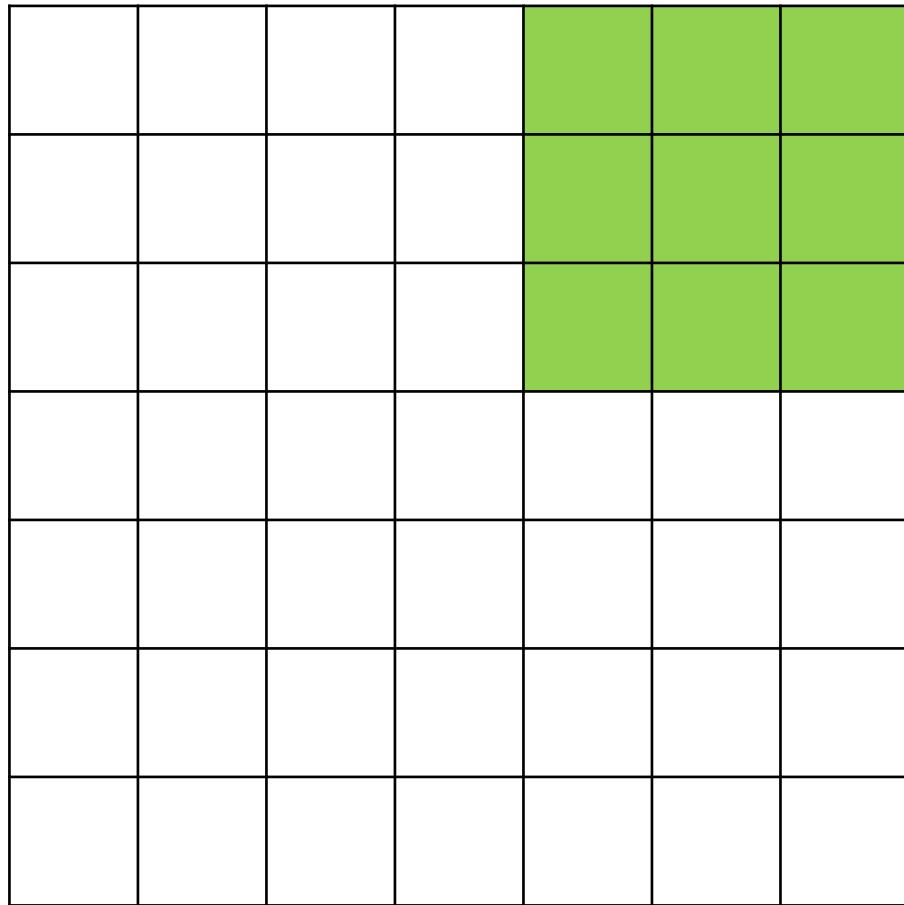
7

Вход  $7 \times 7$  (по  
пространственным  
координатам)

Фильтр  $3 \times 3$

# Посмотрим ближе на пространственные размерности

7



7

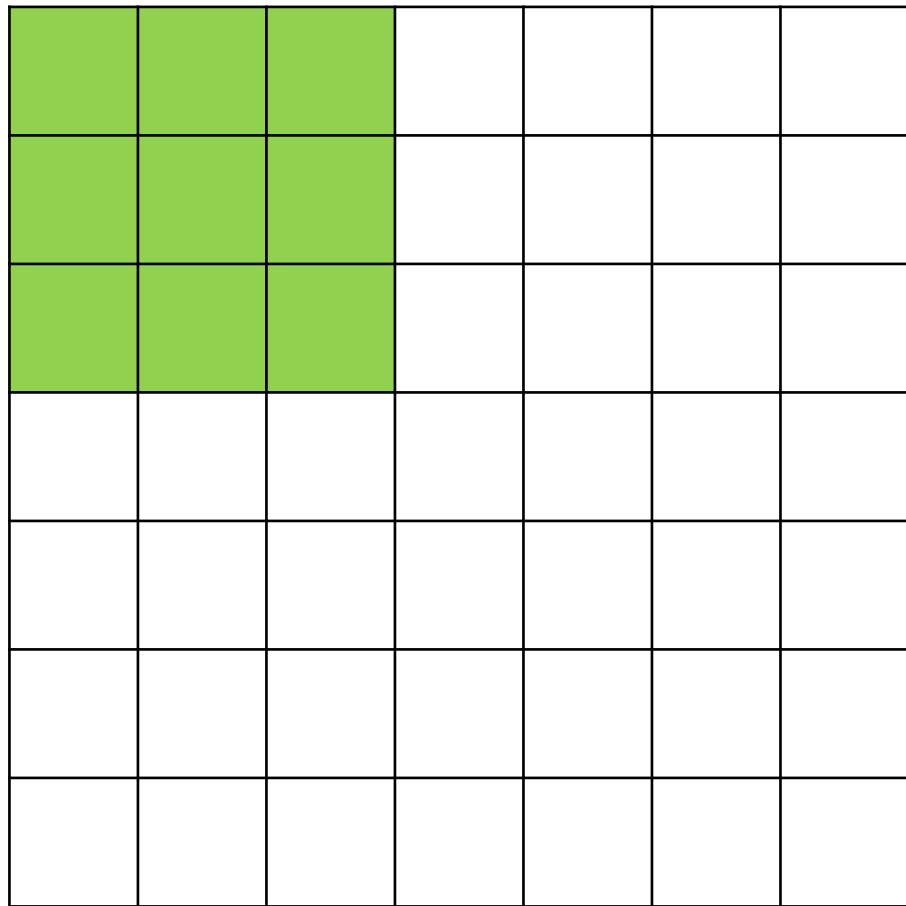
Вход  $7 \times 7$  (по  
пространственным  
координатам)

Фильтр  $3 \times 3$

=> Выход  $5 \times 5$

# Посмотрим ближе на пространственные размерности

7



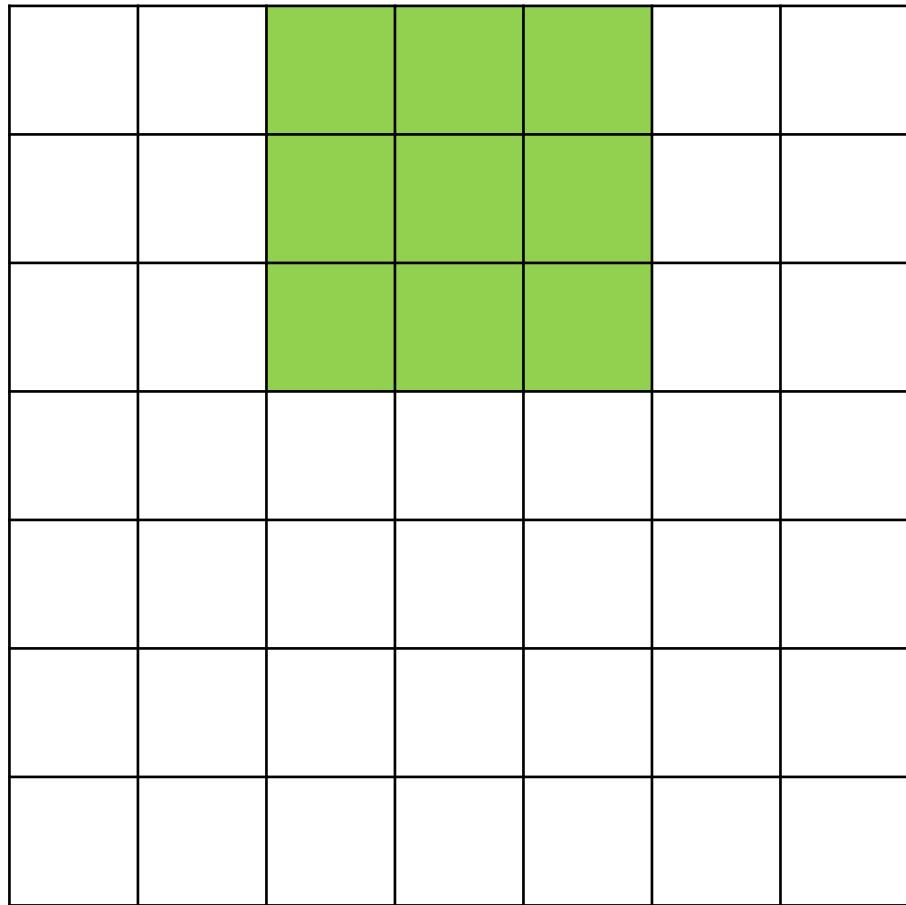
7

Вход  $7 \times 7$  (по пространственным координатам)

Фильтр  $3 \times 3$ , применяемый с  
**шагом (stride) 2**

# Посмотрим ближе на пространственные размерности

7

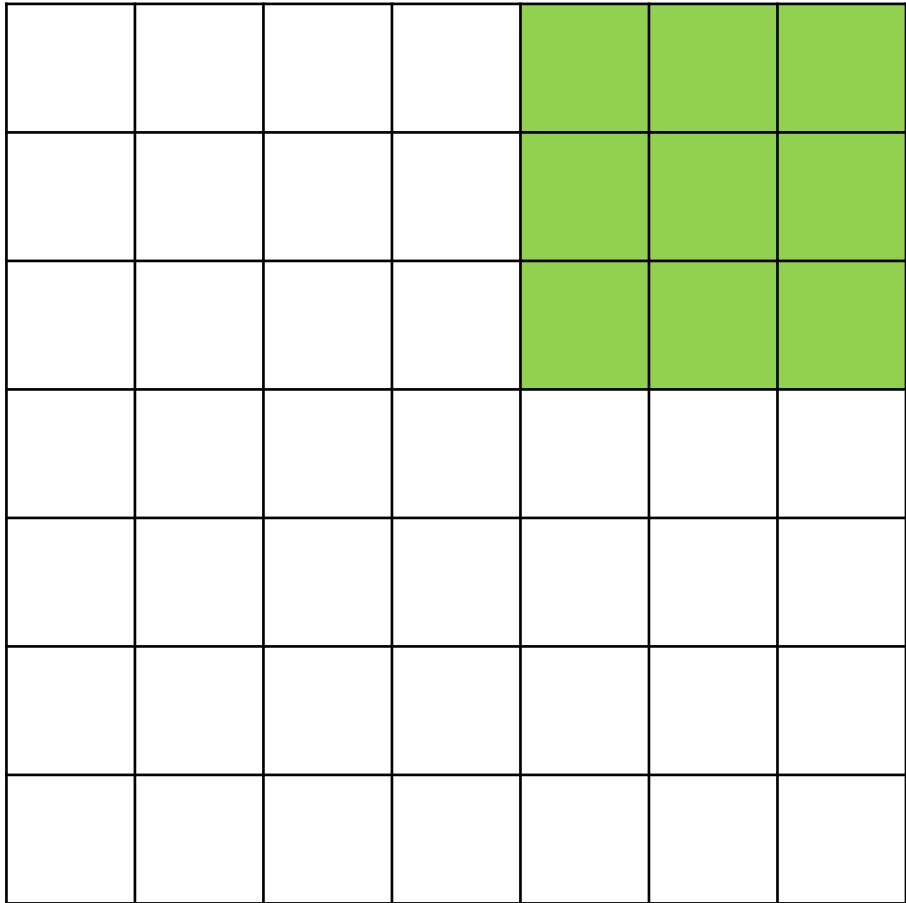


Вход  $7 \times 7$  (по пространственным координатам)

Фильтр  $3 \times 3$ , применяемый с  
**шагом (stride) 2**

# Посмотрим ближе на пространственные размерности

7



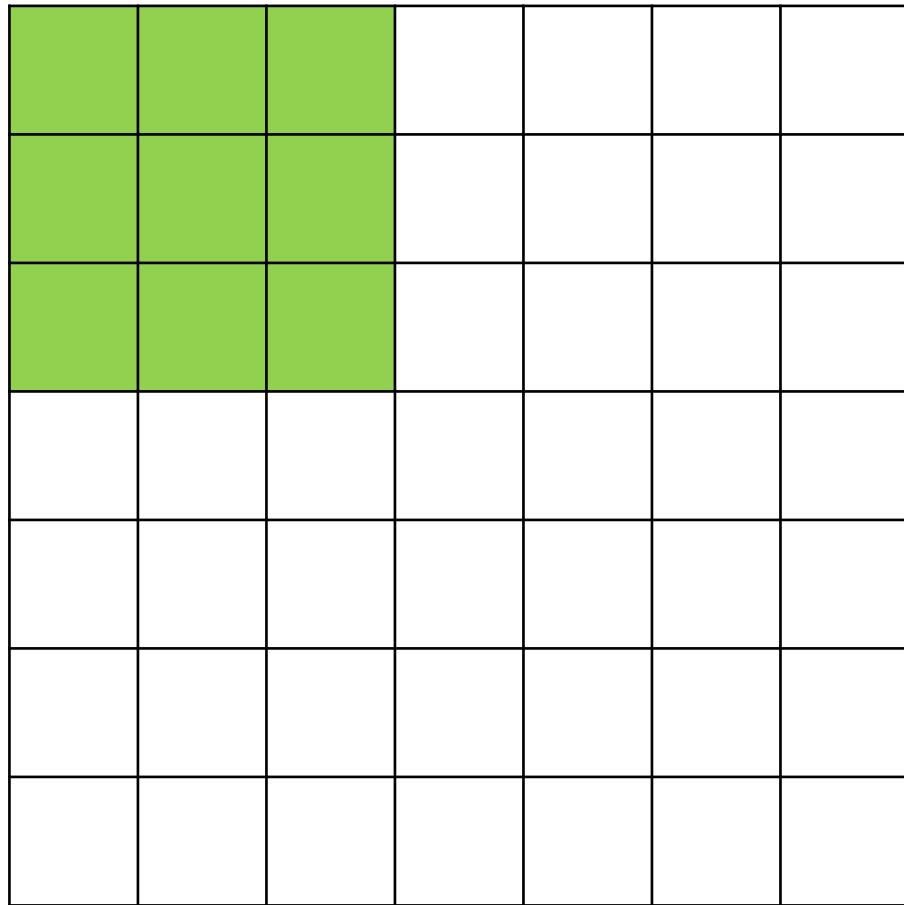
Вход  $7 \times 7$  (по пространственным координатам)

Фильтр  $3 \times 3$ , применяемый с  
**шагом (stride) 2**

=> Вход  $3 \times 3$ !

# Посмотрим ближе на пространственные размерности

7

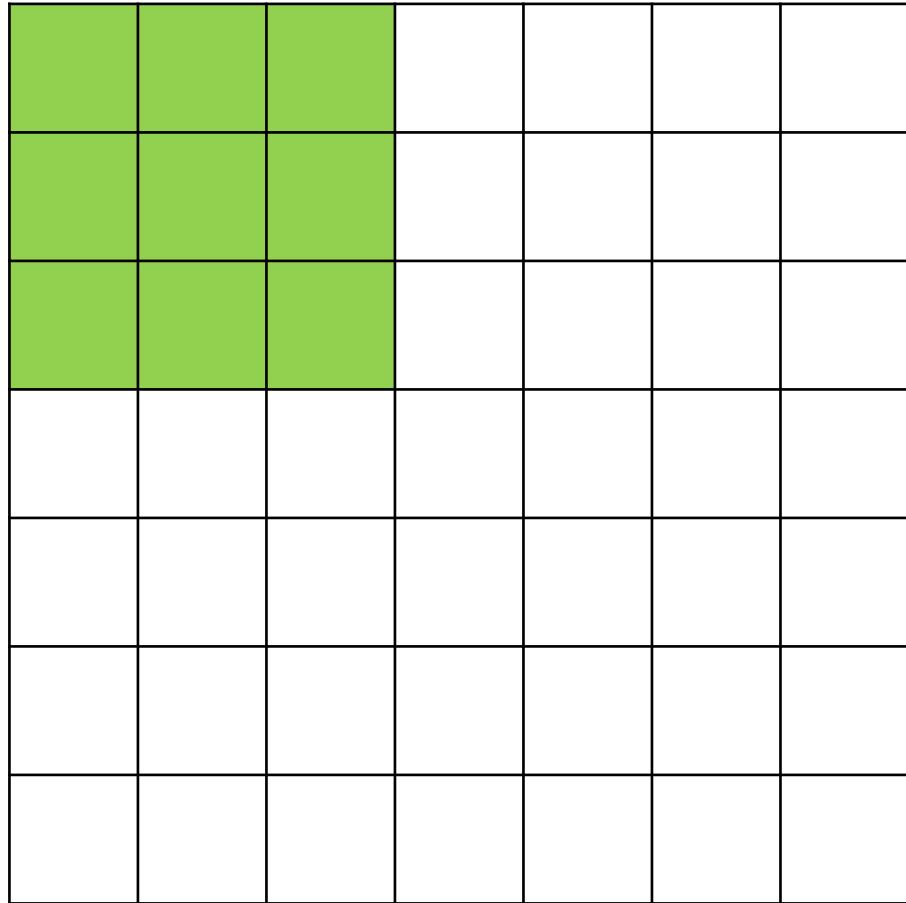


7

Вход  $7 \times 7$  (пространственно)  
Что будет если применить  
**фильтр  $3 \times 3$  с шагом 3?**

# Посмотрим ближе на пространственные размерности

7

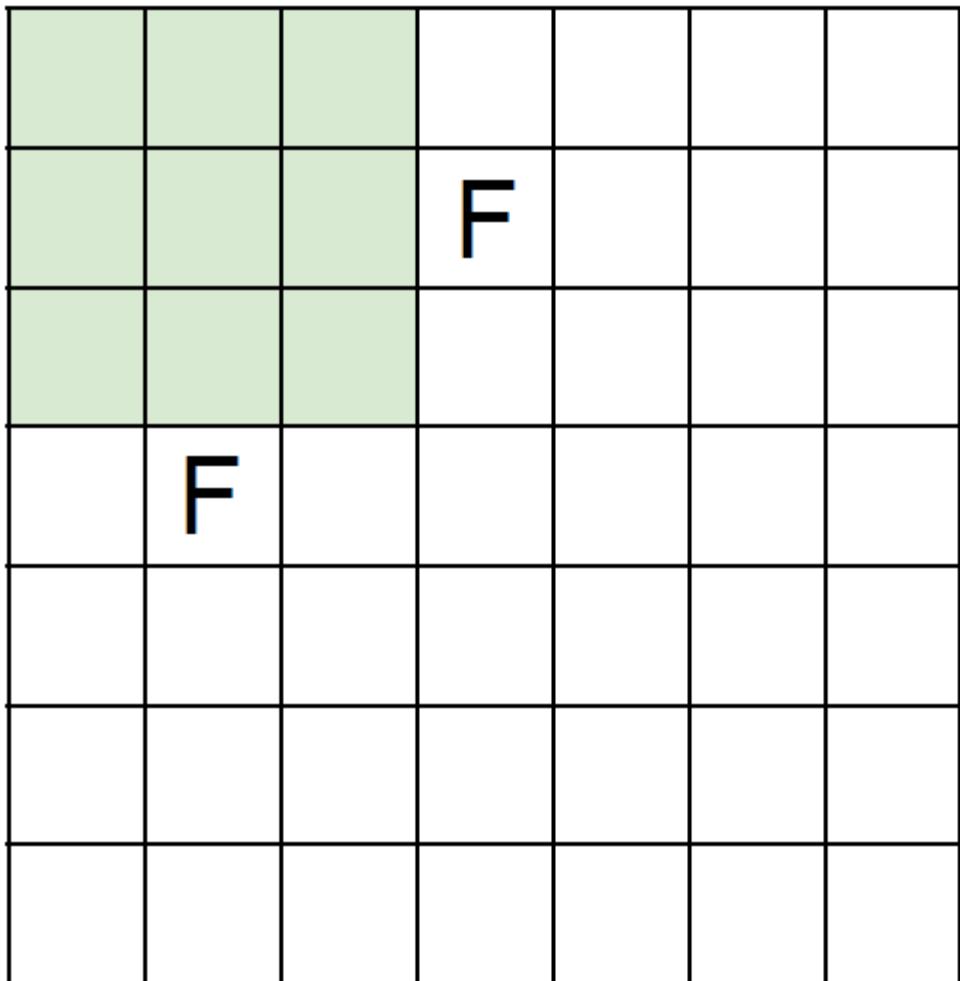


7

Вход  $7 \times 7$  (пространственно)  
Что будет если применить  
фильтр  $3 \times 3$  с шагом 3?

Не поместится!  
Нельзя применить фильтр  $3 \times 3$   
ко входу  $7 \times 7$  с шагом 3

N



Размер выхода:

$$(N - F) / \text{stride} + 1$$

Например  $N=7$ ,  $F=3$

$$\text{Stride 1} \Rightarrow (7-3)/1 + 1 = 5$$

$$\text{Stride 2} \Rightarrow (7-3)/2 + 1 = 3$$

$$\text{Stride 3} \Rightarrow (7-3)/3 + 1 = 2.33 : \backslash$$

N

## На практике: добавляем нулевой padding на границах

0	0	0	0	0	0			
0								
0								
0								
0								

Например, вход размера 7x7

Фильтр 3x3, применяемый с шагом 1

**Паддим границу на 1 пиксель => какой размер выхода?**

Напоминание:  
 $(N - F) / \text{stride} + 1$

## На практике: добавляем нулевой padding на границах

0	0	0	0	0	0			
0								
0								
0								
0								

Например, вход размера  $7 \times 7$

Фильтр  $3 \times 3$ , применяемый с шагом 1

**Паддим границу на 1 пиксель => какой размер выхода?**

**Выход  $7 \times 7$ !**

## На практике: добавляем нулевой padding на границах

0	0	0	0	0	0			
0								
0								
0								
0								

Например, вход размера  $7 \times 7$

Фильтр  $3 \times 3$ , применяемый с шагом 1

**Паддим границу на 1 пиксель => какой размер выхода?**

**Выход  $7 \times 7$ !**

В общем случае очень часто применяют свёрточные слои со stride 1, фильтрами размера  $F \times F$ , и zero-padding размера  $(F-1)/2$  (для сохранения размера входа)

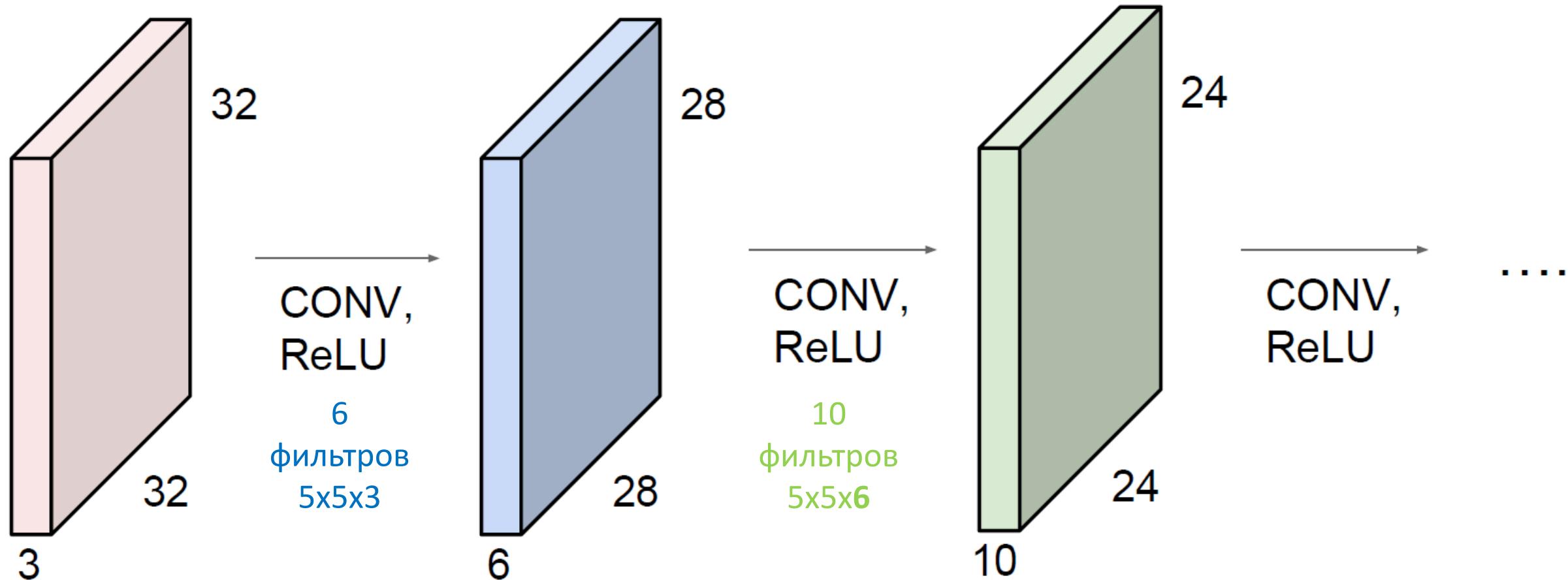
Например  $F=3 \Rightarrow$  zero pad 1

$F=5 \Rightarrow$  zero pad 2

$F=7 \Rightarrow$  zero pad 3

# Вернёмся к старой архитектуре...

Здесь вход размера 32x32 последовательно сворачивался фильтрами 5x5, уменьшая размер входа на каждом шаге (32->28->24...). Быстрое уменьшение работает не всегда хорошо.

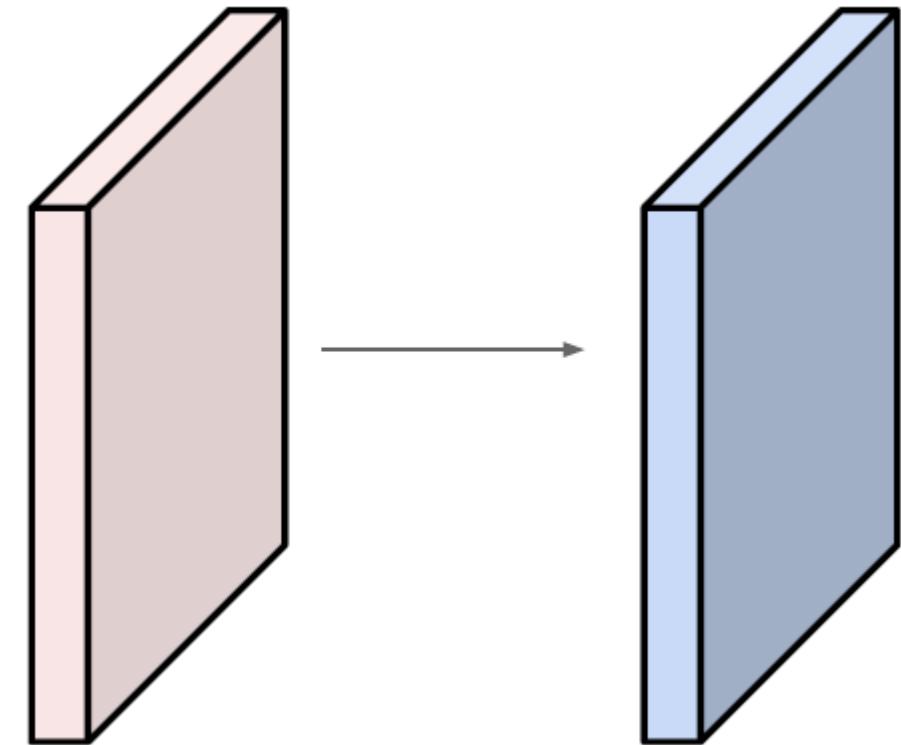


Время примеров:

Входной массив: **32x32x3**

10 фильтров 5x5 с шагом 1,  
padding-ом 2

Размер выходного массива: ?



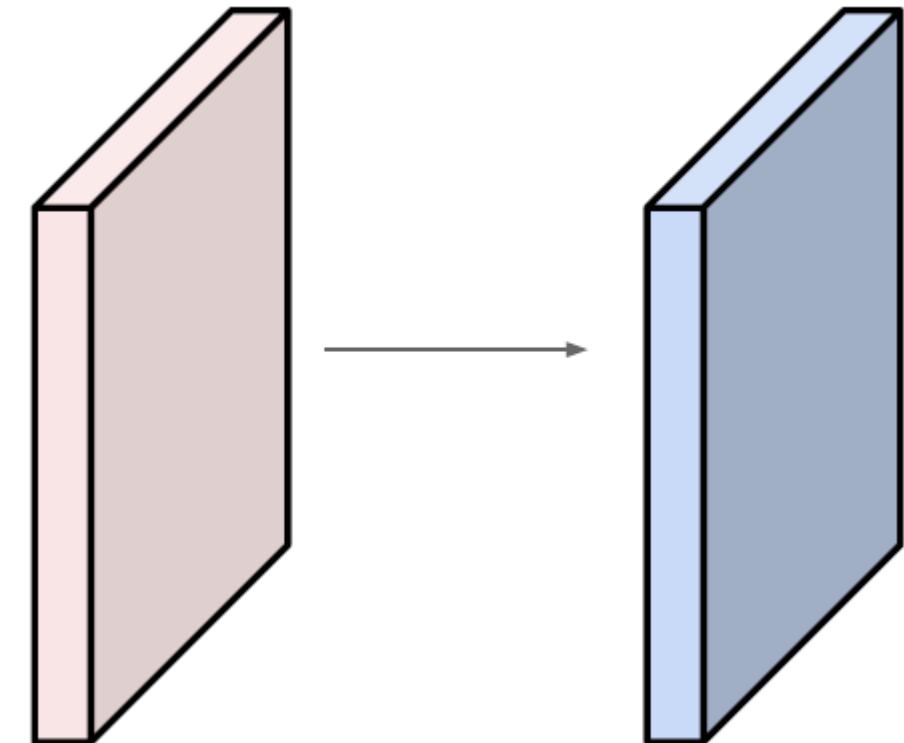
Время примеров:

Входной массив: **32x32x3**

**10** фильтров **5x5** с шагом **1**,  
padding-ом **2**

Размер выходного массива:

$(32 + 2*2 - 5)/1 + 1 = 32$  по  
ширине, то есть **32x32x10**

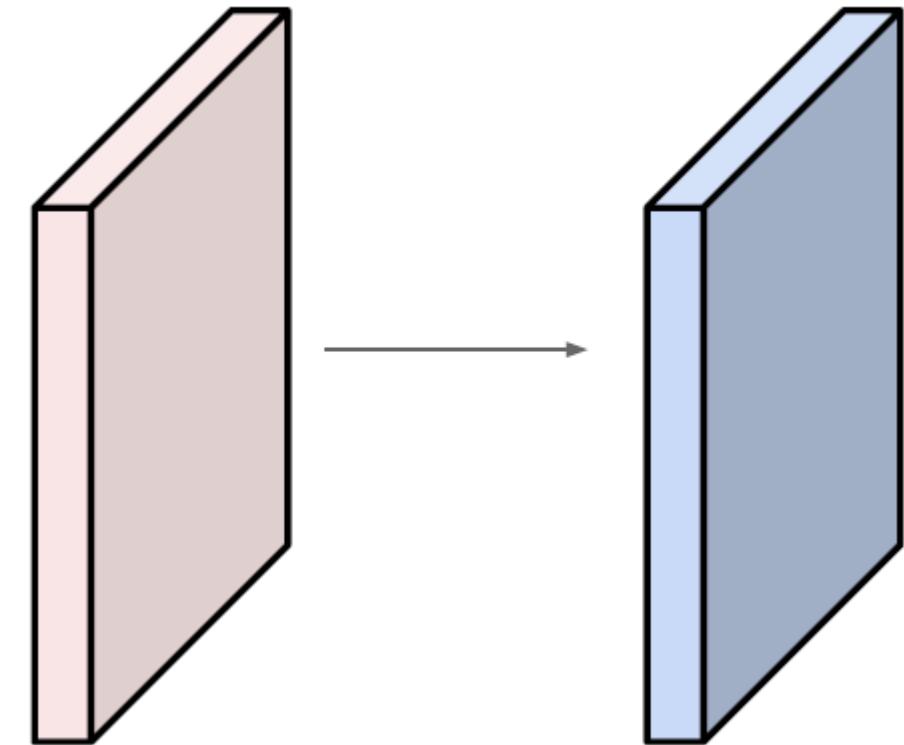


Время примеров:

Входной массив: **32x32x3**

10 фильтров  $5 \times 5$  с шагом 1,  
padding-ом 2

Число параметров этого слоя?



Время примеров:

Входной массив: **32x32x3**

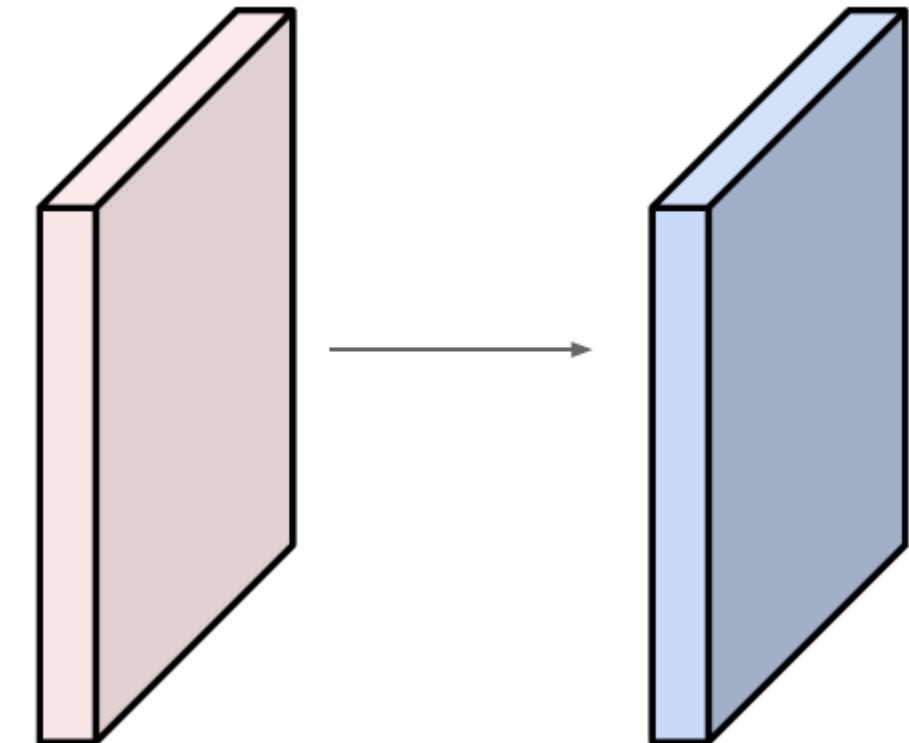
**10** фильтров **5x5** с шагом 1,  
padding-ом 2

Число параметров этого слоя:

В каждом фильтре **5\*5\*3 + 1 =**

**76** параметров =>

**76 \* 10 = 760**



# Свёрточный слой. Итоги

- Принимает вход размера  $W_1 \times H_1 \times D_1$
- Требует 4 гиперпараметра:
  - Число фильтров  $K$
  - Их пространственный размер  $F$
  - Их шаг (stride)  $S$
  - Zero-padding (отступ из нулей)  $P$
- Создаёт выход размера  $W_2 \times H_2 \times D_2$ , где:
  - $W_2 = \frac{W_1 - F + 2P}{S} + 1$
  - $H_2 = \frac{H_1 - F + 2P}{S} + 1$
  - $D_2 = K$
- С учётом разделения параметров, он включает в себя  $F \cdot F \cdot D_1 + 1$  весов на фильтр, в сумме  $(F \cdot F \cdot D_1 + 1) \cdot K$  весов
- В выходном массиве канал с номером  $d$  (размера  $W_2 \times H_2$ ) – это результат применения реальной свёртки фильтра с номером  $d$  с входном массиву с шагом  $S$ , и добавляя bias с номером  $d$

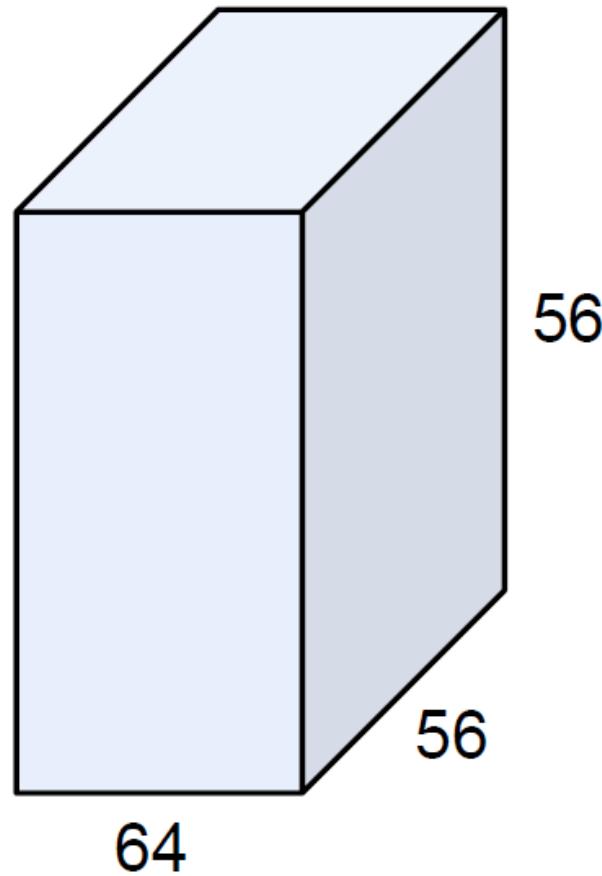
# Свёрточный слой. Итоги

- Принимает вход размера  $W_1 \times H_1 \times D_1$
- Требует 4 гиперпараметра:
  - Число фильтров  $K$
  - Их пространственный размер  $F$
  - Их шаг (stride)  $S$
  - Zero-padding (отступ из нулей)  $P$
- Создаёт выход размера  $W_2 \times H_2 \times D_2$ , где:
  - $W_2 = \frac{W_1 - F + 2P}{S} + 1$
  - $H_2 = \frac{H_1 - F + 2P}{S} + 1$
  - $D_2 = K$
- С учётом разделения параметров, он включает в себя  $F \cdot F \cdot D_1 + 1$  весов на фильтр, в сумме  $(F \cdot F \cdot D_1 + 1) \cdot K$  весов
- В выходном массиве канал с номером  $d$  (размера  $W_2 \times H_2$ ) – это результат применения реальной свёртки фильтра с номером  $d$  с входном массиву с шагом  $S$ , и добавляя bias с номером  $d$

## Частые значения гиперпараметров

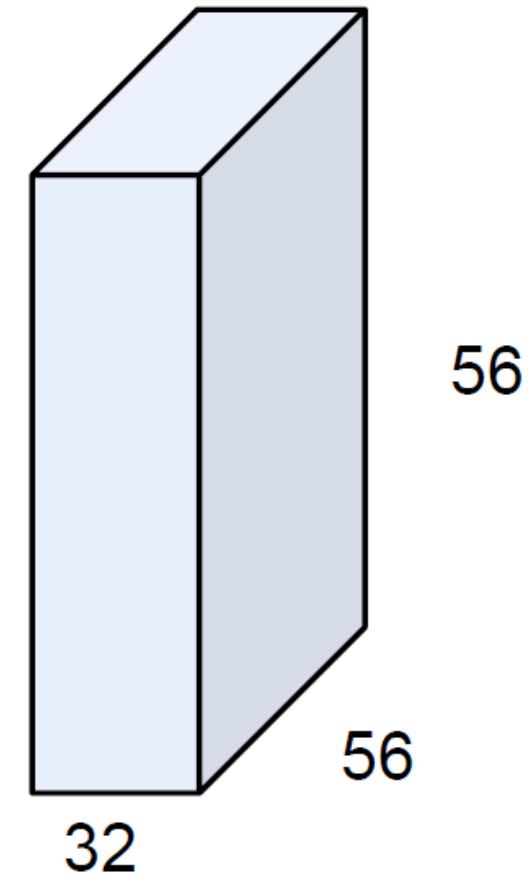
- $K = (\text{степени } 2, \text{ например } 32, 64, 128, 512)$
- $F = 3, S = 1, P = 1$
  - $F = 5, S = 1, P = 2$
  - $F = 5, S = 2, P = ?$  (чтобы помещалось)
  - $F = 1, S = 1, P = 0$

(кстати, свёртки  $1 \times 1$  имеют смысл и применяются довольно часто)



$1 \times 1$  CONV,  
32 filters

(у каждого фильтра размер  
 $1 \times 1 \times 64$ , он производит  
скалярное произведение  
64-мерных векторов входа)



# Пример: CONV2D слой на Torch

CLASS `torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1, bias=True, padding_mode='zeros', device=None, dtype=None)` [\[SOURCE\]](#)

Applies a 2D convolution over an input signal composed of several input planes.

In the simplest case, the output value of the layer with input size  $(N, C_{\text{in}}, H, W)$  and output  $(N, C_{\text{out}}, H_{\text{out}}, W_{\text{out}})$  can be precisely described as:

$$\text{out}(N_i, C_{\text{out}_j}) = \text{bias}(C_{\text{out}_j}) + \sum_{k=0}^{C_{\text{in}}-1} \text{weight}(C_{\text{out}_j}, k) \star \text{input}(N_i, k)$$

where  $\star$  is the valid 2D cross-correlation operator,  $N$  is a batch size,  $C$  denotes a number of channels,  $H$  is a height of input planes in pixels, and  $W$  is width in pixels.

## Parameters

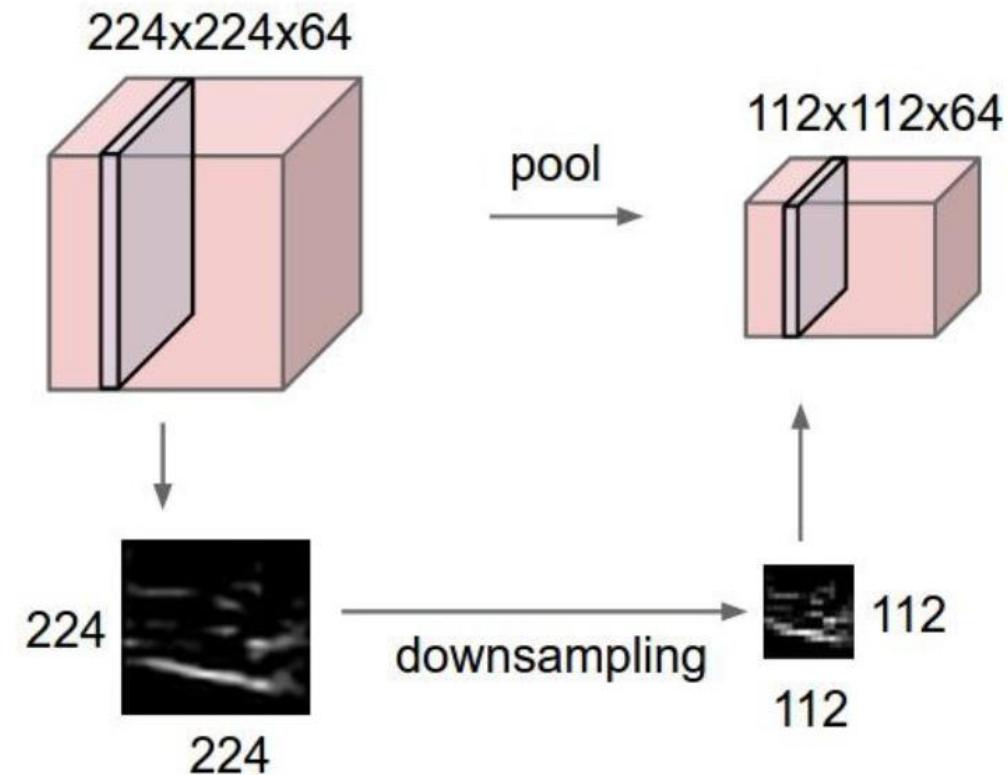
### CONV Layer:

- Принимает вход размера  $W_1 \times H_1 \times D_1$
- Требует 4 гиперпараметра:
  - Число фильтров  $K$
  - Их пространственный размер  $F$
  - Их шаг (stride)  $S$
  - Zero-padding (отступ из нулей)  $P$

- **in\_channels** (`int`) – Number of channels in the input image
- **out\_channels** (`int`) – Number of channels produced by the convolution
- **kernel\_size** (`int` or `tuple`) – Size of the convolving kernel
- **stride** (`int` or `tuple`, `optional`) – Stride of the convolution. Default: 1
- **padding** (`int`, `tuple` or `str`, `optional`) – Padding added to all four sides of the input. Default: 0
- **padding\_mode** (`string`, `optional`) – `'zeros'`, `'reflect'`, `'replicate'` or `'circular'`. Default: `'zeros'`
- **dilation** (`int` or `tuple`, `optional`) – Spacing between kernel elements. Default: 1
- **groups** (`int`, `optional`) – Number of blocked connections from input channels to output channels. Default: 1
- **bias** (`bool`, `optional`) – If `True`, adds a learnable bias to the output. Default: `True`

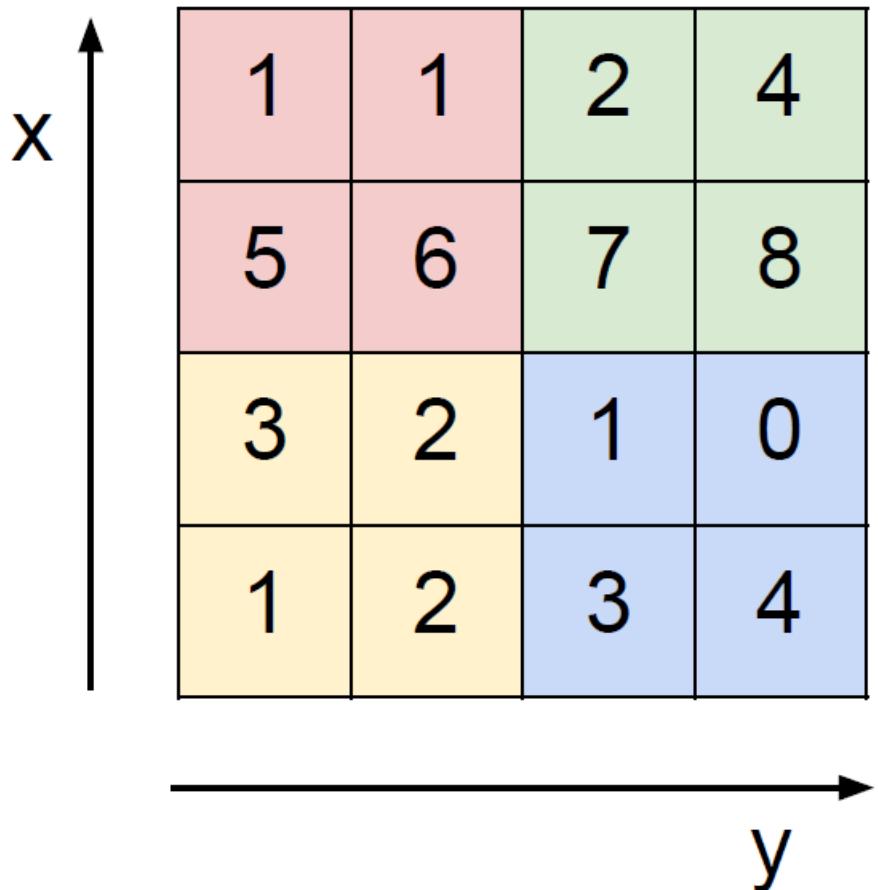
# Pooling layer (Слой субдискретизации/пулинга)

- Уменьшает размер представления
- Обрабатывает каждую карту активации независимо



# MAX POOLING

Один канал входа



Max Pool с фильтром  
2x2 и stride 2

6	8
3	4

# Pooling слой

- Принимает вход размера  $W_1 \times H_1 \times D_1$
- Гиперпараметры:
  - Пространственный размер фильтра  $F$
  - Шаг (stride)  $S$
- Создаёт выход размера  $W_2 \times H_2 \times D_2$ , где:
  - $W_2 = \frac{W_1 - F}{S} + 1$
  - $H_2 = \frac{H_1 - F}{S} + 1$
  - $D_2 = D_1$
- Не имеет обучаемых параметров – это фиксированная функция входа
- Zero padding обычно не используют в pooling слоях (в теории можно)

# Pooling слой

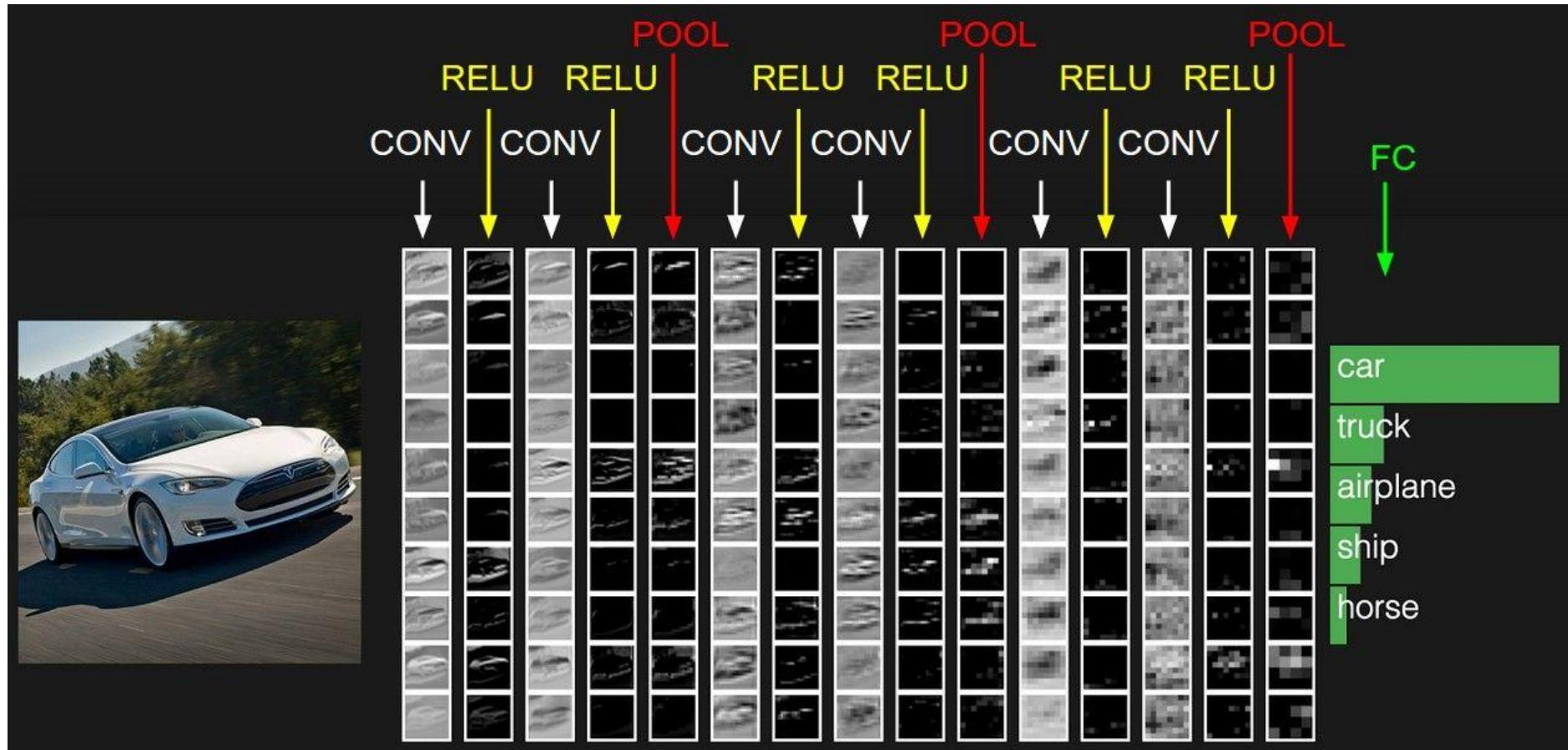
- Принимает вход размера  $W_1 \times H_1 \times D_1$
- Гиперпараметры:
  - Пространственный размер фильтра  $F$
  - Шаг (stride)  $S$
- Создаёт выход размера  $W_2 \times H_2 \times D_2$ , где:
  - $W_2 = \frac{W_1 - F}{S} + 1$
  - $H_2 = \frac{H_1 - F}{S} + 1$
  - $D_2 = D_1$
- Не имеет обучаемых параметров – это фиксированная функция входа
- Zero padding обычно не используют в pooling слоях (в теории можно)

Частые значения гиперпараметров

- $F = 2, S = 2$
- $F = 3, S = 2$
- $F = 3, S = 3$

# Полносвязный (FC) слой

Состоит из нейронов, связанных с каждым элементом входа, как и в обычных нейронных сетях



# Свёрточные нейронные сети – итоги

- Свёрточная сеть – это композиция слоёв CONV, POOL, FC
- Общий тренд в развитии архитектур: меньше фильтры, больше глубина
- Есть тренд в избавлении от слоёв POOL, FC (используют только CONV)
- Типичная архитектура сети для классификации:
  - **$[(CONV-RELU)^*N-POOL?]^*M-(FC-RELU)^*K-SOFTMAX$**
  - Где  $N \sim 5$ ,  $M$  – большое,  $0 <= K <= 2$
  - Последние результаты в исследованиях архитектур меняют эту парадигму: ResNet/GoogLeNet