Convolutional Neural Network - Intuition

Introduction:

- Image can be converted into matrix of pixel values
- These pixel values ranges from 0 to 255
- Dimension of this matrix will be of

  [Image width x Image height x number of channels]

- A grayscale image has one channel

Size [3 x 3 x 1] >> Only one 2D matrix of shape (3 x 3)

0	2	3
9	2	3
2	2	1

- coloured image have three channels (RGB)

Size [3 x 3 x 3] >> Totally three matrix and each of 2D matrix of shape (3 x 3)

Each refers channel Red, Green Blue respectively

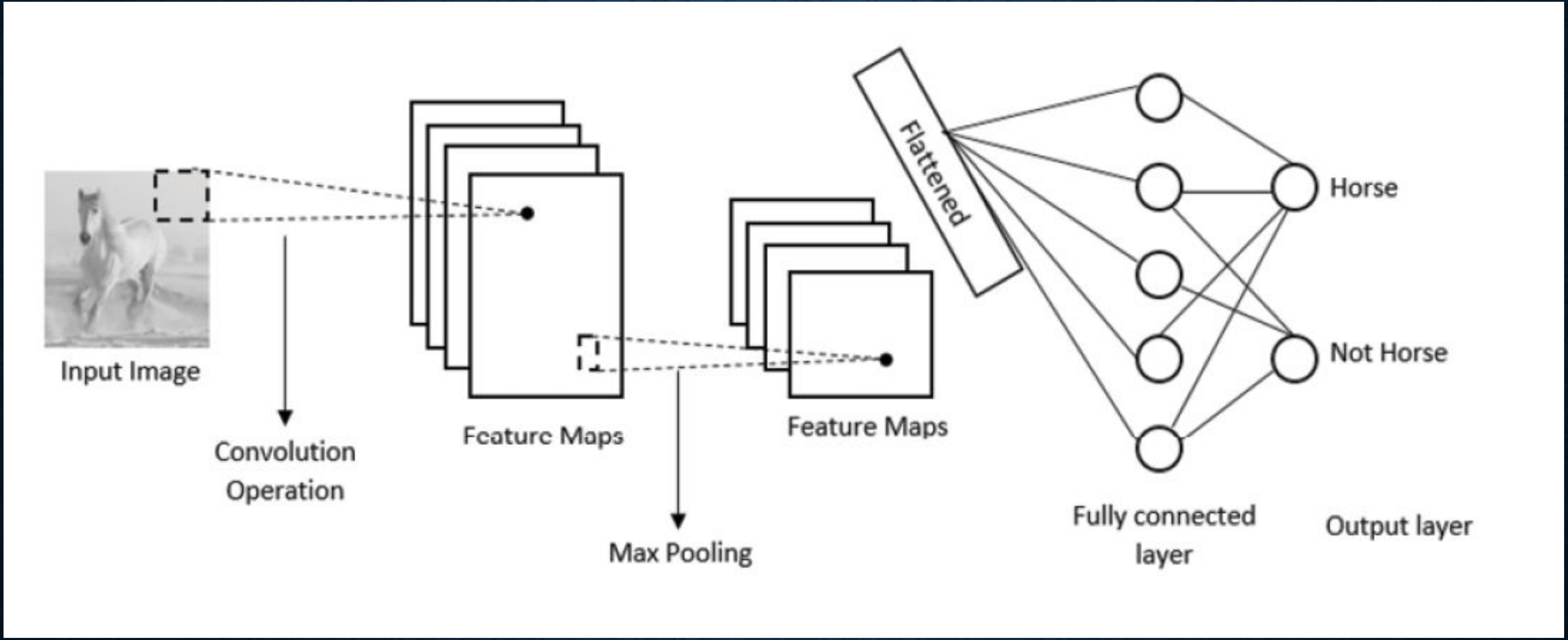
0	2	3
9	2	3
2	2	1

4	2	3
9	7	3
2	2	10

6	2	5
9	13	3
2	2	5

Convolutional Neural Network Architecture:

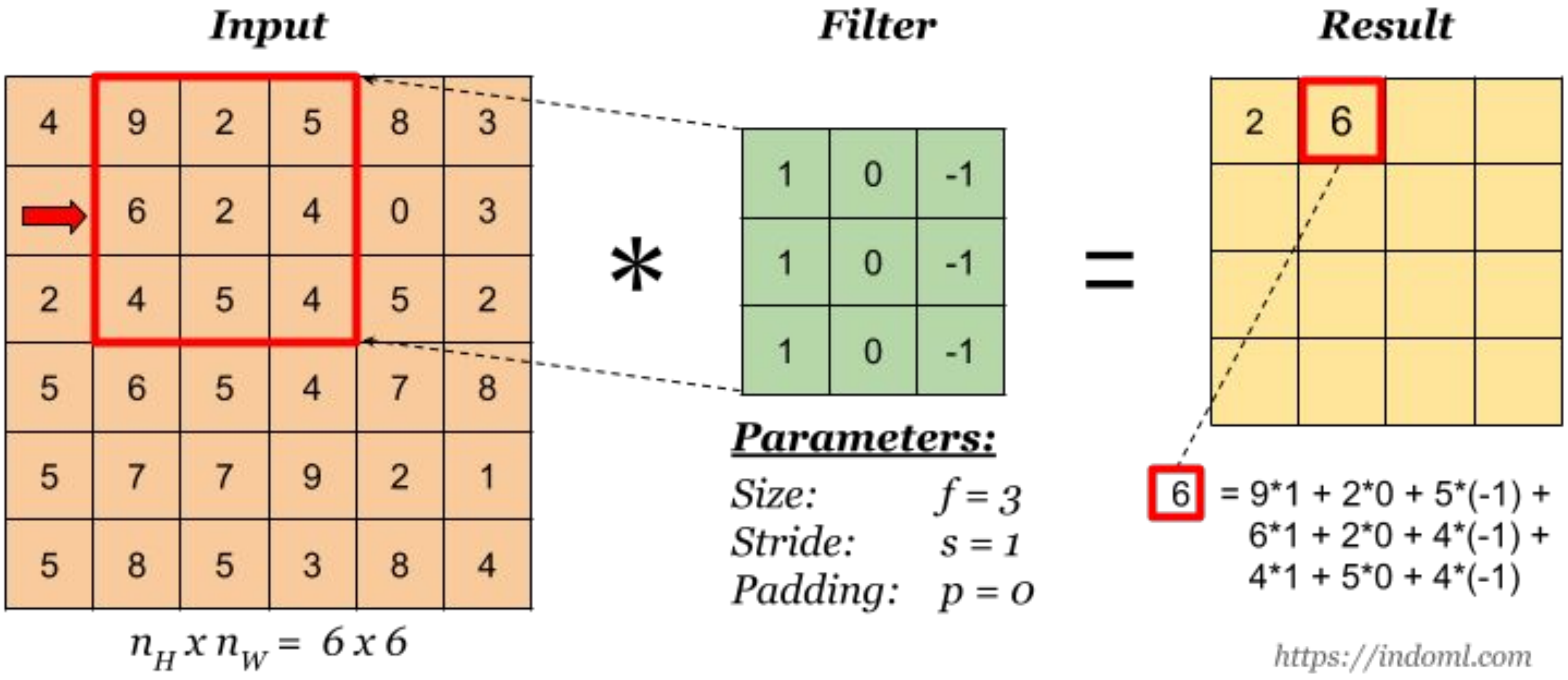
- CNN is extracting features from the image to understand patterns about it.



Patterns are normally : Edges, Shapes, Textures, Curves, Objects, Colours

Convolutional Operation:

- Input image is represented by a matrix of pixel values and there is a filter or kernel matrix which will do convolution
- This filter matrix slide over the input matrix by stride i.e pixel, perform element-wise multiplication, sum the results and produce a single number.



Input

4	9	2	5	8	3
	6	2	4	0	3
2	4	5	4	5	2
5	6	5	4	7	8
5	7	7	9	2	1
5	8	5	3	8	4

$n_H \, x \, n_W = \, 6 \, x \, 6$

$*$

Filter

1	0	-1
1	0	-1
1	0	-1

Parameters:

Size: $f = 3$
Stride: $s = 1$
Padding: $p = 0$

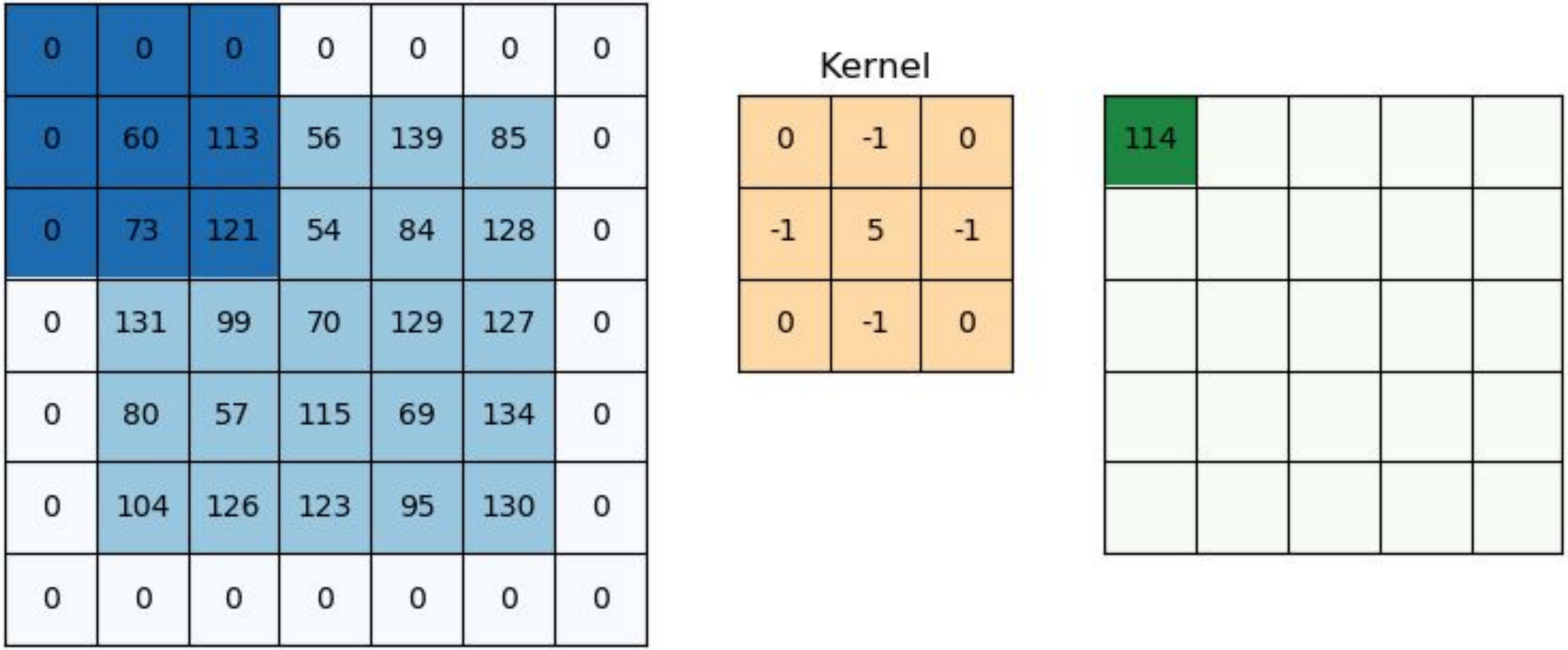
$=$

Result

2	6		

6 = 9*1 + 2*0 + 5*(-1) +
6*1 + 2*0 + 4*(-1) +
4*1 + 5*0 + 4*(-1)

https://indoml.com



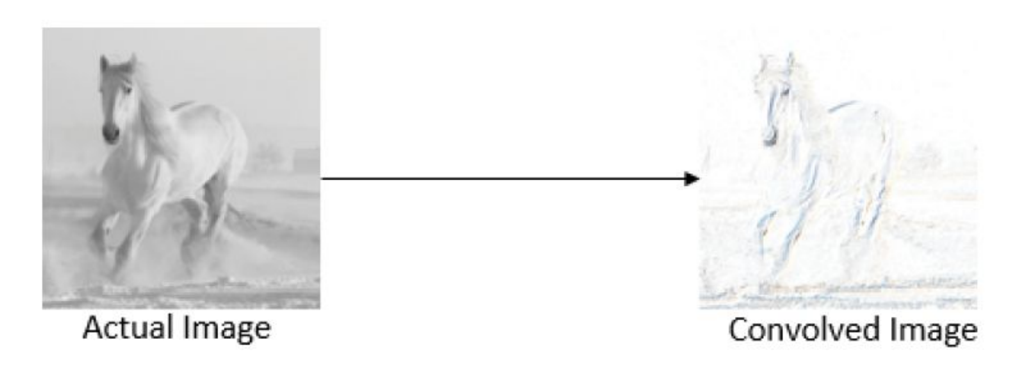
0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

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

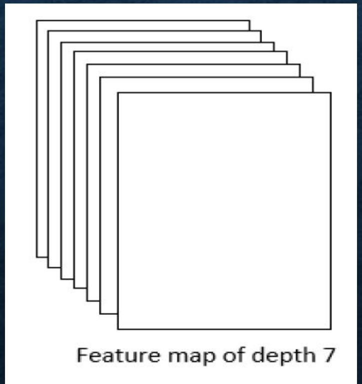
114				

- Feature map representing the extracted features.



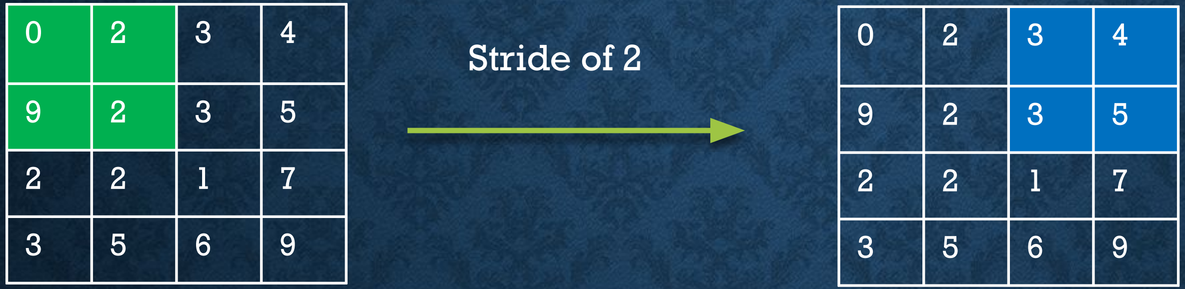
Here filter has detected the edges from the actual image

- Can use multiple filters to extract different features from the image
- Depth of the feature map will be the number of filters
- These filter matrix can be initialized randomly and the optimal values of filter matrix will be learned by back propagation.
- It is expected of us to specify size of the filter and number of filters during building of CNN
- For example if convolution operation is defined with seven filter, there will be seven feature map



Importance of Strides:

- When stride is set to small number, can encode a more detailed representation of the image.
- When stride is set to high value , can encode with less time to compute but not a detailed encoding.
- If stride is set to 2, than slide over the input matrix with the filter matrix by two pixels as below,



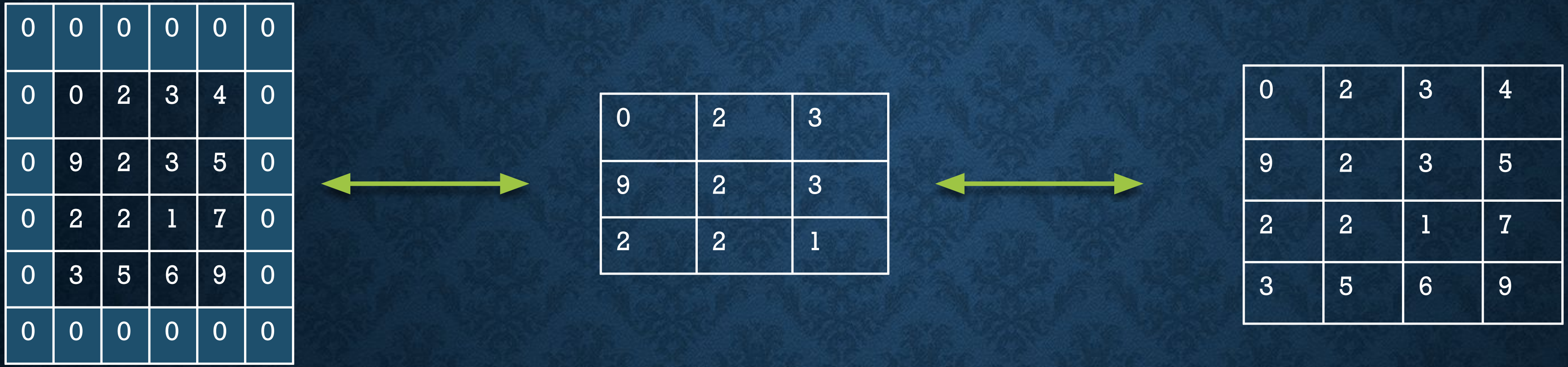
0	2	3	4
9	2	3	5
2	2	1	7
3	5	6	9

Stride of 2

0	2	3	4
9	2	3	5
2	2	1	7
3	5	6	9

Importance of Padding:

- Due to convolving process, the input size is getting smaller and smaller at every time, whenever filter is provided
- While convolving, the information is getting lost at the edge of the image.
- To Avoid this , zero padding pixels are introduced around the edges of image and output image size is not decreased



0	0	0	0	0	0
0	0	2	3	4	0
0	9	2	3	5	0
0	2	2	1	7	0
0	3	5	6	9	0
0	0	0	0	0	0

Input matrix : Before padding 4 x 4
After padding 6 x 6

0	2	3
9	2	3
2	2	1

Filter matrix 3 x 3

0	2	3	4
9	2	3	5
2	2	1	7
3	5	6	9

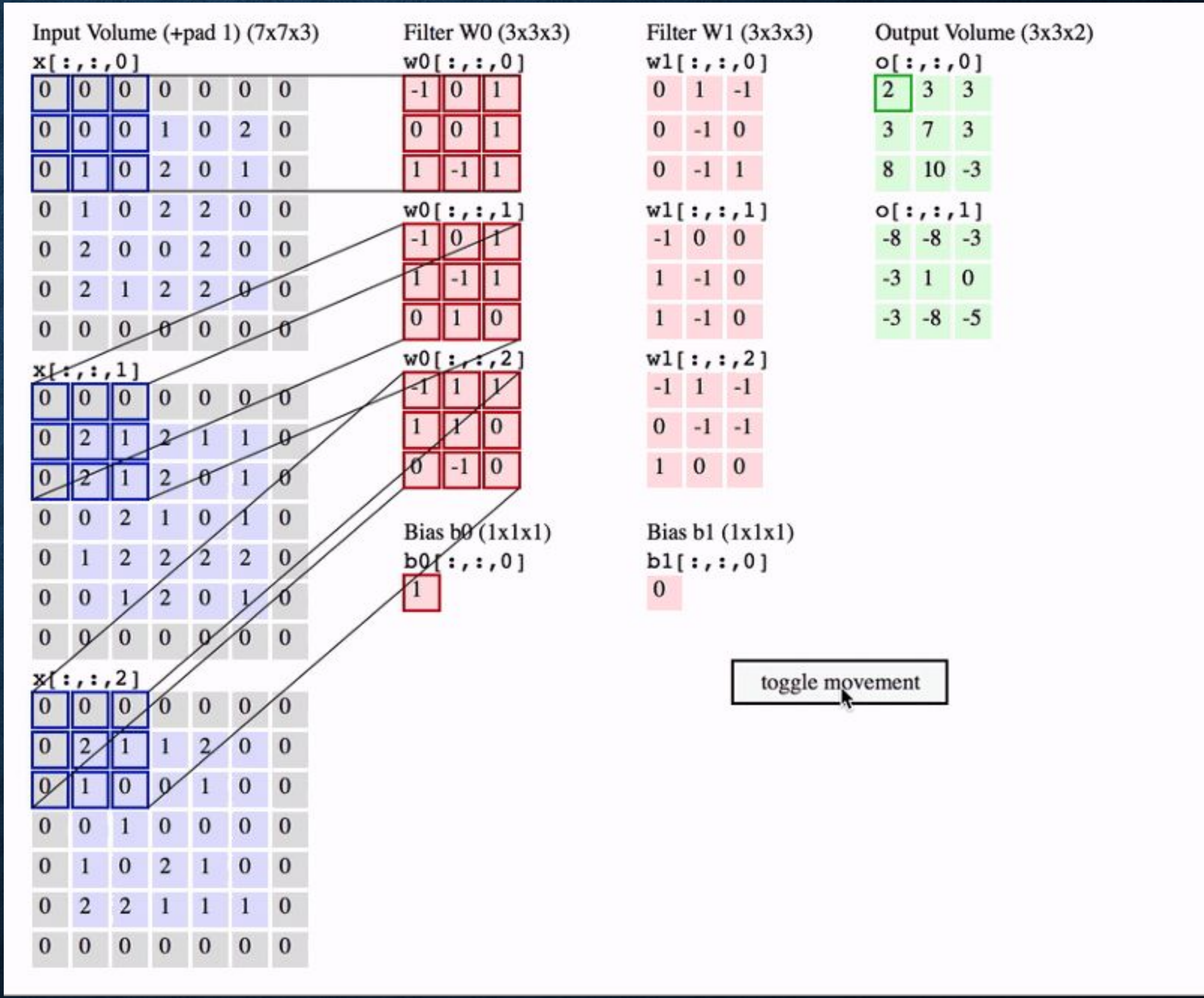
Output matrix 4 x 4

- If padding mode is ‘Valid’, then Convolution layer is not going to pad at all
- If padding mode is ‘Same’ then output size is the same as the input size after padding is applied all edges of input matrix

Formula to calculate output of feature map:-

$o = (i - f + 2*p)/s + 1$

Here,
o - output size
i - input size
f - filter size
p - padding amount
s - number of strides



Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	0	0	1	0	2	0
0	1	0	2	0	1	0
0	1	0	2	2	0	0
0	2	0	0	2	0	0
0	2	1	2	2	0	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	2	1	2	1	1	0
0	2	1	2	0	1	0
0	0	2	1	0	1	0
0	1	2	2	2	2	0
0	0	1	2	0	1	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	2	1	1	2	0	0
0	1	0	0	1	0	0
0	0	1	0	0	0	0
0	1	0	2	1	0	0
0	2	2	1	1	1	0
0	0	0	0	0	0	0

Filter W0 (3x3x3)

w0[:,:,0]

-1	0	1
0	0	1
1	-1	1

w0[:,:,1]

-1	0	1
1	-1	1
0	1	0

w0[:,:,2]

-1	1	1
1	1	0
0	-1	0

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

0	1	-1
0	-1	0
0	-1	1

w1[:,:,1]

-1	0	0
1	-1	0
1	-1	0

w1[:,:,2]

-1	1	-1
0	-1	-1
1	0	0

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

o[:,:,0]

2	3	3
3	7	3
8	10	-3

o[:,:,1]

-8	-8	-3
-3	1	0
-3	-8	-5

Importance of Pooling layers:

- Pooling Operation is also called as ‘Down Sampling’ Or ‘Sub Sampling’ operation.
- The pooling layers reduces spatial dimensions by keeping only the important features

Max Pooling Operation:

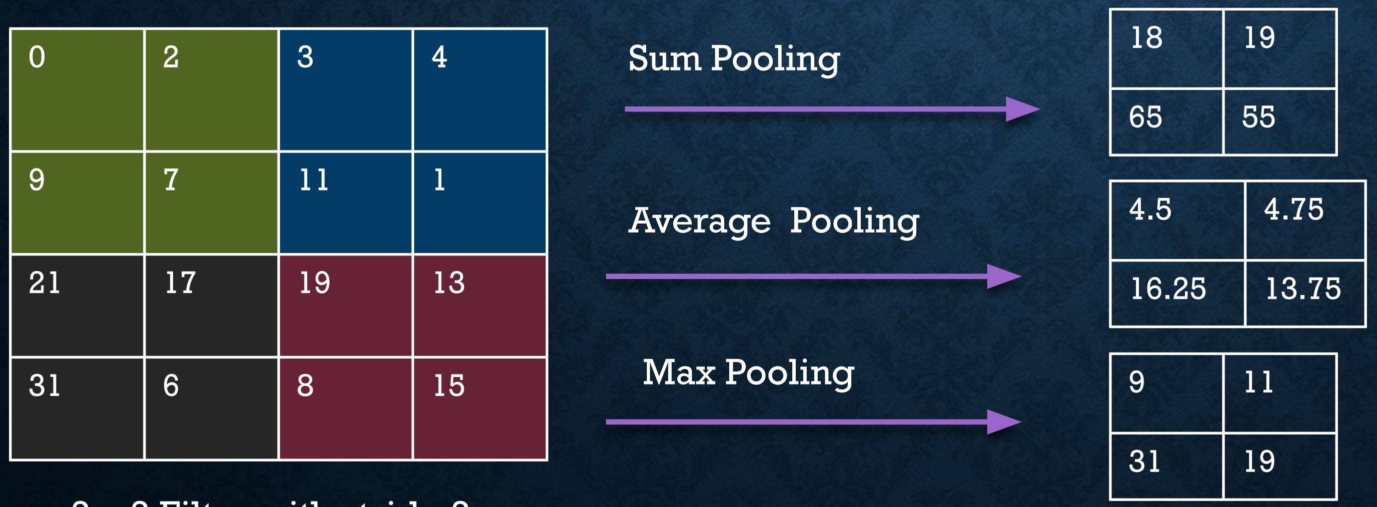
While slide over the filter on input matrix , simply take the maximum value from the filter window

Average Pooling Operation:

While slide over the filter on input matrix, take average value of the input matrix within the filter window

Sum Pooling Operation:

While slide over the filter on input matrix, sum all the values of the input matrix within the filter window



0	2	3	4
9	7	11	1
21	17	19	13
31	6	8	15

Sum Pooling

18	19
65	55

Average Pooling

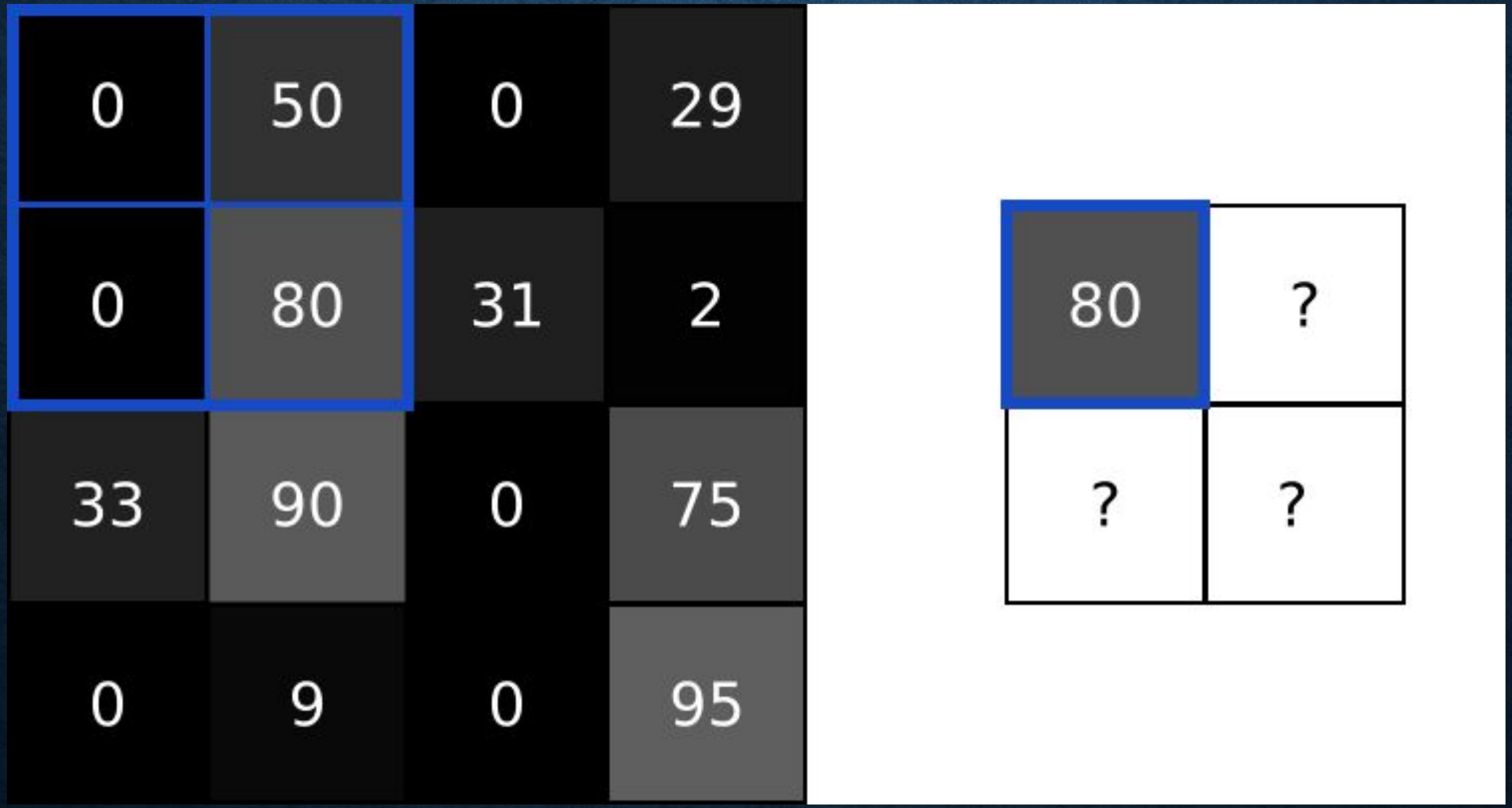
4.5	4.75
16.25	13.75

Max Pooling

9	11
31	19

2 x 2 Filter with stride 2

Which type of pooling is this??

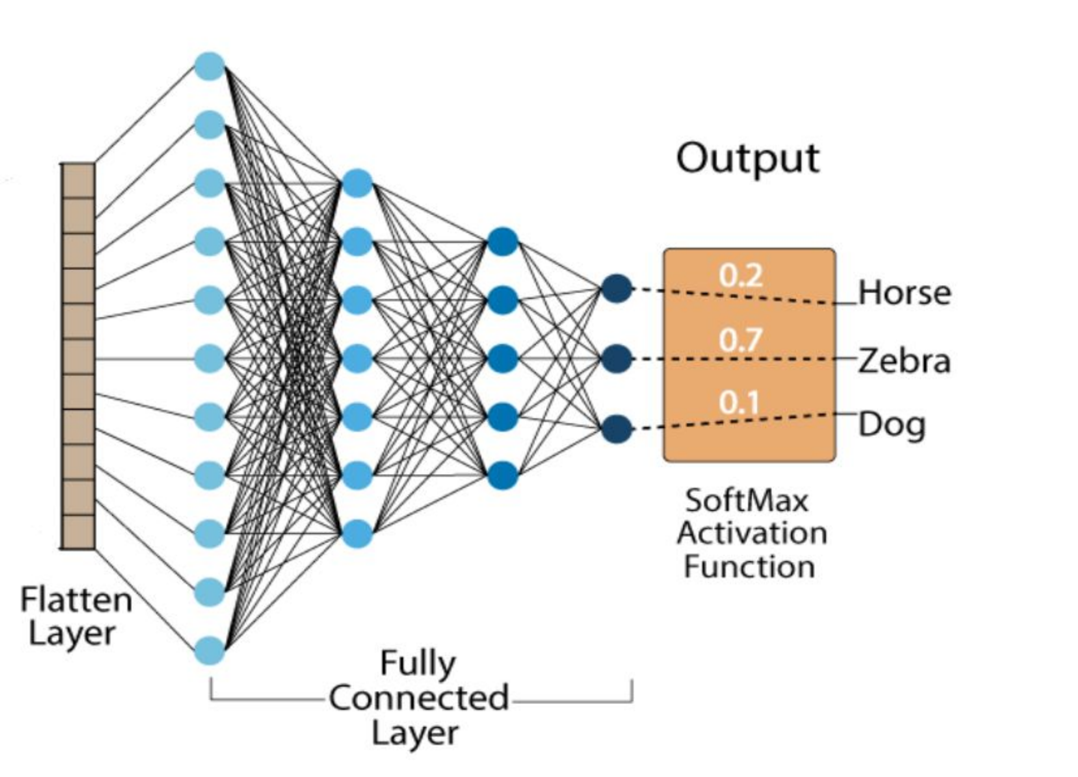


0	50	0	29
0	80	31	2
33	90	0	75
0	9	0	95

80	?
?	?

Fully Connected layers

- Given any image, convolutional layers extract features from the image and produce a feature map.
- Flattening this feature map , produces vector and feed it to the feed forward network
- Feed forward network takes the flattened features and applies an activation function and returns the output
- This output stating whether the image contains which category of features or not .


Output
0.2
Horse
0.7
Zebra
0.1
Dog
SoftMax Activation Function
Flatten Layer
Fully Connected Layer