## horizontal line

Padding and Strides

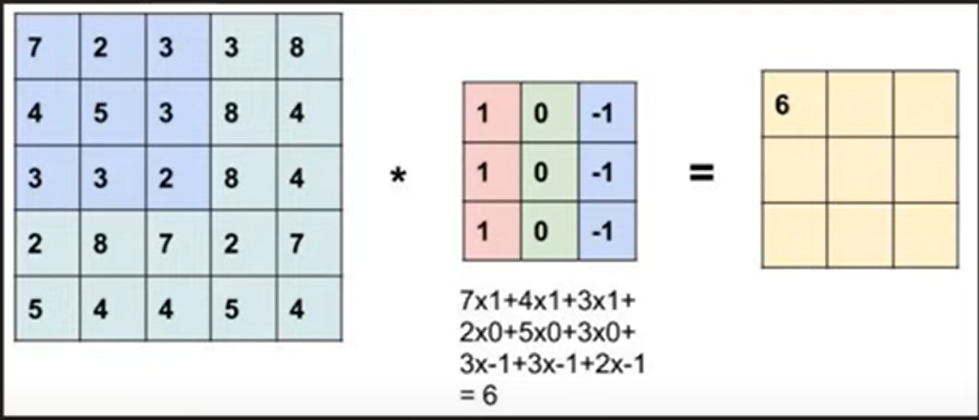
05.04.2025

[Padding and strides](https://colab.research.google.com/drive/1owS5-Km4lD97vyhzHrYkC2oizObWqWWY#scrollTo=65fGVsRL3uZm)

# Padding

Padding is required to prevent loss of useful information in the image. Like in the example below an image of 5X5 convolve with a filter of 3X3 reduced to 3X3 i.e.

1. Only the patterns in the middle are observed so accurately , the border ones are highlighted less to them i.e. not equal focus on all parts of images.
2. Loss of very much useful information.



So in order to prevent this what we can do is to enlarge our image by giving a padding of required size.

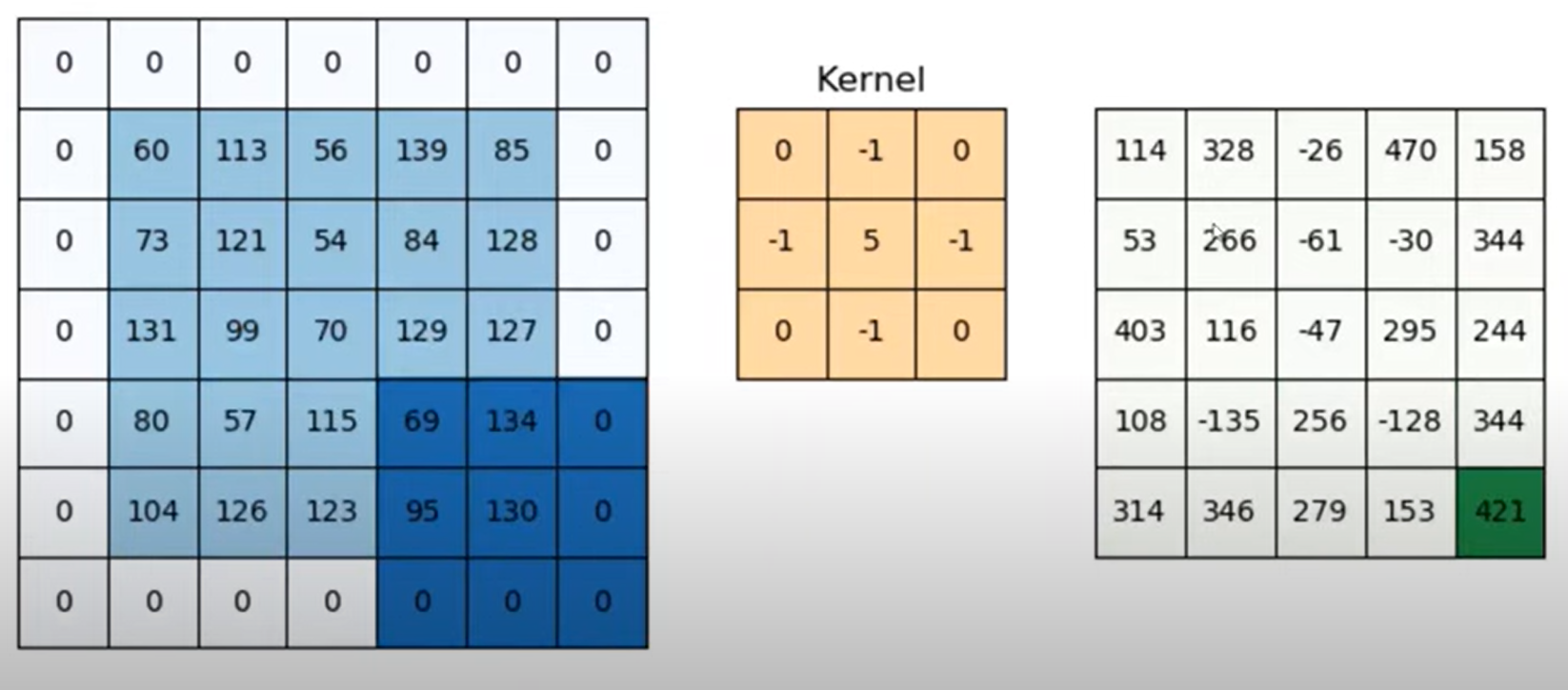
E.g. here we can give a padding of (1) so that the image becomes 7X7. So the image becomes like this . A row and column added of value as 0 .

Now even after convolving the minute patterns are undertaken.

Formula becomes ,

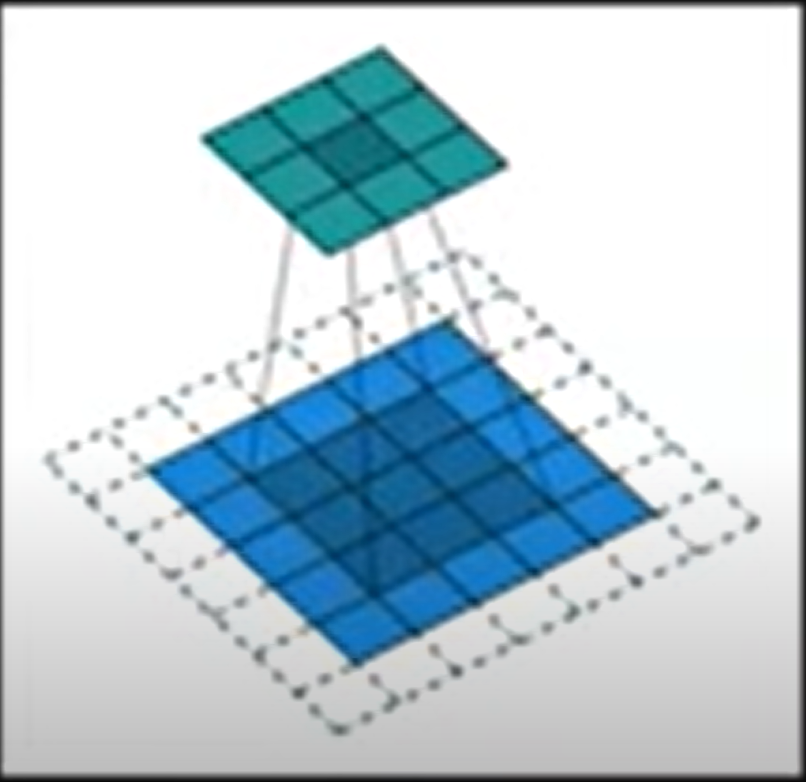
(n+2p-f+1) = image size = (5-2(1)-3+1) = 5

Padding is of 2 types Valid(padding not done) and Same.



# Strides

After one convolve , a shift of one column rightwards and a shift of one row downwards , this shift is a stride of (1,1). On increasing stride feature map decreases and more loss of information , while on decreasing you will get more features.

Feature map size = [(n +2p -f)/2 + 1]

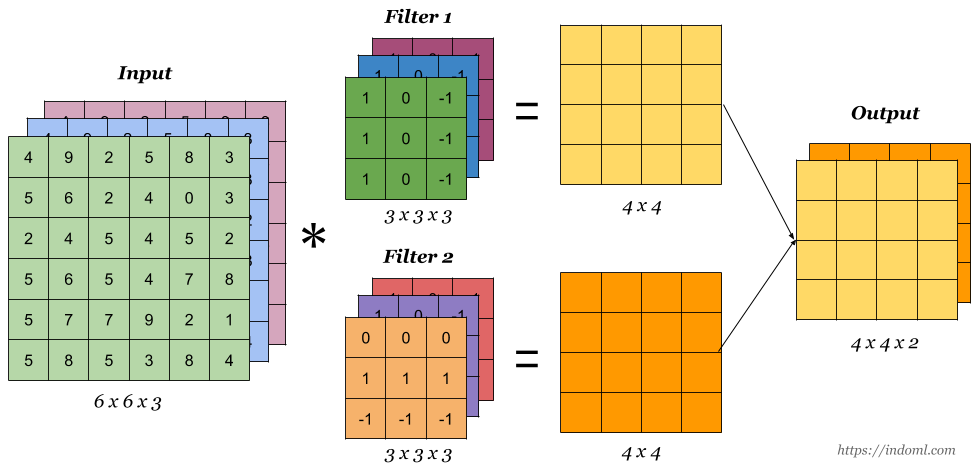
Strides are required for :

1. Capturing much of the high level features ignoring low ones for a large training set.
2. Reducing computing time .

# 

# The problem with Convolution

## Memory issue



Let suppose the image is of 228 X 228 X 3 , and the filter is of 3 X 3 X 3 , so the feature map becomes 226 X 226 . let we have 100 of filters and 32 bits to process them each, so the total size will be :

226 X 226 X 100 X 32 which will be approximately 19 MB for a single image . If we have a training data of about 100 images then it will take approximately 1.5GB. Now this computation is for 1 training data which is very very large to store.

## Translation Variance

It depicts the location dependent feature detection . we want translation invariance i.e. the feature is there doesn't matter its location.

To solve this problem Pooling is used i.e. to down sample your feature map which solves both the issues .

# Pooling

Pooling requires 3 parameters :

1. Size of window (receptive field)
2. Stride (2)
3. Type (max)

Pooling is basically done after applying an activation function like relu on the feature map.

We basically eliminate low level details .

## Advantages of Pooling :

1. Reduced size

(228 X 228 x 3) \* (3 X 3 X 3) —> 226 X 226 X 100 —--> 113 X 113 X 100

1. Translation invariance
2. More Enhanced features (only in case of Max pooling)
3. No need for training , max pooling is just an aggregate function .

## Types of Pooling

1. Max pooling
2. Min pooling
3. Average pooling
4. Global pooling

1. Global max pooling

2. Global average pooling

Global pooling can also be used as a replacement of flatten in the end .

## Disadvantages of Pooling

1. At some places location is an important aspect in object identification.
2. Loss of a lot of information .