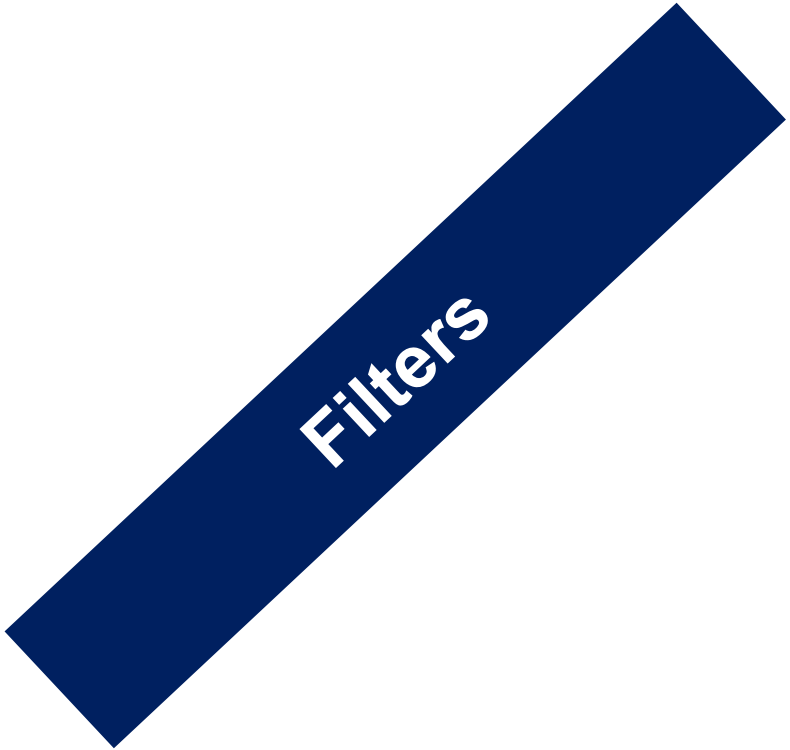
IDC410
A course on Image Processing and Machine Learning
(Lecture 04)

Shashikant Dugad,

IISER Mohali

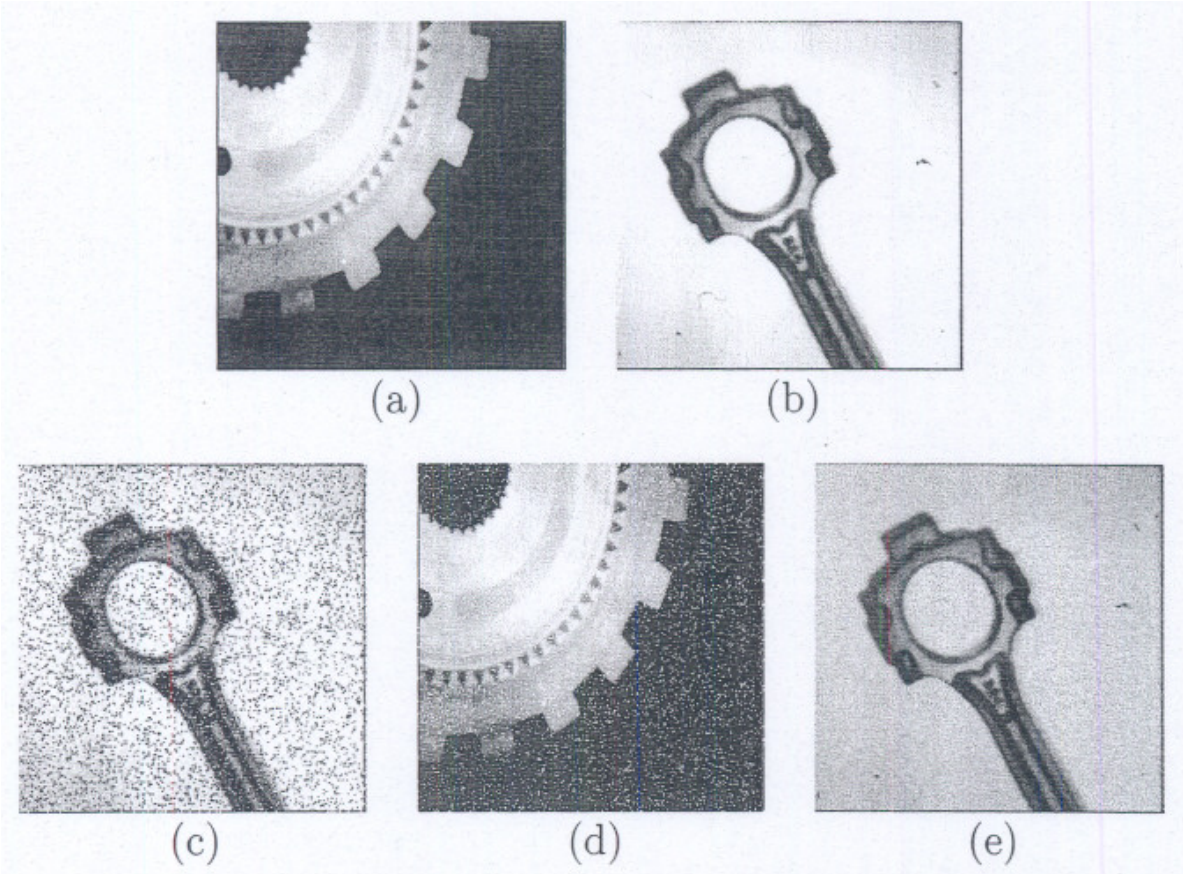

Filters

Filters

- Purpose: To reduce the noise in the image
- Improve the visibility of all features of the image
  - Brightness
  - Contrast
- Types of filters:
  - Mean Filter, Median Filter, Gaussian Filter, Convolution Filter ….
- Concepts of filters used neural network
- Purpose: Reduce noise, extract features, create images that are rich in feature of interest, reduce size etc.

Noise types

- Images are often corrupted by random variations in intensity values, called *noise.*
- Some common types of noise are *salt and pepper* noise, *impulse* noise, and *Gaussian* noise.
  - Salt and pepper noise contains random occurrences of both black and white intensity values.
  - Impulse noise contains only random occurrences of white intensity values.
  - Gaussian noise contains variations in intensity that are drawn from gaussian or normal distribution and is a very good model for many kinds of sensor noise, such as the noise due to camera electronics



Images corrupted by salt and pepper, impulse, and Gaussian noise. (a) & (b) Original images. (c) Salt and pepper noise. (d) Impulse noise. (e) Gaussian noise.

Mean Filter

- The mean filter is implemented by a local averaging operation where the value of each pixel is replaced by the average of all the values in the local neighbourhood where *M* is the total number of pixels in the neighbourhood.

- If f(i,j) is an initial image and h(i, j) is the final image after applying mean filter of say 3 x 3 neighbourhood then, h(i, j) can be obtained as,

$$h(i,j) = \frac{1}{9} \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} f(k,l)$$

Mean Filter

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Smoothening Filter

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

Mean Filter Result

Original Image

MF: 3 x 3

MF: 5 x 5

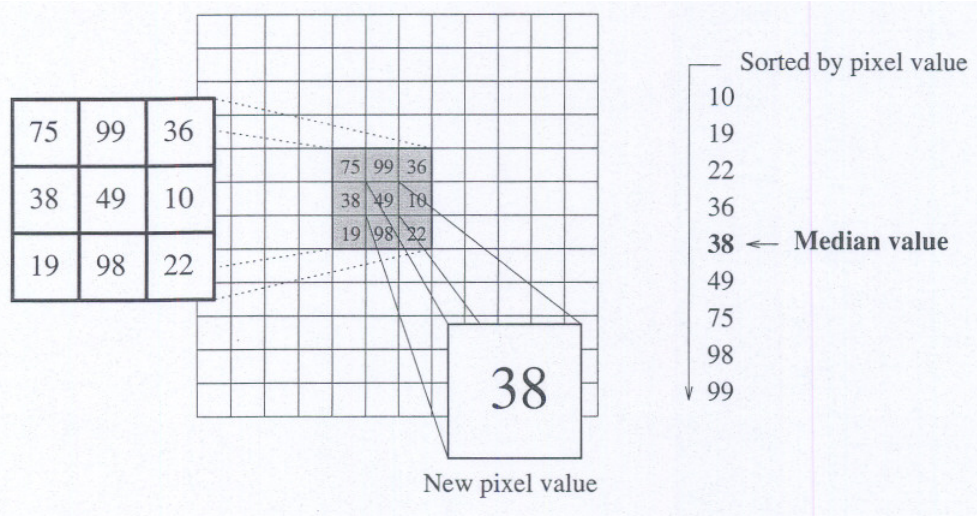
MF: 7 x 7

Median Filter

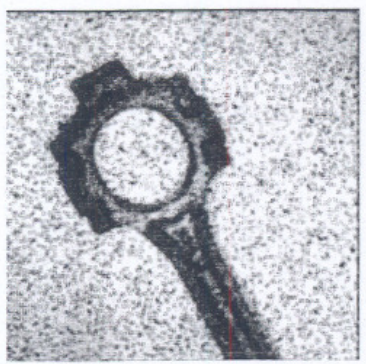
- Local averaging operations tend to blur sharp discontinuities in intensity values in an image.
- An alternative approach is to replace each pixel value with the median of the gray values in the local neighbourhood.
- Filters using this technique are called *median filters.*
- Median filters effective in removing salt and pepper and impulse noise while retaining image details because they do not depend on values which are significantly different from typical values in the neighbourhood.

Median Filter Implementation

- Median filters approach is more like mean filter
- For example, take a 3 x 3 window and compute the median of the pixels in each window cantered around [i, j]
- Sort the pixels into ascending order by gray level.
- Select the value of the middle pixel as the new value for pixel [i,j].



Median Filter Results



Original Image

MF: 3 x 3

MF: 5 x 5

MF: 7 x 7

Gaussian Filter

- Gaussian filters are a class of smoothening filters where, the weights are chosen according to the shape of a Gaussian function. The Gaussian smoothing filter is a very good filter for removing noise drawn from a normal distribution.!
- The zero-mean Gaussian function in 1-dimension and 2-dimension are:

$$g(x) = e^{-\frac{x^2}{2\sigma^2}}, \qquad g(i,j) = e^{-\frac{i^2+j^2}{2\sigma^2}}$$

- Gaussian filter properties: a) Rotational symmetry, b) Single lobe, c) Spread of smoothening governed by sigma, d) Separable.

$$g(i,j) \bigstar f(i,j) = \sum_{k=1}^{m}\sum_{l=1}^{n} g(k,l) * f(i-k, j-l) = \sum_{k=1}^{m}\sum_{l=1}^{n} e^{-\frac{k^2+l^2}{2\sigma^2}} * f(i-k, j-l)$$

$$= \sum_{k=1}^{m} e^{-\frac{k^2}{2\sigma^2}} \sum_{l=1}^{n} e^{-\frac{l^2}{2\sigma^2}} * f(i-k, j-l)$$

Gaussian Filter

- Pascal Tringles (coefficient of following equation) can be used for small size filters

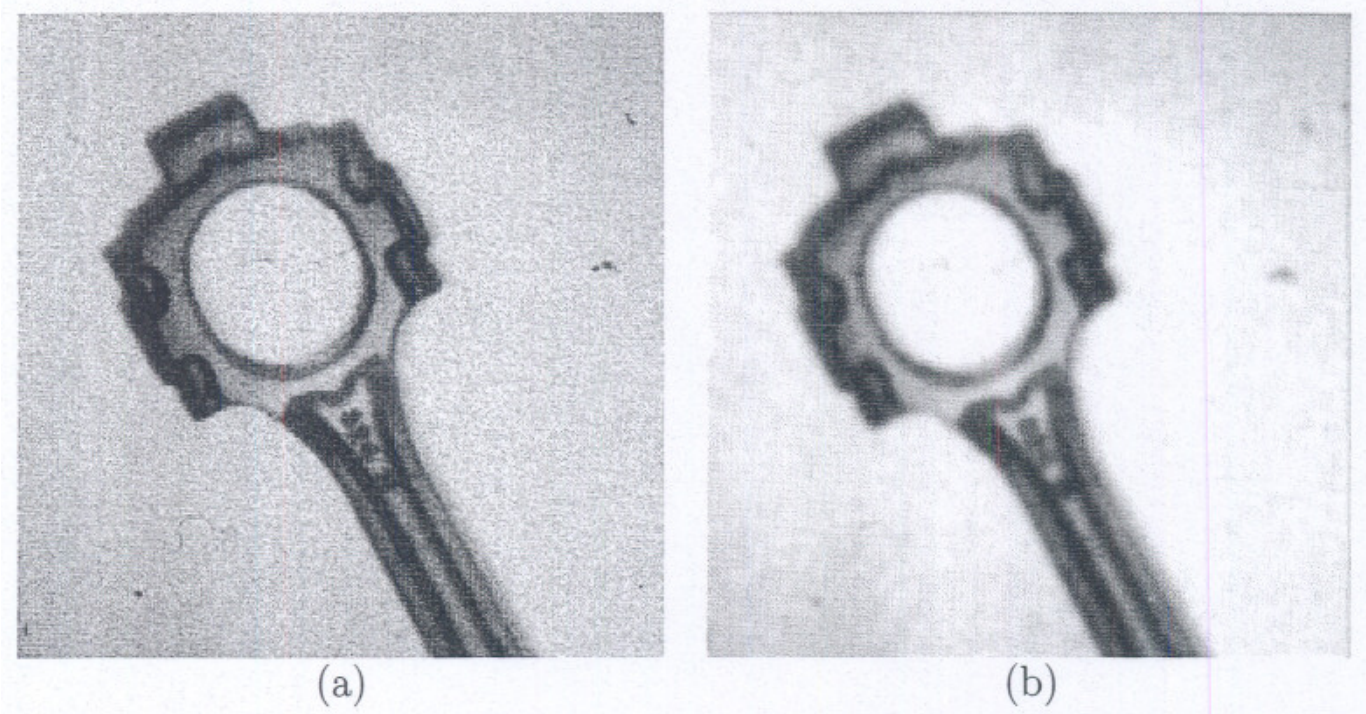
$$(x+y)^n = \sum_{r=0}^{n} C_r^n \, x^{n-r} \, y^r$$

- For n=1: [1, 1], n=2: [1,2,1], n=4: [1, 4, 6, 4, 1]
- Seperatibility property can be used to apply two dimensional filter using 1-dimensional filter
- For large n, gaussian function can be used: n=7, $\sigma^2 = 2$

$[i,j]$	−3	−2	−1	0	1	2	3
−3	.011	.039	.082	.105	.082	.039	.011
−2	.039	.135	.287	.368	.287	.135	.039
−1	.082	.287	.606	.779	.606	.287	.082
0	.105	.368	.779	1.000	.779	.368	.105
1	.082	.287	.606	.779	.606	.287	.082
2	.039	.135	.287	.368	.287	.135	.039
3	.011	.039	.082	.105	.082	.039	.011

$[i,j]$	−3	−2	−1	0	1	2	3
−3	1	4	7	10	7	4	1
−2	4	12	26	33	26	12	4
−1	7	26	55	71	55	26	7
0	10	33	71	91	71	33	10
1	7	26	55	71	55	26	7
2	4	12	26	33	26	12	4
3	1	4	7	10	7	4	1

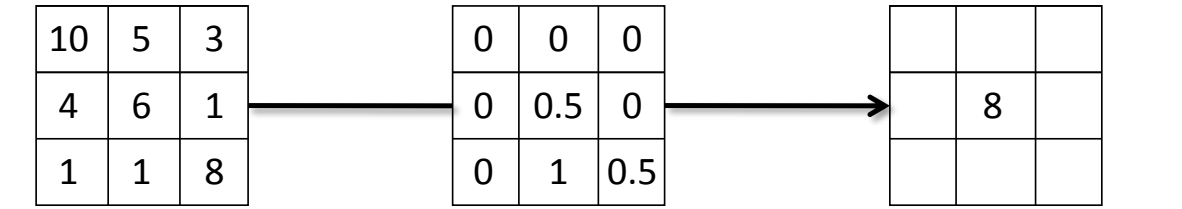
Result of Gaussian Filter



The results of smoothing using the 7x7 Gaussian mask. (a) Original image corrupted by Gaussian noise. (b) Smoothed image.

Linear Filtering

- **Linear Filtering: Replace each pixel by a linear combination of its neighbours and self**
  - **cross-correlation, convolution**
- **The prescription for the linear combination is called the "kernel" (or "mask", "filter")**



- **Same Kernel applied across entire image are referred as Space Invariant Filters**

Cross-correlation

- Let F be the input image and H be the kernel (of size 2k+1 x 2k+1), and G be the output image

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u,j+v]$$

- This is called a cross-correlation operation:

$$G = H \otimes F$$

Convolution

- **Same as cross-correlation, except that the kernel is "flipped" (horizontally and vertically)**

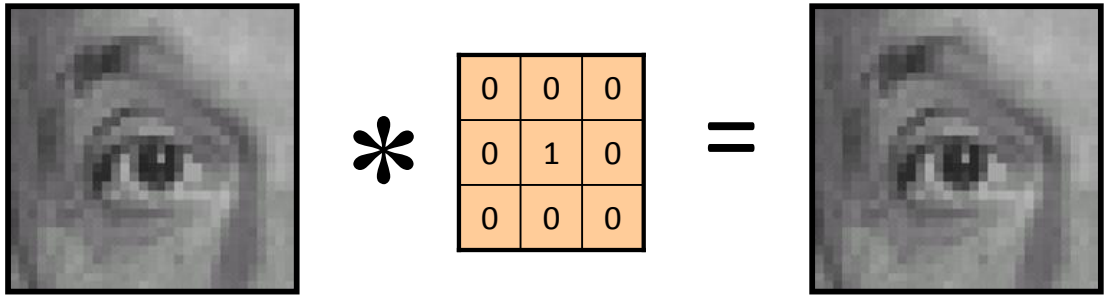
$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u,j-v]$$

- **This is called a convolution operation:**

$$G = H * F$$

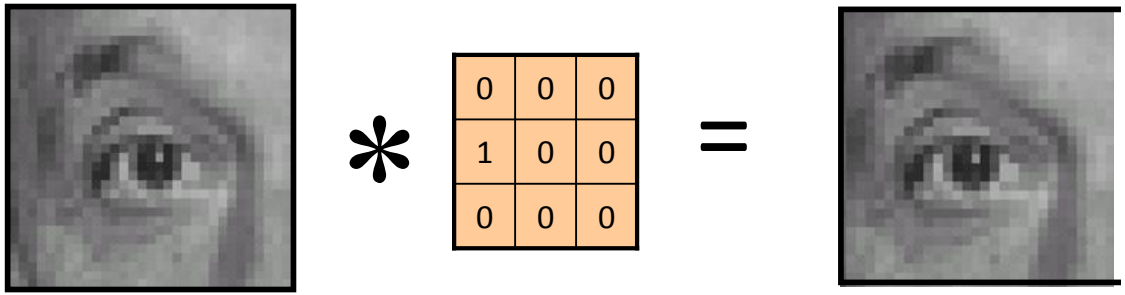
- **Convolution is commutative and associative**

Convolution



Original

Identical image

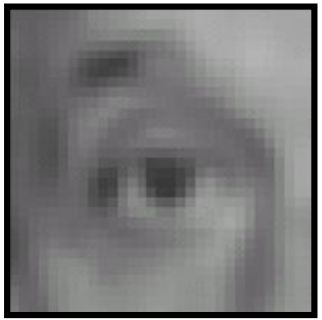


Original

Shifted left
By 1 pixel

Convolution

$$* \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$



Original

Blur (with a mean filter)

$$* \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right) =$$

Original

Sharpening filter
(accentuates edges)

Properties of Convolution Filter

- **Notation: b = c ★ a**
- **Convolution is a multiplication-like operation**
  - **Commutative: a ★ b = b ★ a**
  - **Associative: a ★ (b ★ c) = (a ★ b) ★ c**
  - **Distributes over addition: a ★ (b + c) = (a ★ b) + (a ★ c)**
  - **Scalars factor out: αa ★ b = a ★ αb = α(a ★ b)**
  - **Identity: unit impulse e = […, 0, 0, 1, 0, 0, …]: a ★ e = a**
- **Usefulness of associativity**
  - **Often apply several filters one after another: (((a ★ b1) ★ b2) ★ b3)**
  - **This is equivalent to applying one filter: a * (b1 ★ b2 ★ b3)**

Padding and Strides

- In Convolutional Neural Networks (CNNs), stride refers to the step size by which the filter/kernel moves across the input image during the convolution operation in horizontal and vertical direction
  - Stride defines how big of steps filters should take (i.e., how many pixels our filters should skip) while sliding over the image
  - Minimum value of stride = 1
  - Smaller strides (1 or 2) offer detailed feature extraction, while larger strides (3+) aid in down-sampling.
- CNN is like a collection of small, overlapping magnifying glasses called filters. These filters scan over different parts of a image to find interesting features, like edges, shapes, or colours. These filters slide or convolve over the entire image as defined by the stride.
- The kernel size, stride value can substantially reduce the size of output im the output size and computational efficiency of the network, influencing feature extraction and spatial dimensions of image.

Movement of Filter

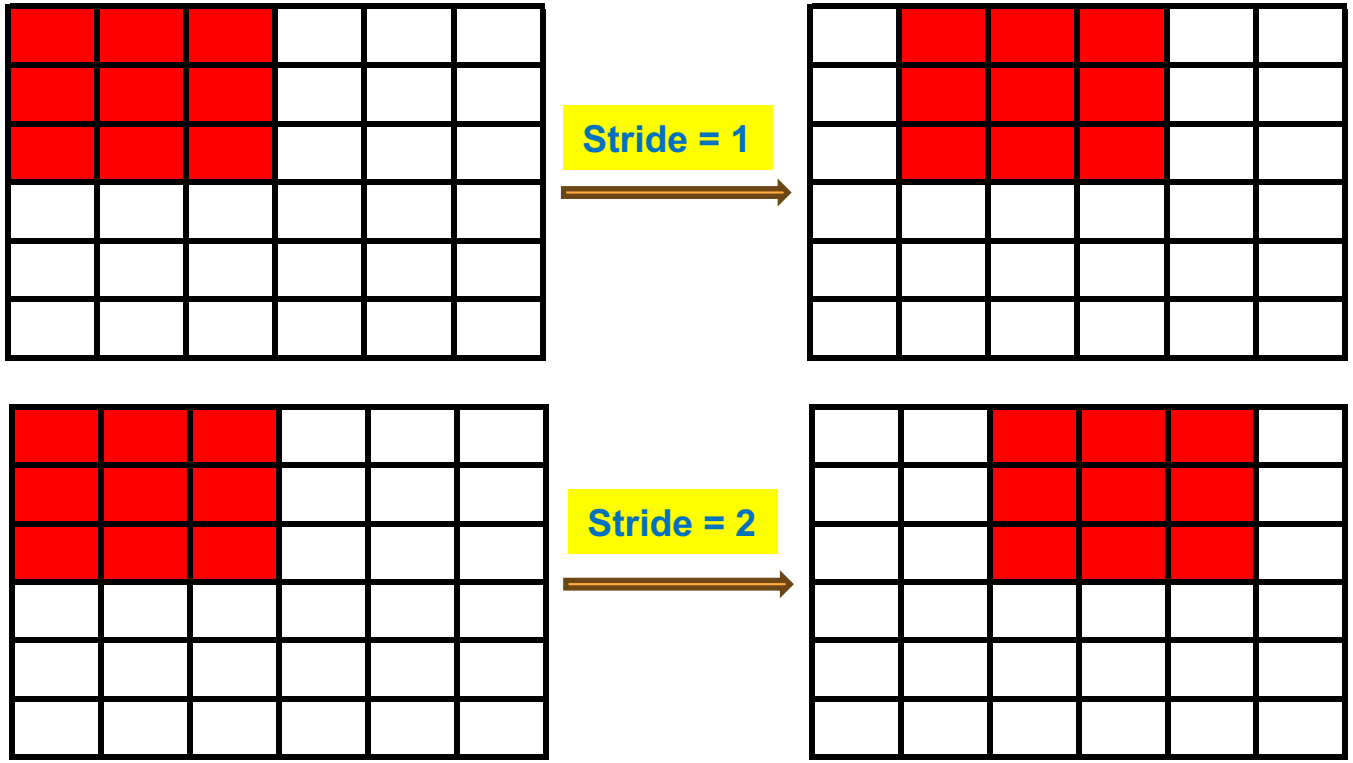


Image Padding

Input Image (5 x 5)

22	145	23	167	67
45	110	45	119	29
78	99	78	88	112
145	100	99	38	164
223	110	23	45	29

Pad = 1

Output Image (7 x 7) [Pad = 1]

0	0	0	0	0	0	0
0	22	145	23	167	67	0
0	45	110	45	119	29	0
0	78	99	78	88	112	0
0	145	100	99	38	164	0
0	223	110	23	45	29	0
0	0	0	0	0	0	0

Input Image (5 x 5)

22	145	23	167	67
45	110	45	119	29
78	99	78	88	112
145	100	99	38	164
223	110	23	45	29

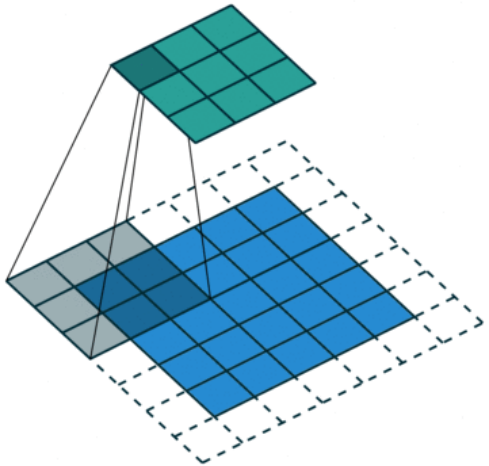
Pad = 2

Output Image (9 x 9) [Pad = 2]

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	22	145	23	167	67	0	0
0	0	45	110	45	119	29	0	0
0	0	78	99	78	88	112	0	0
0	0	145	100	99	38	164	0	0
0	0	223	110	23	45	29	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

- Padding means adding extra columns and rows with ZERO pixel intensity before doing any operations.
- Helps in keeping the spatial info intact, particularly prevents data loss at the edges hence stabilises training
- It also helps to keep the output size consistent with the input and makes training more stable.

Padding and Strides



- Image Dimension: N_{row} X M_{col}
- Padding: P Strides: S_{row}, $S_{col.}$ Dilation: D_{row}, D_{col}
- Kernal Size: K_{row}, K_{col}

$$N_{row}^{out} = \frac{N_{row}^{in} + 2\,X\,P \quad - D_{row}\,X\,(K_{row} - 1) - 1}{S_{row}} + 1$$

$$M_{col}^{out} = \frac{M_{col}^{in} + 2\,X\,P \quad - D_{col}\,X\,(K_{col} - 1) - 1}{S_{col}} + 1$$