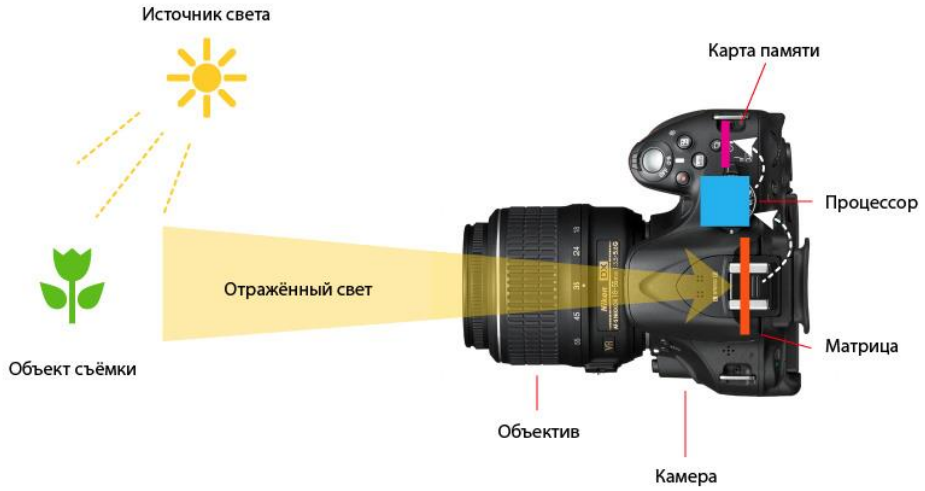
Введение в Computer Vision

Большая часть материала взята из учебника по CV
http://vision.stanford.edu/teaching/cs131_fall1718/files/cs131-class-notes.pdf

Изображение

Как работает фотоаппарат?



1. Изначально стекло закрывает собой матрицу, находясь в стандартном положении
2. После, лучи попадают на матовое стекло, проходя к оптической системе. Здесь изображение переворачивается на 90 градусов, чтобы отобразиться на матрице под правильным углом
3. После того, как пользователь нажимает на кнопку, делающую снимок, зеркало переходит во второе положение. В это время отодвигается затвор, а изображение проецируется на матрицу камеры
4. Последним этапом, который проходит снимок, является считывание информации и её отображение на экране фотокамеры

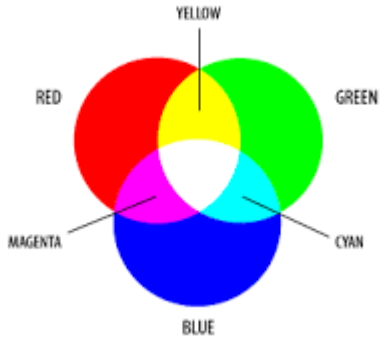
Изображение

Почему это важно?

Изображение

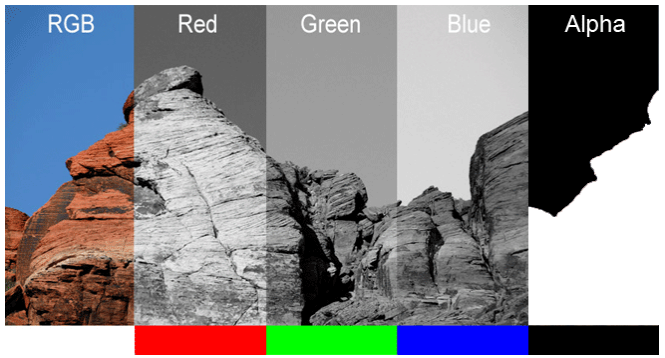
Как представлено в компьютере?

RGB



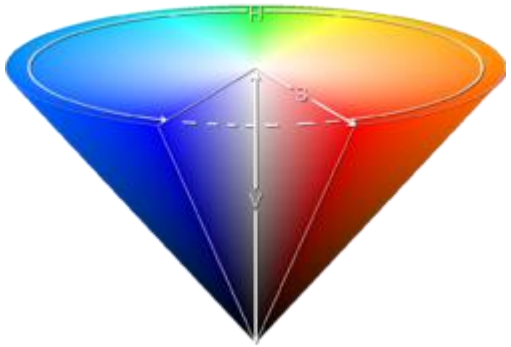
Самый распространённый формат кодирования изображения. При смешении всех каналов получают белый цвет.

RGB + alpha



Попытка улучшить RGB с помощью добавления еще одного канала Alpha, отвечающего
за непрозрачность накладываемого пикселя

HCV



HSV (Hue, Saturation, Value — тон, насыщенность, значение). Цвет, представленный в HSV, зависит от устройства, на которое он будет выведен. Разработан со основателем студии Pixar.

Нейронные сети

Как вы думаете, почему для обработки изображений не применяют нейронные сети, состоящие из линейных слоев?

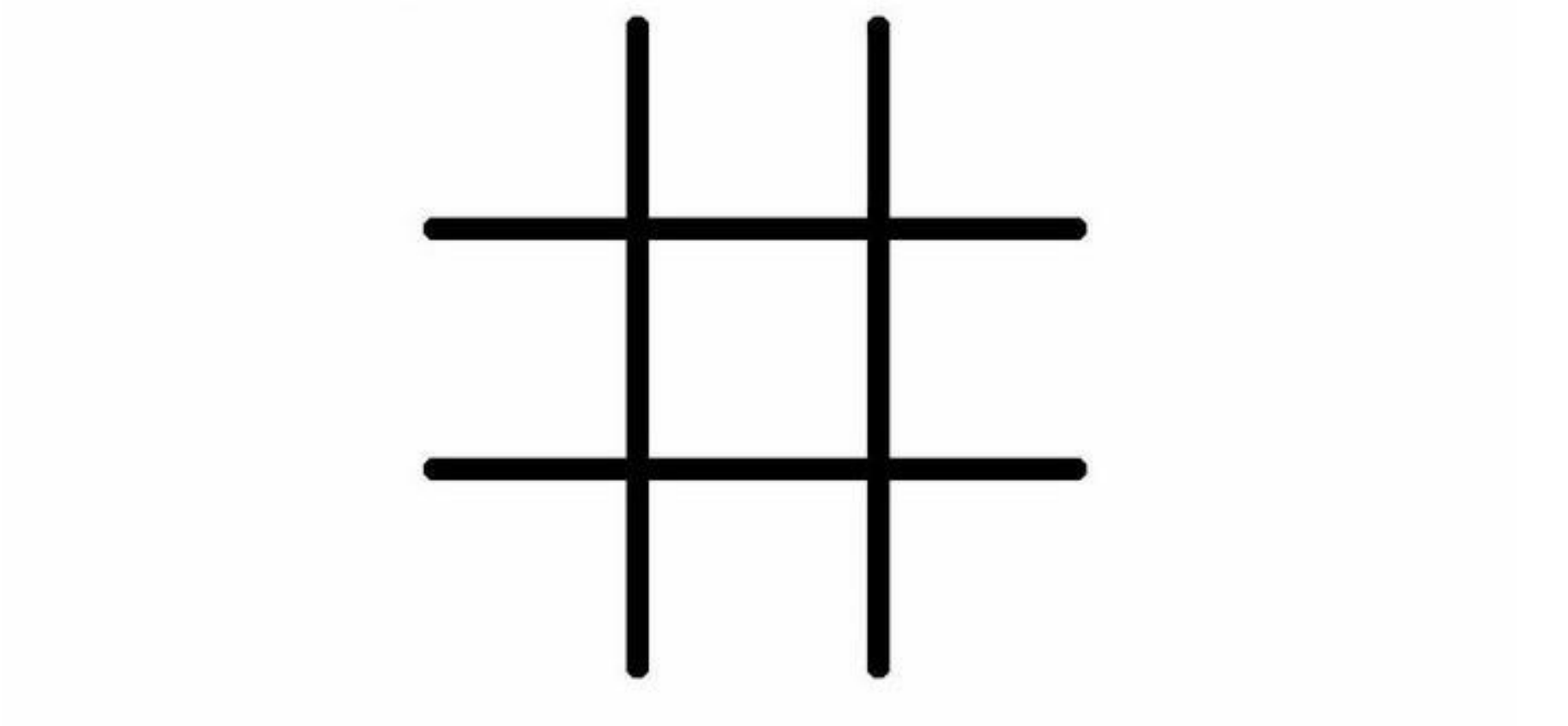
Посчитаем сколько параметров потребуется, чтобы обработать изображение 224x224x3.
Первый слой – линейный. На вход поступает 224x224x3 пикселей. Пусть на выходе мы ожидаем 1024 значения.
Тогда количество параметров в первом слое: 224*224*3*1024 = 154140672.

Как модель, построенная на линейных сетях отреагирует, если мы немного изменим изображение, например перевернем?

Что можно сделать, чтобы упростить вычисления?

Нейронные сети

Новый слой



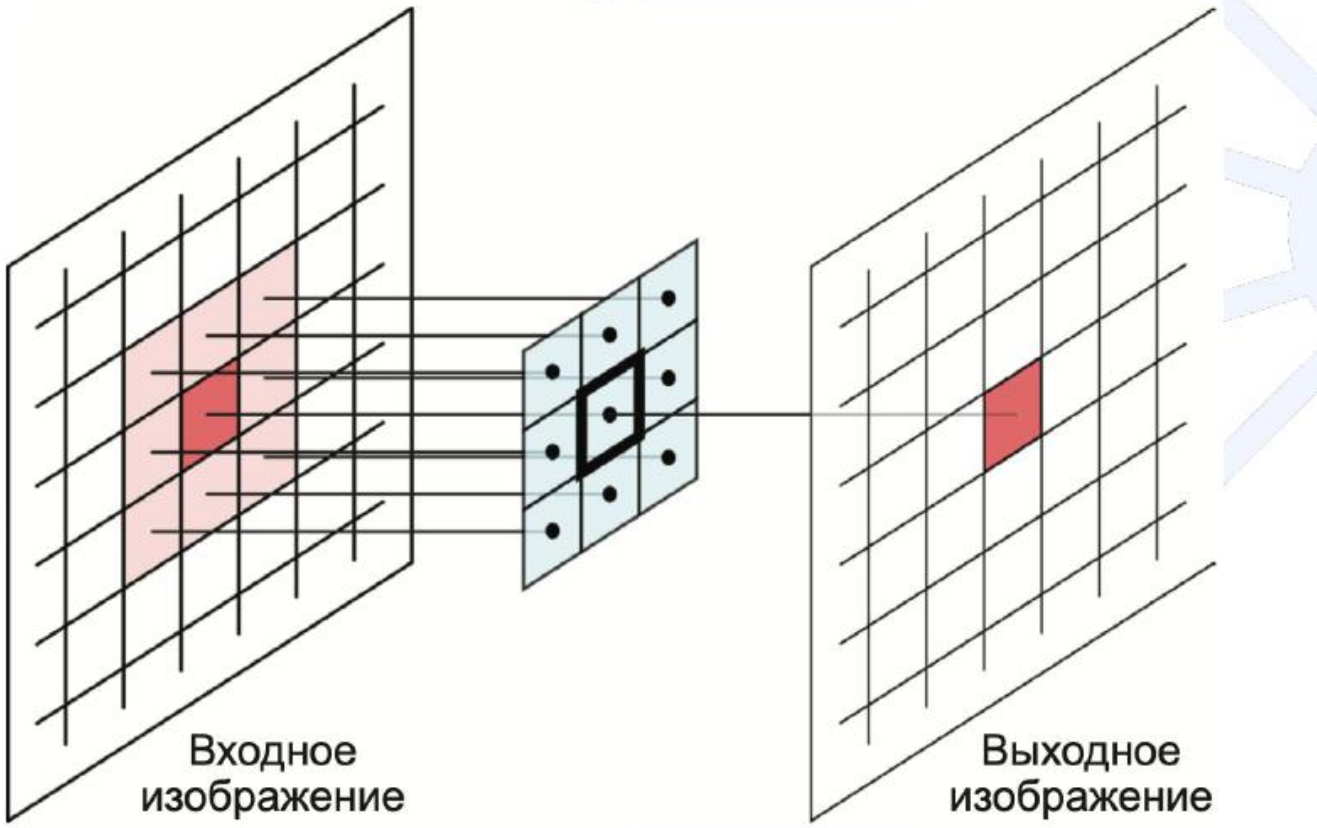
Нейронные сети

Сверточный слой

Свёртка (кросс-корреляция):

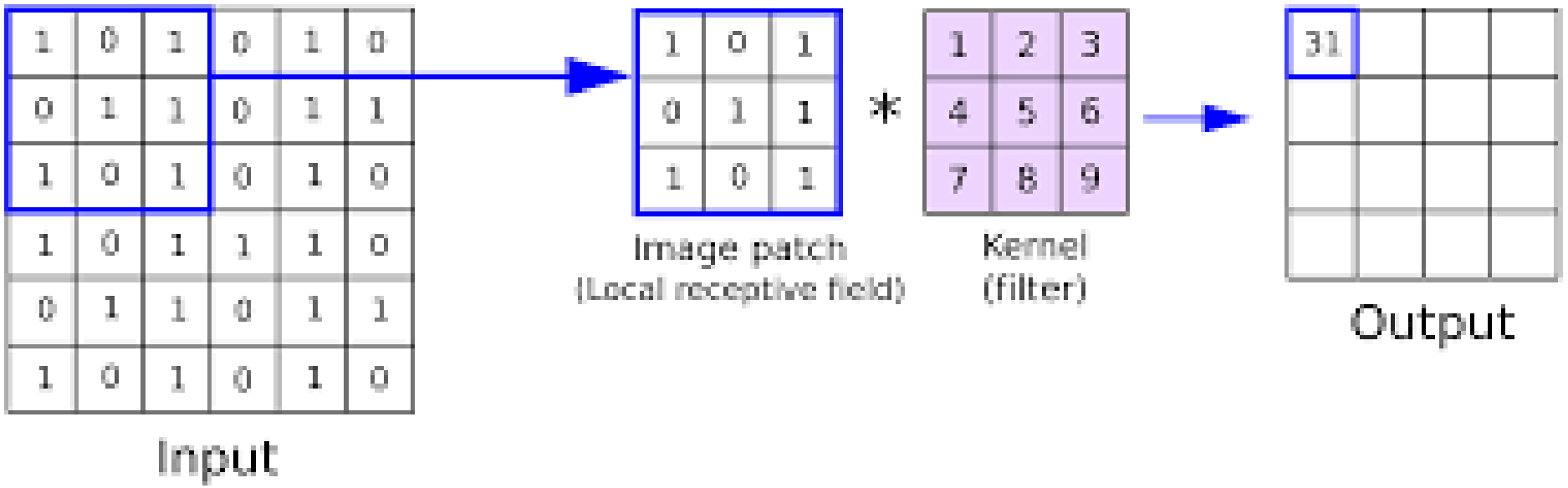
$$(K * x)(i,j,k) = \sum_{u=i-\delta}^{i+\delta} \sum_{v=j-\delta}^{j+\delta} \sum_{c} x(u,v,c) K(u-i+\delta, v-j+\delta, c, k)$$

- Гораздо меньше параметров (kernel, filter)
- Фильтры не зависят от размеров картинки



Нейронные сети

Сверточный слой



Реализация свёртки

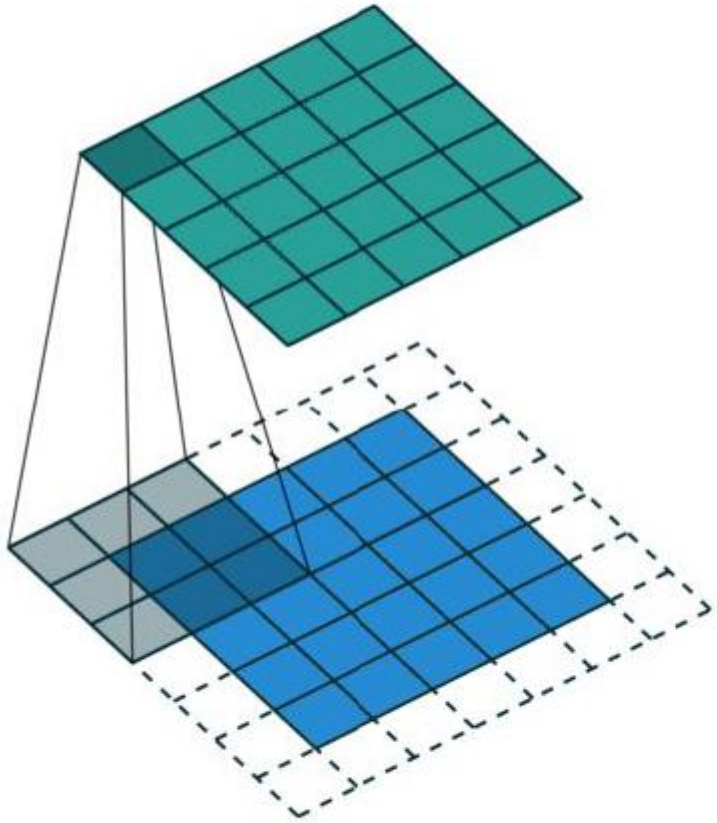
Свёртка – линейная операция
Основная идея – использовать матричное умножение:

$$\begin{pmatrix} k_1 & k_2 \\ k_3 & k_4 \end{pmatrix} * \begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix} \Rightarrow \begin{pmatrix} k_1 & k_2 & 0 & k_3 & k_4 & 0 & 0 & 0 & 0 \\ 0 & k_1 & k_2 & 0 & k_3 & k_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & k_1 & k_2 & 0 & k_3 & k_4 & 0 \\ 0 & 0 & 0 & 0 & k_1 & k_2 & 0 & k_3 & k_4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{pmatrix} = \begin{pmatrix} k_1x_1 + k_2x_2 + k_3x_4 + k_4x_5 \\ k_1x_2 + k_2x_3 + k_3x_5 + k_4x_6 \\ k_1x_4 + k_2x_5 + k_3x_7 + k_4x_8 \\ k_1x_5 + k_2x_6 + k_3x_8 + k_4x_9 \end{pmatrix}$$

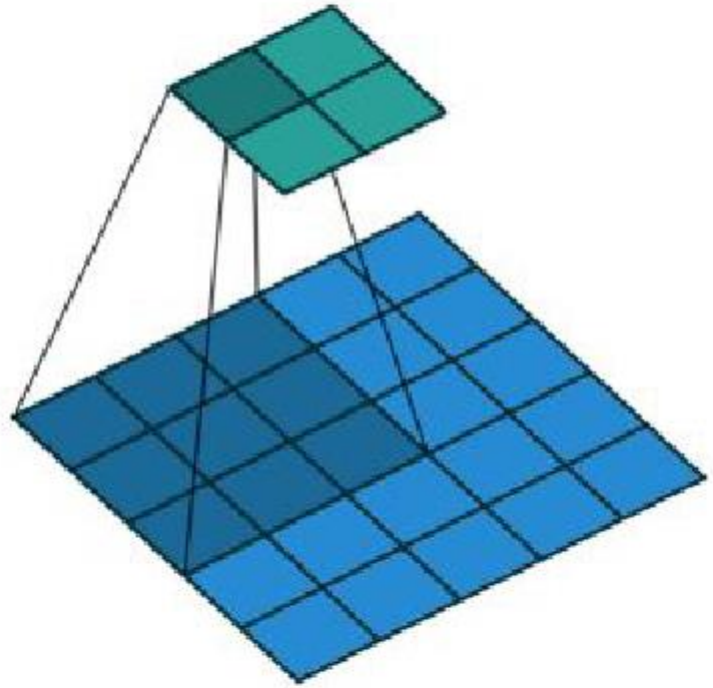
Очень эффективные реализации: NVIDIA cuDNN, Nervana kernels
Специализация: метод Винограда для сверток 3x3, преобразования Фурье для больших свёрто

Padding / Striding / Dilation

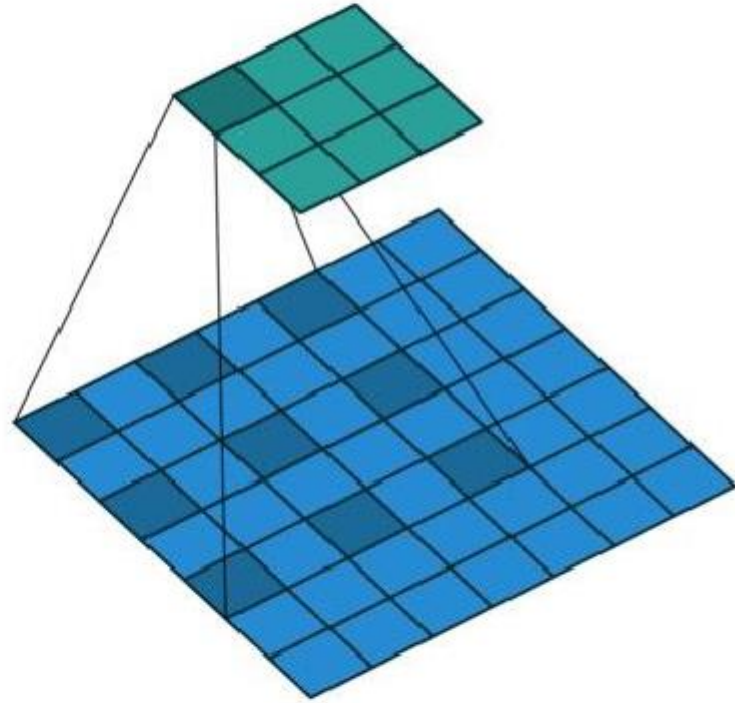
Padding – обработка границ (нет, нули, зеркальный):



Striding - большие значения понижают разрешение:



Dilation - увеличить область зависимости:



Размер выхода сети:

$$L_{out} = floor((L_{in} + 2pad - dil(kernel - 1) - 1)/stride + 1)$$

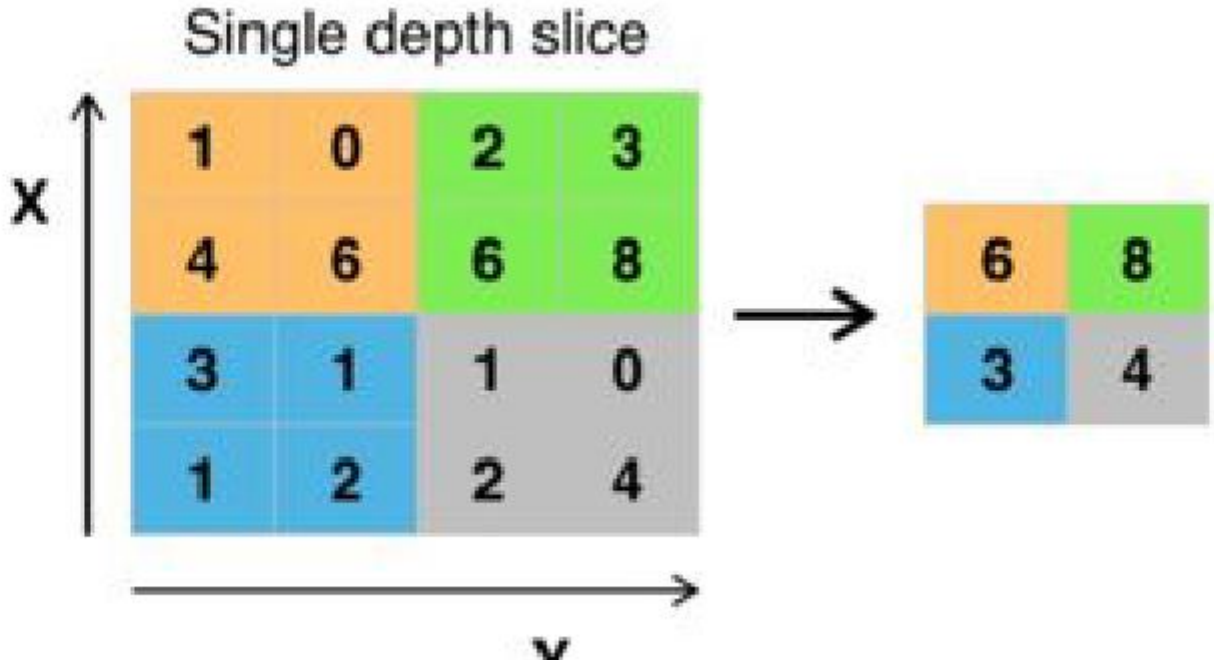
Pooling

Pooling – агрегация признаков (max, sum)

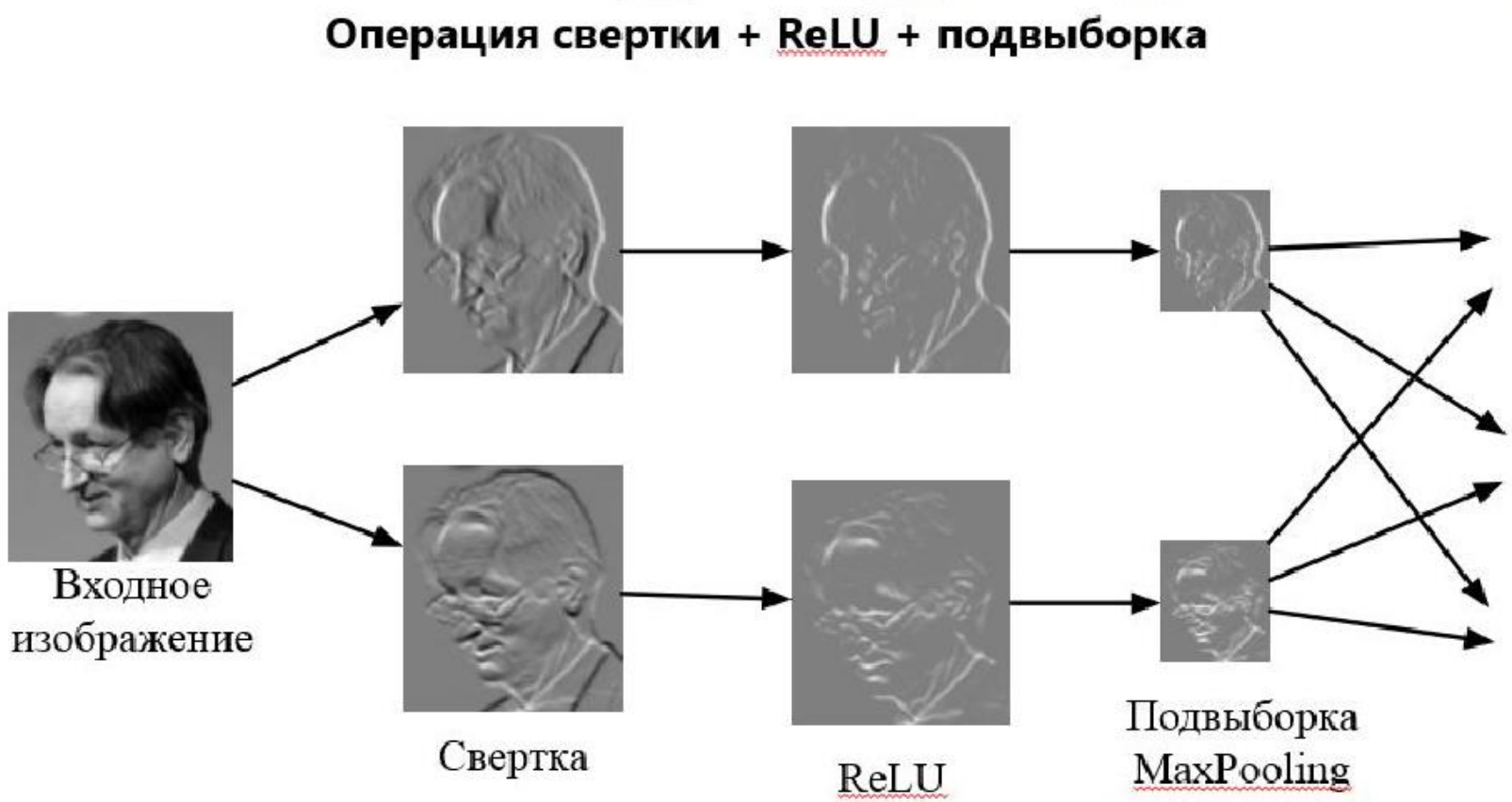
Pooling слой агрегирует соседние активации (пространственно или из разных фильтров)

Max-pooling обеспечивает инвариантность к небольшим сдвигам (иногда полезно, иногда нет

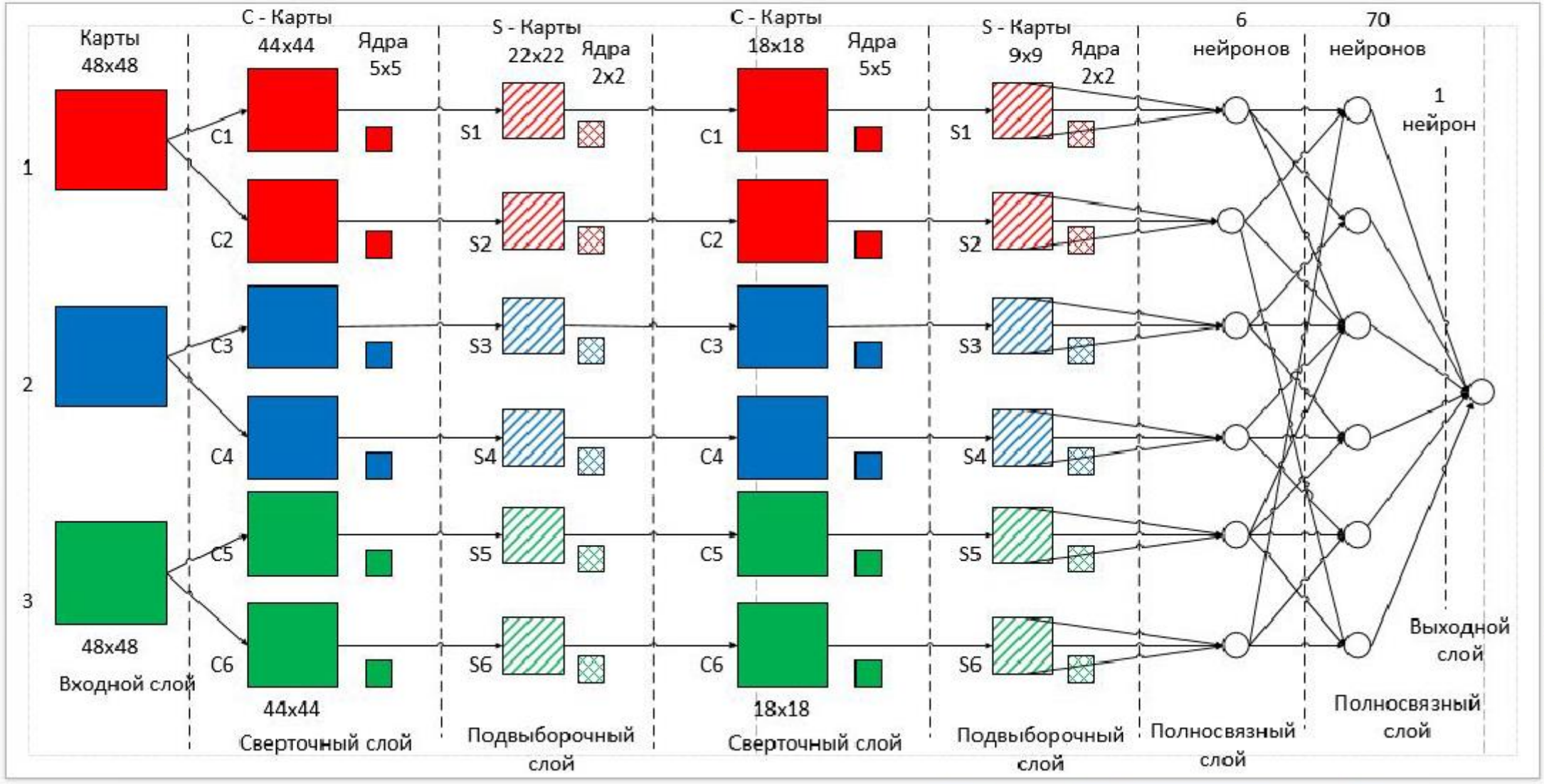
Дифференцирование – вернуть градиент в позиции максимумов



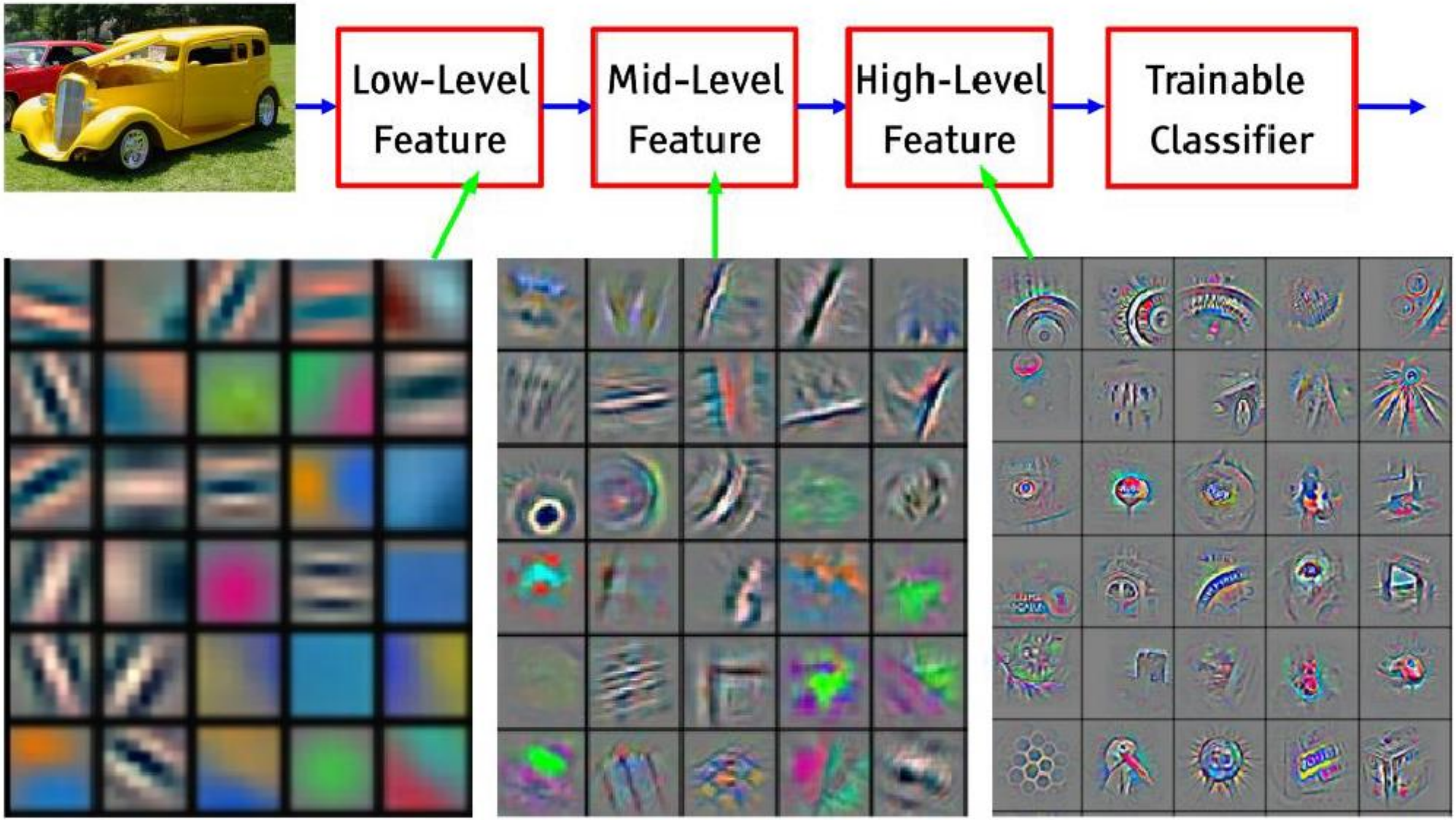
Свёртка + ReLU + MaxPooling



Топология CNN

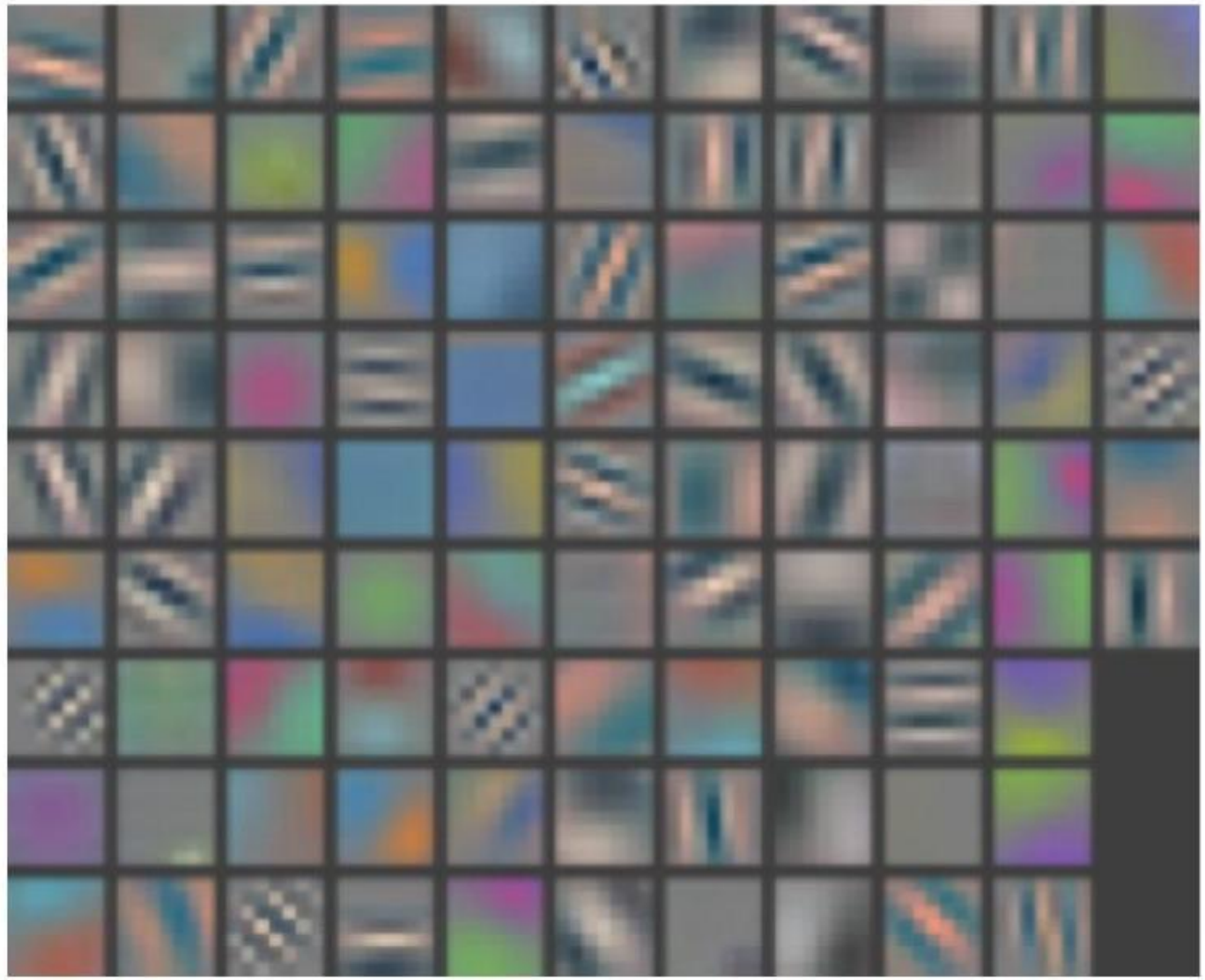


Визуализация слоев CNN



Визуализация слоев CNN

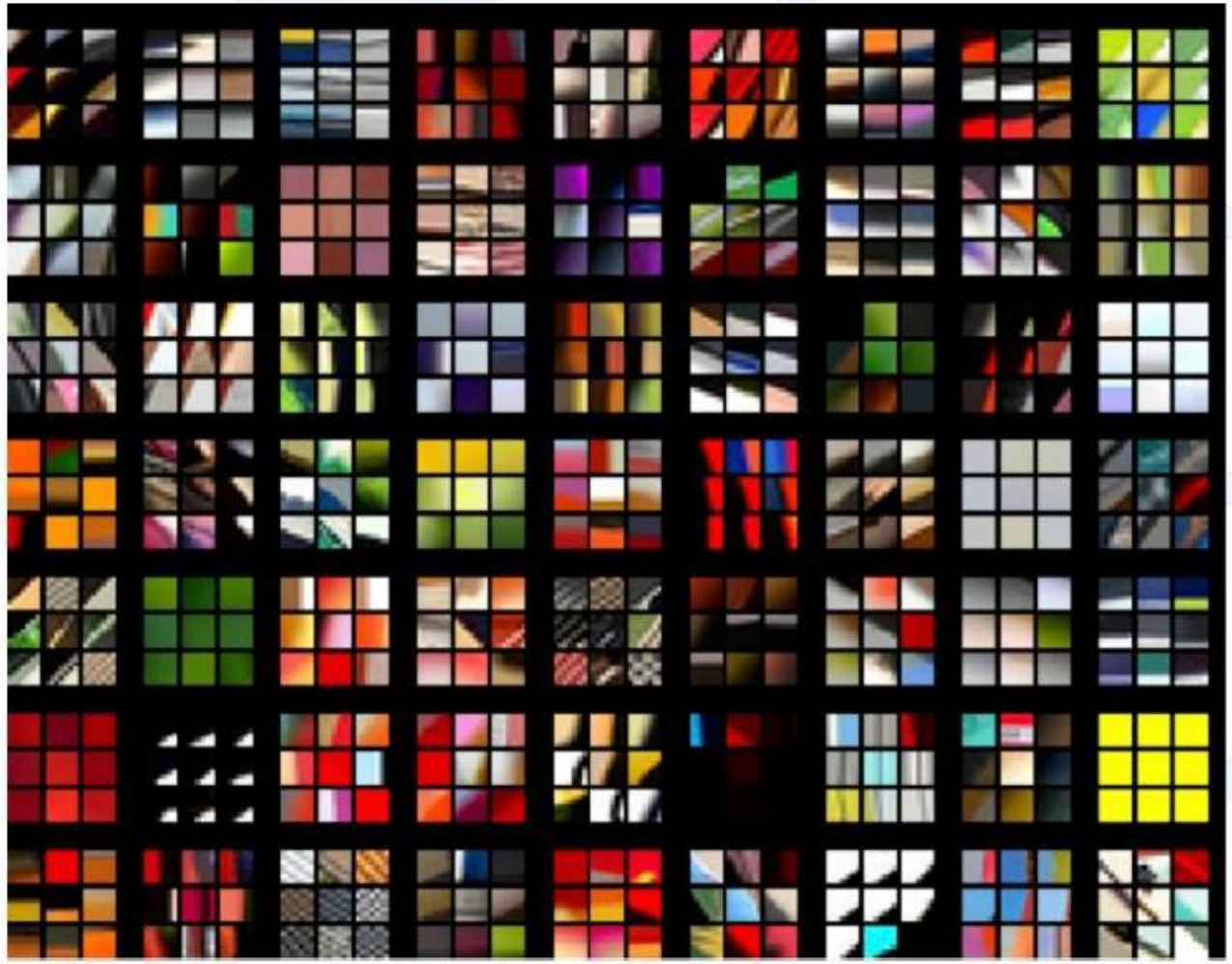
Фильтры первого слоя:



Визуализация слоев CNN

Оставшиеся слои надо «спроецировать» на пиксели
Один из способов – искать патчи, дающие наибольший отклик

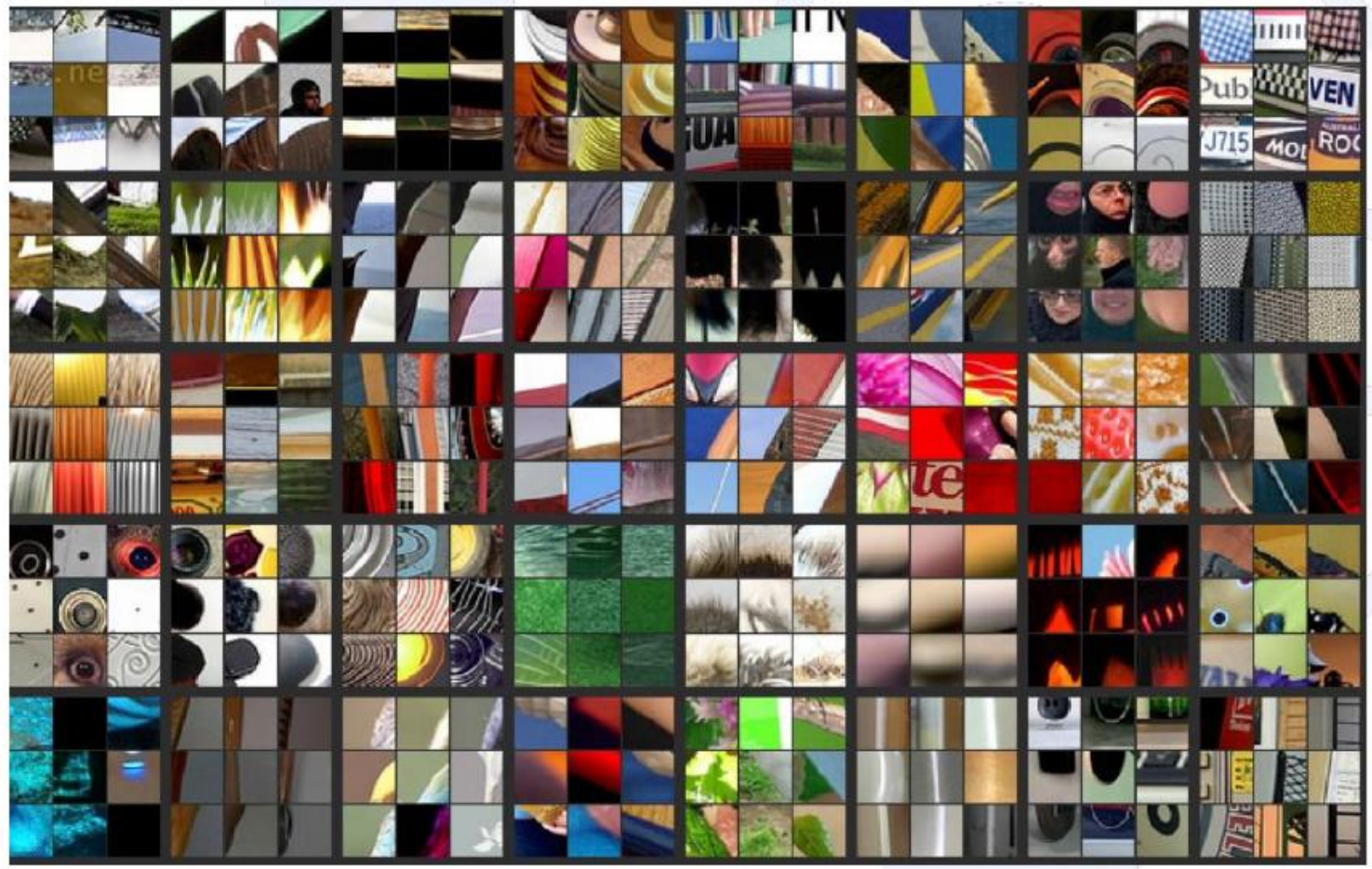
Слой 1:



Визуализация слоев CNN

Оставшиеся слои надо «спроецировать» на пиксели
Один из способов – искать патчи, дающие наибольший отклик

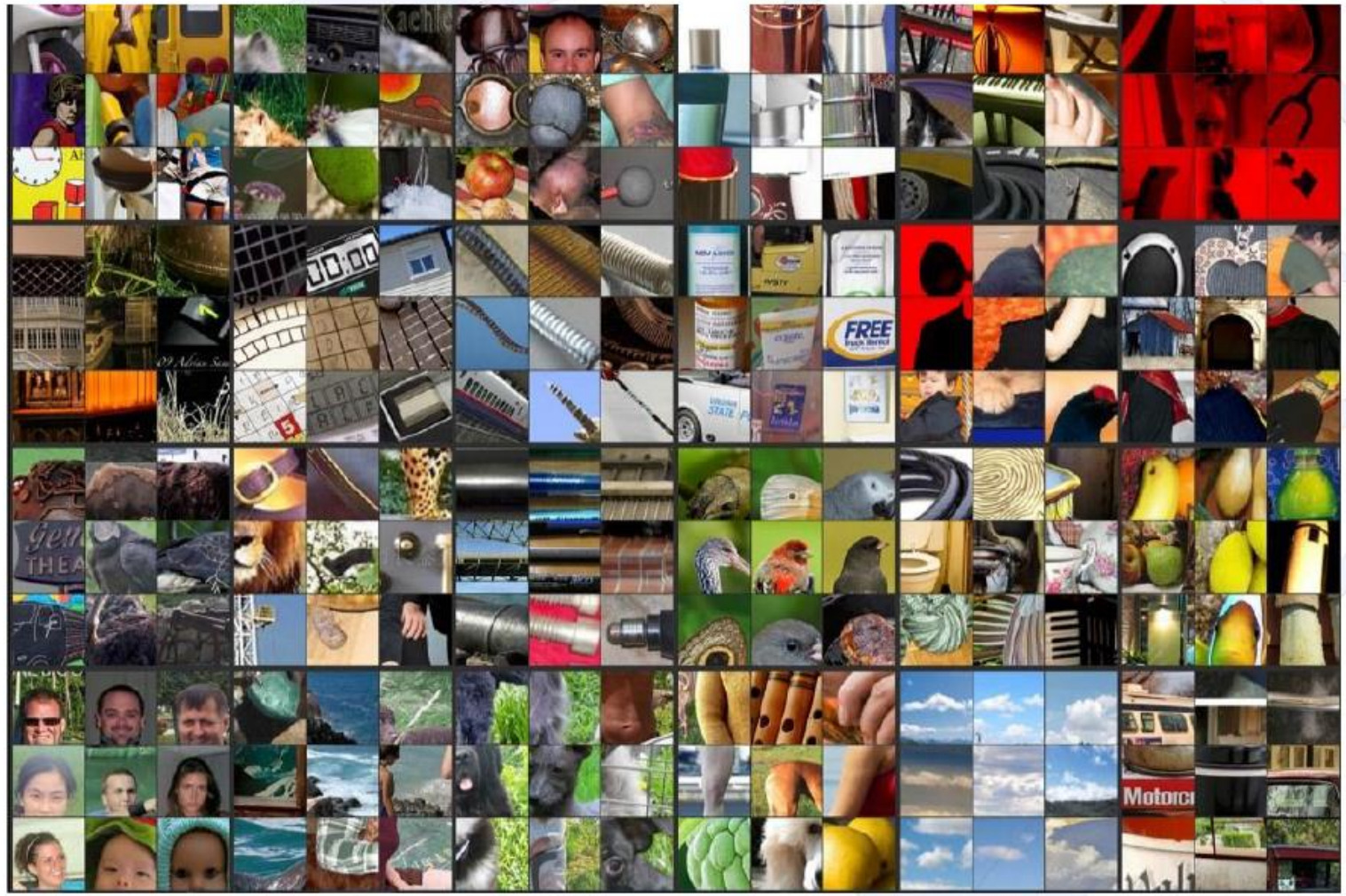
Слой 2:



Визуализация слоев CNN

Оставшиеся слои надо «спроецировать» на пиксели
Один из способов – искать патчи, дающие наибольший отклик

Слой 3:



Визуализация слоев CNN

Оставшиеся слои надо «спроецировать» на пиксели
Один из способов – искать патчи, дающие наибольший отклик

Слой 4: