CNN | Introduction to Padding

During convolution, the size of the output feature map is determined by the size of the input feature map, the size of the kernel, and the stride. if we simply apply the kernel on the input feature map, then the output feature map will be smaller than the input. This can result in the loss of information at the borders of the input feature map. In Order to preserve the border information we use padding.

What Is Padding

padding is a technique used to preserve the spatial dimensions of the input image after convolution operations on a feature map. Padding involves adding extra pixels around the border of the input feature map before convolution.

This can be done in two ways:

- **Valid Padding**: In the valid padding, no padding is added to the input feature map, and the output feature map is smaller than the input feature map. This is useful when we want to reduce the spatial dimensions of the feature maps.
- **Same Padding**: In the same padding, padding is added to the input feature map such that the size of the output feature map is the same as the input feature map. This is useful when we want to preserve the spatial dimensions of the feature maps.

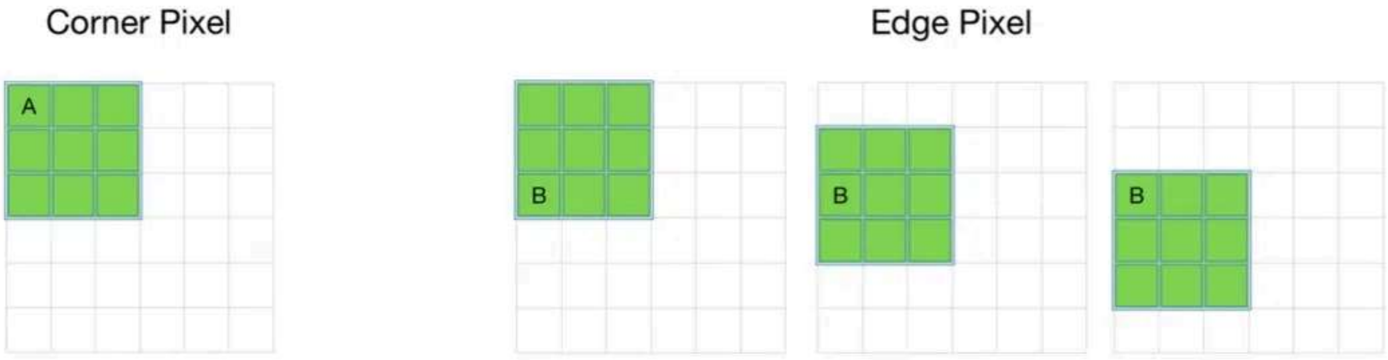
The number of pixels to be added for padding can be calculated based on the size of the kernel and the desired output of the feature map size. The most common padding value is zero-padding, which involves adding zeros to the borders of the input feature map.

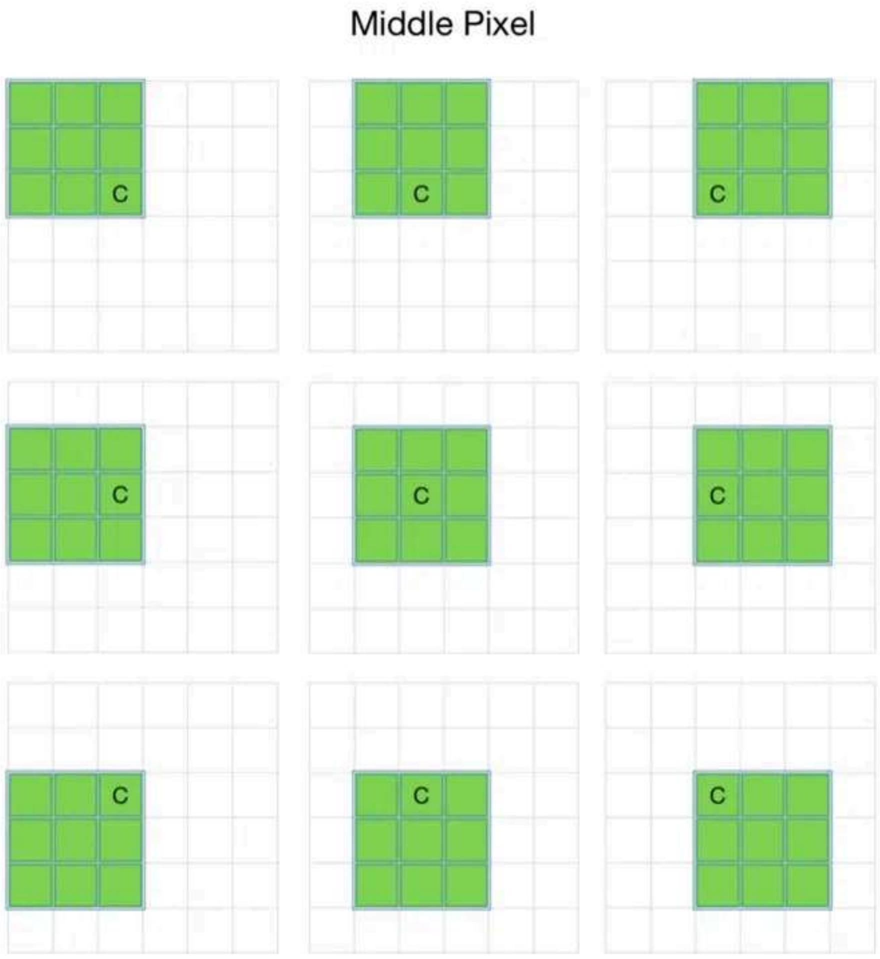
Padding can help in reducing the loss of information at the borders of the input feature map and can improve the performance of the model. However, it also increases the computational cost of the convolution operation. Overall, padding is an important technique in CNNs that helps in preserving the spatial dimensions of the feature maps and can improve the performance of the model.

Problem With Convolution Layers Without Padding

- For a grayscale (n x n) image and (f x f) filter/kernel, the dimensions of the image resulting from a convolution operation is **(n – f + 1) x (n – f + 1)**.
  For example, for an (8 x 8) image and (3 x 3) filter, the output resulting after the convolution operation would be of size (6 x 6). Thus, the image shrinks every time a convolution operation is performed. This places an upper limit to the number of times such an operation could be performed before the image reduces to nothing thereby precluding us from building deeper networks.

- Also, the pixels on the corners and the edges are used much less than those in the middle. For example,



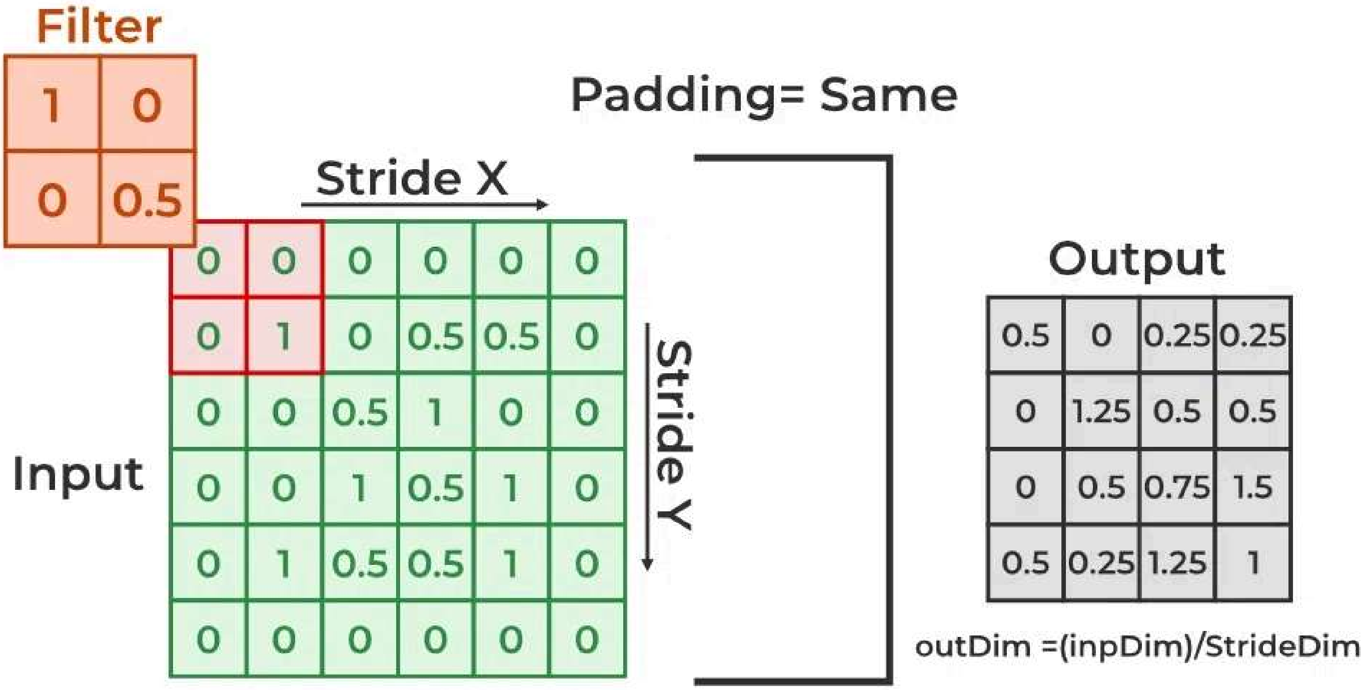


Padding in convulational neural network

- Clearly, pixel A is touched in just one convolution operation and pixel B is touched in 3 convolution operations, while pixel C is touched in 9 convolution operations. In general, pixels in the middle are used more often than pixels on corners and edges. Consequently, the information on the borders of images is not preserved as well as the information in the middle.

Effect Of Padding On Input Images

Padding is simply a process of adding layers of zeros to our input images so as to avoid the problems mentioned above through the following changes to the input image.



Filter

1	0
0	0.5

Input

0	0	0	0	0	0
0	1	0	0.5	0.5	0
0	0	0.5	1	0	0
0	0	1	0.5	1	0
0	1	0.5	0.5	1	0
0	0	0	0	0	0

Output

0.5	0	0.25	0.25
0	1.25	0.5	0.5
0	0.5	0.75	1.5
0.5	0.25	1.25	1

padding in convolutional network

Padding prevents the shrinking of the input image.

> **p** = number of layers of zeros added to the border of the image,
>
> then **(n x n)** image —> **(n + 2p) x (n + 2p)** image after padding.
>
> **(n + 2p) x (n + 2p) * (f x f)** ——> **outputs (n + 2p – f + 1) x (n + 2p – f + 1)** images

For example, by adding one layer of padding to an (8 x 8) image and using a (3 x 3) filter we would get an (8 x 8) output after performing a convolution operation.

This increases the contribution of the pixels at the border of the original image by bringing them into the middle of the padded image. Thus, information on the borders is preserved as well as the information in the middle of the image.

Types of Padding

Valid Padding: It implies no padding at all. The input image is left in its valid/unaltered shape. So

> $$(nn)(ff)(nf+1)(nf+1)$$

> where, nxn is the dimension of input image
>
> fxf is kernel size
>
> n-f+1 is output image size
>
> * represents a convolution operation.

Same Padding: In this case, we add 'p' padding layers such that the output image has the same dimensions as the input image.
So,

> **[(n + 2p) x (n + 2p) image] * [(f x f) filter] —> [(n x n) image]**

which gives **p = (f – 1) / 2** (because n + 2p – f + 1 = n).

So, if we use a (3 x 3) filter on an input image to get the output with the same dimensions. the 1 layer of zeros must be added to the borders for the same padding. Similarly, if (5 x 5) filter is used 2 layers of zeros must be appended to the border of the image.