

# Машинное обучение

Лекция 13

Анализ изображений. Свёрточные нейронные сети.

Михаил Гуцин

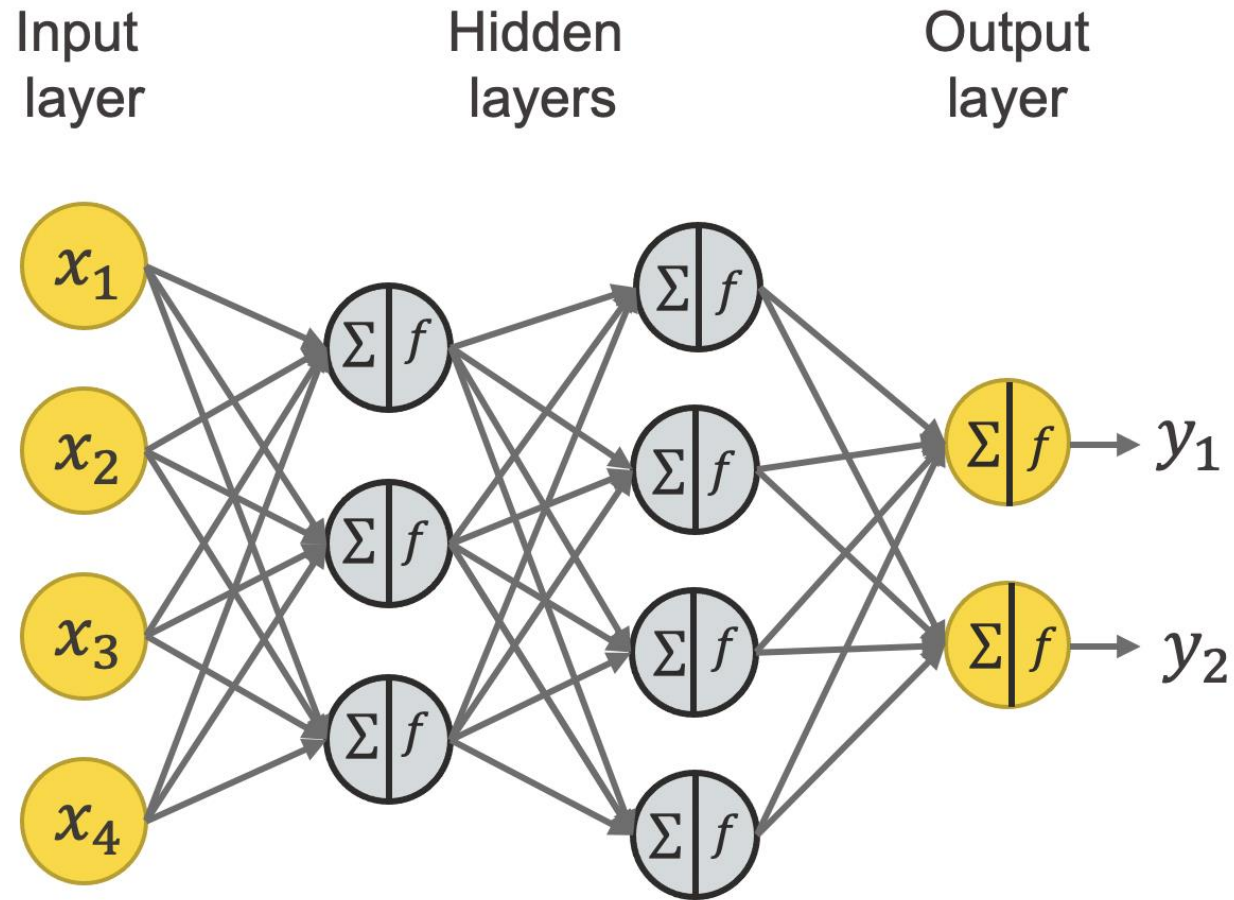
[mhushchyn@hse.ru](mailto:mhushchyn@hse.ru)

НИУ ВШЭ, 2024



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ

# На прошлой лекции



# Как компьютер видит изображения

---



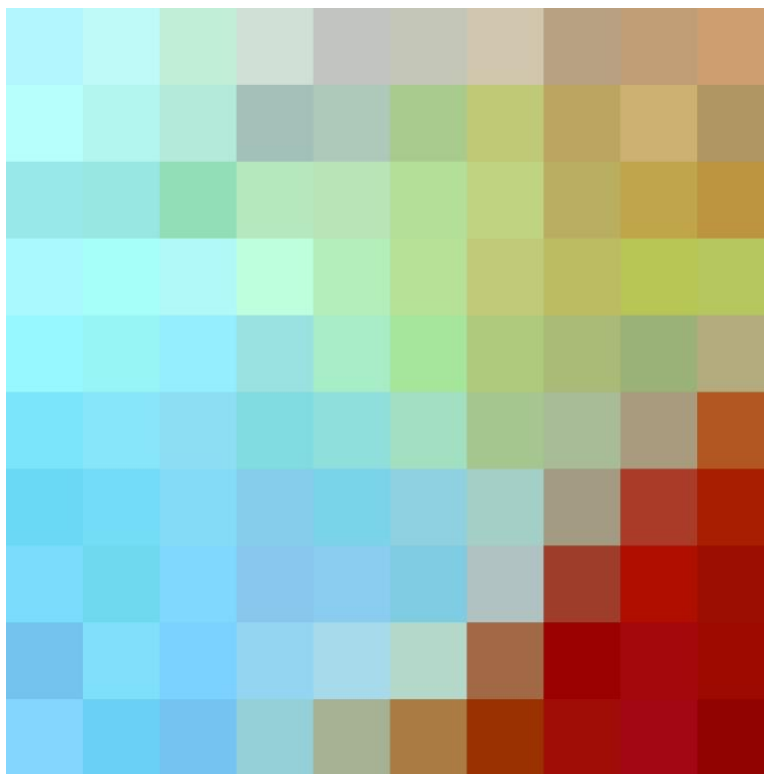
# Изображение



# Пиксели

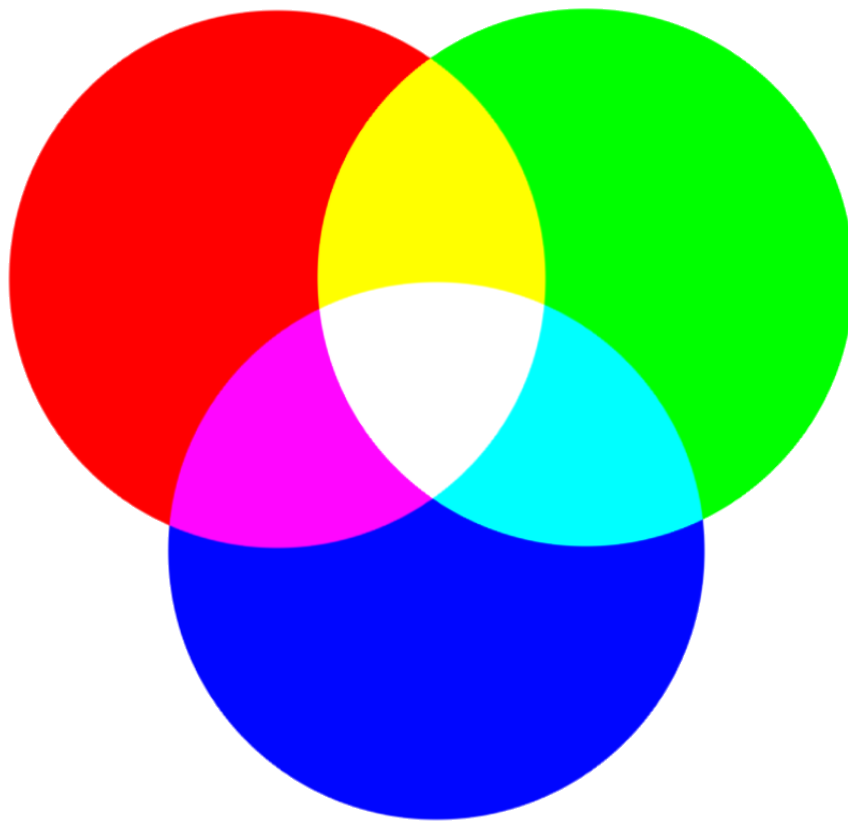


# Пиксели



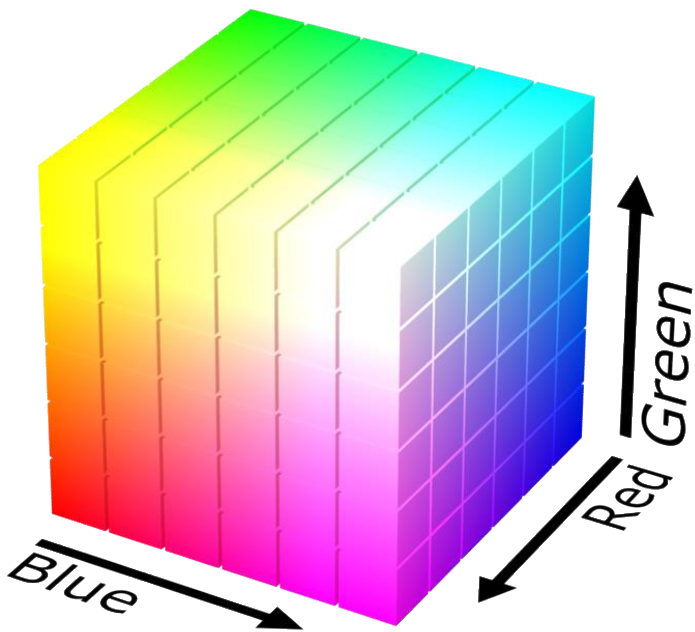
- ▶ Изображение состоит из матрицы пикселей
- ▶ Каждый пиксель имеет свой цвет

# Цветовая модель



- ▶ Цвета пикселей представляются в виде комбинации красного (**R**), зеленого (**G**) и синего (**B**) цветов
- ▶ Такой способ называется RGB цветовой моделью

# Цветовая модель



- ▶ Любой цвет задается тремя числами от 0 до 255:

(**R**, **G**, **B**)



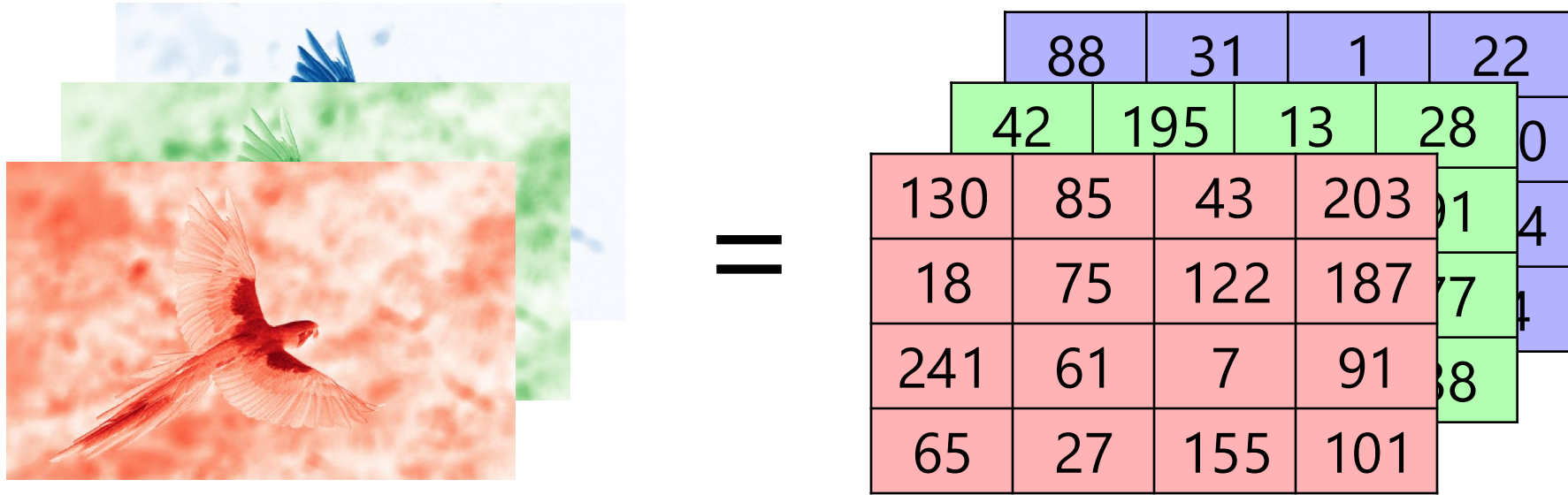
# Примеры

(0, 0, 0)	(150, 150, 150)	(255, 255, 255)
(255, 0, 0)	(0, 255, 0)	(0, 0, 255)
(255, 64, 255)	(255, 252, 121)	(148, 23, 81)

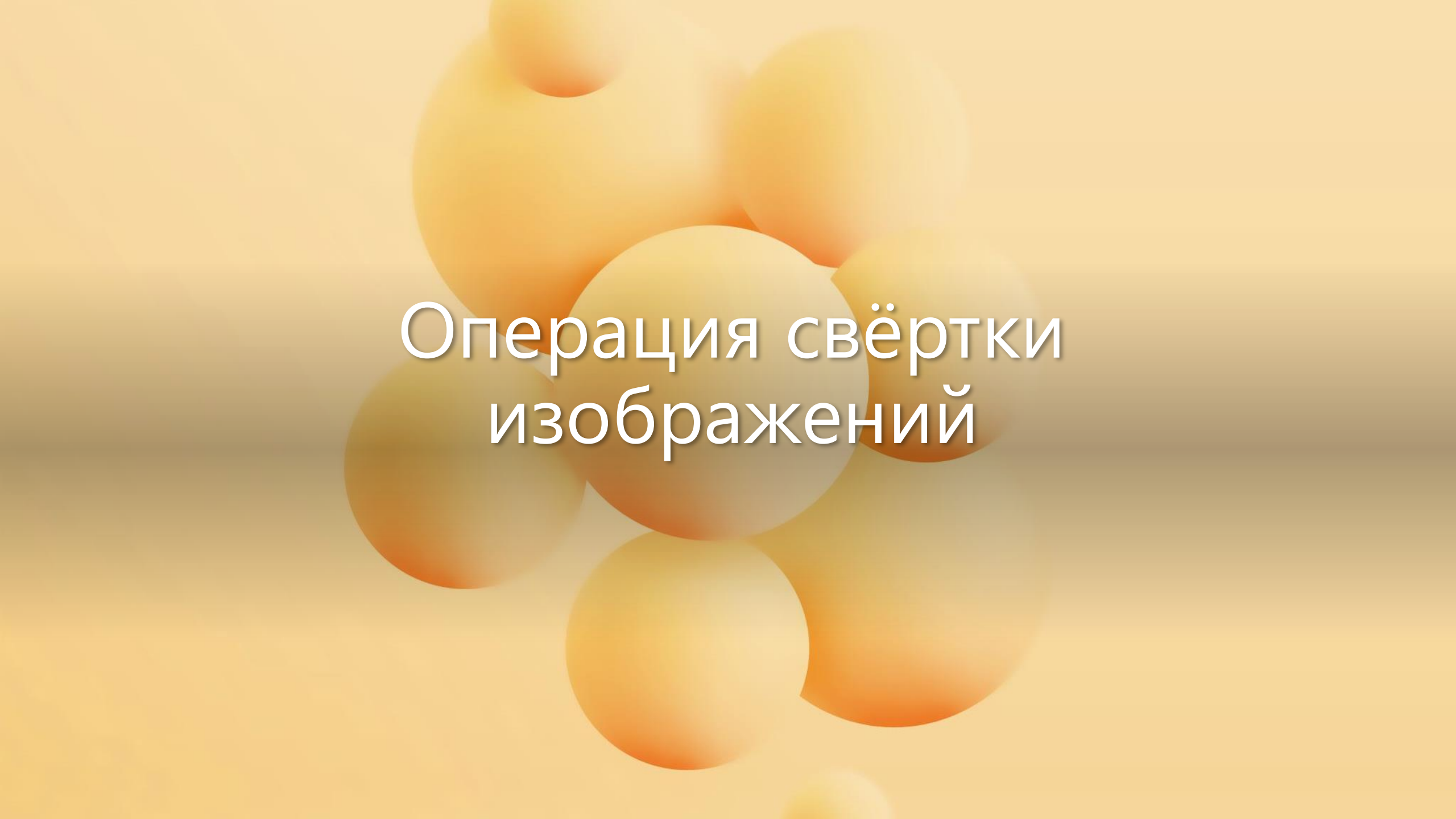
# Разложение изображений



# Каналы изображений



- ▶ Изображение представляется в виде трех матриц
- ▶ Каждая матрица соответствует одному из цветовых **каналов**

The background of the slide is a light beige color, decorated with several overlapping circles in shades of orange and yellow. These circles are of varying sizes and are positioned in a way that they appear to be floating or clustered together, creating a soft, abstract pattern.

# Операция свёртки изображений



# Мотивация

- ▶ Изображение в HD разрешении имеет 1280 x 720 пикселей
- ▶ Каждый пиксель имеет 3 цветовых канала
- ▶ Общее количество признаков  $N$ :

$$N = 1280 * 720 * 3 = 2\,764\,800$$

- ▶ Слишком большая размерность для полносвязной сети

# Пример



=

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

- Рассмотрим изображение с одним каналом

# Операция свертки

Входное  
изображение

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

Ядро  
3x3

0	1	0
-	2	-
1		1
0	1	0

\*

=

Свернутое  
изображение

9			

$$\begin{aligned} &3 * 0 + 3 * 1 + 5 * 0 - \\ &3 * 1 + 4 * 2 - 0 * 1 + \\ &5 * 0 + 1 * 1 + 2 * 0 = 9 \end{aligned}$$

# Операция свертки

Входное  
изображение

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

Ядро  
3x3

0	1	0
-	2	-
1		1
0	1	0

\*

=

Свернутое  
изображение

9	2		

$$\begin{aligned} &3 * 0 + 5 * 1 + 1 * 0 - \\ &4 * 1 + 0 * 2 - 1 * 1 + \\ &1 * 0 + 2 * 1 + 3 * 0 = 2 \end{aligned}$$



# Операция свертки

Входное  
изображение

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

Ядро  
3x3

0	1	0
-	2	-
1		1
0	1	0

\*

=

Свернутое  
изображение

9	2	2	

$$\begin{aligned} &5 * 0 + 1 * 1 + 2 * 0 - \\ &0 * 1 + 1 * 2 - 4 * 1 + \\ &2 * 0 + 3 * 1 + 2 * 0 = 2 \end{aligned}$$

# Операция свертки

Входное  
изображение

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

Ядро  
3x3

0	1	0
-1	2	-1
0	1	0

\*

=

Свернутое  
изображение

9	2	2	6
-1	4	8	7
-4	8	8	0
2	9	7	4

**Свертка** - это процесс сложения соседних элементов изображения, взвешиваемых ядром.

# Пример: детектор вертикальных границ

Входное  
изображение

2	2	2	0	0	0
2	2	2	0	0	0
2	2	2	0	0	0
2	2	2	0	0	0
2	2	2	0	0	0
2	2	2	0	0	0

Ядро  
3x3

1	0	-1
1	0	-1
1	0	-1

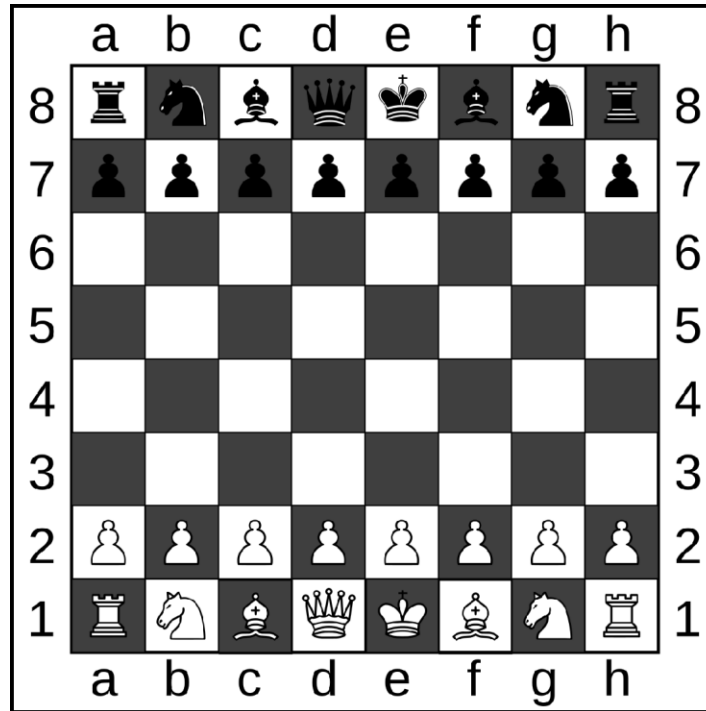
\*

=

Свернутое  
изображение

0	6	6	0
0	6	6	0
0	6	6	0
0	6	6	0

# Пример: детектор горизонтальных границ



<https://testingweb.r-cms.jp/>

\*

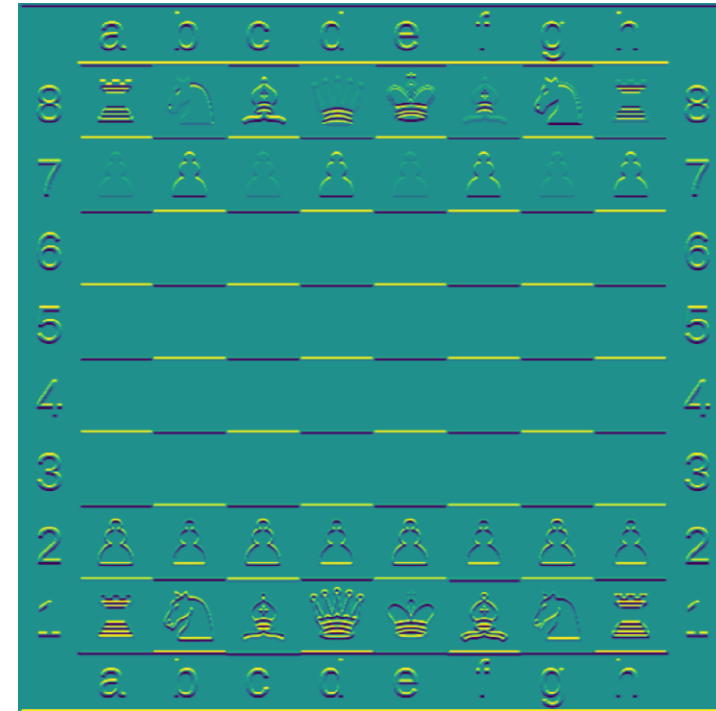
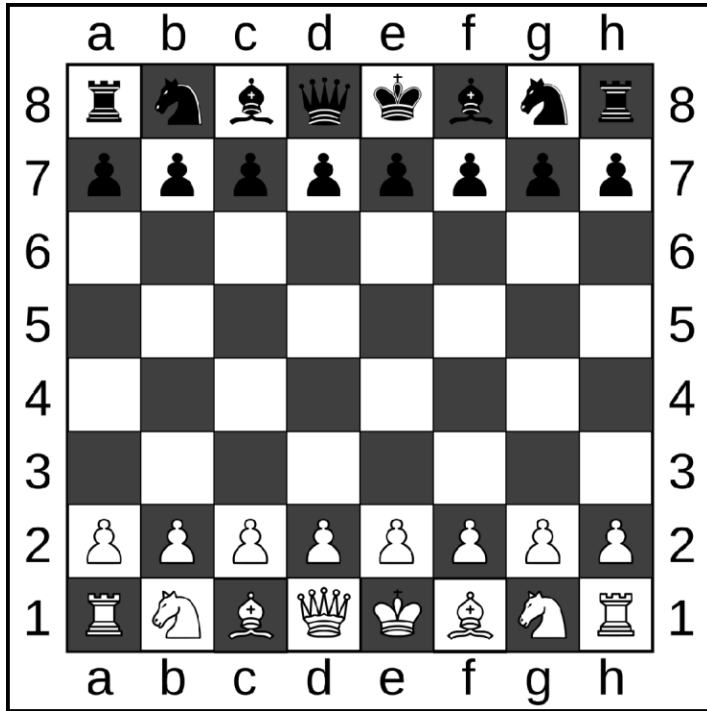
Оператор  
Собеля

1	2	1
0	0	0
-	-	-
1	2	1

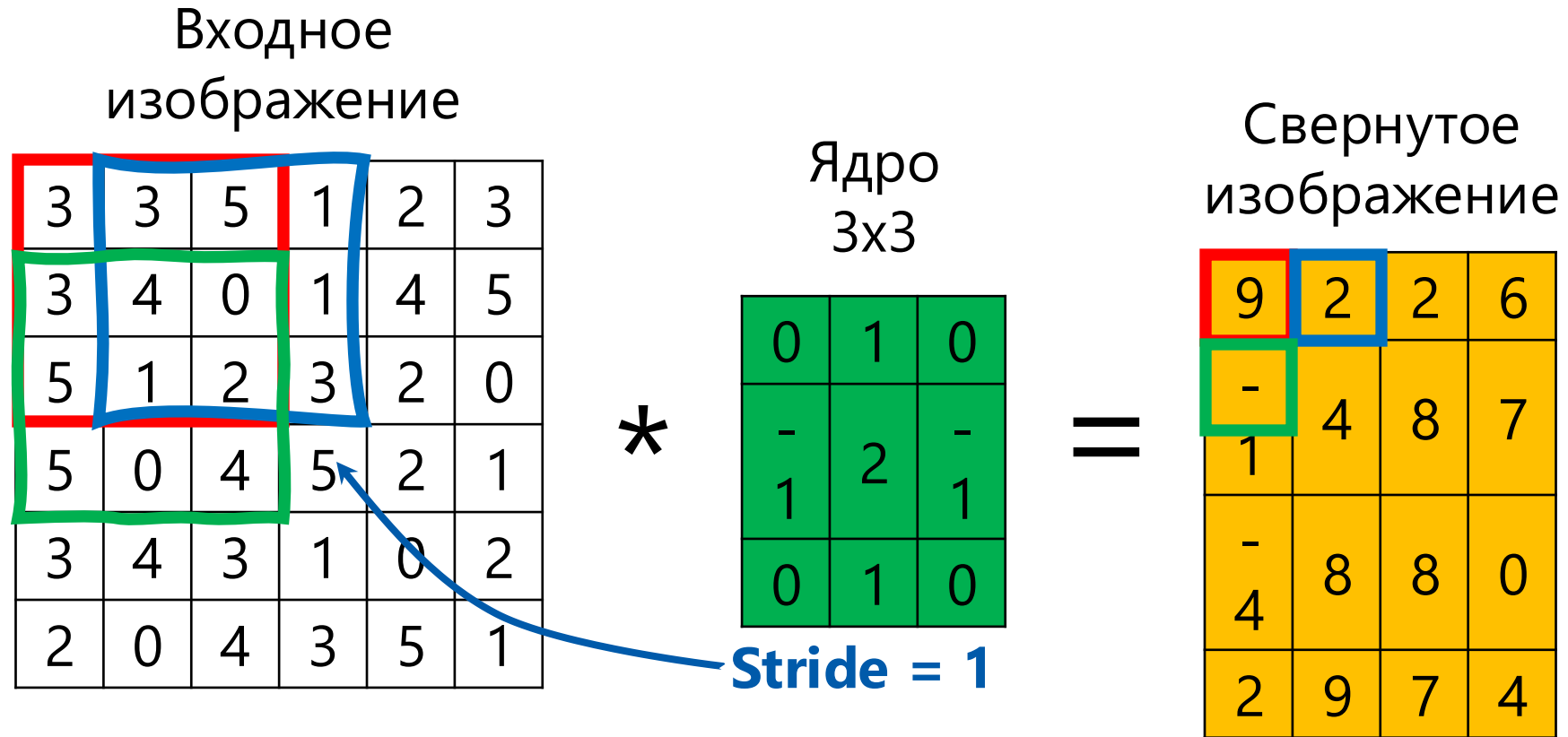
Применим оператор Собеля для обнаружения горизонтальных линий.



# Пример: детектор горизонтальных границ



# Шаг свертки



**Шаг свертки (stride)** - это величина сдвига ядра между соседними операциями свертки изображения.

# Шаг свертки

Входное  
изображение

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

Ядро  
3x3

0	1	0
-	2	-
1	2	1
0	1	0

\*

Stride = 2

=

Свернутое  
изображение

9	2
-	8
4	

# Дополнение изображения

Входное  
изображение

4	4	3	2
2	3	4	4
1	1	3	2
1	2	2	2

Ядро  
3x3

0	1	0
-	2	-
1		1
0	1	0

\*

=

Свернутое  
изображение

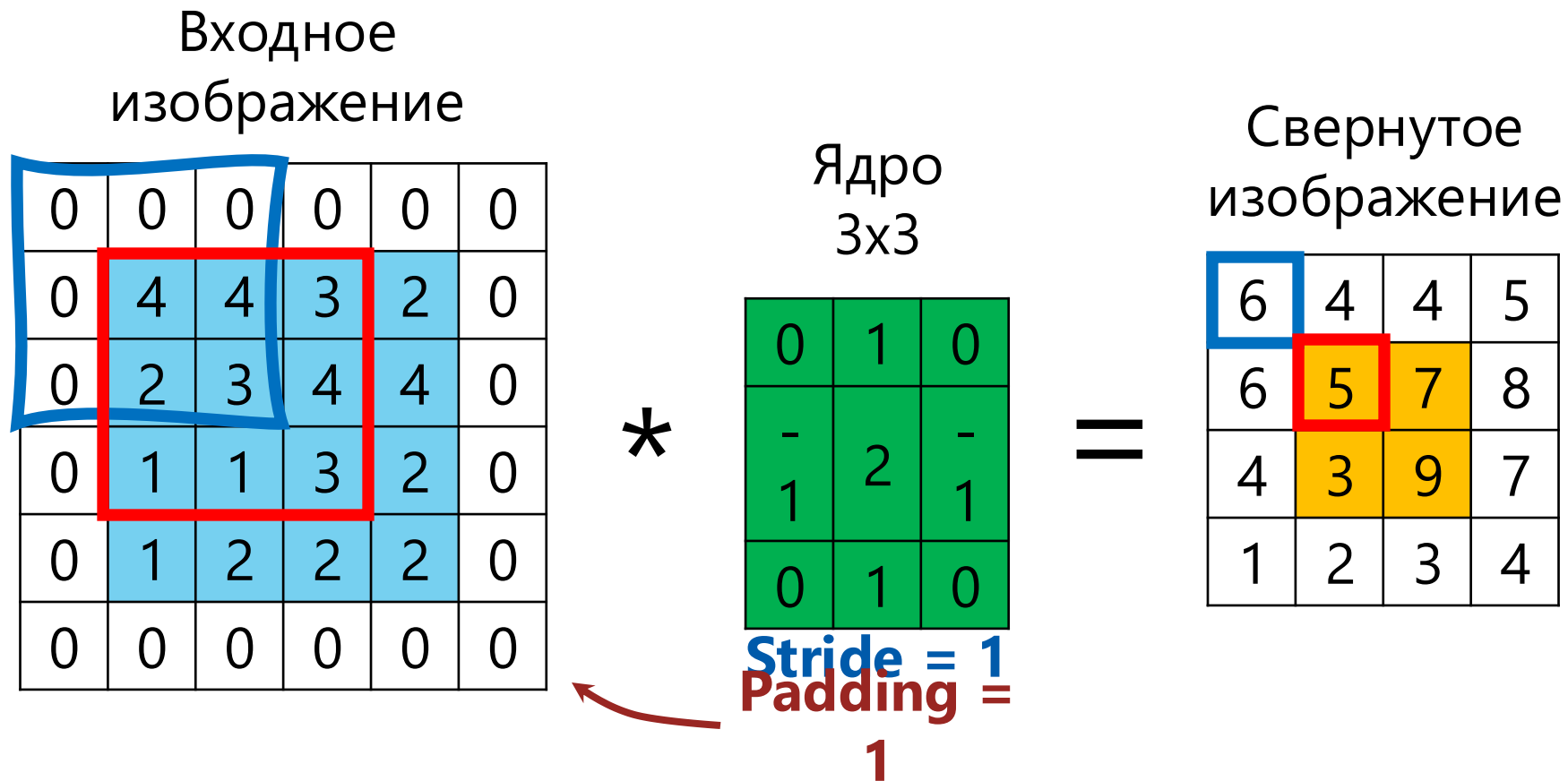
5	7
3	9

Stride = 1

- ▶ Свертка изображения всегда уменьшает его размер.
- ▶ Как этого можно избежать?

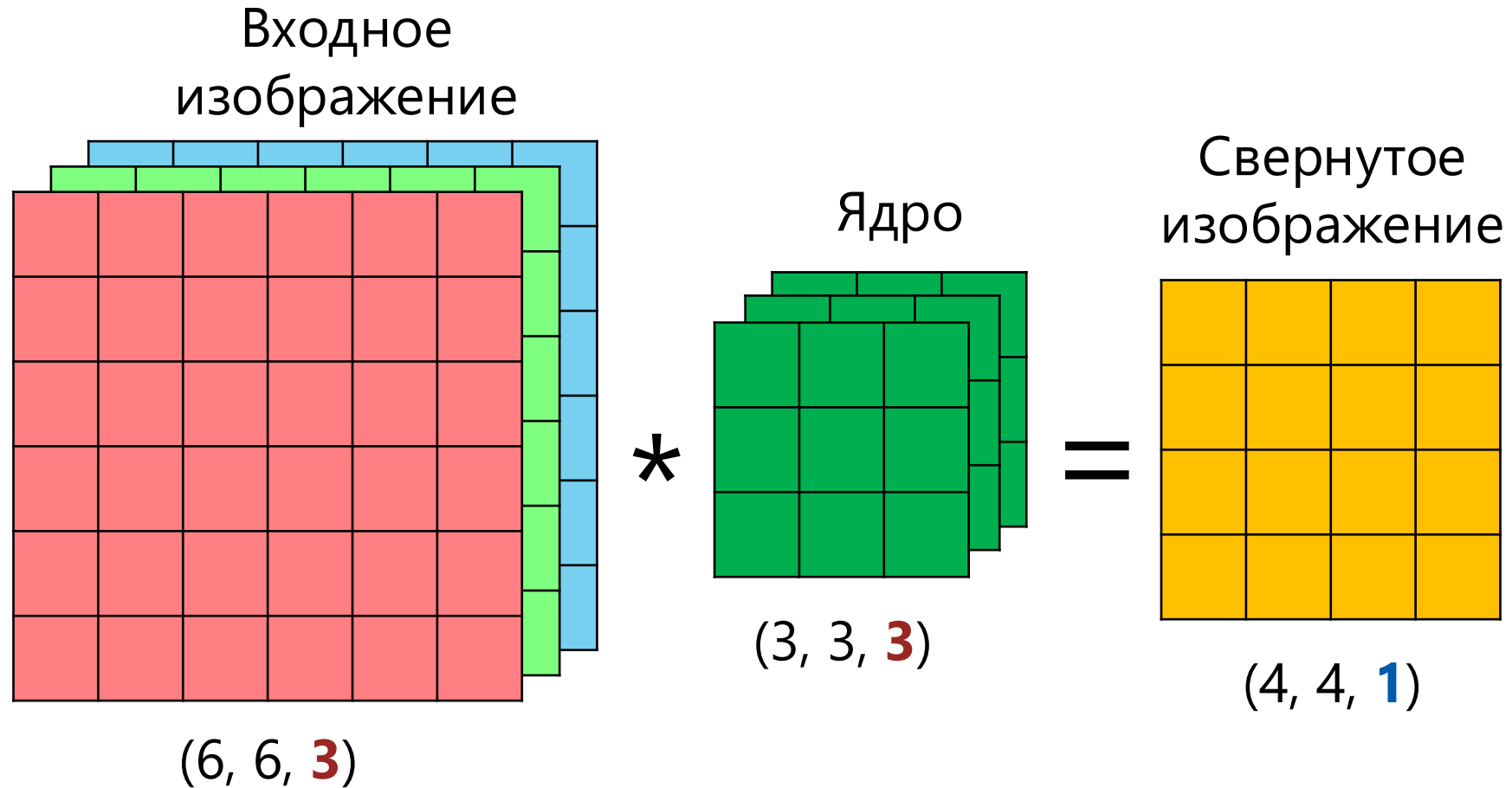


# Дополнение изображения



**Дополнение изображения (padding)** - это искусственное расширение изображения по краям.

# Свертка многоканального изображения



Результат свертки – **одноканальное изображение**.

Операция объединения  
пикселей

# Объединение по максимуму

Входное  
изображение

3	3	5	1	2	3
3	4	0	1	4	5
5	1	2	3	2	0
5	0	4	5	2	1
3	4	3	1	0	2
2	0	4	3	5	1

Ядро  
3x3

	Max()	

Свернутое  
изображение

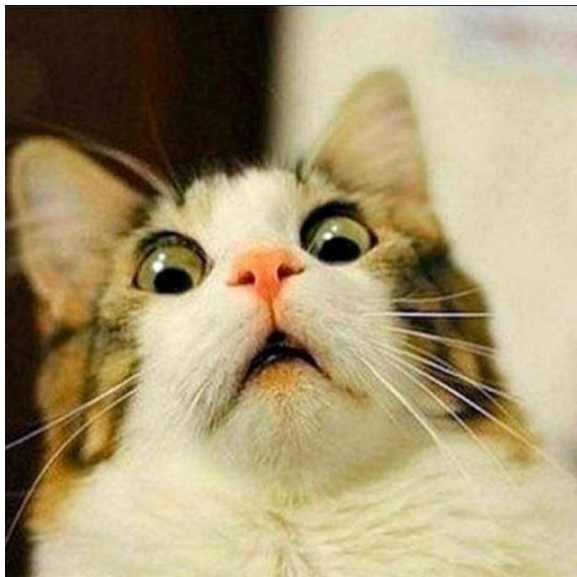
5	5
5	5

- ▶ Объединение по максимуму (MaxPooling) возвращает максимальное значение соседних пикселей.
- ▶ Выбирают шаг так, чтобы ядра не пересекались.

# Свёрточные нейронные сети

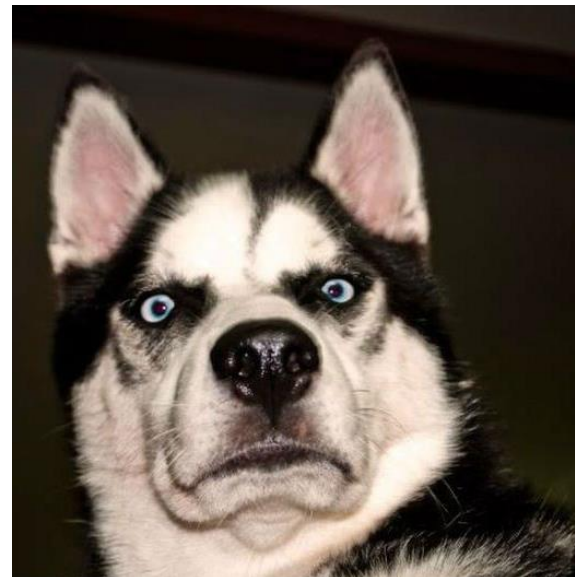
The background of the image is a dark blue field filled with a complex network of glowing, thin blue lines. These lines intersect and branch out, creating a web-like structure. Small, bright orange-yellow dots are scattered throughout the image, often positioned at the intersections of the blue lines, giving the impression of nodes or data points in a network. The overall effect is one of dynamic, interconnected energy.

# Задача



<https://www.facebook.com/katzenbest>

VS



<https://sobakibalabaki.com>

- Рассмотрим задачу бинарной классификации изображений.
- Как подать изображение на вход нейронной сети?



# Выпрямление изображения

Изображение

$x_1$	$x_2$	$x_3$
$x_4$	$x_5$	$x_6$
$x_7$	$x_8$	$x_9$

(3, 3, 1)

**Значения пикселей** в каждом канале изображения –  
входные признаки для классификатора

# Выпрямление изображения

Изображение

$x_1$	$x_2$	$x_3$
$x_4$	$x_5$	$x_6$
$x_7$	$x_8$	$x_9$

(3, 3, 1)



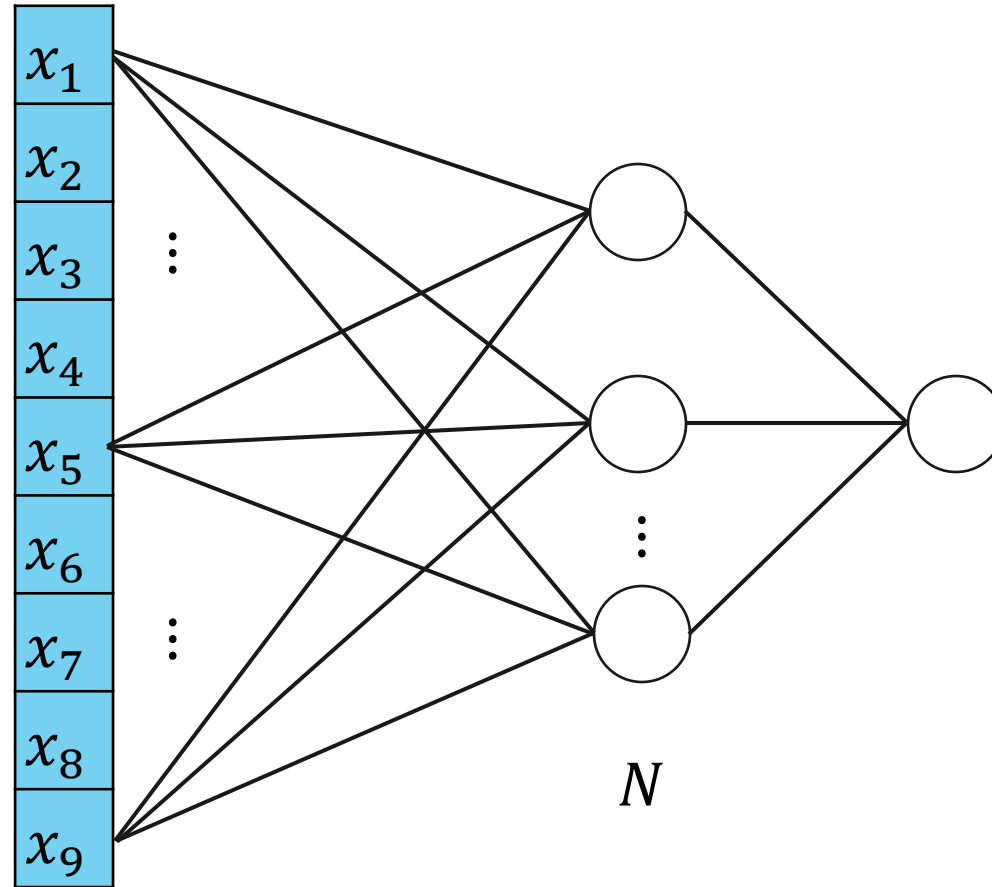
Flattening

$x_1$
$x_2$
$x_3$
$x_4$
$x_5$
$x_6$
$x_7$
$x_8$
$x_9$

Вектор

Развернем значения пикселей во всех каналах в вектор.

# Полносвязная нейронная сеть



Количество весов сети:

$$9 * N + N = 10 * N$$

# Свертка изображения

Изображение

$x_1$	$x_2$	$x_3$
$x_4$	$x_5$	$x_6$
$x_7$	$x_8$	$x_9$

(3, 3, **1**)

\*

Ядро

$w_1$	$w_2$
$w_3$	$w_4$

(2, 2, **1**)

=

Свернутое  
изображение

$z_1$	$z_2$
$z_3$	$z_4$

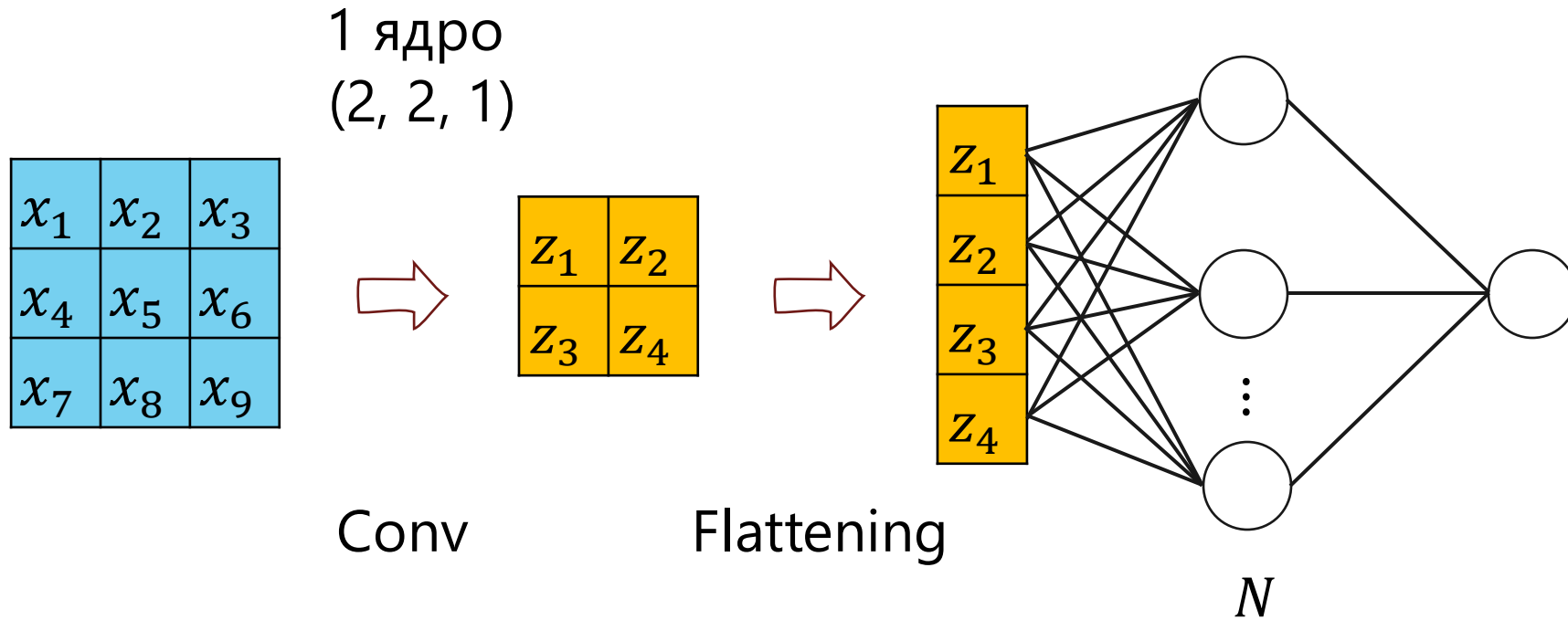
(2, 2, **1**)

Ядро свертки как нейрон:

$$z_1 = \mathbf{w}_1 x_1 + \mathbf{w}_2 x_2 + \mathbf{w}_3 x_4 + \mathbf{w}_4 x_5$$

Веса  $\mathbf{w}_i$  получим в процессе обучения.

# Простая сверточная нейронная сеть



Количество весов сети:

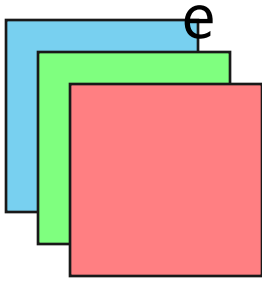
$$4 + 4 * N + N = 5 * N + 4$$

# Преимущества сверток

- ▶ Значительно уменьшают количество весов сети
- ▶ Сети быстрее обучаются
- ▶ Требуют меньше данных
- ▶ Достигают лучшего качества

# Сверточные нейронные сети

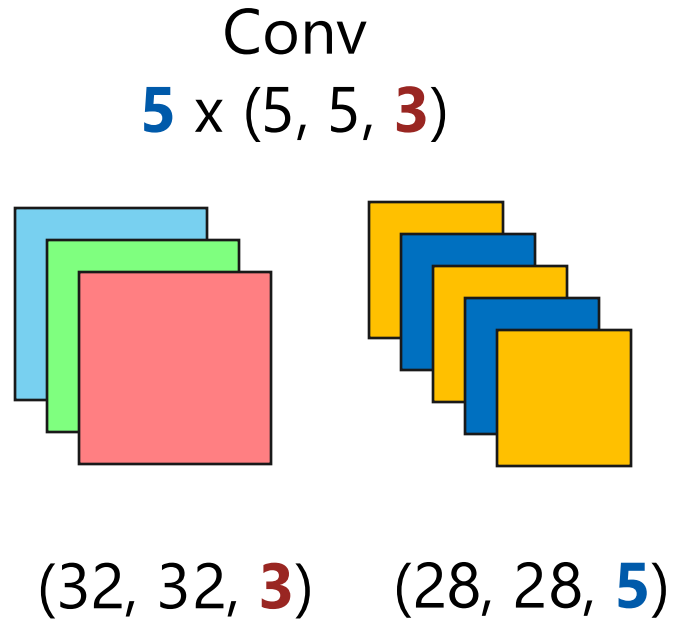
Входное  
изображение



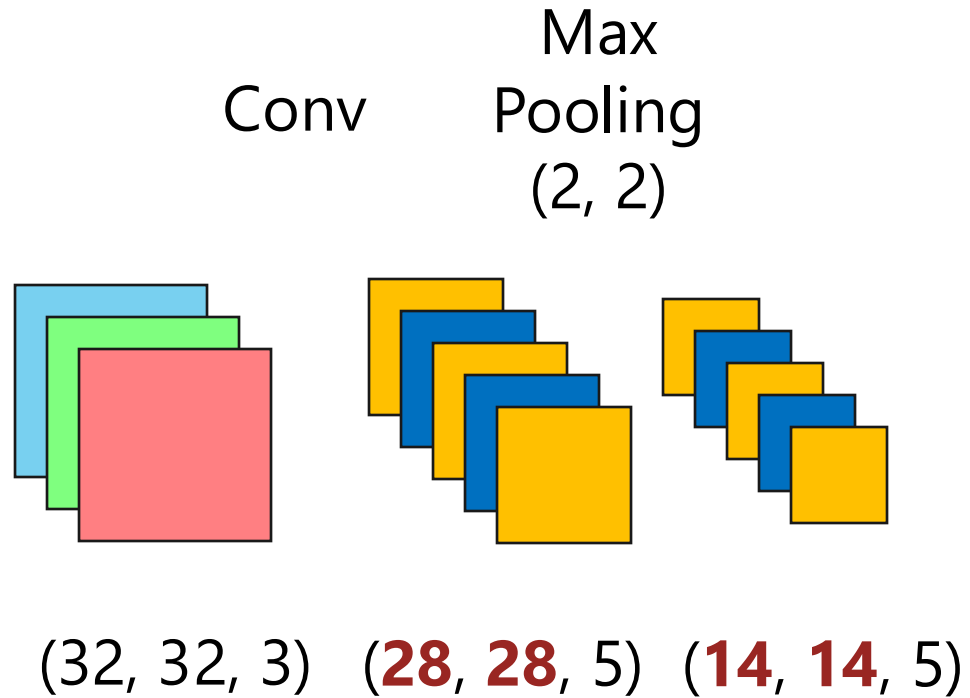
(32, 32, 3)



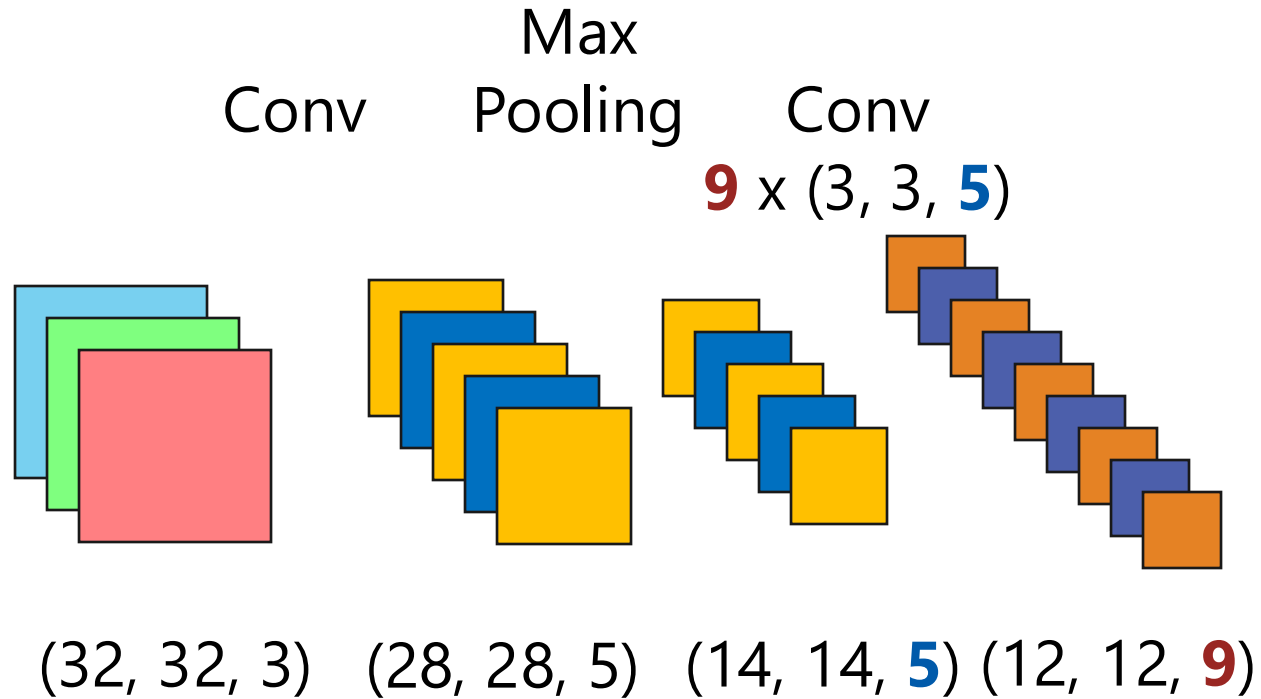
# Сверточные нейронные сети



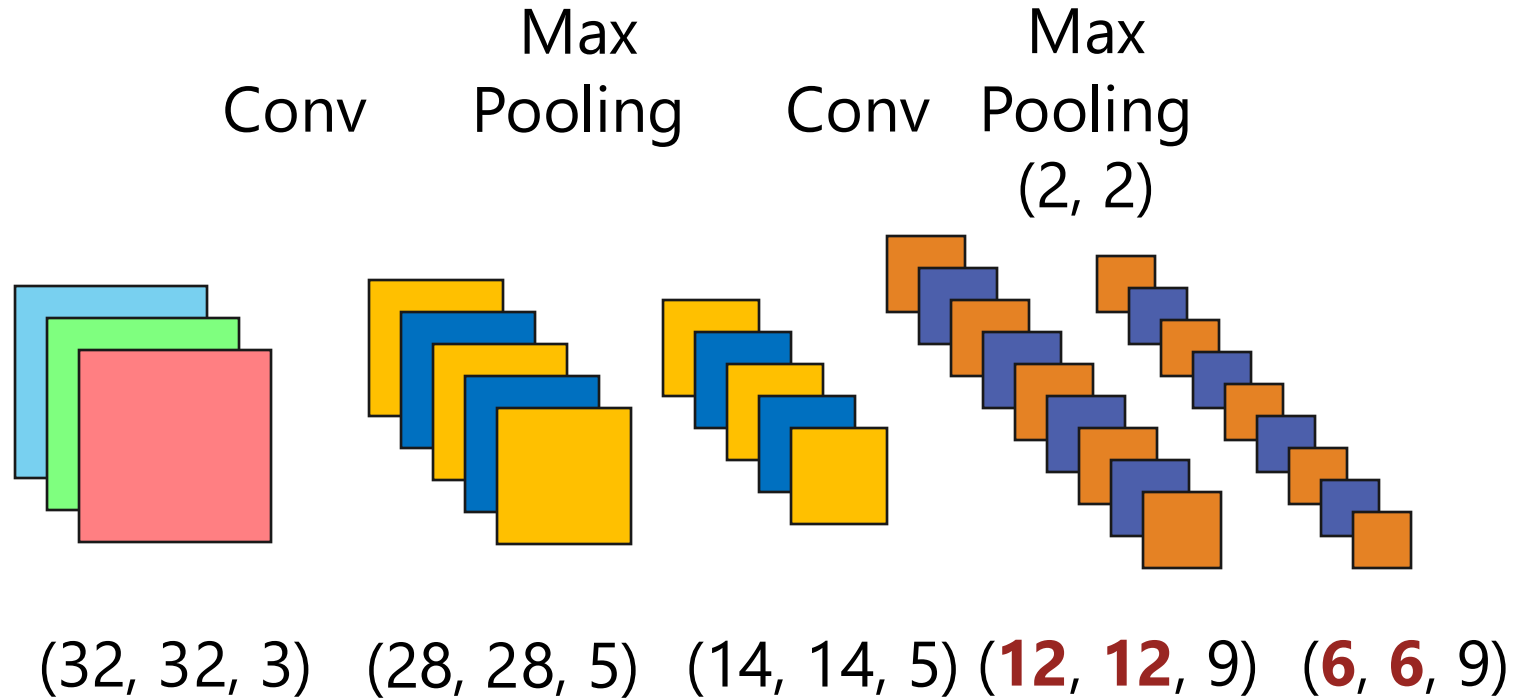
# Сверточные нейронные сети



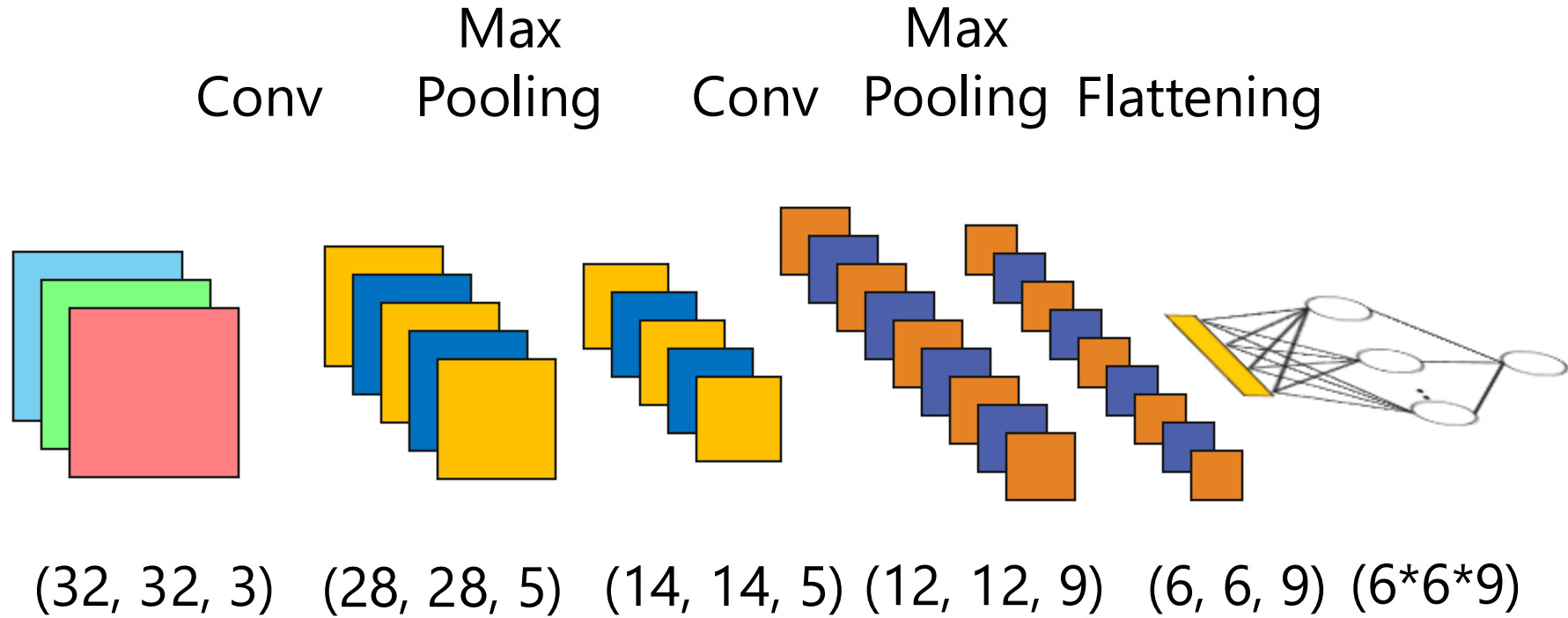
# Сверточные нейронные сети



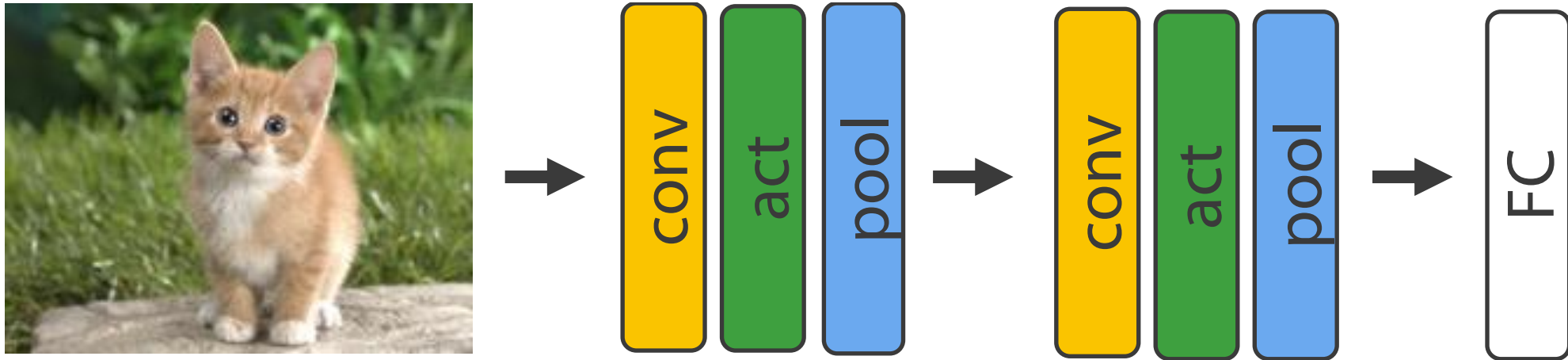
# Сверточные нейронные сети



# Сверточные нейронные сети



# Вычислительный граф

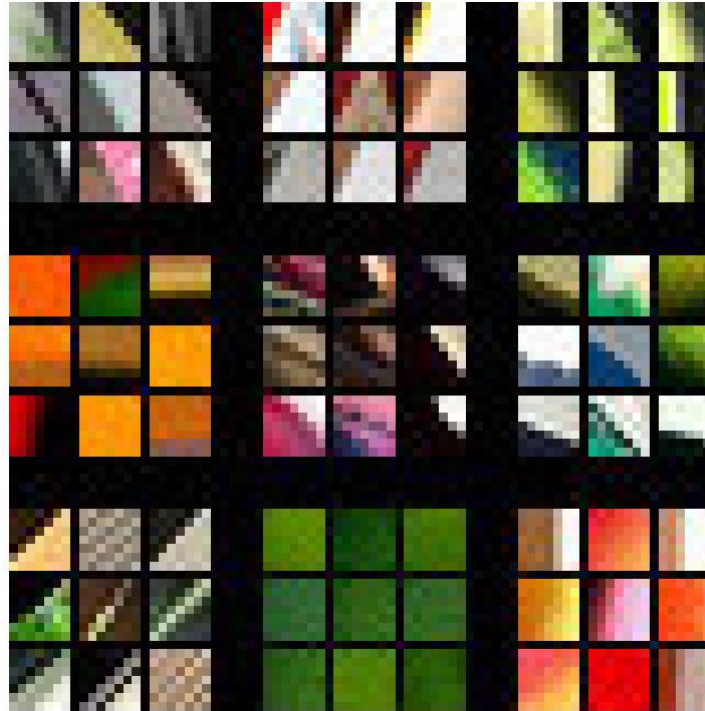


# Типичная архитектура

- ▶ Последовательное применение комбинаций **свёрточный слой** -> **нелинейность** -> **pooling**
- ▶ Выпрямление (**flattening**) выхода последней комбинации
- ▶ Серия полносвязных слоёв

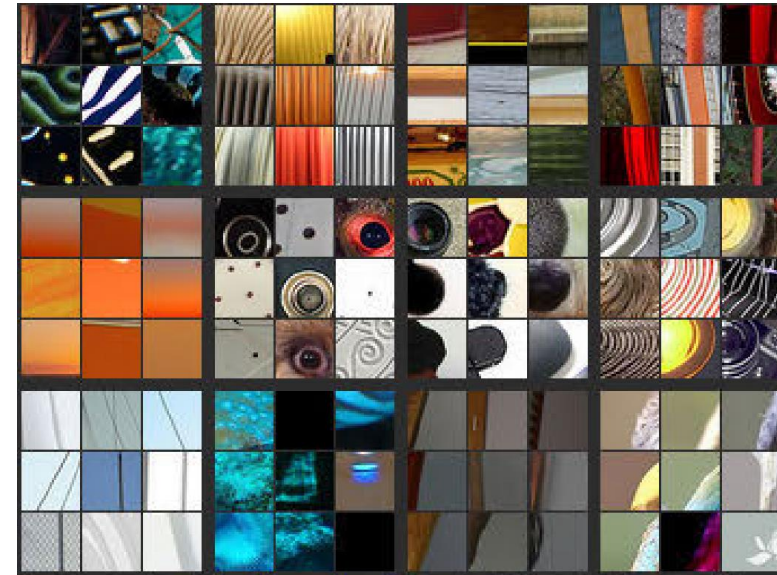
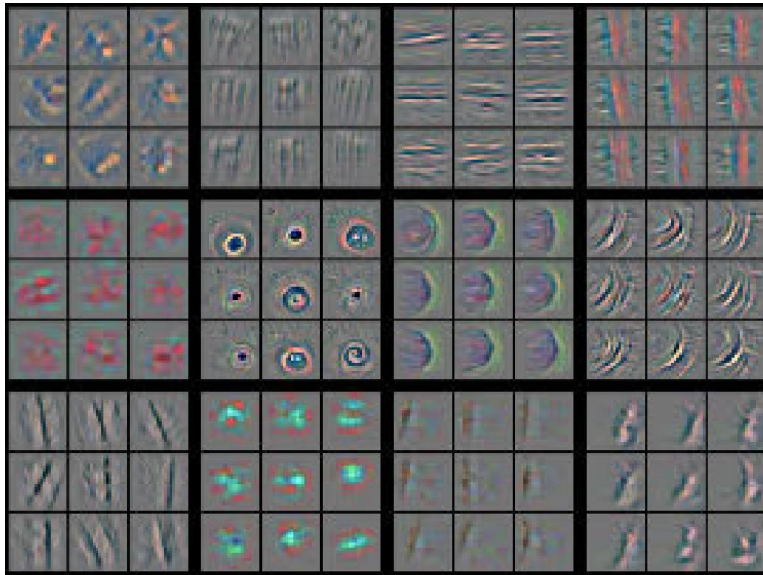


# Сверточный слой 1



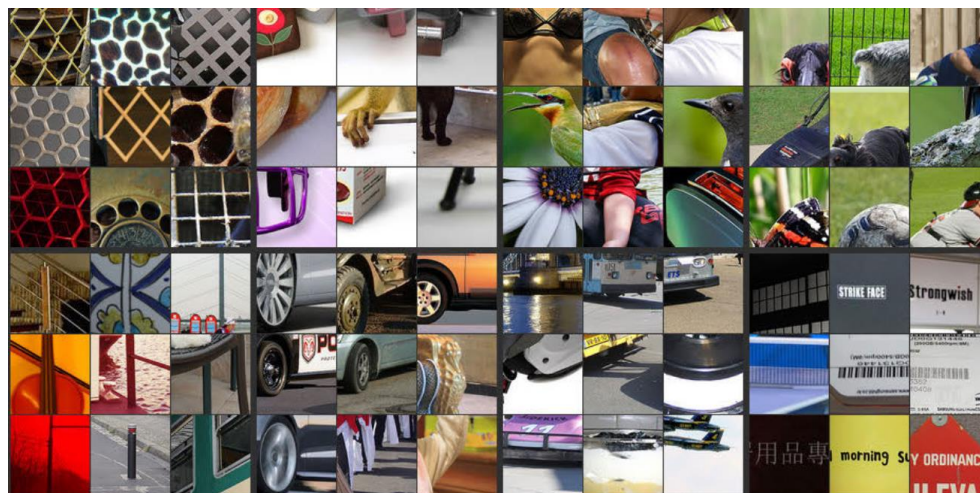
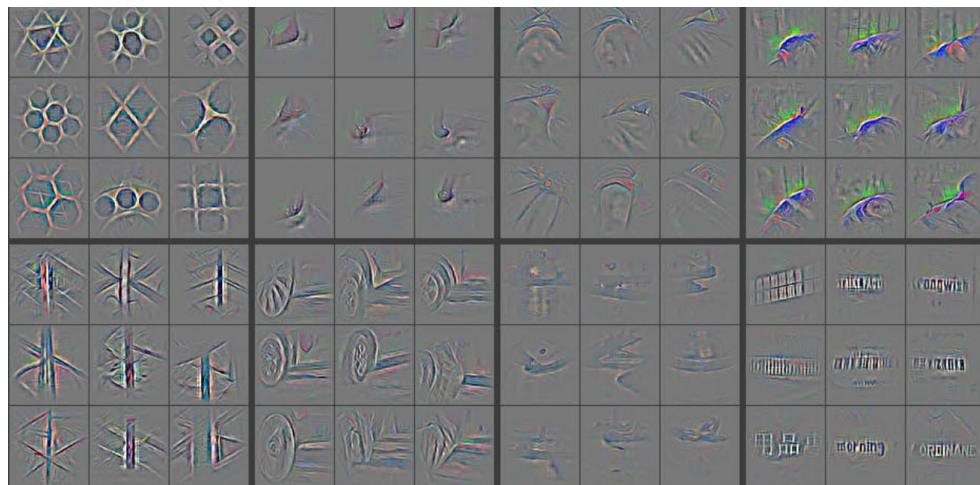
На первом слое сеть учится находить простые шаблоны: прямые линии, границы, цвета.

# Сверточный слой 2



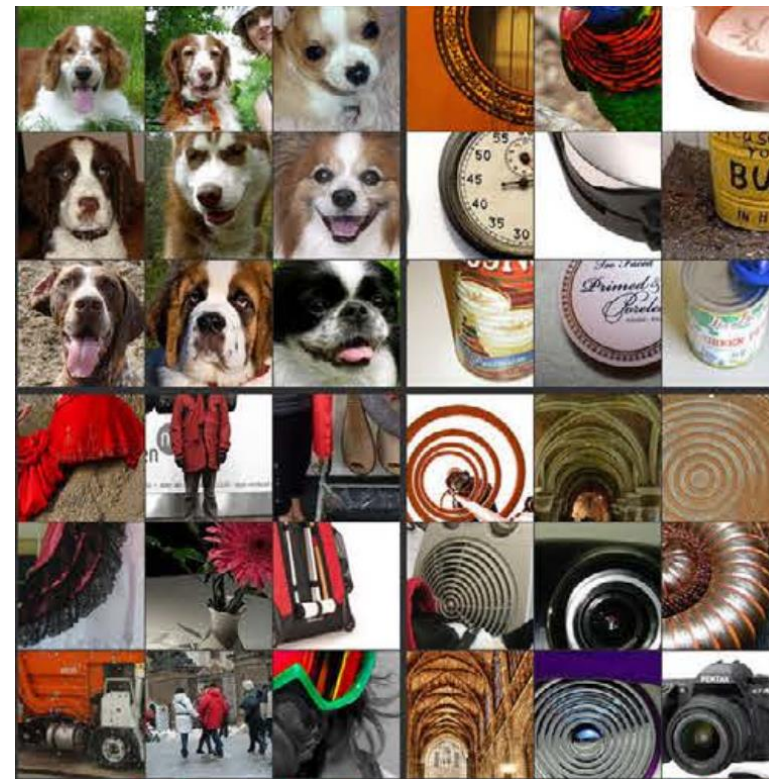
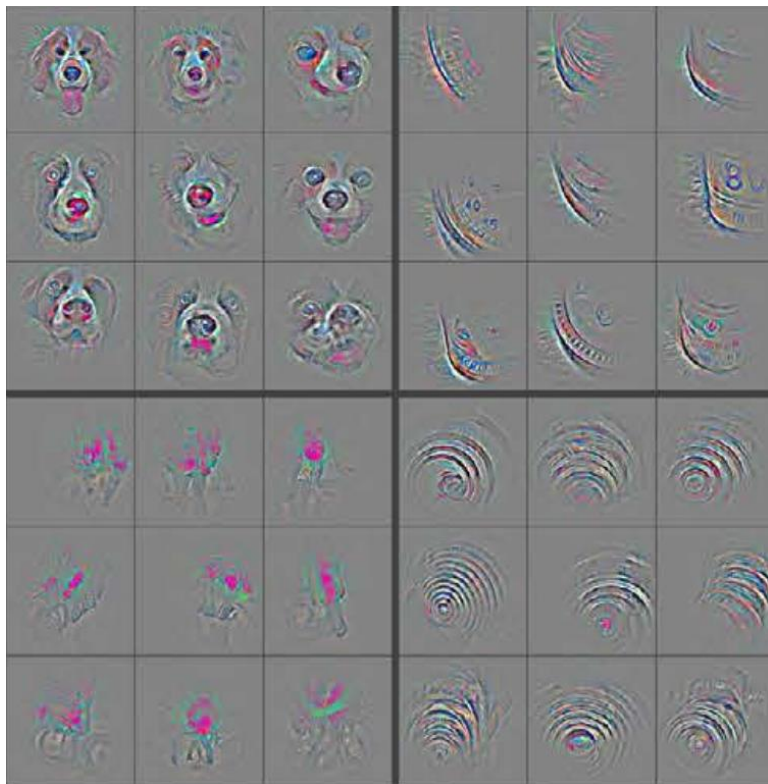
На втором слое – более сложные шаблоны: комбинации линий, окружности.

# Сверточный слой 3

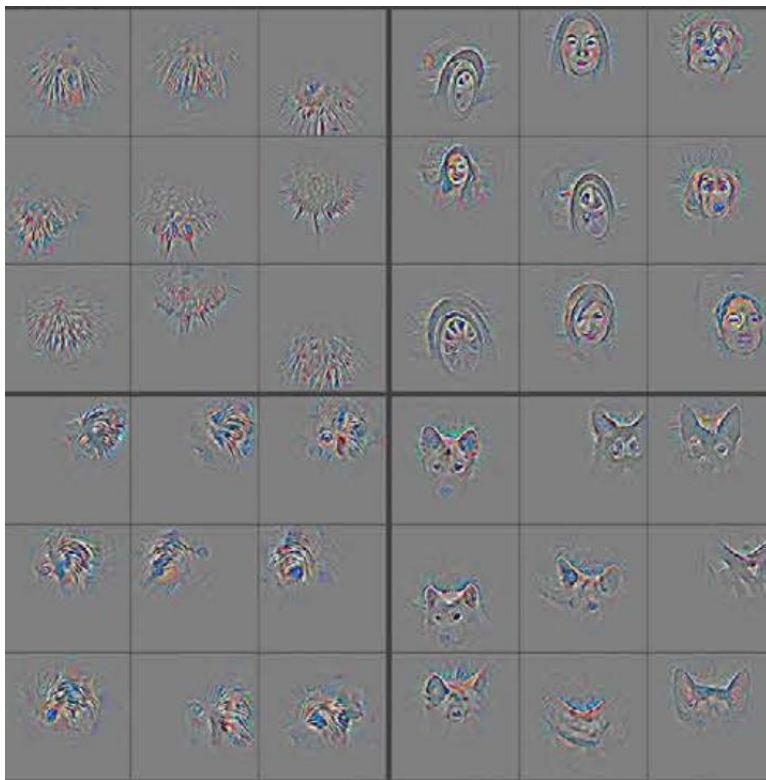




# Сверточный слой 4



# Сверточный слой 5



# Заключение



# Резюме

