KARPOV.COURSES >>>
КОНСПЕКТ

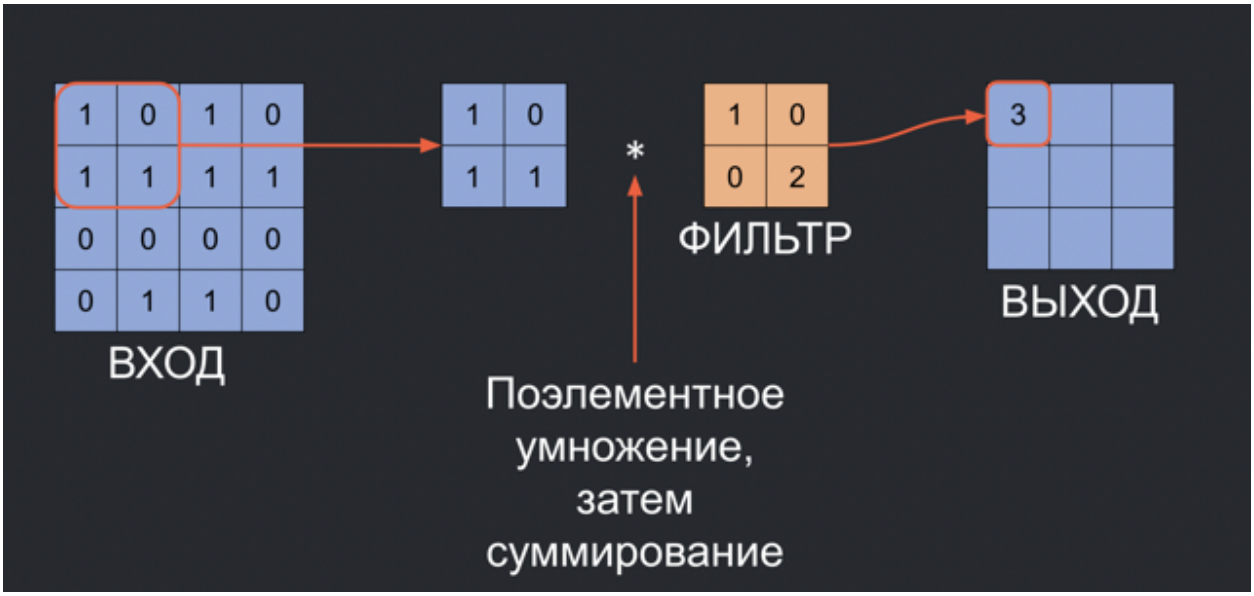
Конспект > 3 урок > Сверточные нейронные сети

>Свертка

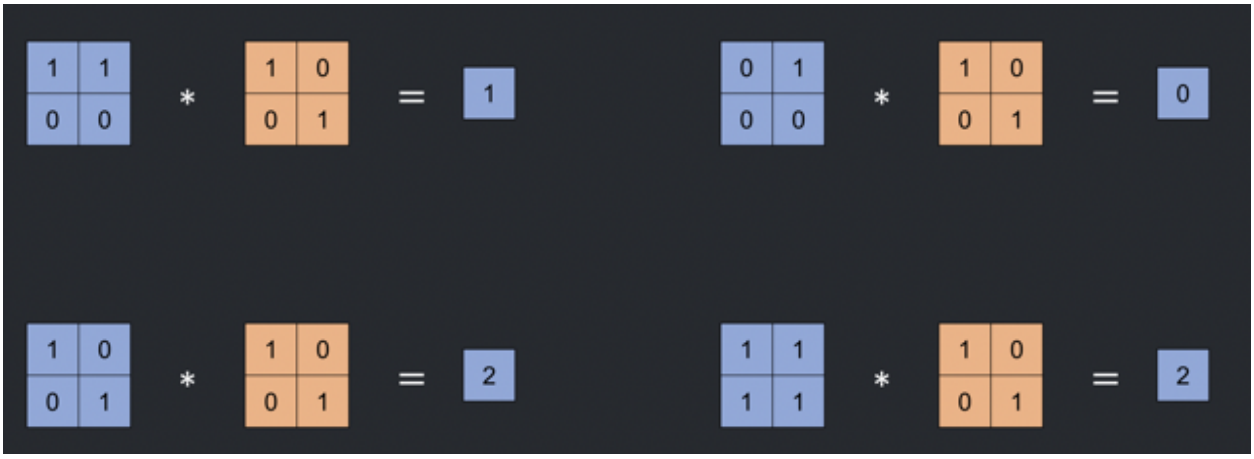
В чем заключается операция свертки?

Например, имеется входное изображение и некий фильтр. Идея состоит в том, чтобы проходить фильтром по изображению и выполнять поэлементное

умножение, а затем суммирование над каждым участком изображения. Результат операций записываются в выходную матрицу. Значение в ядре свертки (фильтре) - обучаемое. Соответственно, это будет являться параметрами.



Какой смысл имеет свертка?



Рассмотрим на примере: есть фильтр, у которого единицы находятся в диагонали. Посмотрим, какой будет результат после применения свертки к разным видам изображения.

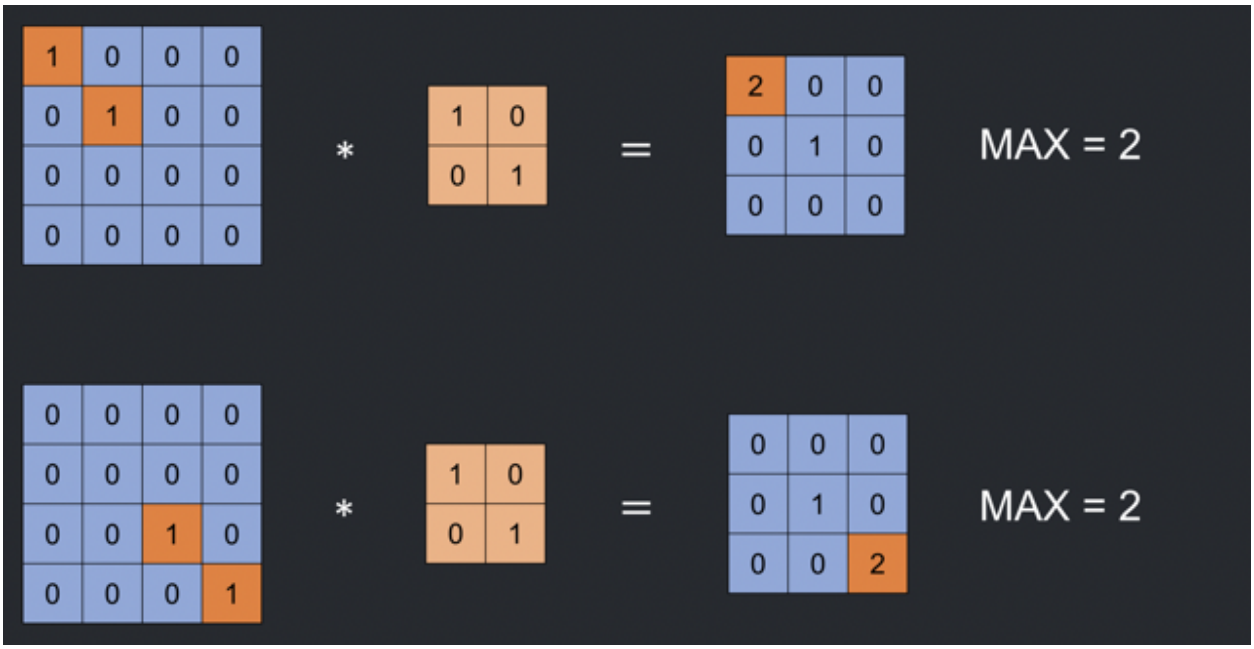
Из примера выше можно увидеть, что в случае, когда во входном изображении присутствуют единицы в диагоналях - выходной результат максимальный и наоборот.

Можно сделать вывод: число на выходе говорит, насколько паттерн присутствует в изображении.

Свертка помогает искать геометрические шаблоны в данных и при этом они будут оптимальные с точки зрения минимазации функции ошибки.

Максимум свертки инвариантен к сдвигам

Еще одно важное наблюдение, которое может пригодиться в дальнейшем:



Рассмотрим два входных изображения:

Есть фильтр, который детектирует единицы по диагонали, на изображениях такие диагонали расположены по разным углам. После применения свертки к этим изображениям на выходе получаются два других изображения. И если взять максимум от них, то в обоих случаях результат будет равен 2.

Максимум свертки инвариантен к положению искомого паттерна.

>Ликбез: Grayscale и RGB

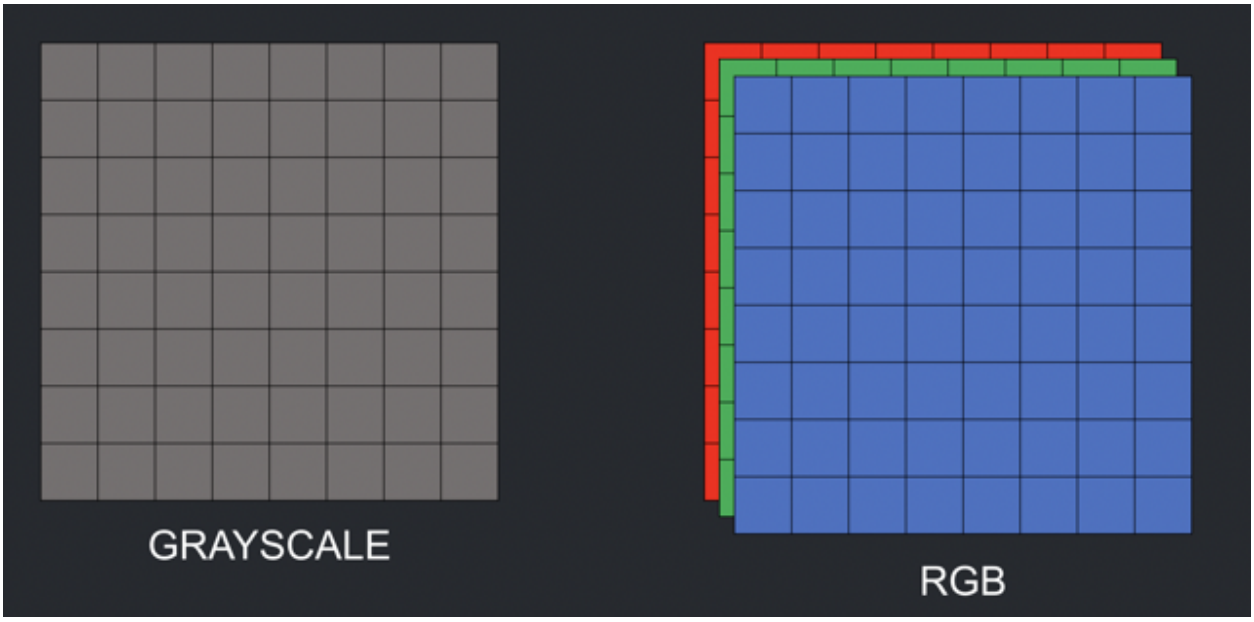
До этого в примерах были рассмотрены только одномерные изображения. Однако чаще всего изображения являются трехмерными.

На курсе будем рассматривать изображения в двух цветовых пространствах: Grayscale и RGB.

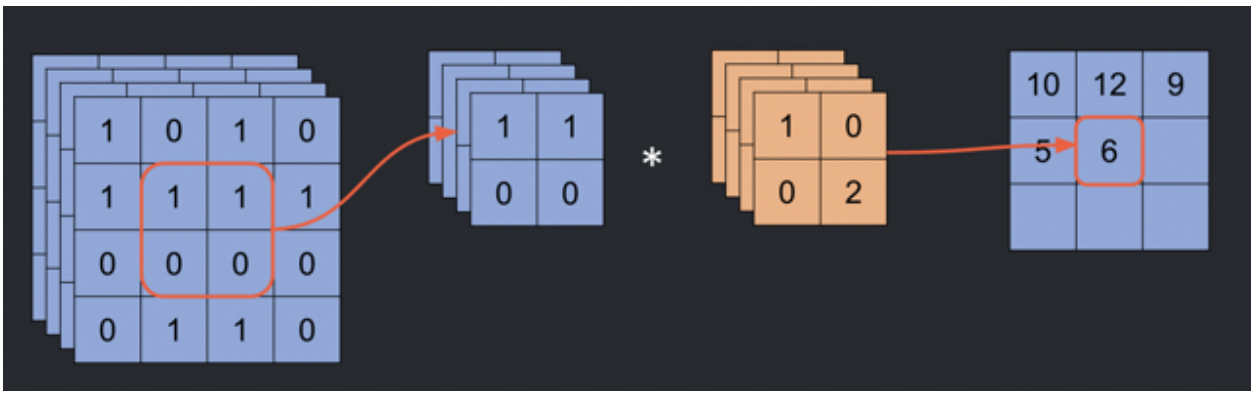
Grayscale - один канал, который представляет собой интенсивность белого.

RGB - три канала, которые представляют собой интенсивность красного, зеленого и синего соответственно.

Каждое изображение так же представляется в виде матрицы. Только в случае с Grayscale - это матрица размером 1xMxN, а в случае с RGB - 3xMxN.



Что происходит со сверткой при RGB?



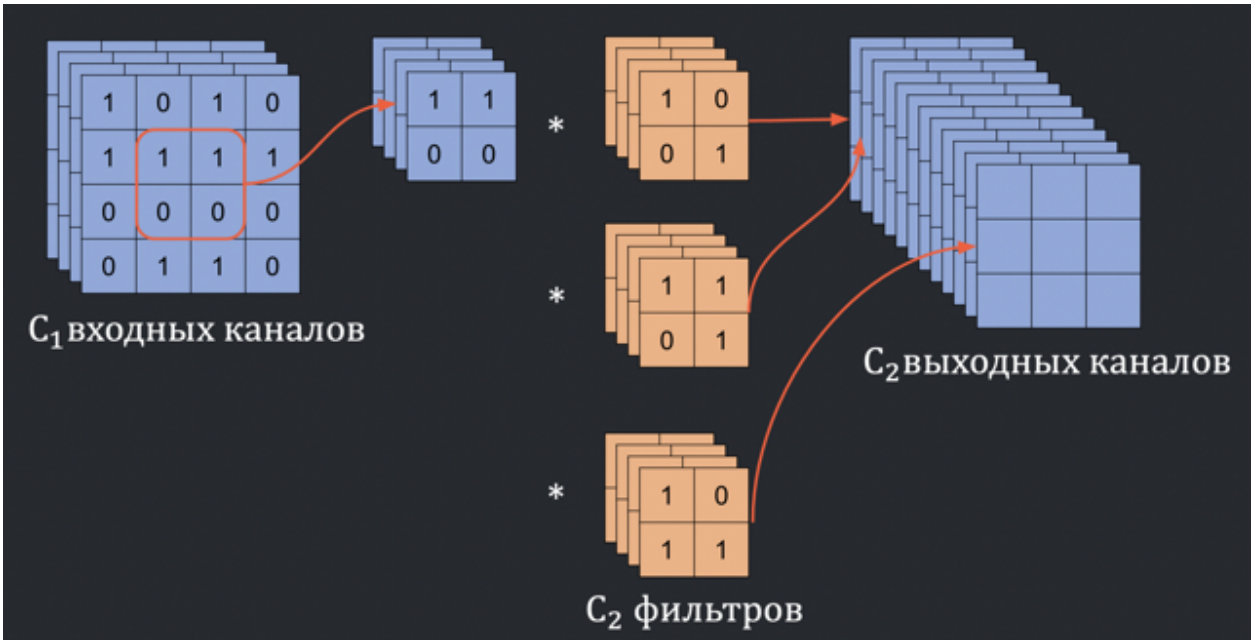
При добавлении каналов у фильтра также появляются дополнительные размерности: появляется одно ядро на каждый канал. Идея та же: происходит

поэлементное умножение, а затем суммирование, только уже для трех каналов.

>Сверточный слой

Как можно задать сверточный слой?

На вход подается изображение, у которого c1 каналов, и мы знаем, что фильтр выделяет какой-то определенный паттерн. Мы хотим выделять множество различных паттернов: нам нужны несколько различных фильтров. Соответсвенно, если у нас есть c2 фильтров, результатом будет c2 выходных изображений с добавленными на выходе каналами.



Сколько будет параметров в слое?

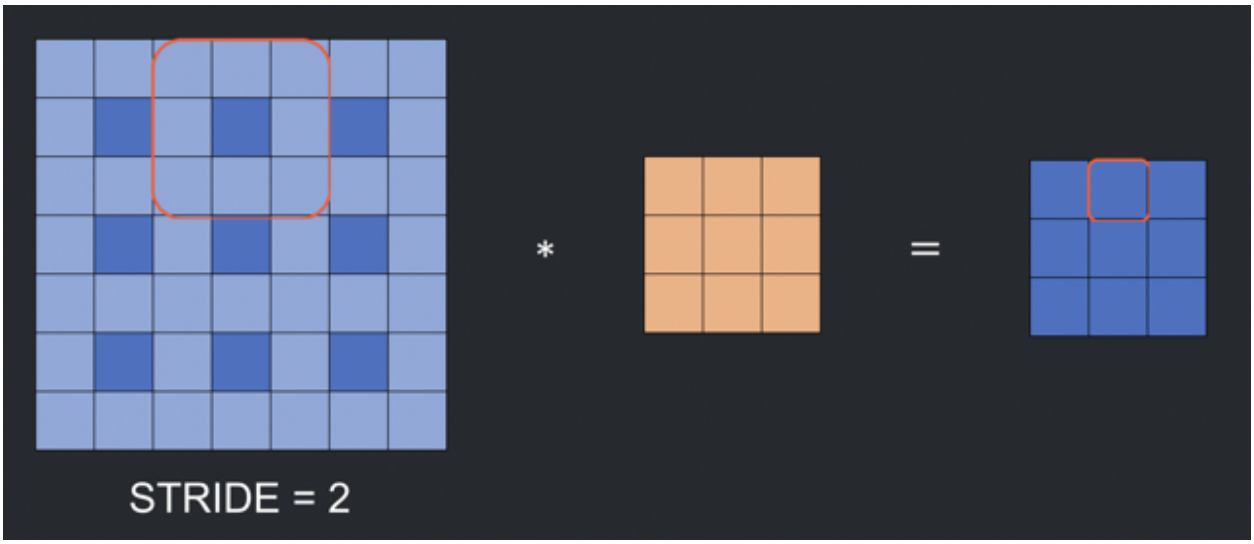
Каждый фильтр имеет размер ядра в квадрате - n^2.

Количество входных каналов - C_1

Количество выходных каналов - C_2

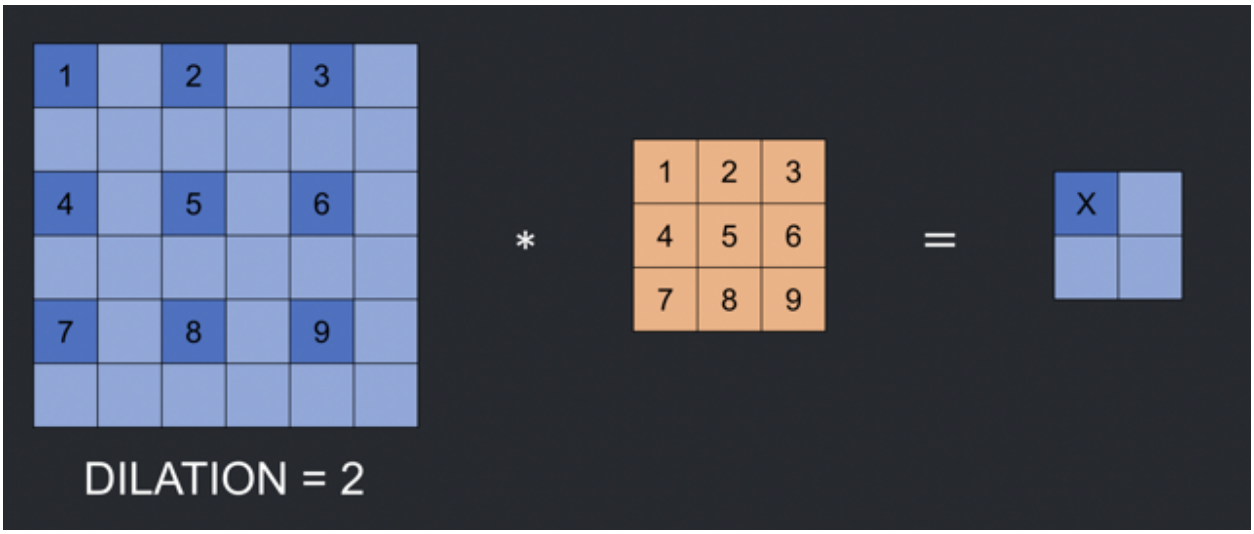
$(C_1 * n^2 + 1) * C_2$ - количество параметров

>Гиперпараметры



Идея в том, чтобы двигать окно применения свертки не на соседнюю область (на 1), а на какое-то большее число (Например, stride=2)

Расширенная свертка



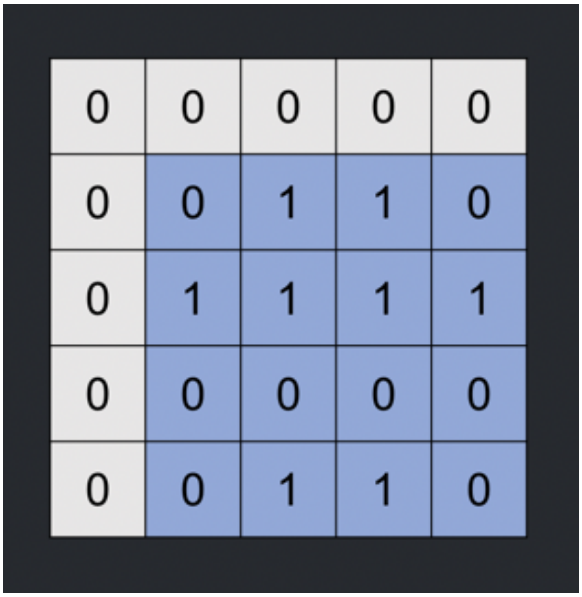
Идея в том, чтобы выделять окно, равное не размеру фильтра, а немного больше, и применять фильтр, например, через один, в зависимости от значения dilation.

Паддинг

На предыдущих шагах при применении свертки размер изображения уменьшался. Кроме того, не учитывается, что происходит на краях картинки. Что можно с этим

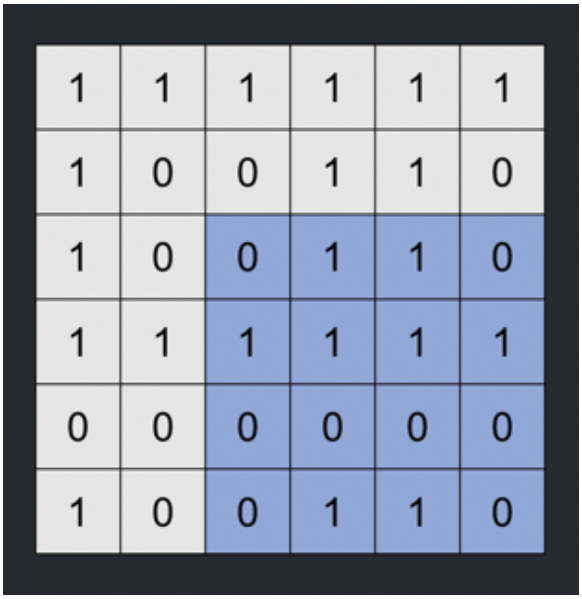
сделать? Идея в дополнении картинки значениями по краям.

Самый простой способ: добавить нули - zero padding



Но такой способ порождает другую проблему: фильтр может научиться детектировать край и на основе этого будет переобучаться.

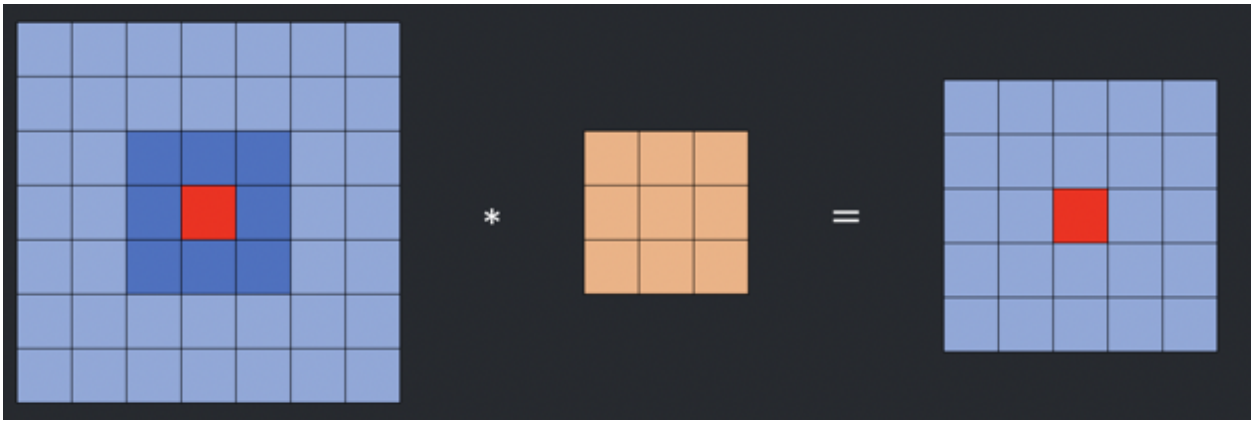
Другая идея: вместо нулей сделать отражение - reflection padding.



Replication padding - подолжение картинки значениями крайних пикселей

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

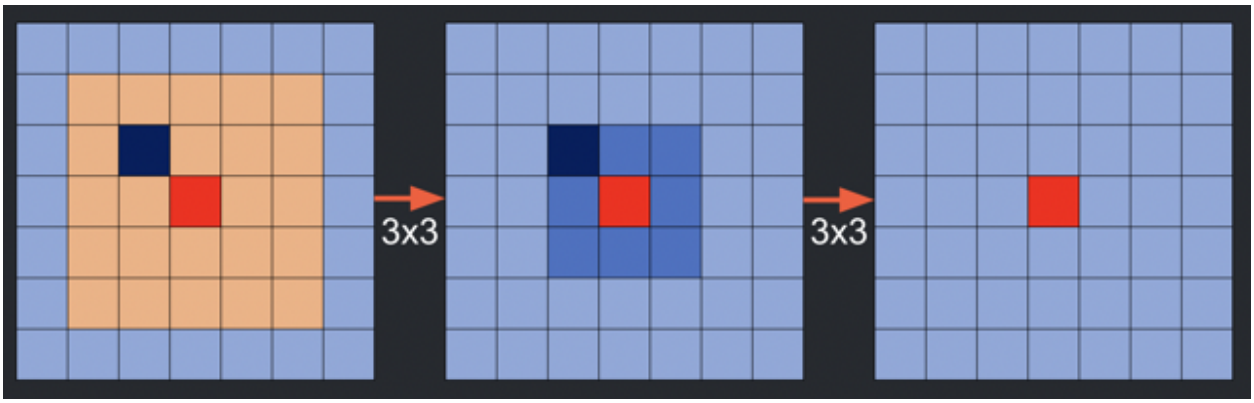
>Поле восприятия



Представим, что применили сверточный слой к изображению. На изображении возьмем произвольный пиксель (красный). От какого числа пикселей на исходном изображении будет зависеть красный пиксель?

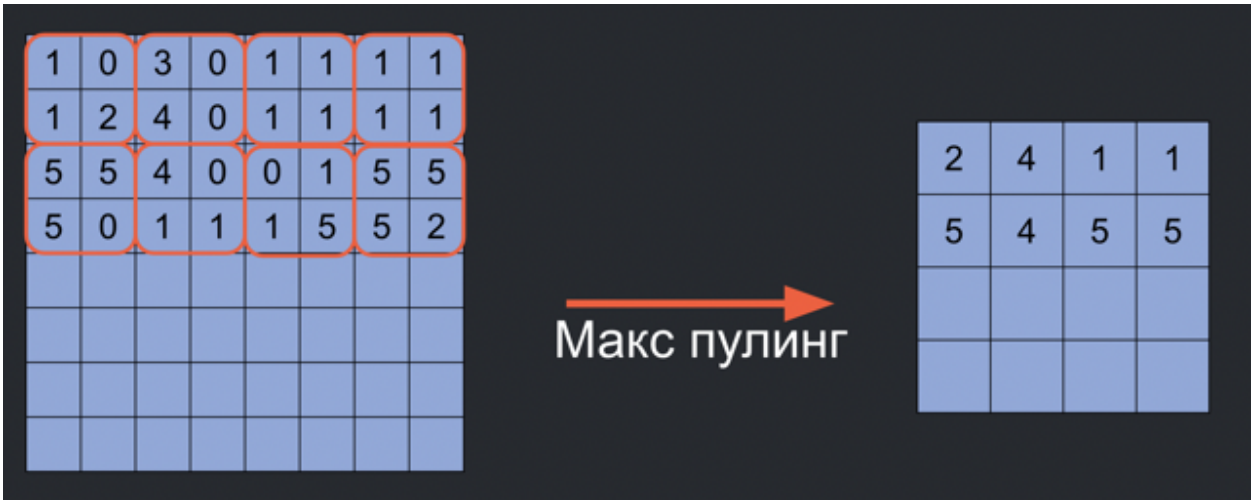
Если фильтр был 3x3, соответсвенно, чтобы получить красный пиксель, нужно было задействовать 9 пикселей (область 3x3).

Как считать поле восприятия?



Давайте использовать два фильтра: свернуть с ядром 3x3, а затем результат еще раз свернуть ядром 3x3. Красный пискель будет зависеть от области 3x3 на промежуточном результате, а каждый из этих пикселей будет зависеть от своих областей 3x3. В результате получится область 5x5.

Пулинг



Идея: разбить изображение на части и по этим частям посчитать статистики (max или avg, например).

Max pooling вычисляет максимум этой части, а avg pooling - среднее значение.

Пулинг значительно сокращает размер изображения и не имеет параметров.