CNN

Stands for
Convolutional Neural Network (ConvNN)

CNN

- Convolutional neural network (CNN) is a regularized type of feed-forward neural network that learns feature engineering by itself via filters (or kernel) optimization.

CNN applications examples:

- image and video recognition
- image classification,
- medical image analysis,
- natural language processing

Architecture

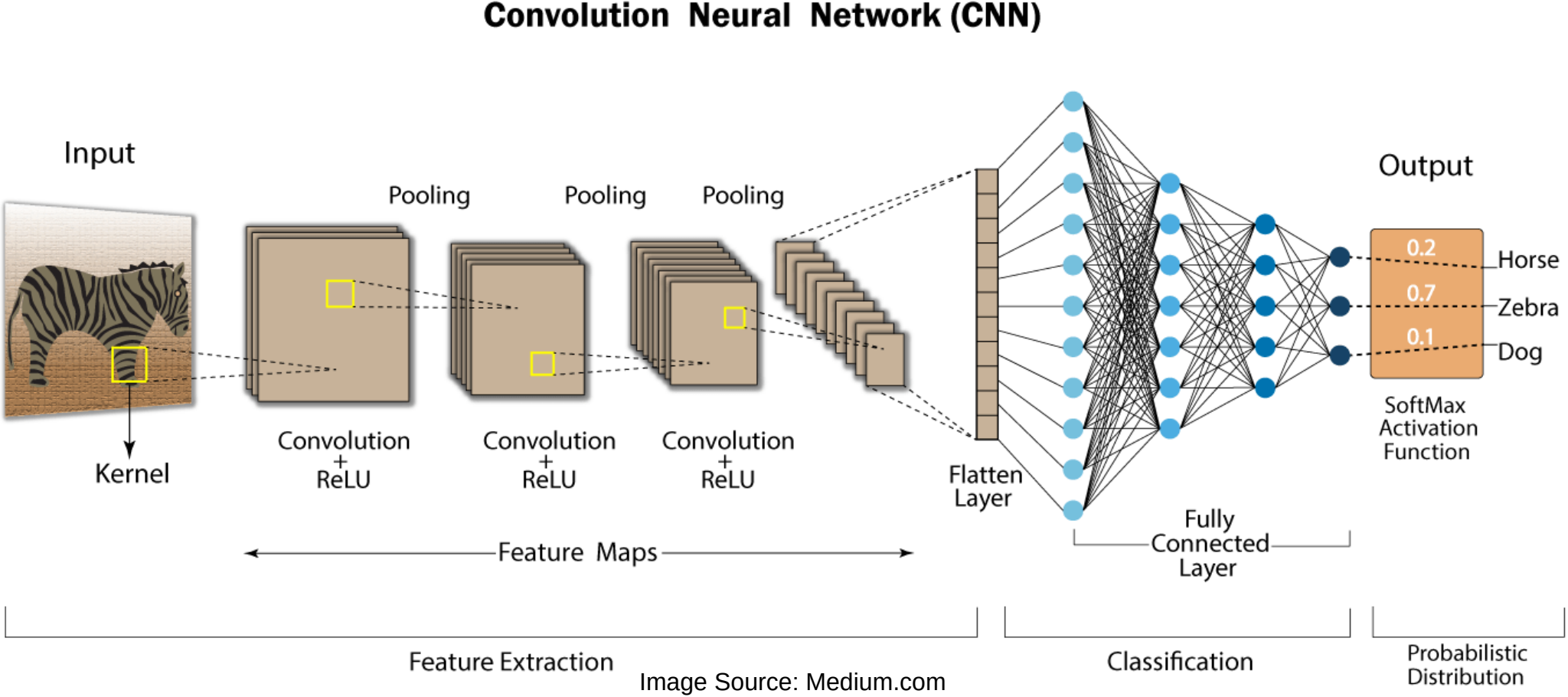
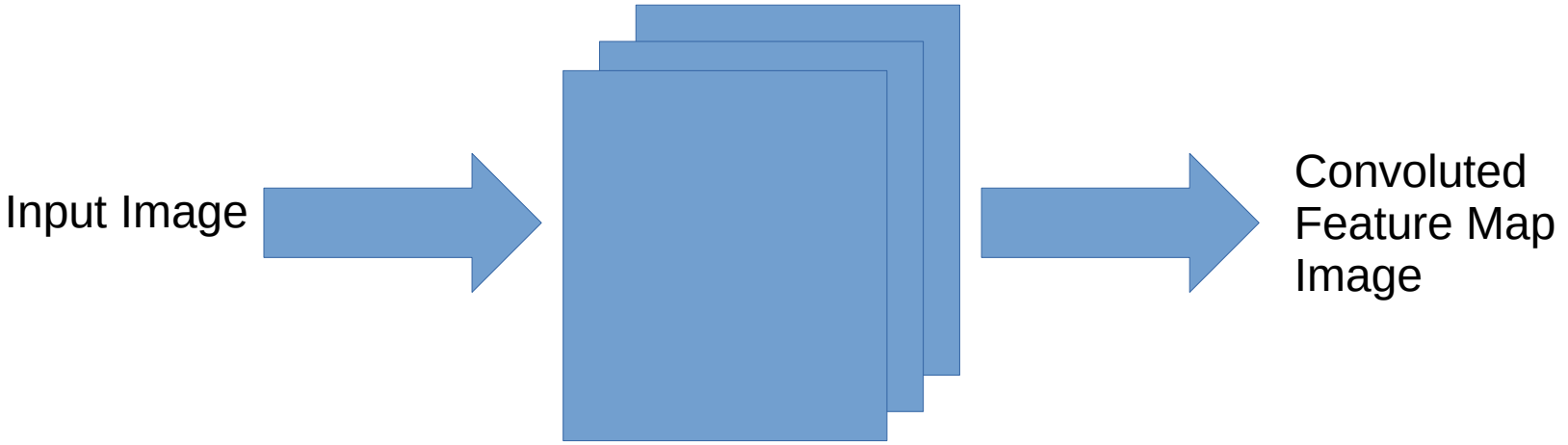
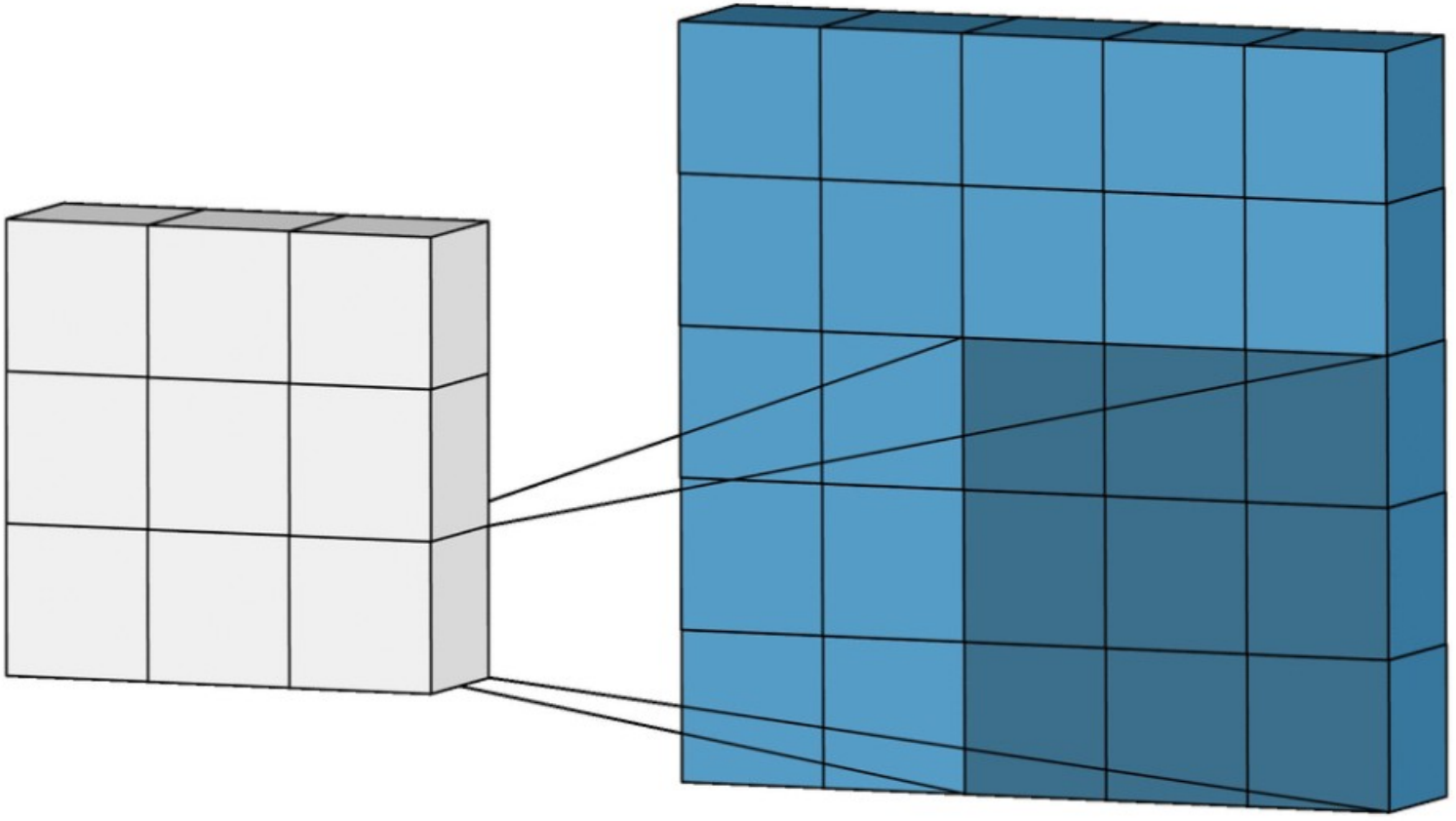


Image Source: Medium.com

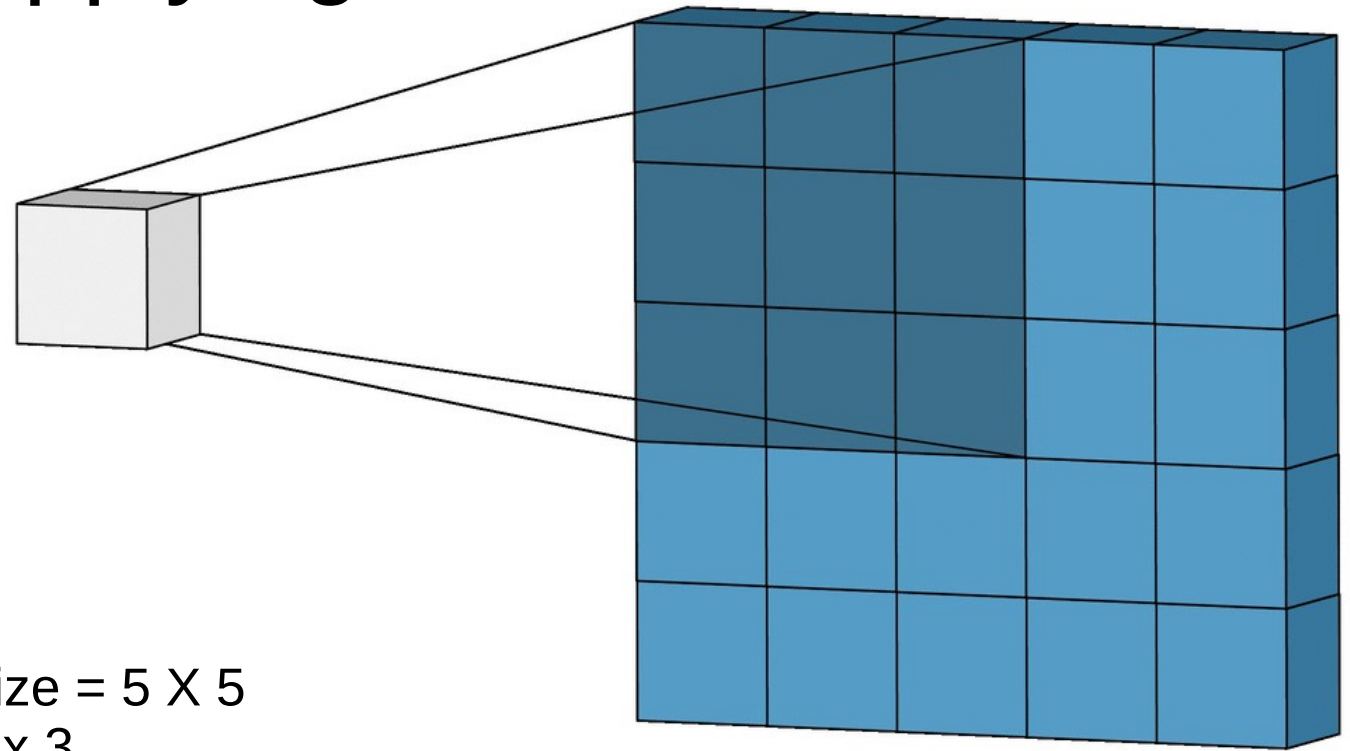
Convolutional Layer

The Primary purpose of Convolution is to extract features from input image

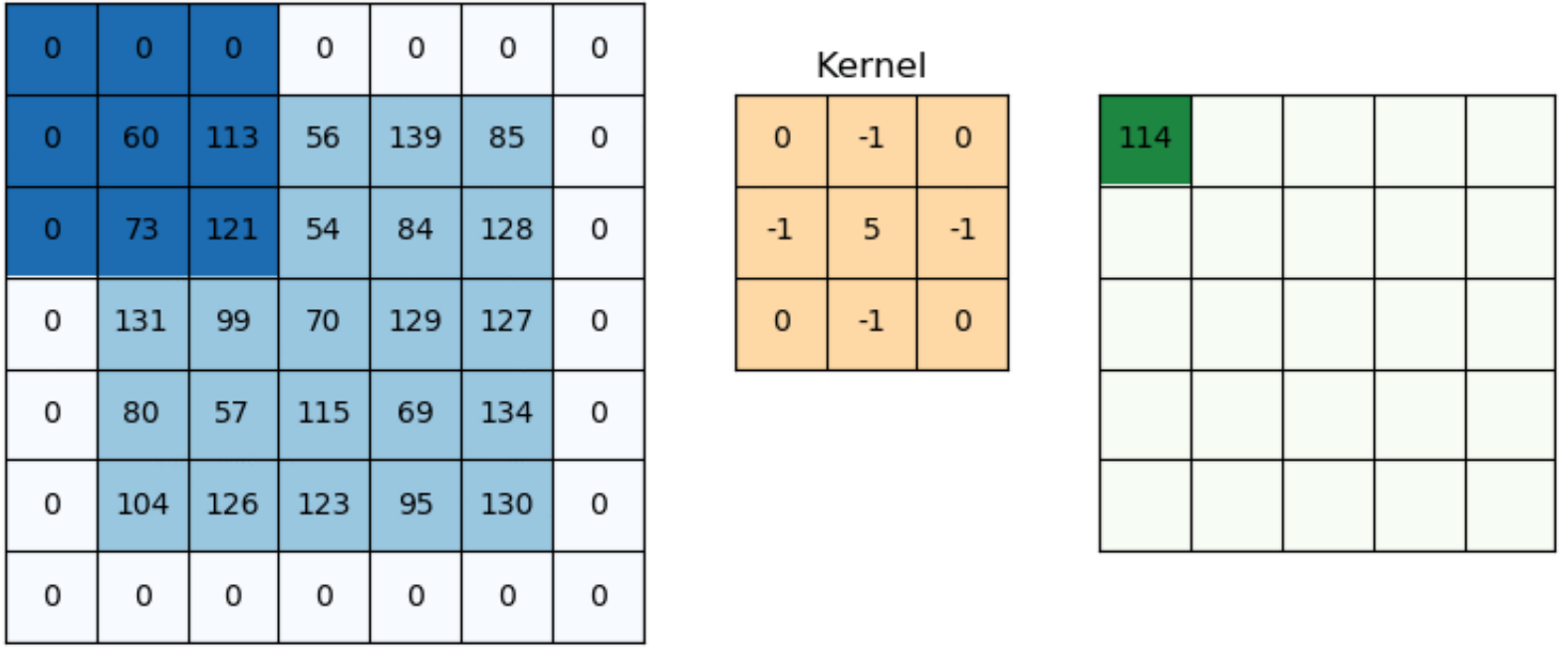


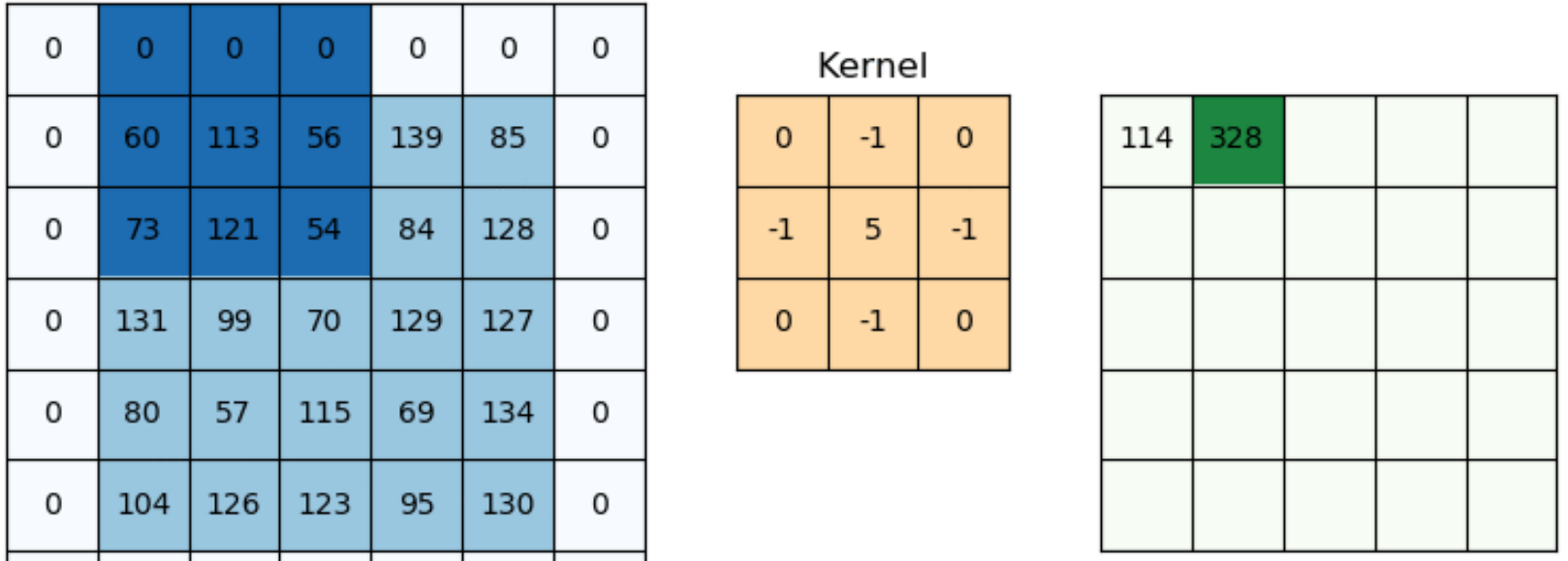


Applying filters Convolutional layer cont..



Input Image Size = 5 X 5
Filter Size = 3 x 3
Stride = 1
Padding = 0
Feature map size(output) = 3 x 3


0 0 0 0 0 0 0
0 60 113 56 139 85 0
0 73 121 54 84 128 0
0 131 99 70 129 127 0
0 80 57 115 69 134 0
0 104 126 123 95 130 0
0 0 0 0 0 0 0
Kernel
0 -1 0
-1 5 -1
0 -1 0
114


0 0 0 0 0 0 0
0 60 113 56 139 85 0
0 73 121 54 84 128 0
0 131 99 70 129 127 0
0 80 57 115 69 134 0
0 104 126 123 95 130 0
0 0 0 0 0 0 0
Kernel
0 -1 0
-1 5 -1
0 -1 0
114 328

Applying filters Convolutional layer cont..

0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

Image Source: vitalflux.com

Input Image Size = 5 X 5
Filter Size = 3 x 3
Stride = 1
Padding = 1
Feature map size(output) = 5 x 5

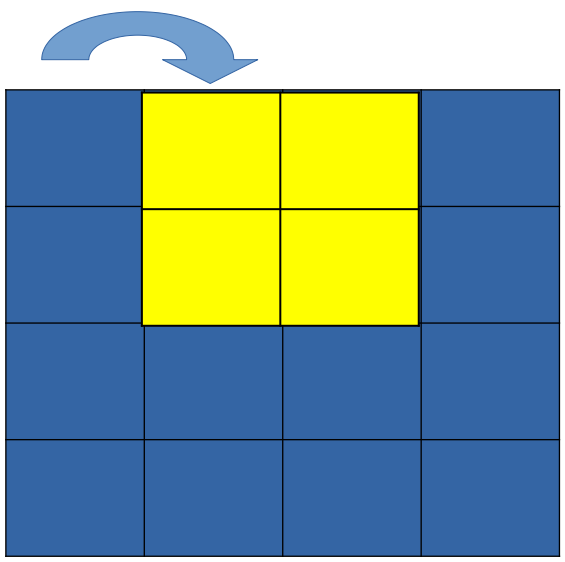
Feature map size Convolutional layer cont..

Depends on:

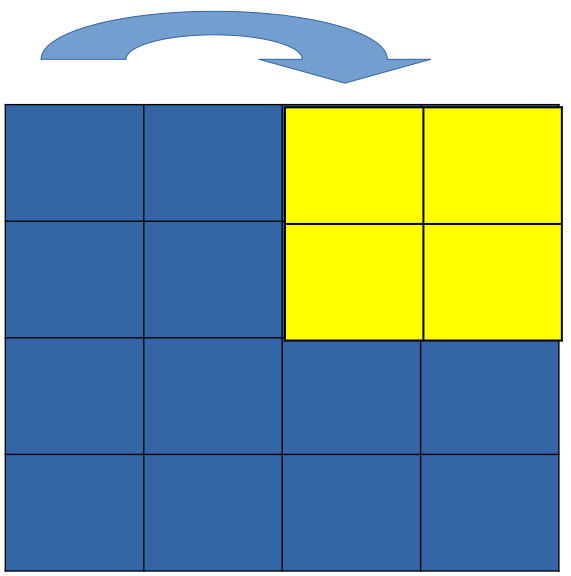
- Input Image Size n×n
- Filter Size f×f
- Stride s
- Padding p

Stride Convolutional layer cont..

- **Stride:** It denotes how many steps we are moving the filter at each step. [default is 1]



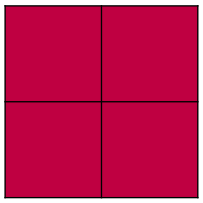
Stride = 1



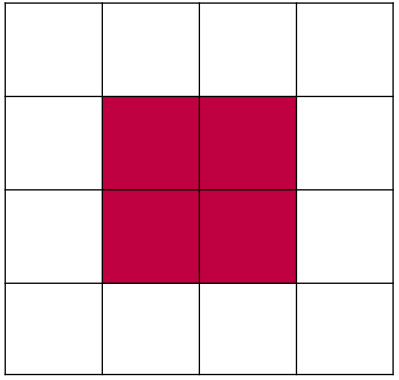
Stride = 2

Padding Convolutional layer cont..

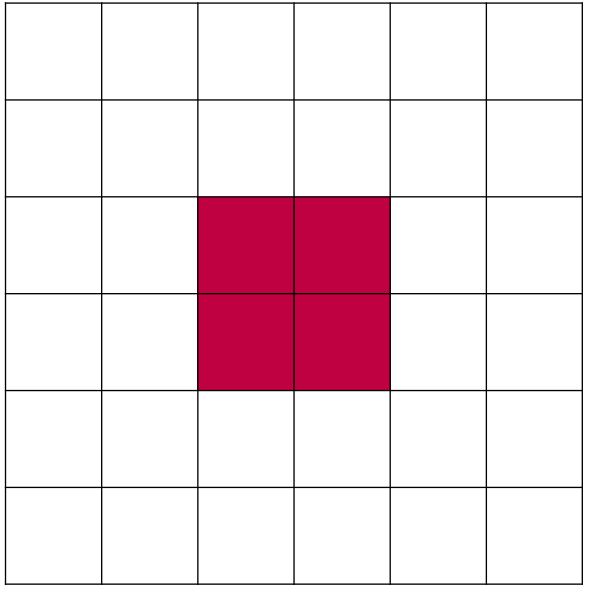
- Padding is a process of adding (row or column) to the input at each side



Padding = 0



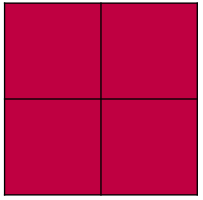
Padding = 1



Padding = 2

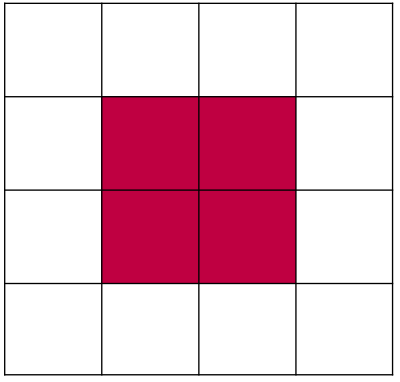
Padding Types Convolutional layer cont..

Valid Padding



Padding = 0

Same Padding



Padding = 1

Calculate Padding Convolutional layer cont..

$$p = \frac{n * s - n + f - s}{2}$$

$$let\ s = 1\ ,\ p = \frac{f - 1}{2}$$

Calculate the feature map size

$$feature\ map = \frac{n - f + 2p}{s} + 1$$

n×*n* image
f×*f* filter

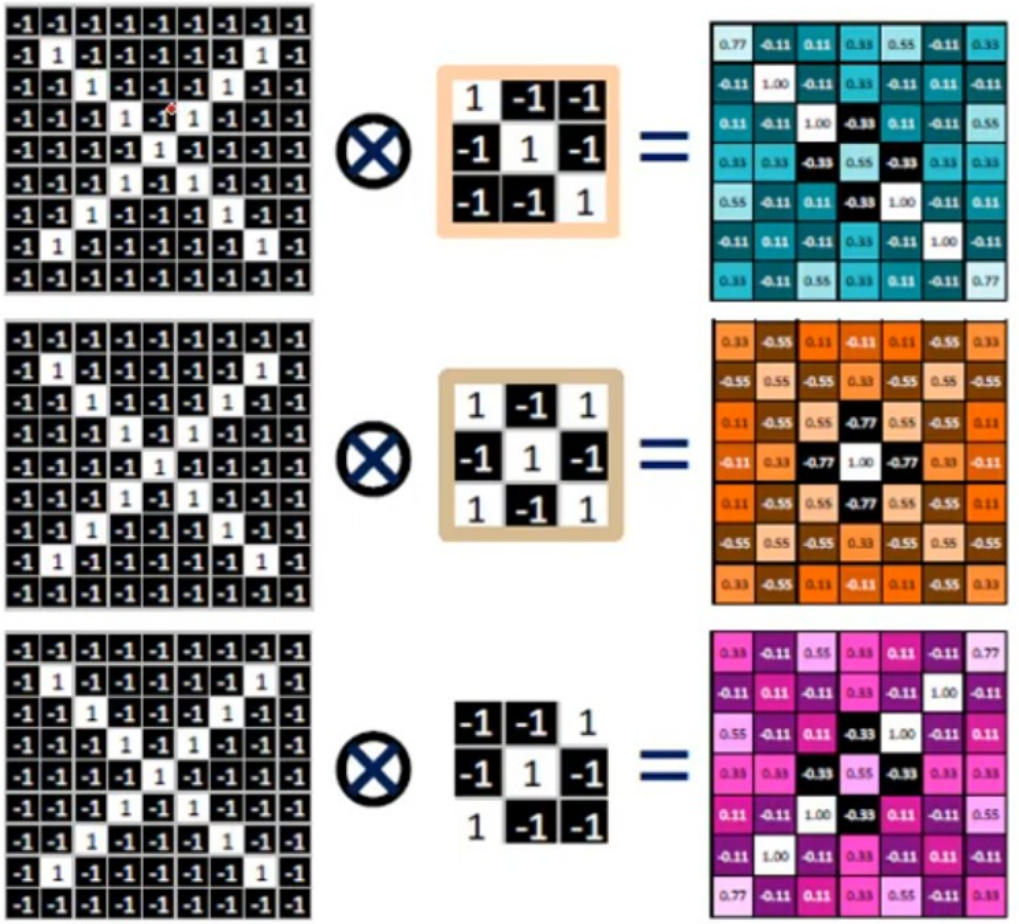
Padding *p*
Stride *s*

Convolutional Layer – Filters – Output Feature Map

- Output Feature Map of One complete convolution:
  - Filters: 3
  - Filter Size: 3 X 3
  - Stride: 1
- Conclusion:
  - Input Image:

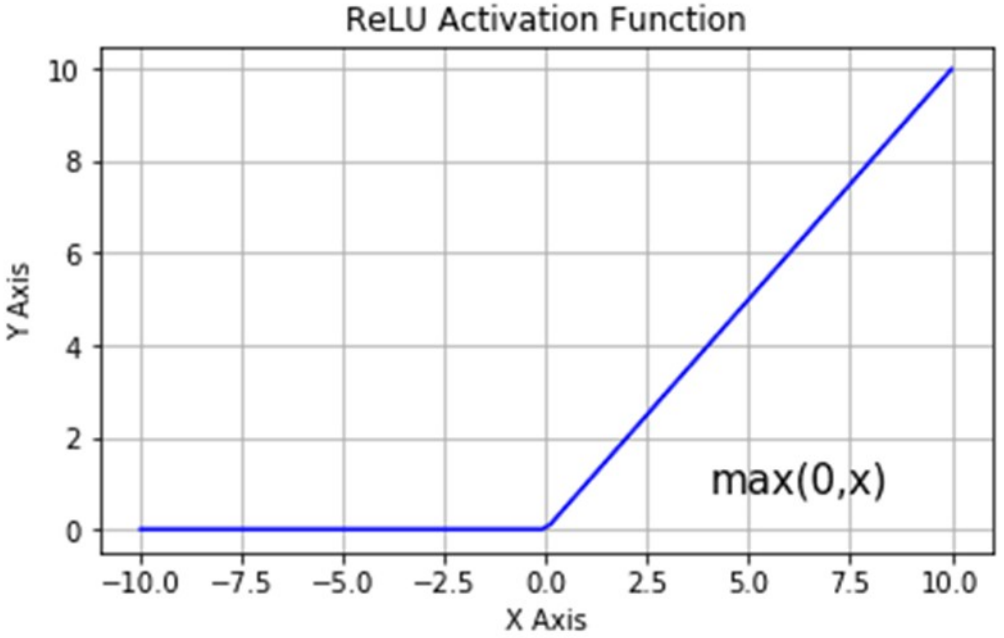
    9 X 9
  - Output of Convolution:

    7 X 7 X 3

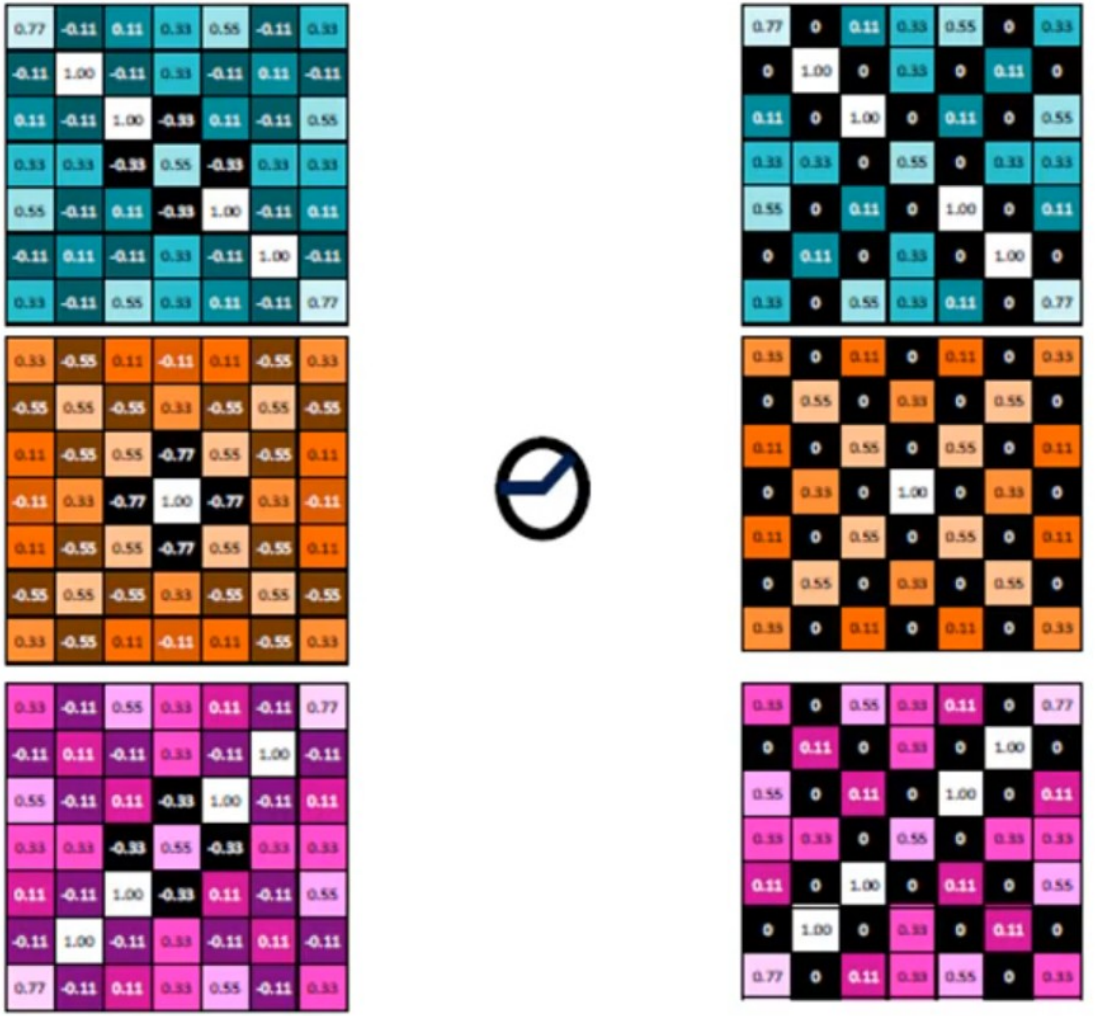


ReLU layer

- Applying max(0, x) on the previous feature map layers
- This one does not change size unlike the previous one



Relu Layer


0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
0.77 0 0.11 0.33 0.55 0 0.33
0 1.00 0 0.33 0 0.11 0
0.11 0 1.00 0 0.11 0 0.55
0.33 0.33 0 0.55 0 0.33 0.33
0.55 0 0.11 0 1.00 0 0.11
0 0.11 0 0.33 0 1.00 0
0.33 0 0.55 0.33 0.11 0 0.77
0.33 -0.55 0.11 -0.11 0.11 -0.55 0.33
-0.55 0.55 -0.55 0.33 -0.55 0.55 -0.55
0.11 -0.55 0.55 -0.77 0.55 -0.55 0.11
-0.11 0.33 -0.77 1.00 -0.77 0.33 -0.11
0.11 -0.55 0.55 -0.77 0.55 -0.55 0.11
-0.55 0.55 -0.55 0.33 -0.55 0.55 -0.55
0.33 -0.55 0.11 -0.11 0.11 -0.55 0.33
0.33 0 0.11 0 0.11 0 0.33
0 0.55 0 0.33 0 0.55 0
0.11 0 0.55 0 0.55 0 0.11
0 0.33 0 1.00 0 0.33 0
0.11 0 0.55 0 0.55 0 0.11
0 0.55 0 0.33 0 0.55 0
0.33 0 0.11 0 0.11 0 0.33
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
0.33 0 0.55 0.33 0.11 0 0.77
0 0.11 0 0.33 0 1.00 0
0.55 0 0.11 0 1.00 0 0.11
0.33 0.33 0 0.55 0 0.33 0.33
0.11 0 1.00 0 0.11 0 0.55
0 1.00 0 0.33 0 0.11 0
0.77 0 0.11 0.33 0.55 0 0.33

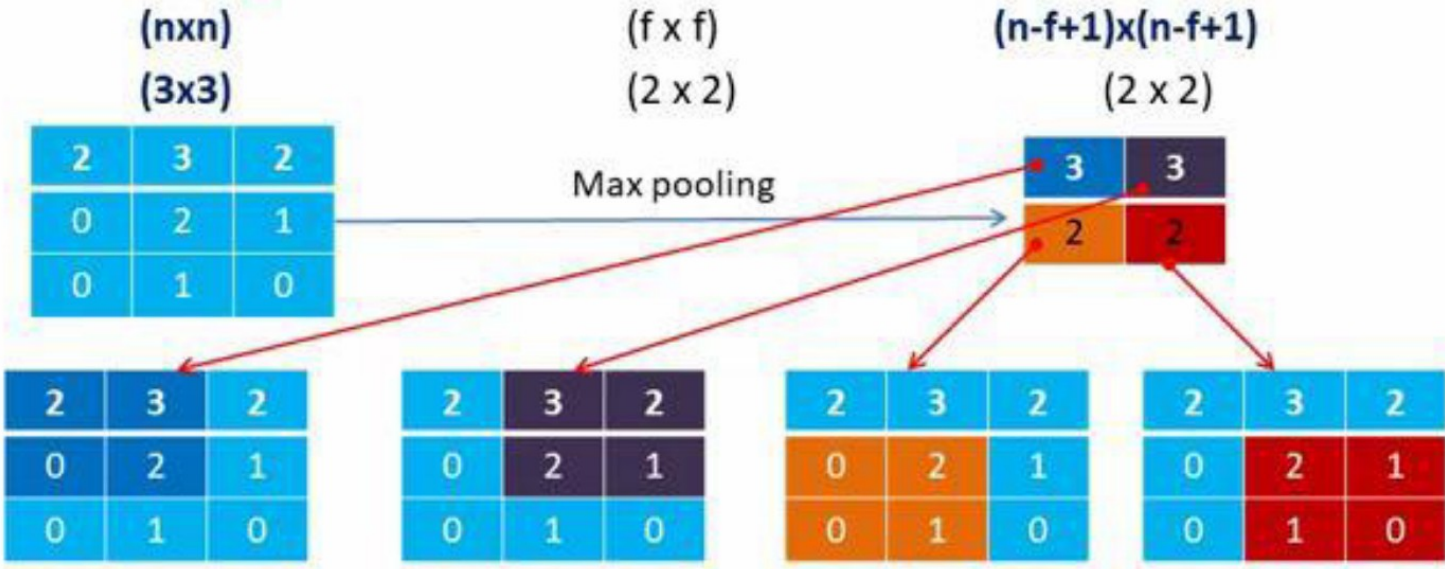
Pooling layer

- Its purpose is to gradually shrink the representation’s spatial size to reduce the number of parameters and computations in the network.
- The pooling layer treats each feature map separately.

Pooling layers

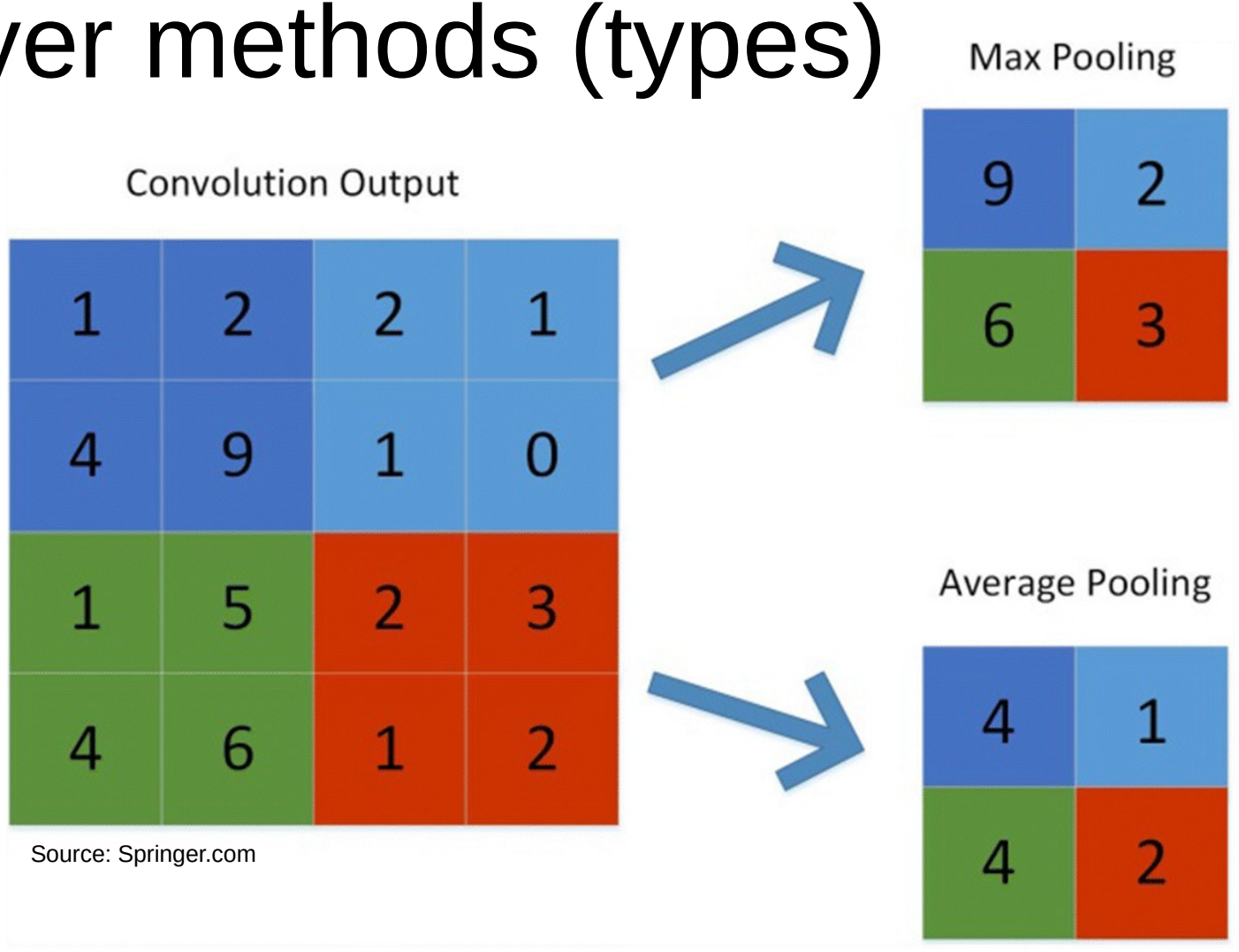
- Pooling layers would reduce the number of parameters when the inputs are too large.
- Pooling also called down sampling which reduces the dimensionality of each map but retains important information.
- There are three types of pooling namely, Max Pooling, Average Pooling, Sum Pooling.

Max Pooling:



Pooling layer methods (types)

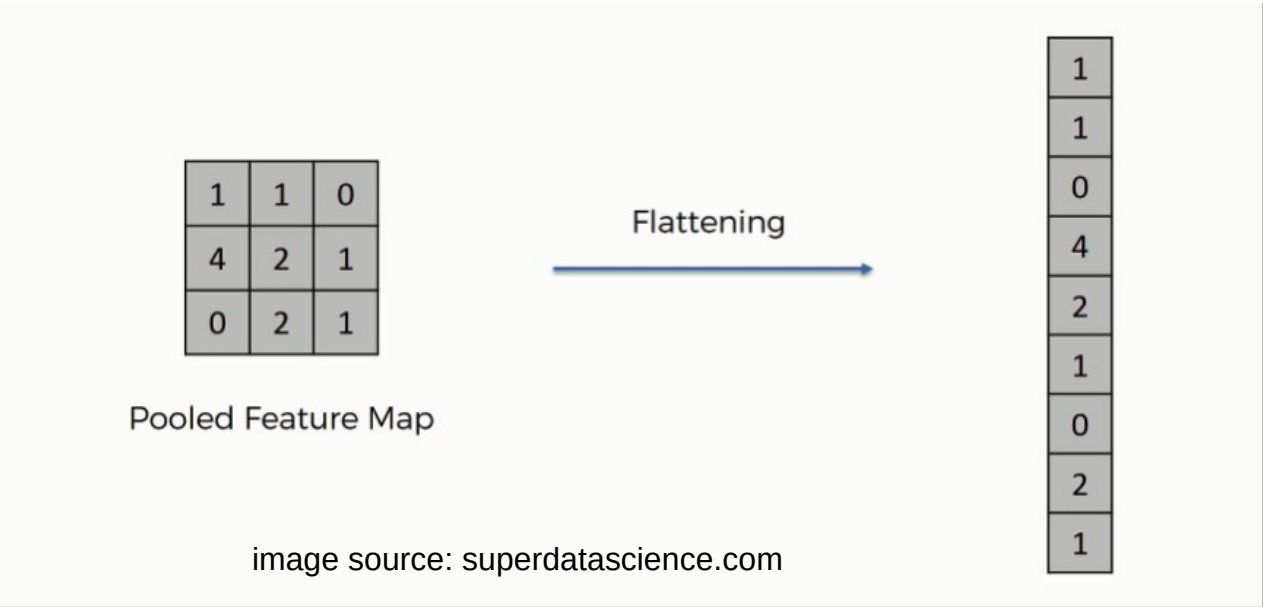
- **Max pooling**
- **Average Pooling**



Source: Springer.com

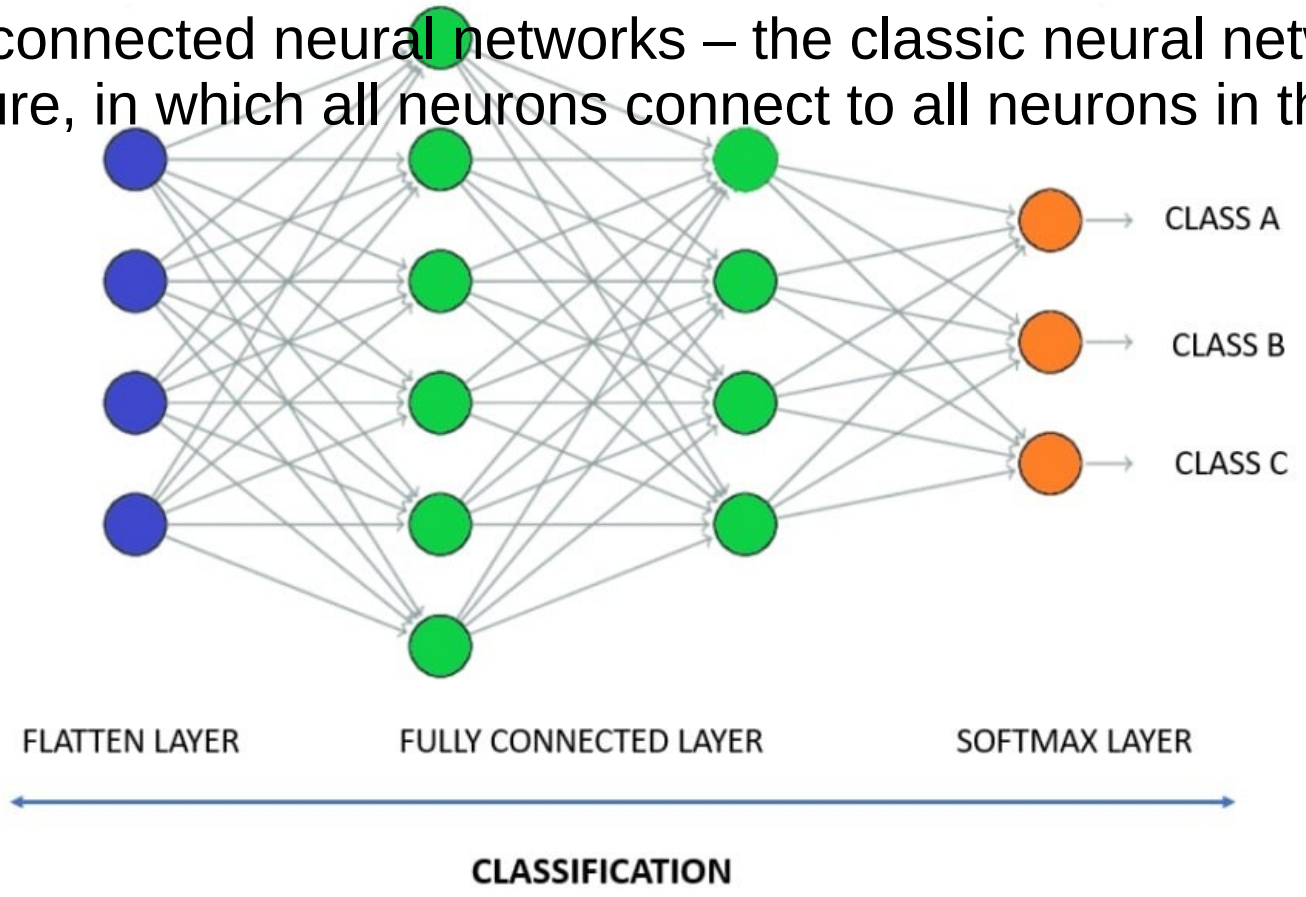
Flatten Layer

- **Flatten** is used to flatten the input. For example, if flatten is applied to layer having input shape as (3,3), then the output shape of the layer will be (9)



Fully Connected layer

- Is a fully connected neural networks – the classic neural network architecture, in which all neurons connect to all neurons in the next layer.



MNIST Example

- MNIST is a subset of a larger set available from NIST (it's copied from http://yann.lecun.com/exdb/mnist/)
- The MNIST database of handwritten digits has a training set of 60,000 examples, and a test set of 10,000 examples.

Resources

- Convolutional neural network [wikipedia]
- Feed-forward neural network [wikipedia]
- Basics of CNN in Deep Learning
- Intuitively Understanding Convolutions for Deep Learning
- Convolutional Neural Network CNN- الشبكات العصبية الملتفة by dr. Ahmed Yousry [youtube]
- What is ReLU and Sigmoid activation function?
- Convolutional Neural Networks (CNN): Step 3 - Flattening
- Feature map size calculate in CNN | Stride, Padding | Deep Learning Animation [youtube]
- Fully Connected Layers in Convolutional Neural Networks
- Deep Neural Networks: Padding
- #02 Convolutional neural network : MNIST Dataset (99% accuracy) [youtube]
- Deep Learning with Keras [youtube]
- AhmedIbrahimai/How-Convolutional-Neural-Networks-CNNs-Works [github]