

Глубинное обучение

Лекция 2
Свёрточные сети

Александр Шабалин

Полносвязные сети для
изображений

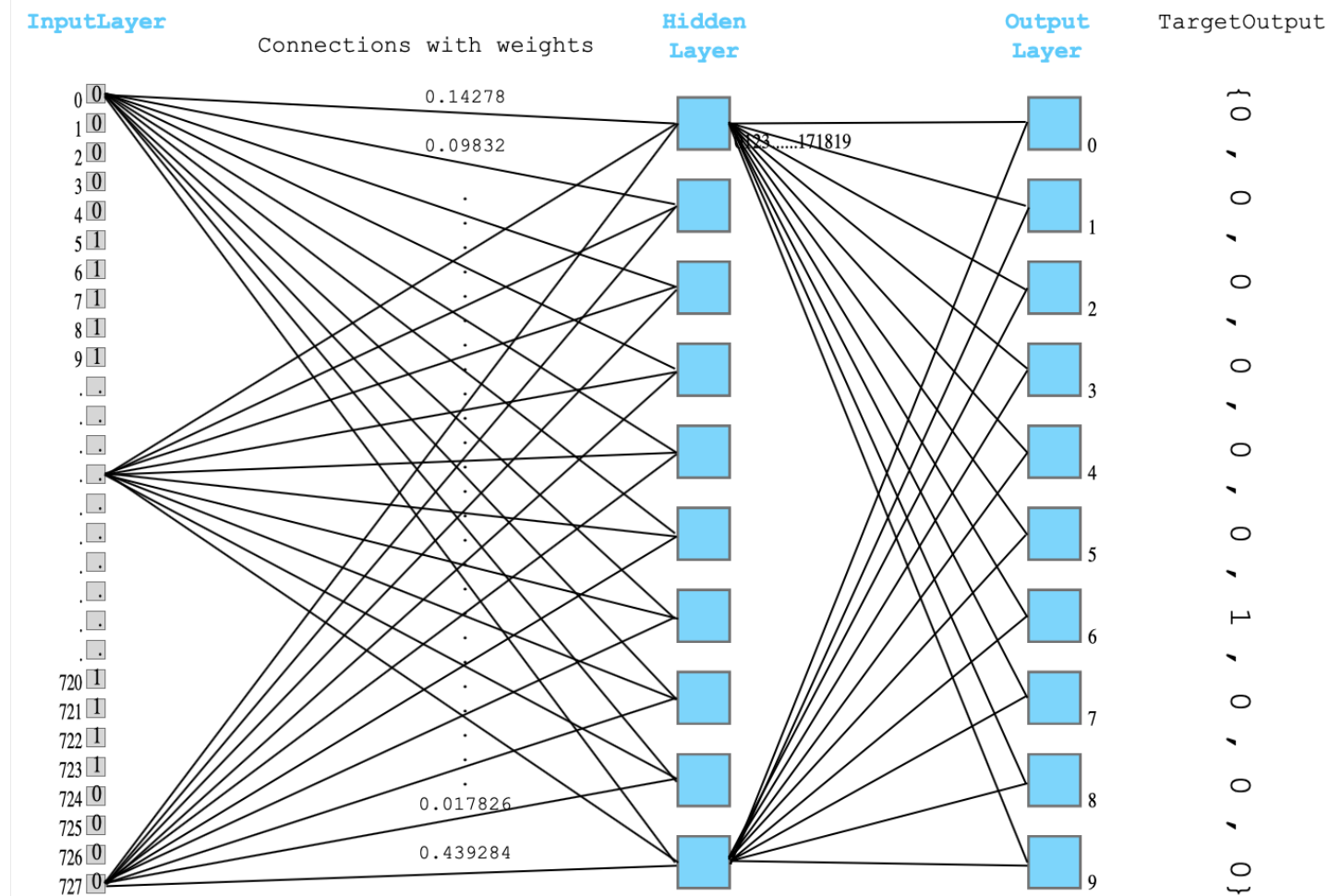
MNIST

- Изображения 28 x 28
- 60.000 объектов в обучающей выборке



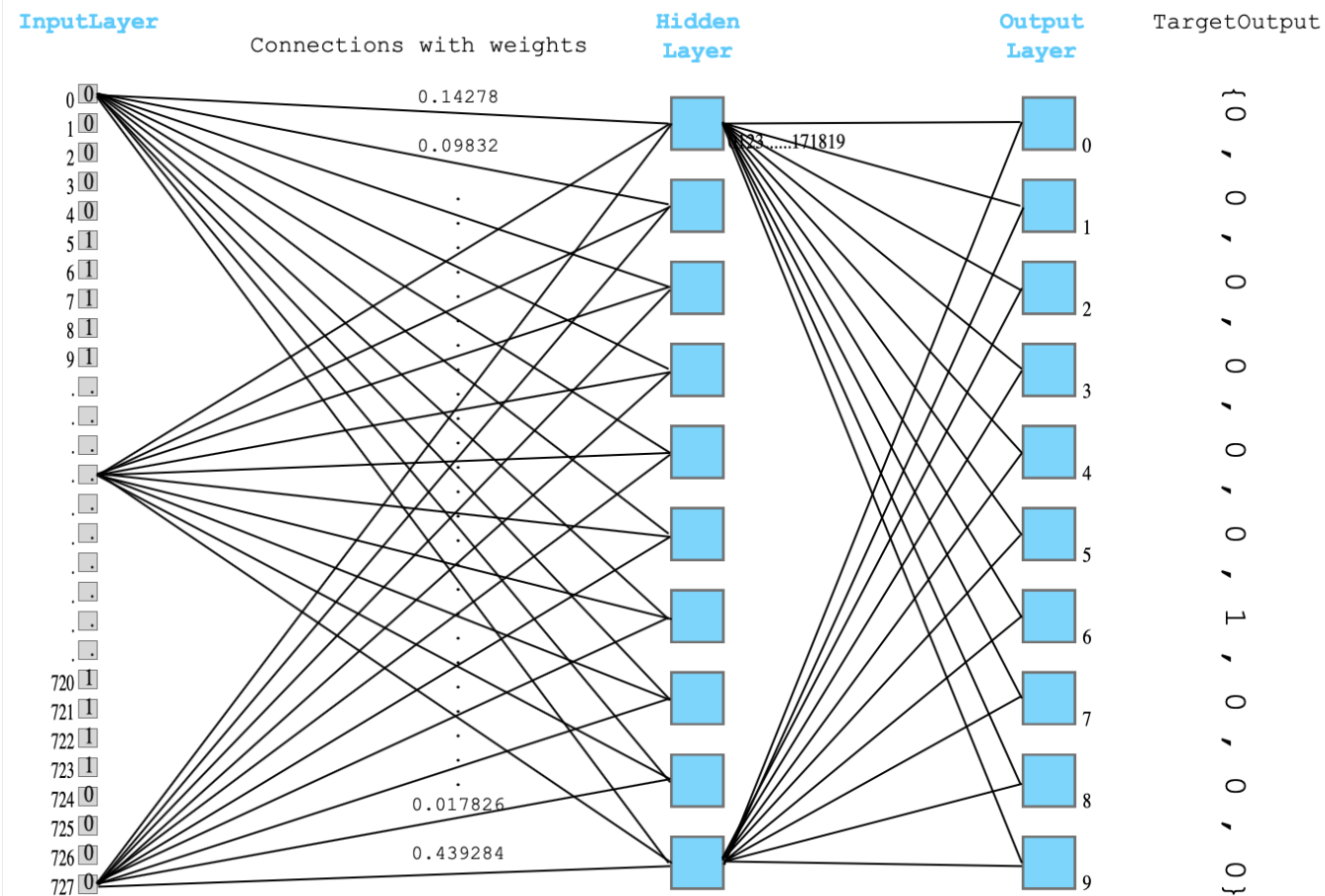
MNIST

- Что может выучить полносвязная сеть?



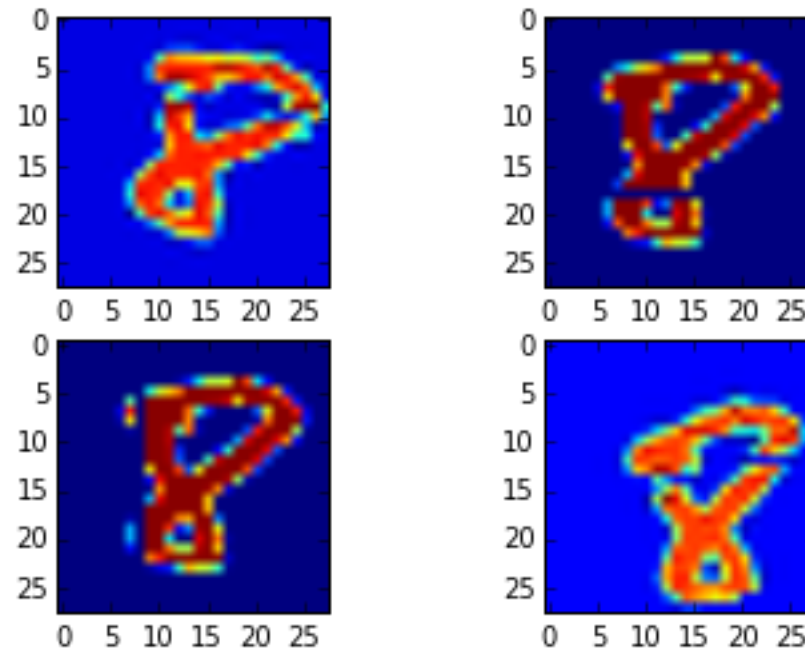
MNIST

- Каждый нейрон может детектировать цвет конкретного набора пикселей



MNIST

- Если немного сдвинуть цифру, то нейрон уже не будет на неё реагировать



Число параметров

- Вход: $28 * 28 = 784$ пикселя
- Скрытый слой: 1000 нейронов
- Выход: 10 нейронов (по одному на каждый класс)
- Весов между входным и скрытым слоем:

$$(784 + 1) * 1000 = 785.000$$

- Весов между скрытым слоем и выходом:

$$(1000 + 1) * 10 = 10.010$$

- Всего параметров: 795.010

Число параметров

- Качество полносвязных сетей часто низкое
- Но можно добиться хорошего качества увеличивая число параметров (с аугментацией)

Table 1: Error rates on MNIST test set.

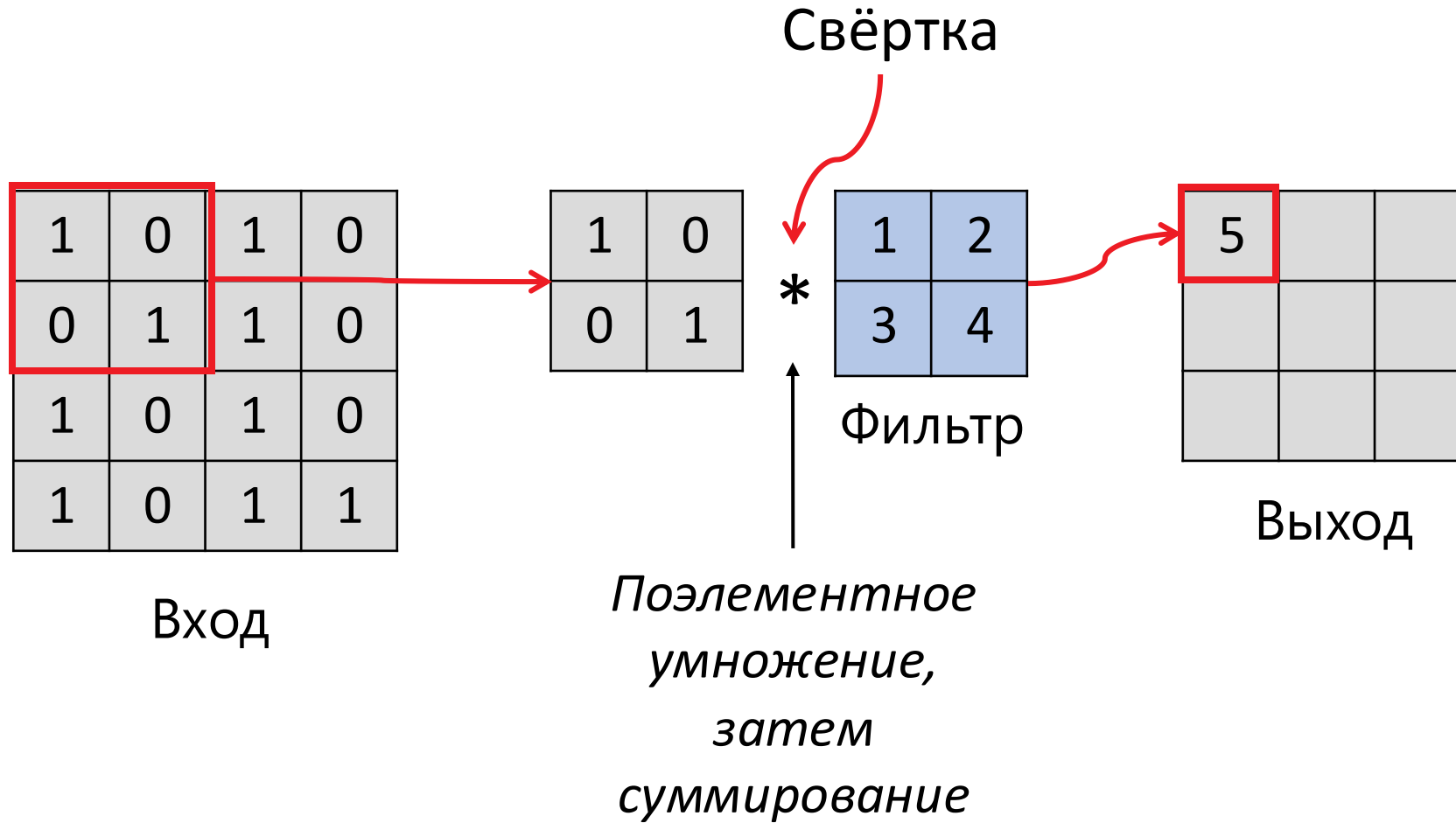
ID	architecture (number of neurons in each layer)	test error for best validation [%]	best test error [%]	simulation time [h]	weights [milions]
1	1000, 500, 10	0.49	0.44	23.4	1.34
2	1500, 1000, 500, 10	0.46	0.40	44.2	3.26
3	2000, 1500, 1000, 500, 10	0.41	0.39	66.7	6.69
4	2500, 2000, 1500, 1000, 500, 10	0.35	0.32	114.5	12.11
5	9×1000 , 10	0.44	0.43	107.7	8.86

Полносвязные сети для изображений

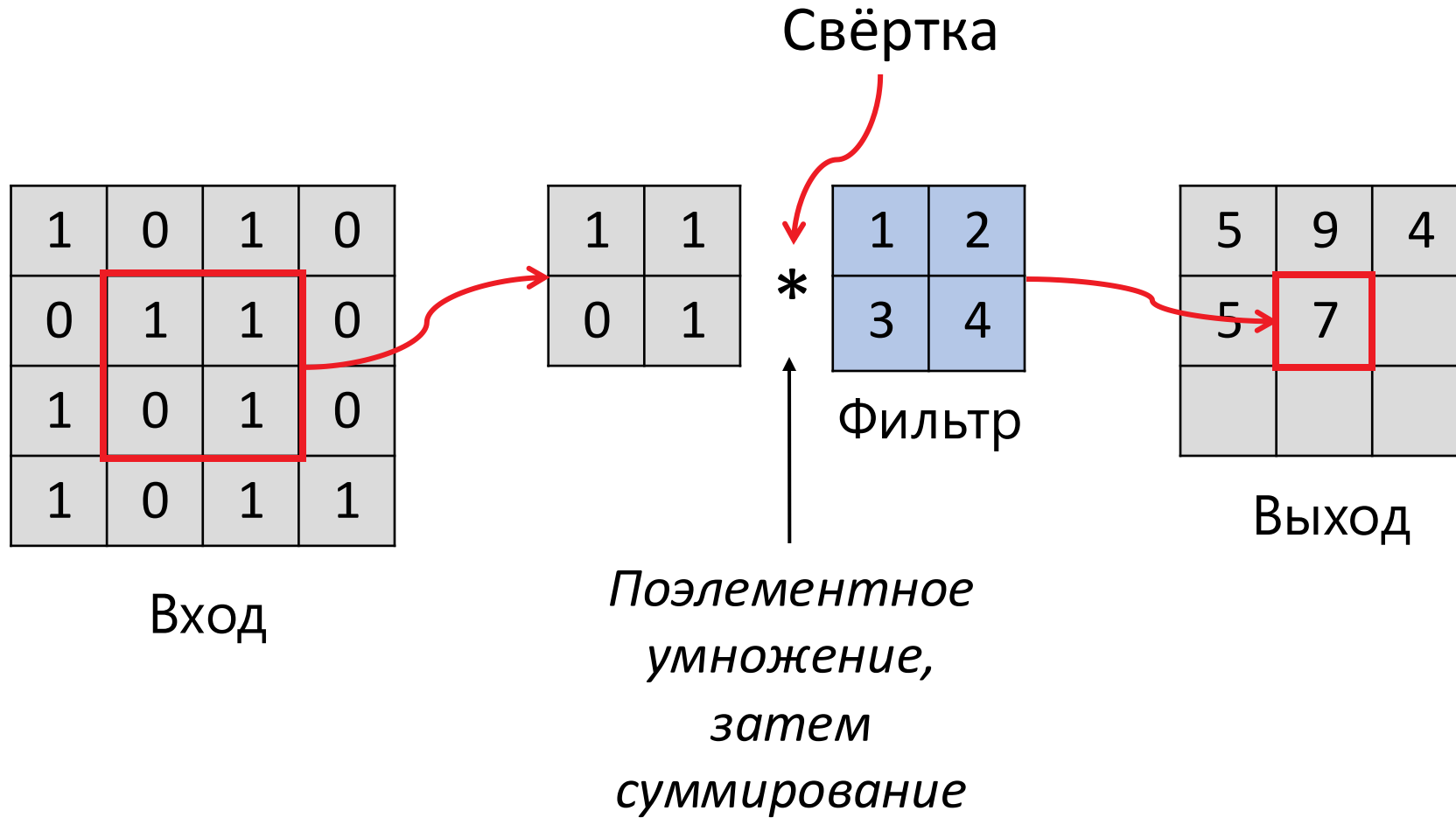
- Очень много параметров
- Легко переобучаются
- Не учитывают специфику изображений (сдвиги, небольшие изменения формы и т.д.)

Сверточные слои
а.к.а. Свёртки

Свёрточный слой



Свёрточный слой



Свёртка

$$\begin{array}{|c|c|} \hline 1 & 1 \\ \hline 0 & 1 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} = \boxed{2}$$

$$\begin{array}{|c|c|} \hline 1 & 1 \\ \hline 1 & 1 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} = \boxed{2}$$

$$\begin{array}{|c|c|} \hline 1 & 2 \\ \hline 3 & 0 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} = \boxed{1}$$

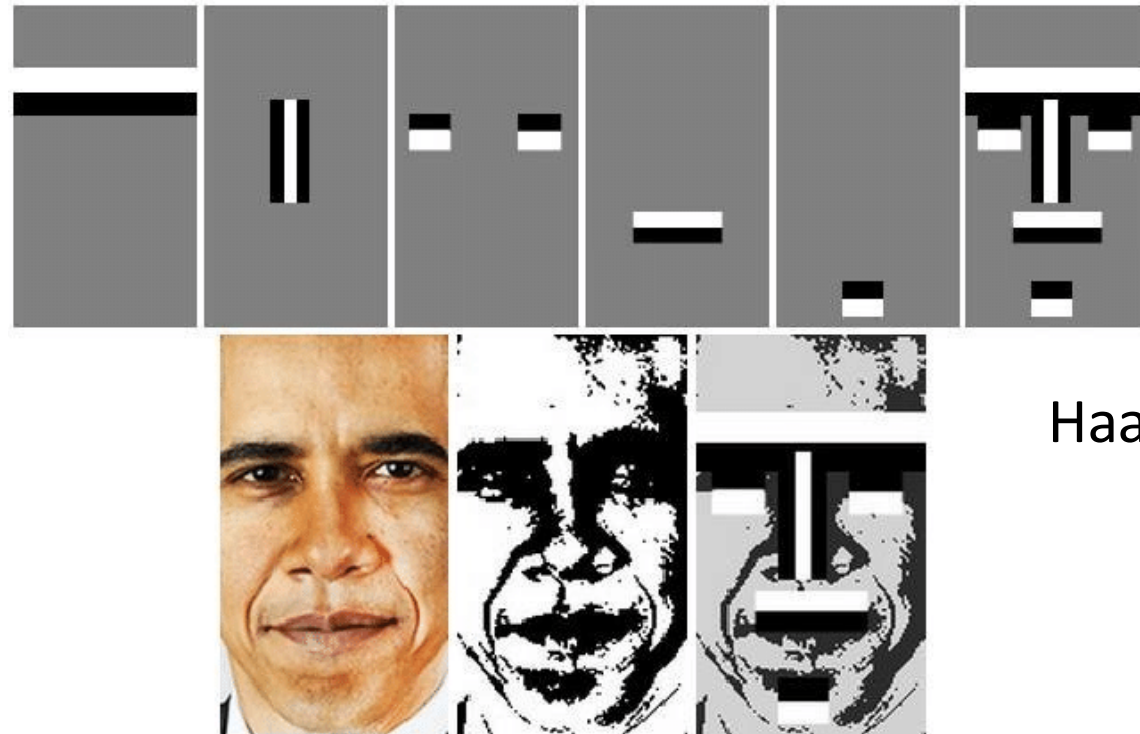
$$\begin{array}{|c|c|} \hline 0 & 2 \\ \hline 3 & 0 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} = \boxed{0}$$

$$\begin{array}{|c|c|} \hline 3 & 0 \\ \hline 0 & 3 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} = \boxed{6}$$

$$\begin{array}{|c|c|} \hline 5 & 0 \\ \hline 0 & 5 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array} = \boxed{10}$$

Свёртка

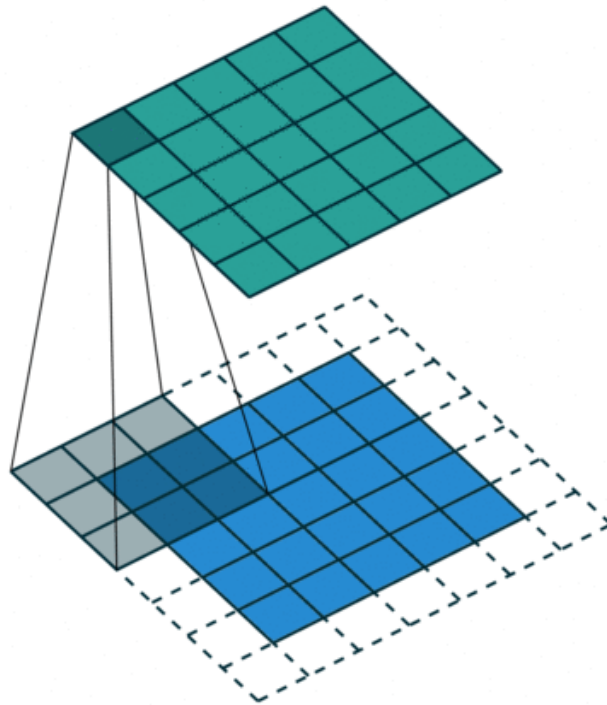
- Операция свёртки выявляет наличие на изображении паттерна, который задаётся фильтром
- Чем сильнее на участке изображения представлен паттерн, тем больше будет значение свёртки



Haar features

Свёртка

- Результат свёртки изображения с фильтром — новое изображение



Максимум свёртки инвариантен к сдвигам

0	0	0	0
0	0	0	0
0	0	1	0
0	0	0	1

Вход

*

1	0
0	1

Фильтр

=

0	0	0
0	1	0
0	0	2

Выход

Max = 2

Не меняется

1	0	0	0
0	1	0	0
0	0	0	0
0	0	0	0

Вход

*

1	0
0	1

Фильтр

=

2	0	0
0	1	0
0	0	0

Выход

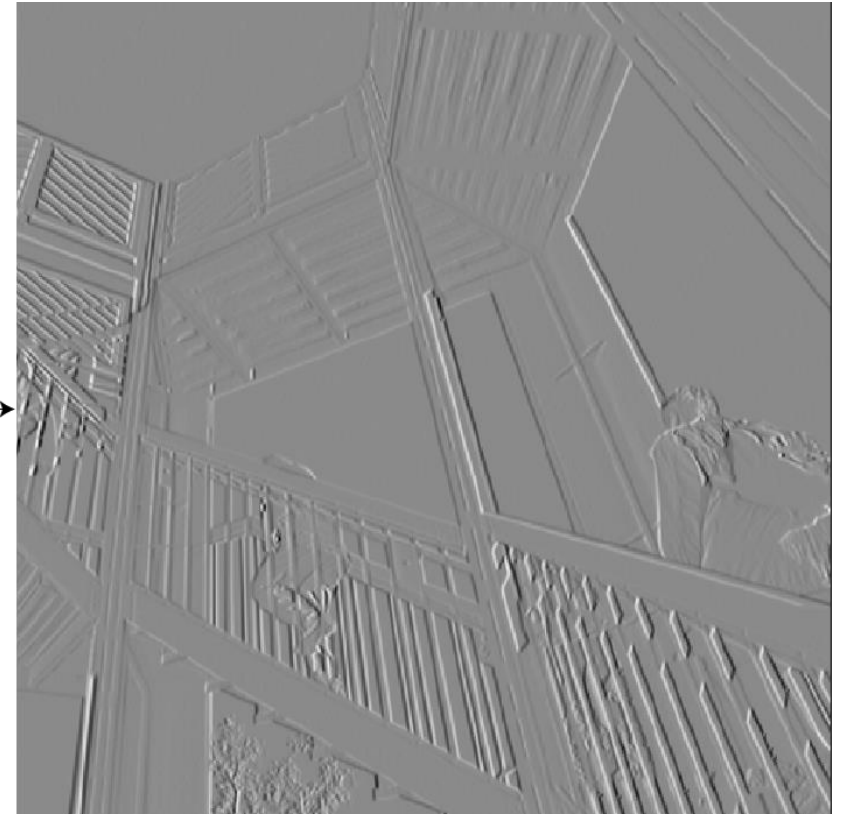
Max = 2

Свёртки в компьютерном зрении



$$\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

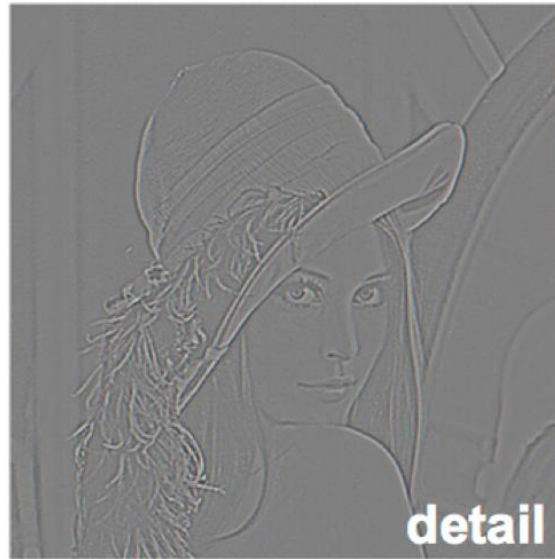
Horizontal Sobel kernel



Свёртки в компьютерном зрении



+



=



•0	•0	•0
•0	•1	•0
•0	•0	•0

+

•0	•0	•0
•0	•1	•0
•0	•0	•0

$-\frac{1}{9}$

•1	•1	•1
•1	•1	•1
•1	•1	•1

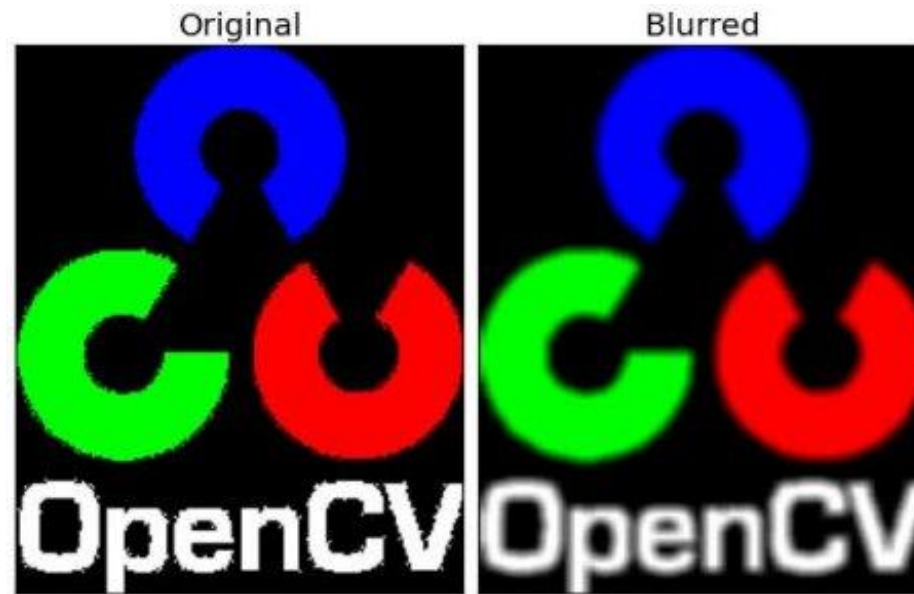
=

•0	•0	•0
•0	•2	•0
•0	•0	•0

$-\frac{1}{9}$

•1	•1	•1
•1	•1	•1
•1	•1	•1

Свёртки в компьютерном зрении



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

Формальная запись свёртки

$$\text{Im}^{out}(x, y) = \sum_{i=-d}^d \sum_{j=-d}^d (K(i, j) \text{Im}^{in}(x + i, y + j) + b)$$

Формальная запись свёртки

$$\text{Im}^{out}(x, y) = \sum_{i=-d}^d \sum_{j=-d}^d (K(i, j) \text{Im}^{in}(x + i, y + j) + b)$$

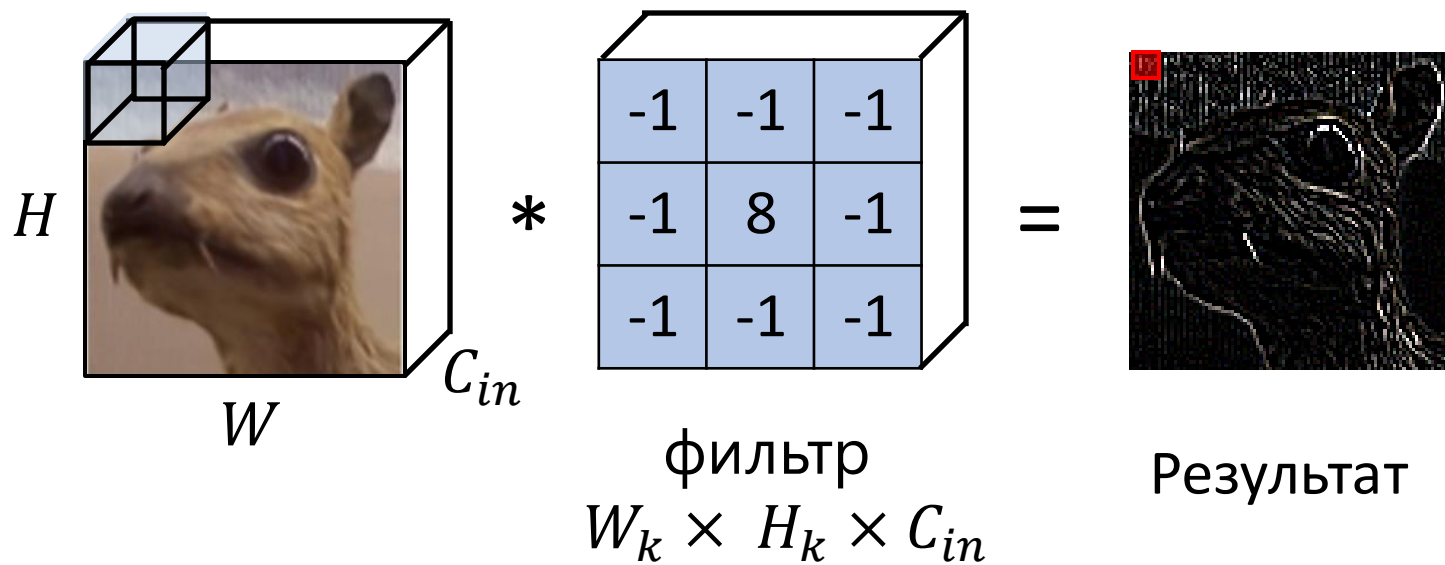
- Пиксель в результирующем изображении зависит только от небольшого участка исходного изображения (local connectivity)
- Веса одни и те же для всех пикселей результирующего изображения (shared weights)

Формальная запись свёртки

- Обычно исходное изображение цветное!
- Это означает, что в нём несколько каналов (R, G, B)
- Учтём в формуле:

$$\text{Im}^{out}(x, y) = \sum_{i=-d}^d \sum_{j=-d}^d \sum_{c=1}^c (K(i, j, c) \text{Im}^{in}(x + i, y + j, c) + b)$$

Свёртка



Свёртка

- Одна свёртка выделяет конкретный паттерн на изображении
- Нам интересно искать много паттернов
- Сделаем результат трёхмерным:

$$\text{Im}^{out}(x, y, t) = \sum_{i=-d}^d \sum_{j=-d}^d \sum_{c=1}^C (K_t(i, j, c) \text{Im}^{in}(x + i, y + j, c) + b_t)$$

Число параметров

$$\text{Im}^{out}(x, y, t) = \sum_{i=-d}^d \sum_{j=-d}^d \sum_{c=1}^C (\textcolor{red}{K}_t(\textcolor{red}{i}, \textcolor{red}{j}, \textcolor{red}{c}) \text{Im}^{in}(x + i, y + j, c) + \textcolor{red}{b}_t)$$

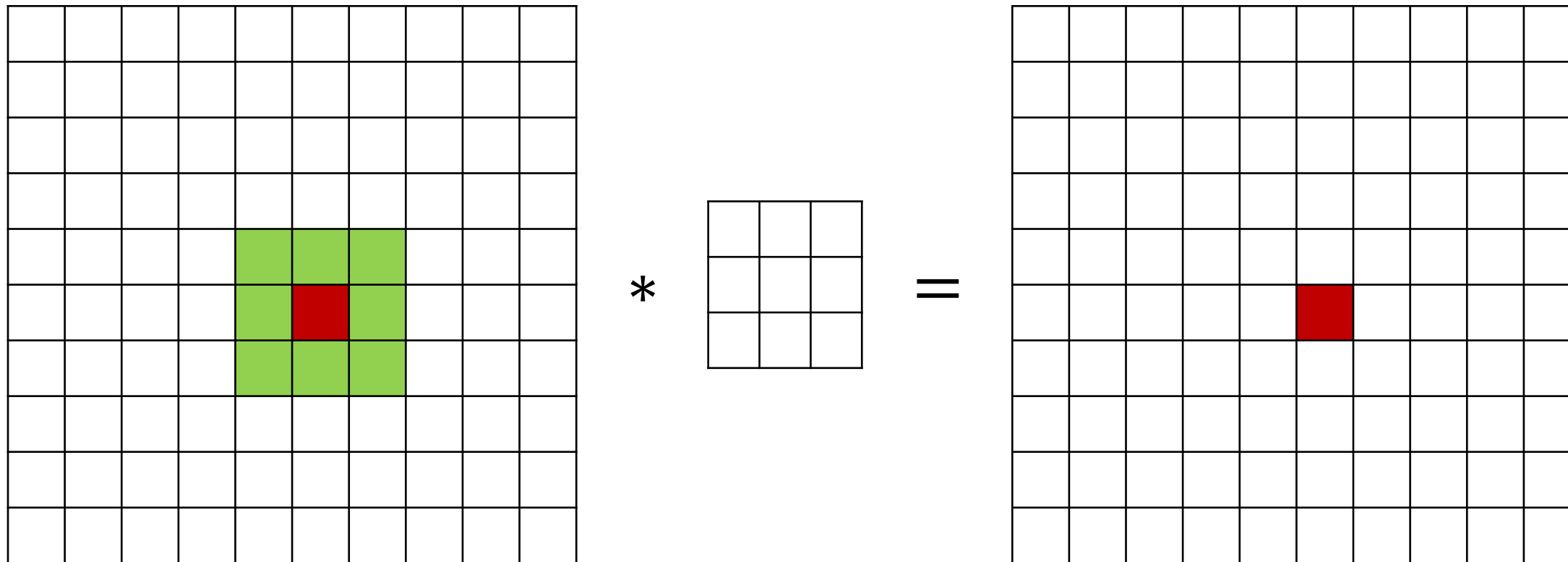
- Обучается только фильтр
- $((2d + 1)^2 * C + 1) * T$ параметров
- Как из этого сделать модель — обсудим позже

Receptive field
a.k.a Поле восприятия

Receptive field

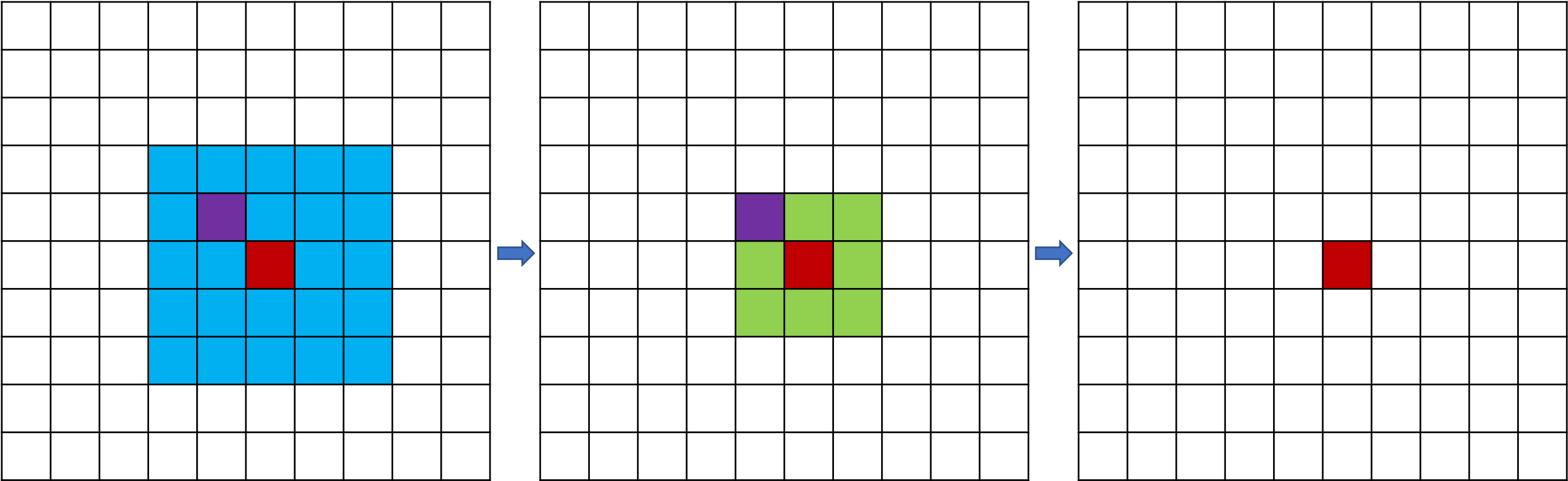
- Возьмём пиксель в итоговом изображении (после свёрточных слоёв)
- От какой части входного изображения зависит значение в этом пикселе?

Receptive field



Поле восприятия: 3 x 3

Receptive field



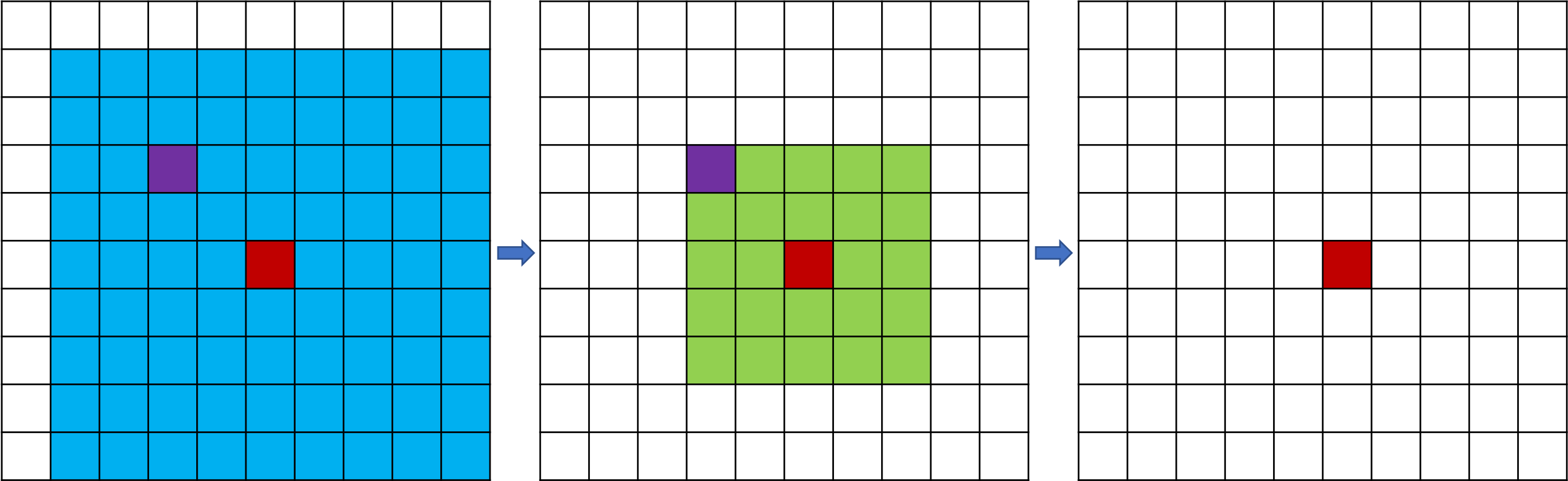
Поле восприятия: 5 x 5

Receptive field

Поле восприятия для свёртки 3 x 3:

- После 1 свёрточного слоя: 3 x 3
- После 2 свёрточных слоев: 5 x 5
- После 3 свёрточных слоёв: 7 x 7

Receptive field



Поле восприятия: 9 x 9

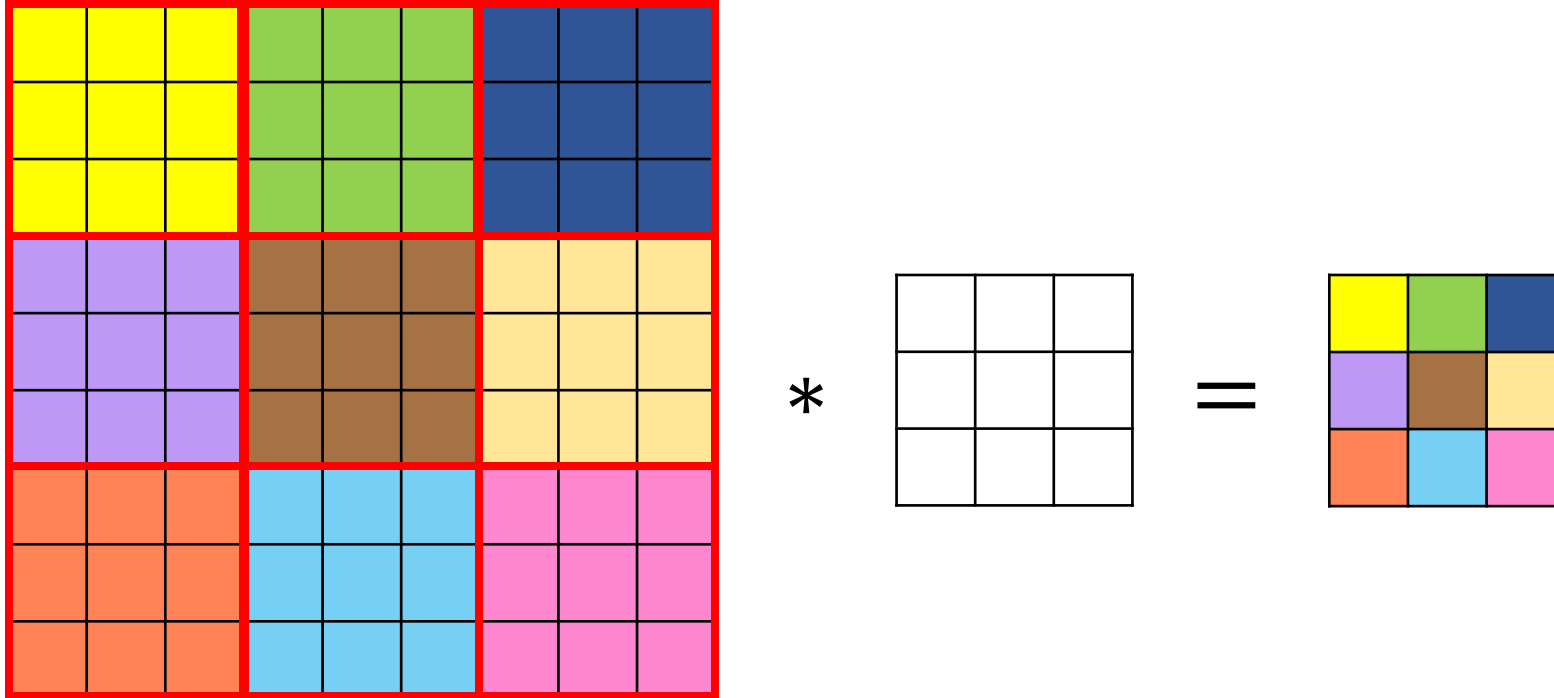
Receptive field

Поле восприятия для свёртки 5 x 5:

- После 1 свёрточного слоя: 5 x 5
- После 2 свёрточных слоев: 9 x 9
- После 3 свёрточных слоёв: 13 x 13

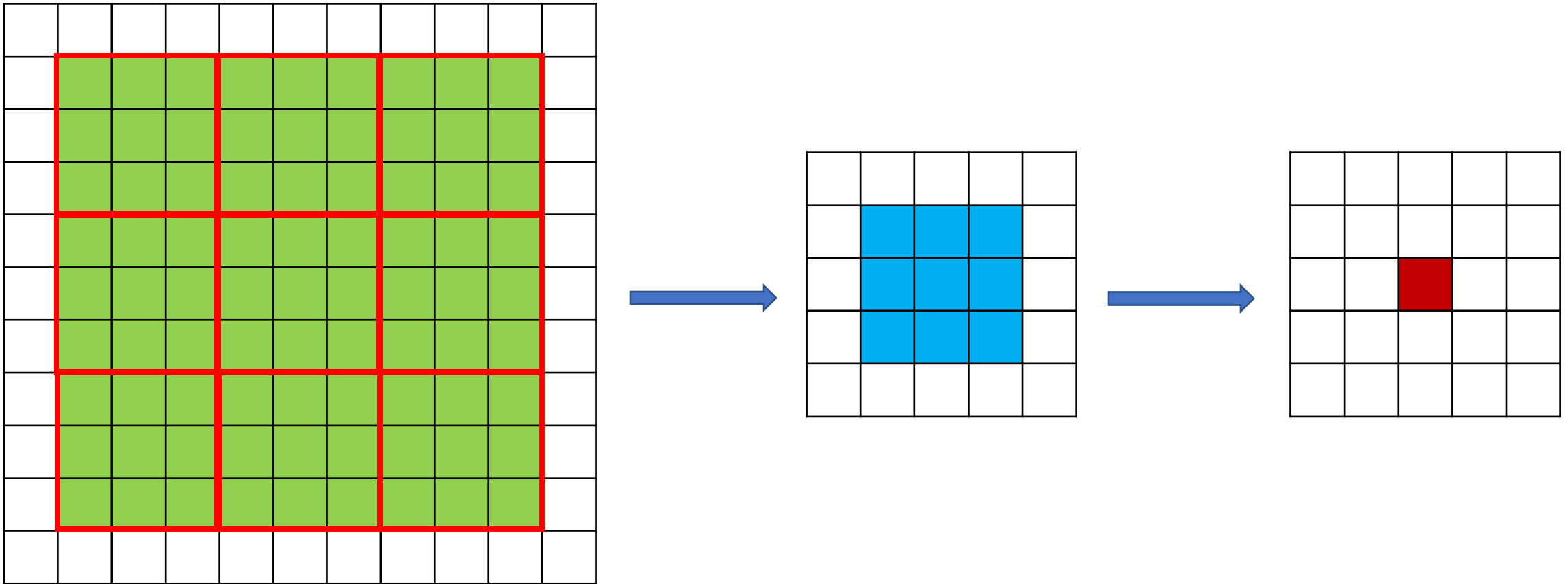
Нужно очень много слоёв, если изображение размера 512 x 512

Свёртки с пропусками (strides)



$$s = 3$$

Свёртки с пропусками (strides)



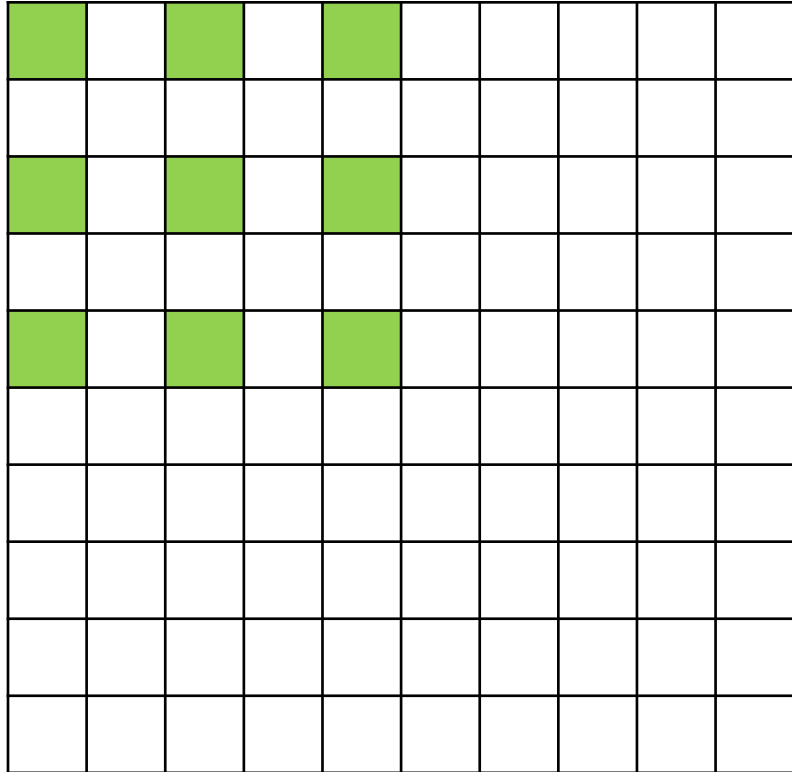
Поле восприятия: 9 x 9

Свёртки

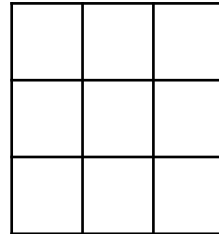
Подробности про подсчёт размера поля с интерактивными
визуализациями

<https://distill.pub/2019/computing-receptive-fields/>

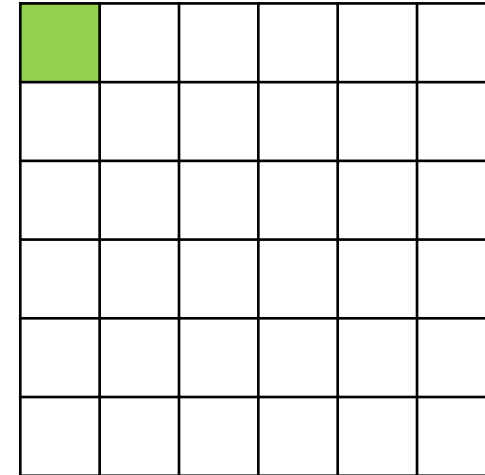
Dilated convolutions («раздутые» свёртки)



*

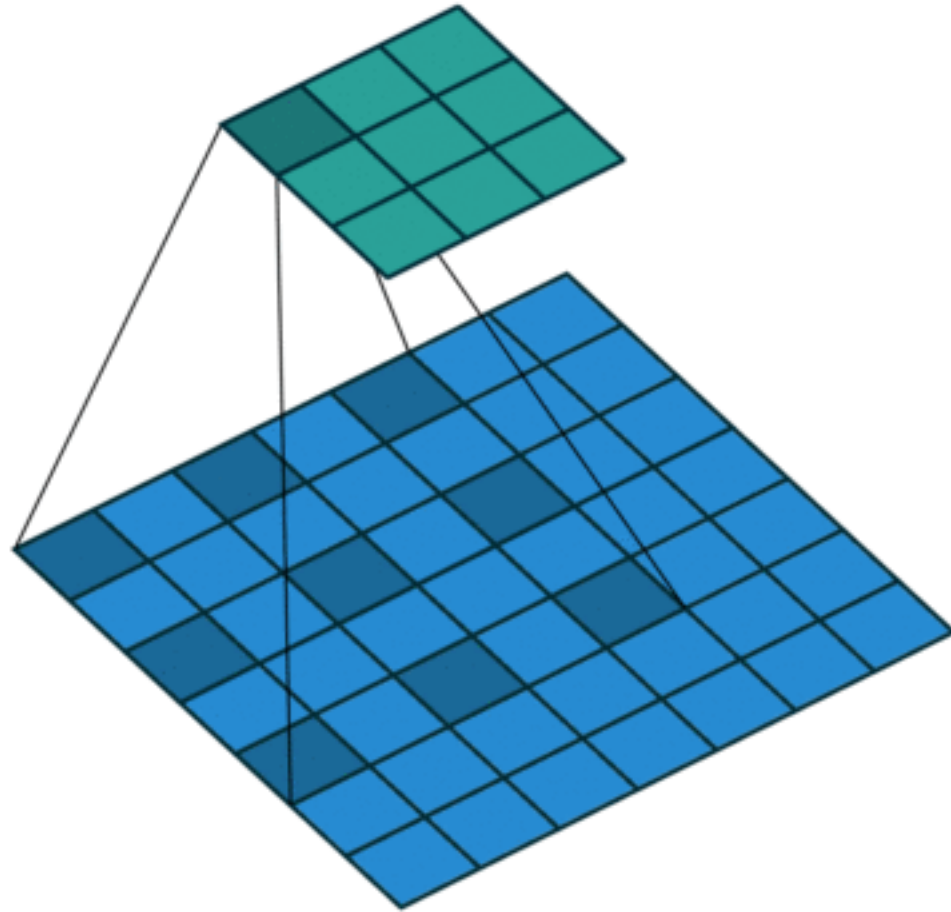


=

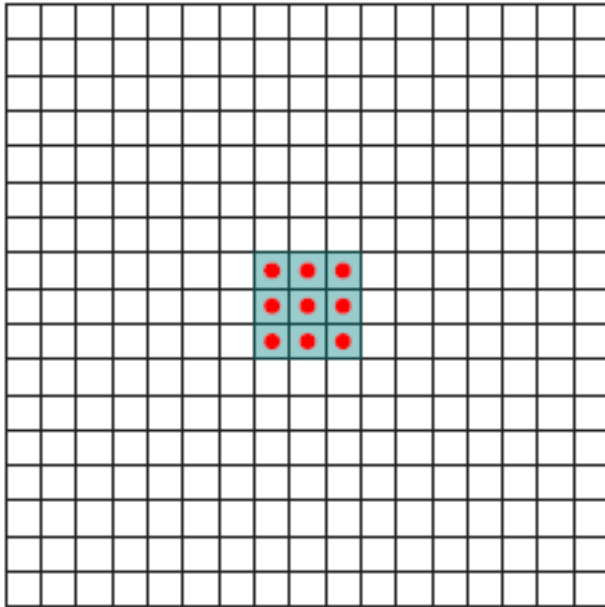


$$l = 2$$

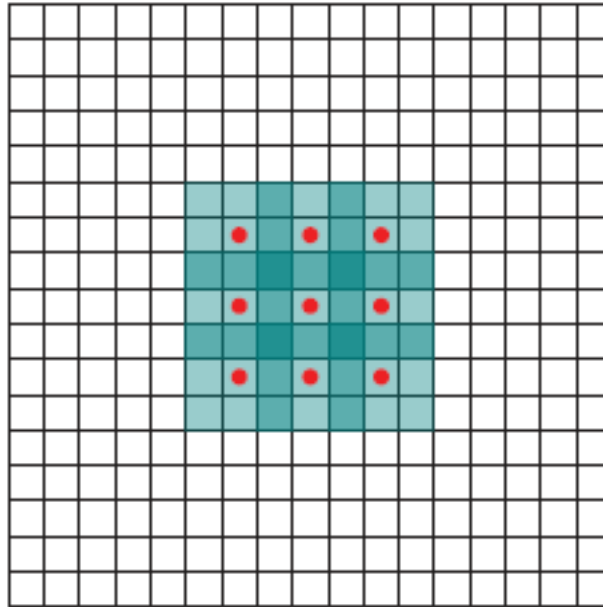
Dilated convolutions («раздутые» свёртки)



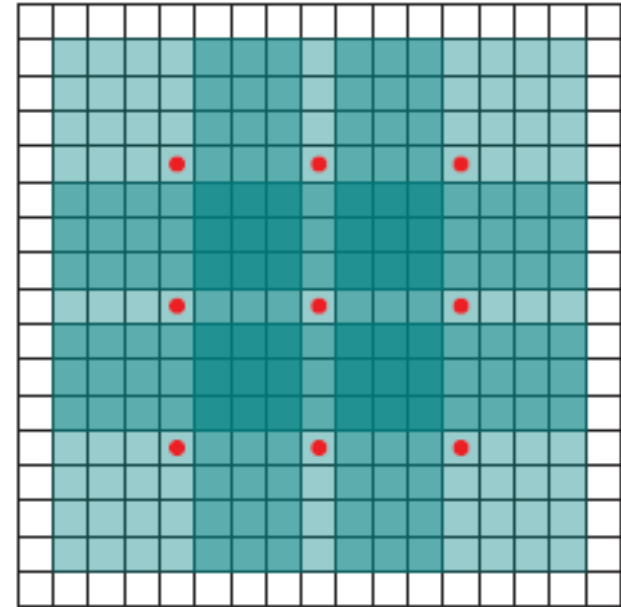
Dilated convolutions («раздутые» свёртки)



$l = 1$

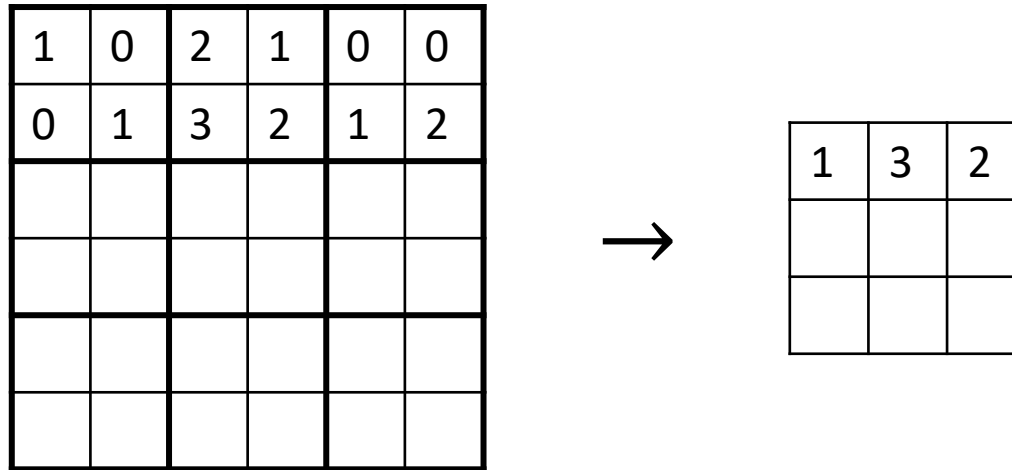


$l = 2$



$l = 4$

Pooling



Max-pooling с фильтром 2x2

Pooling

- Разбивает изображение на участки $n \times t$ и считает некоторую статистику в каждом участке (обычно максимум)
- Существенно сокращает размер изображения (значит, увеличивает поле восприятия следующих слоёв)
- Не имеет параметров

Зачем это всё?

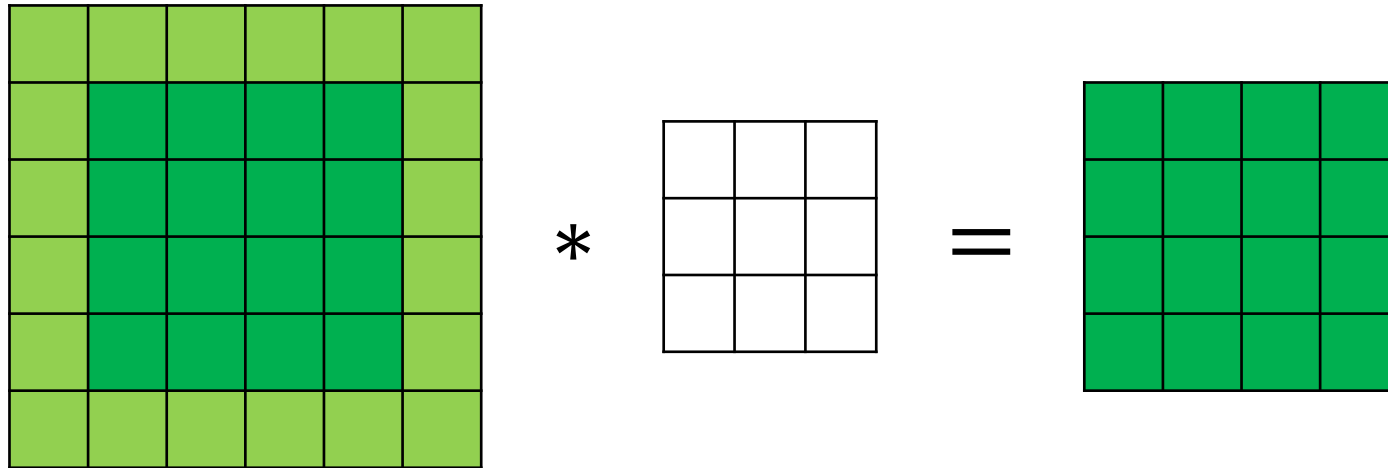
- Важно следить за тем, чтобы последние свёрточные слои имели размер поля восприятия, сравнимый со всей картинкой

Padding

Свёртки

- Если применять свёртку по формуле, то выходное изображение будет меньше входного

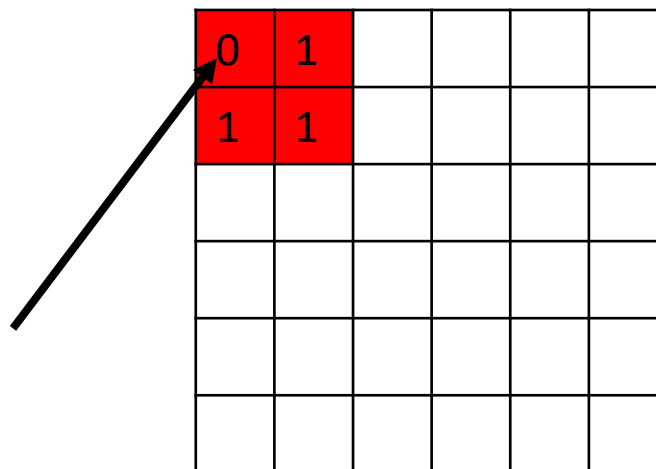
Свёртки



Valid mode

- При честном подсчёте свёрток пиксели на краях не оказывают большого влияния на результат

Не увидим, что фильтр имеет хороший отклик, потому что не можем поставить центр в этот пиксель



0	1				
1	1				

*

0	0	1
0	0	1
1	1	1

Zero padding

0	0	0	0	0	0	0	0
0							0
0							0
0							0
0							0
0							0
0							0
0	0	0	0	0	0	0	0

[illegible]

Zero padding

- Добавляем по границам нули так, чтобы посчитанная после этого свёртка в `valid mode` давала изображение такого же размера, как исходное
- Есть риск, что модель научится понимать, где на изображении края — можем потерять инвариантность

Reflection padding

Отзеркалим ближайшие к границе пиксели

[illegible]

[illegible]

Reflection padding

- Не получится легко находить края изображения
- Но теперь модель может начать находить зеркальные отражения и подбирать фильтры под них

Replication padding

Продублируем ближайшие к границе пиксели

[illegible]

[illegible]

Replication padding

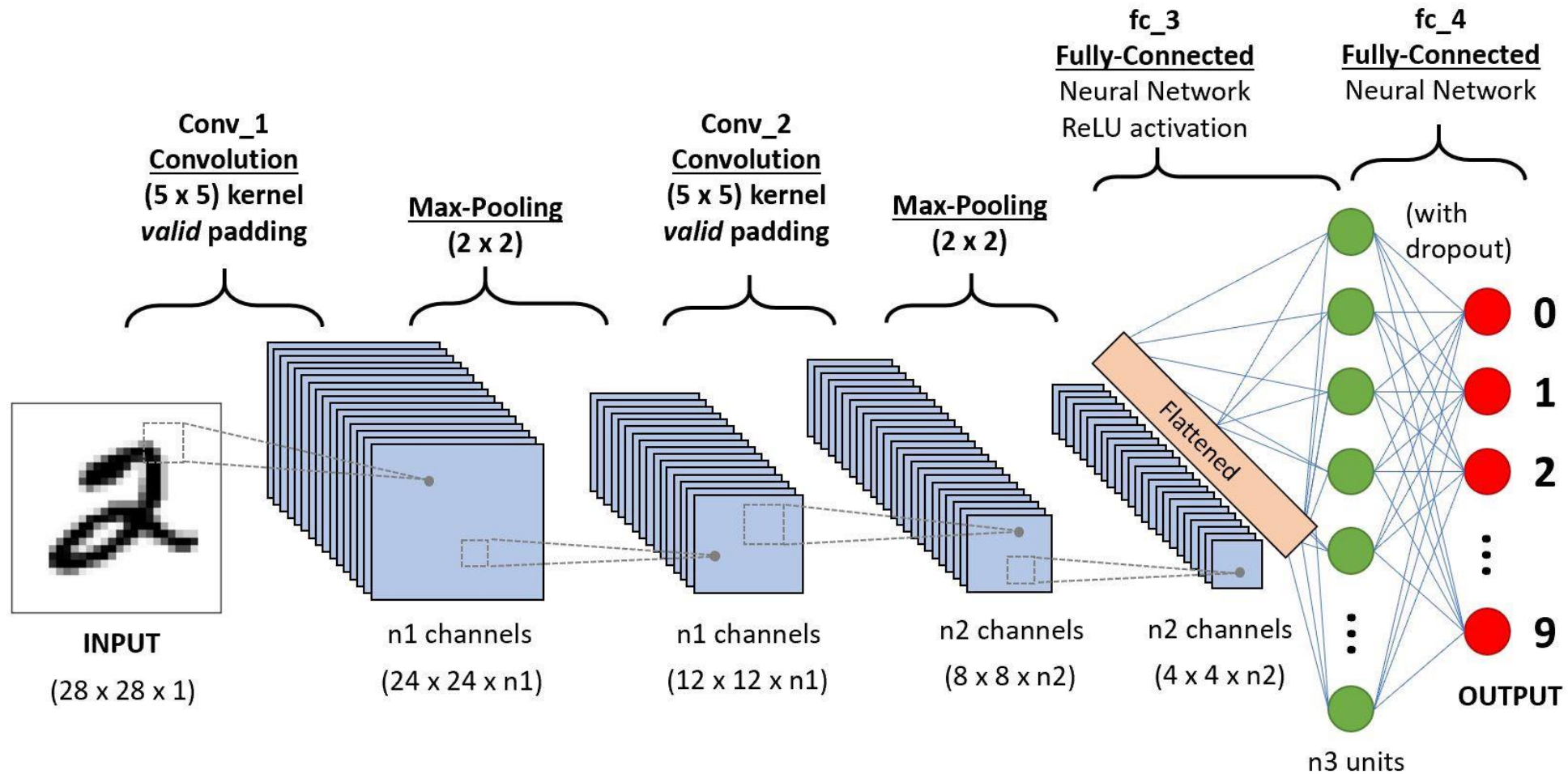
- Пиксель на границе равен ближайшему пикселю из изображения
- Модель всё ещё может настроиться под паттерны, которые возникают из-за такого паддинга

Резюме

- Паддинг позволяет контролировать размер выходных изображений
- Паддинг позволяет учитывать даже объекты на краях
- Разные типы паддингов допускают разные способы переобучения под края

Структура свёрточных сетей

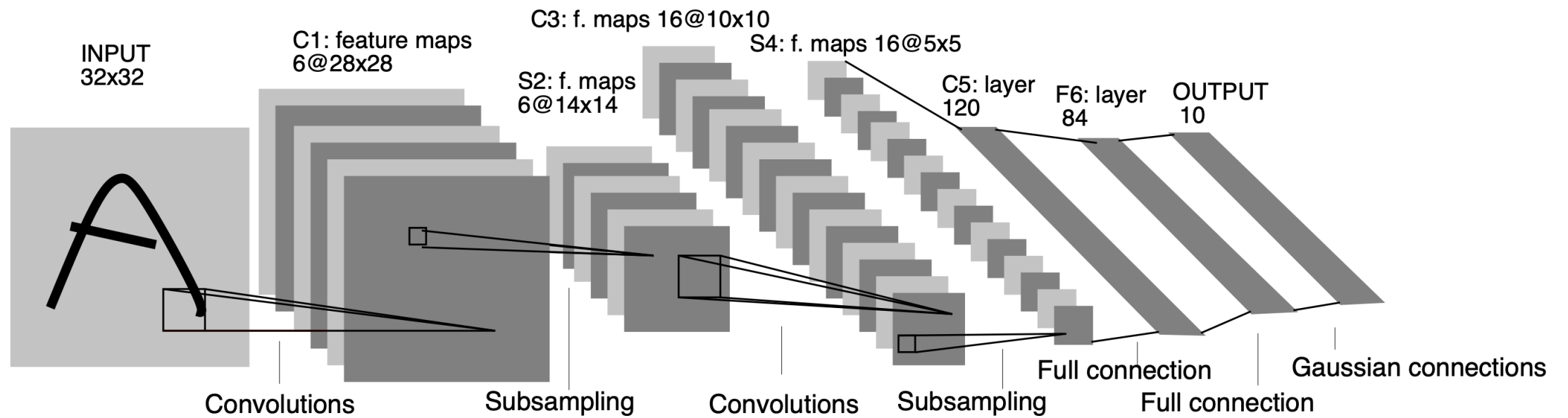
Типичная архитектура



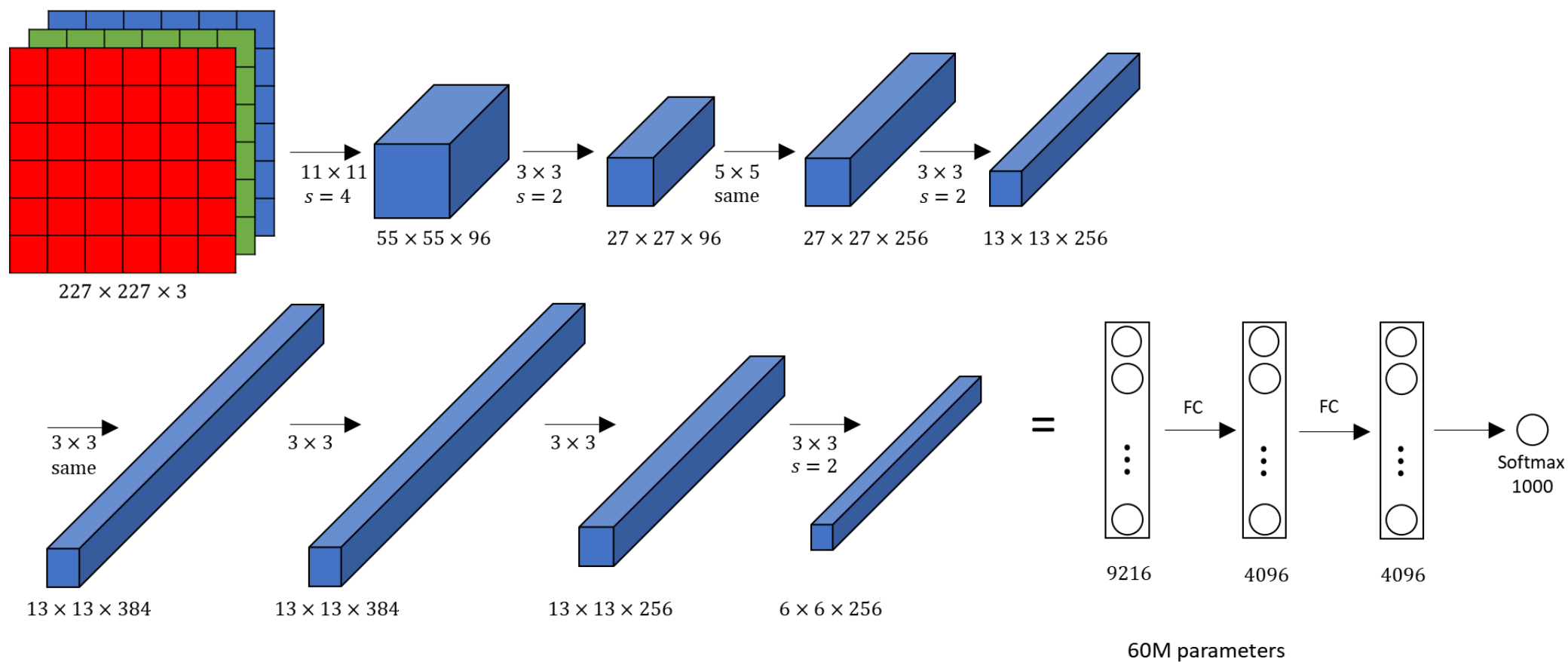
Типичная архитектура

- Последовательное применение комбинаций вида «свёрточный слой -> нелинейность -> pooling» или «свёрточный слой -> нелинейность»
- Выпрямление (flattening) выхода очередного слоя
- Серия полносвязных слоёв

LeNet



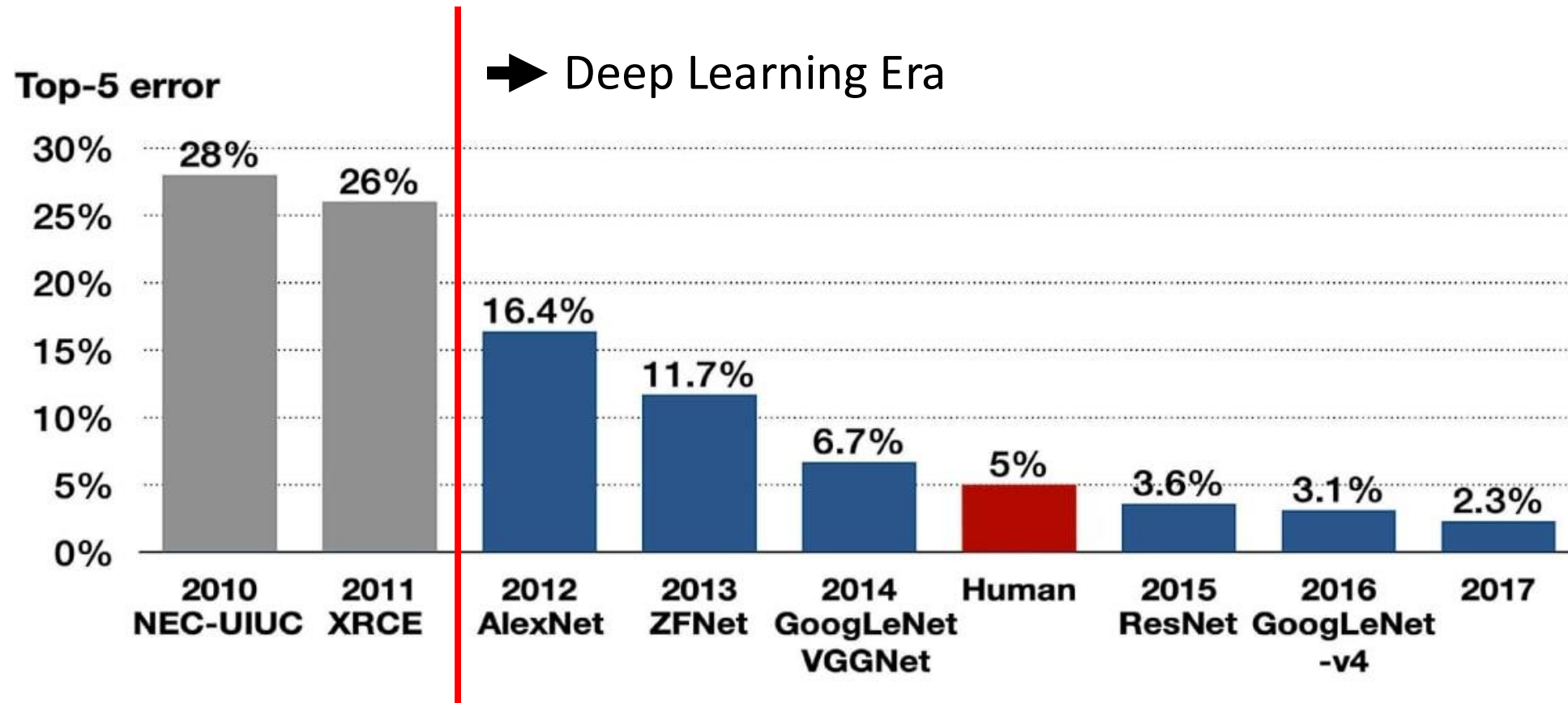
AlexNet



ImageNet

Large Scale Visual Recognition Challenge (ILSVRC)

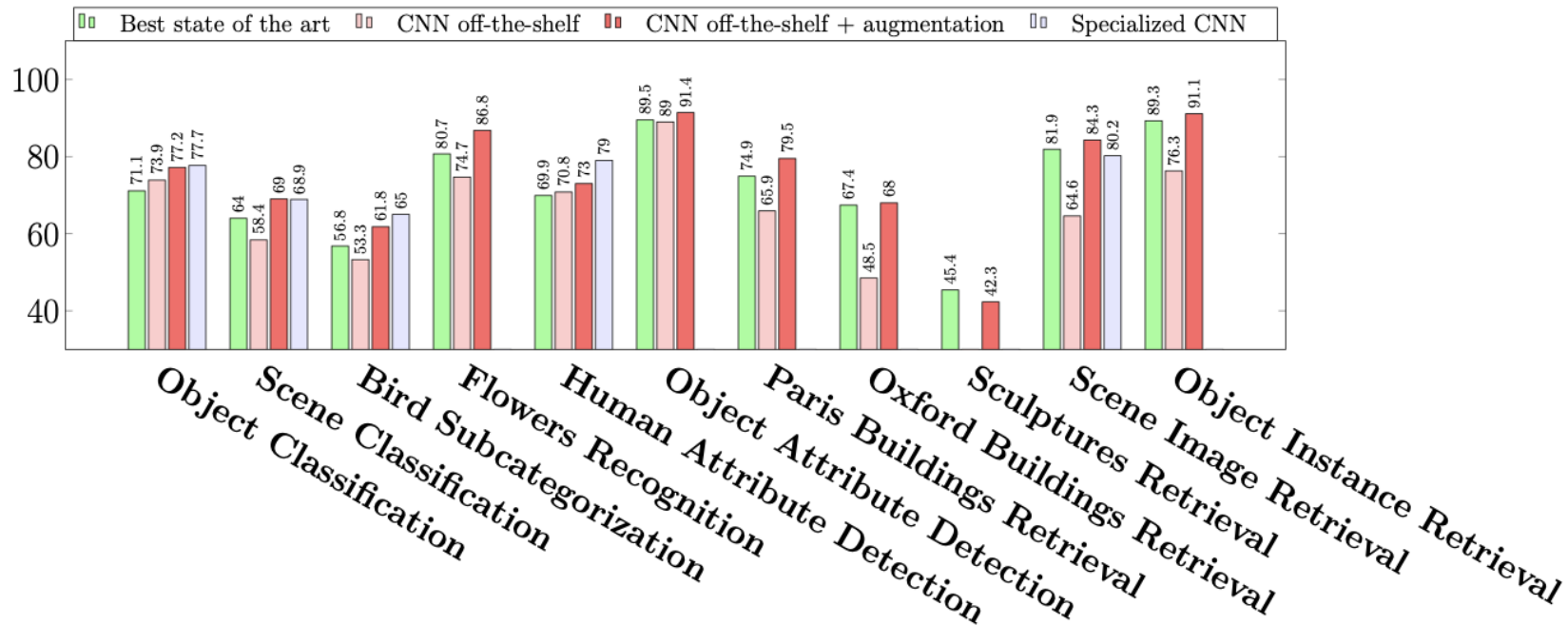
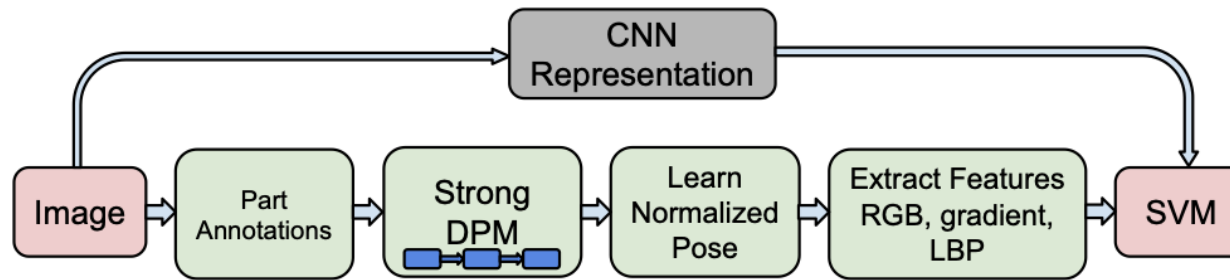
- 14M+ картинок 256x256
- 1000 классов



Представления с последних слоёв

- Важное наблюдение: выходы полносвязных слоёв являются хорошими признаковыми описаниями изображений
- Полезны во многих задачах
- Например, поиск похожих изображений

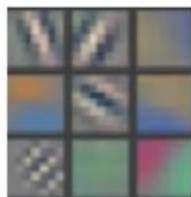
Представления с последних слоёв



Представления с последних слоёв

- Не интерпретируется (в отличие от классического компьютерного зрения)
- По смыслу — «индикаторы» наличия каких-то паттернов

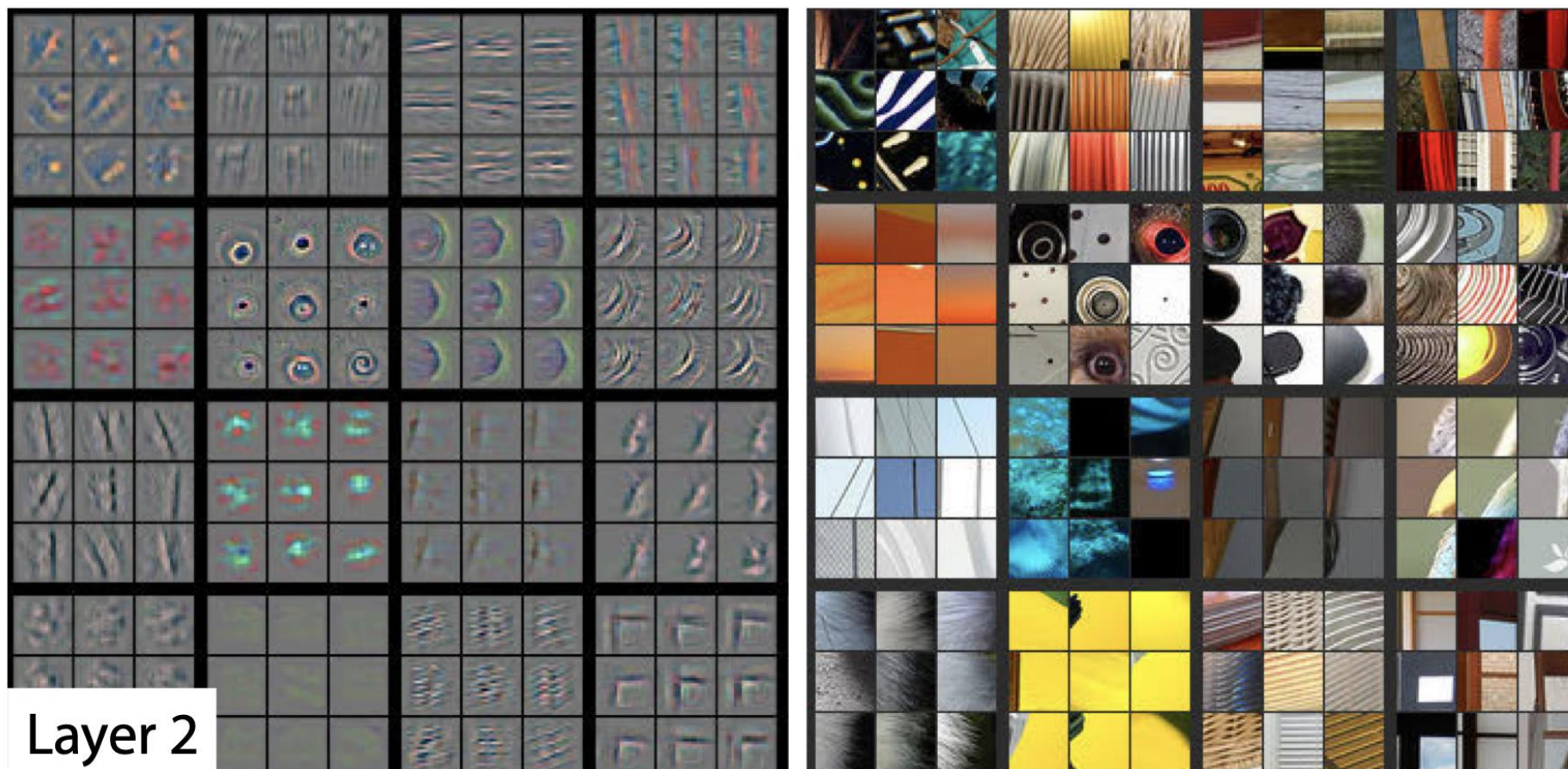
Представления с последних слоёв



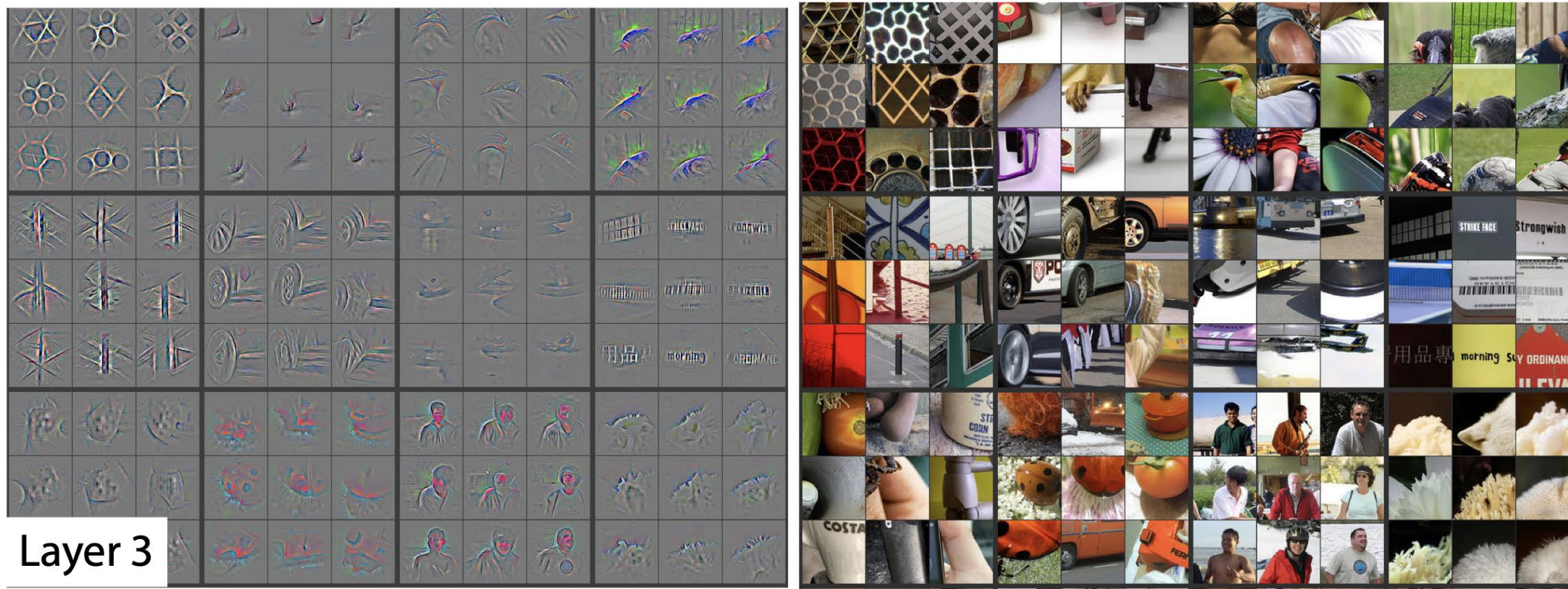
Layer 1



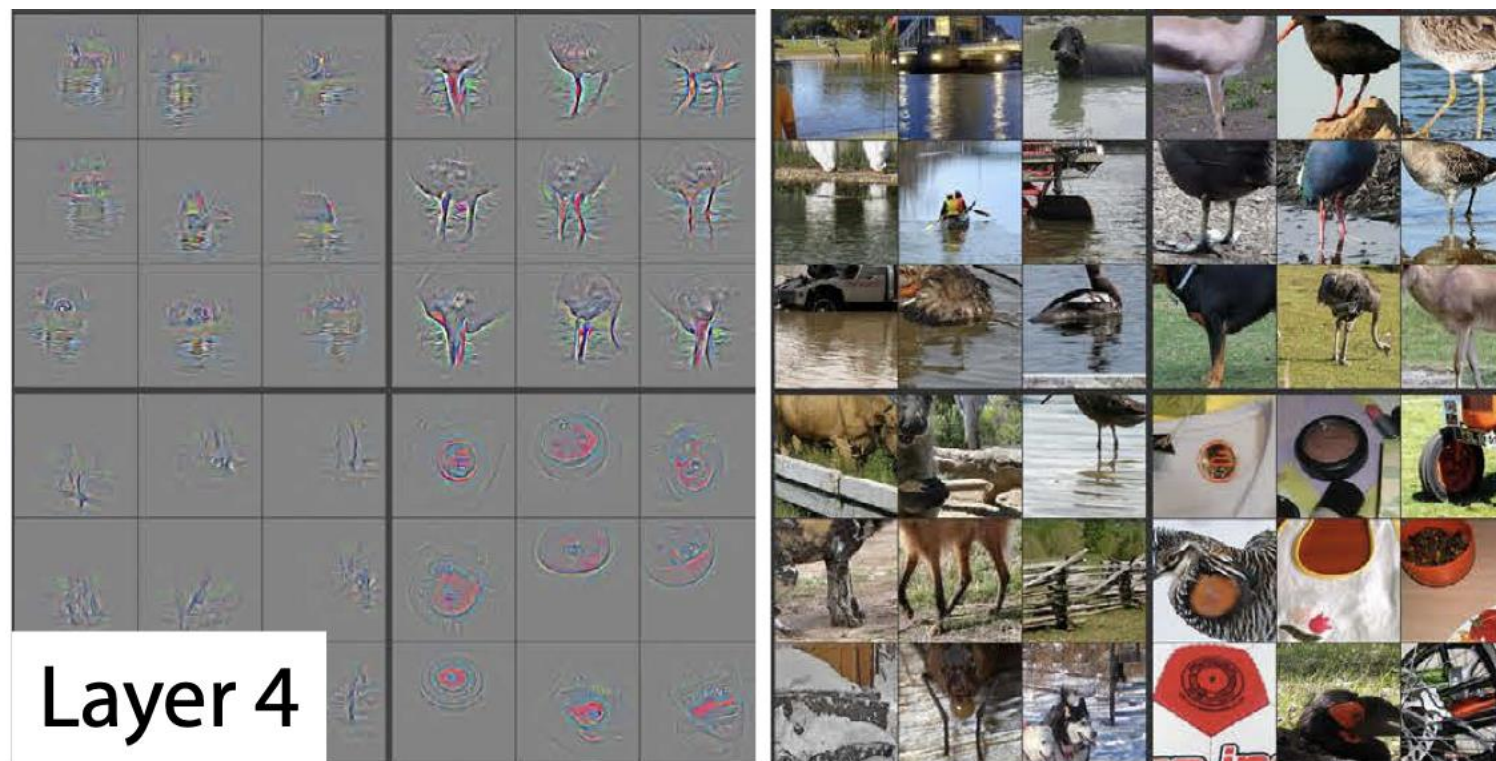
Представления с последних слоёв



Представления с последних слоёв



Представления с последних слоёв



Представления с последних слоёв

