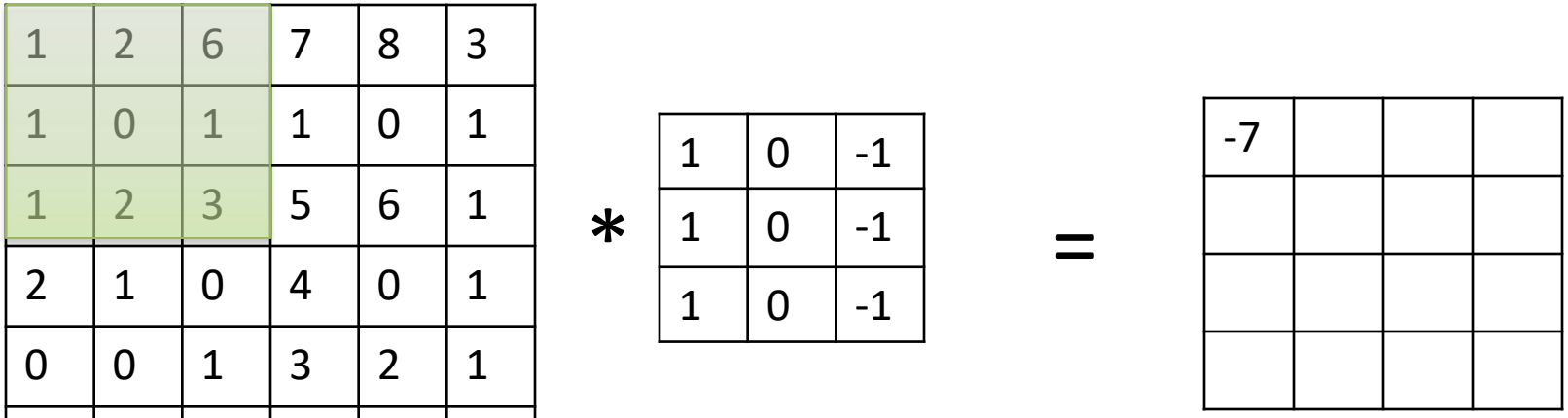
Convolution Operation

MUKESH KUMAR



1*1+2*0+6*-1 + 1*1+0*0+1*-1 + 1*1 +2*0+3*-1

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

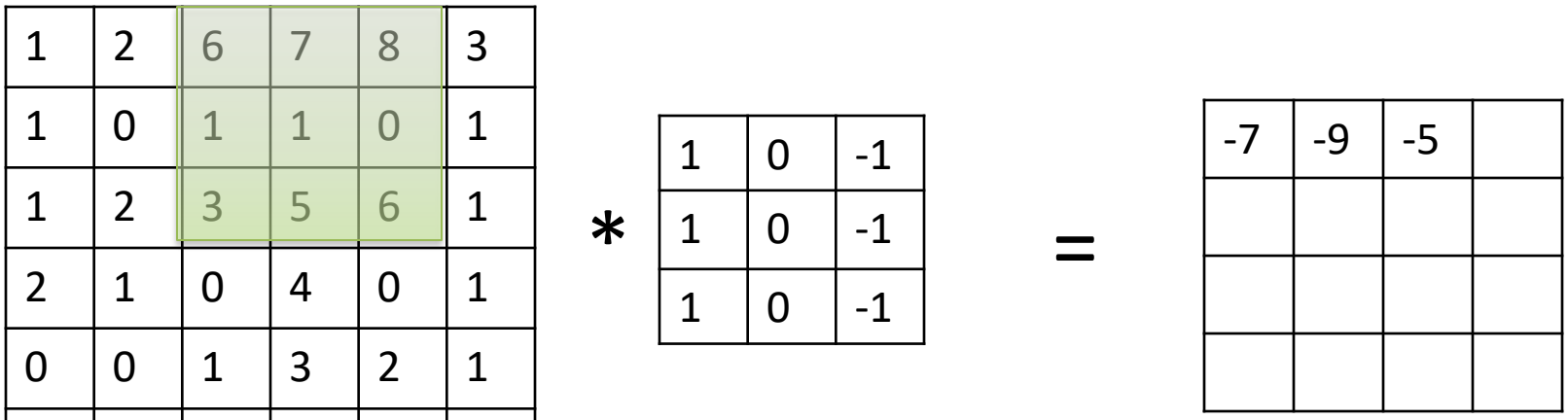
*

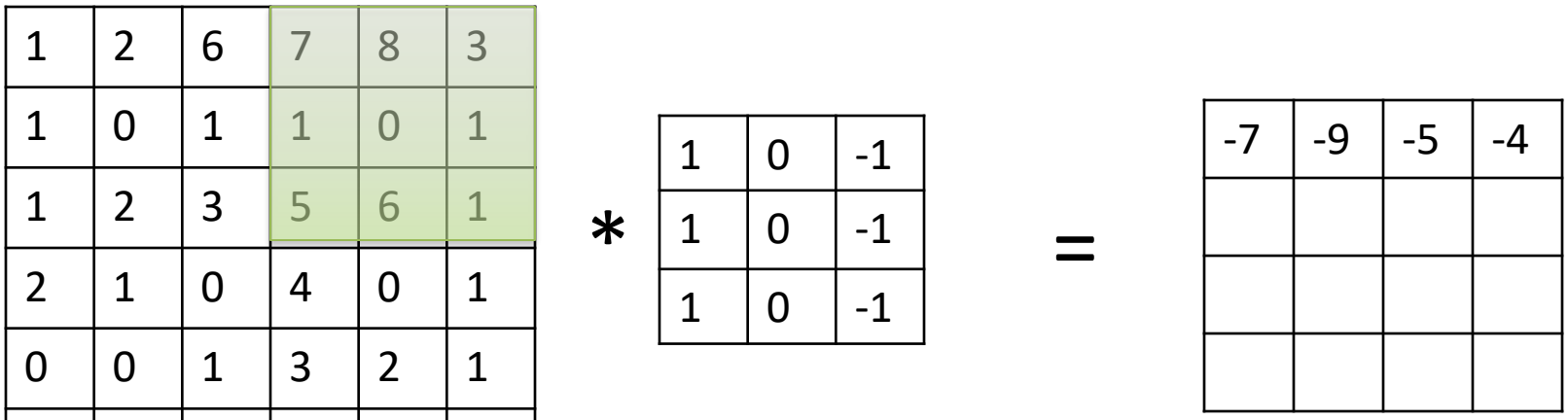
1	0	-1
1	0	-1
1	0	-1

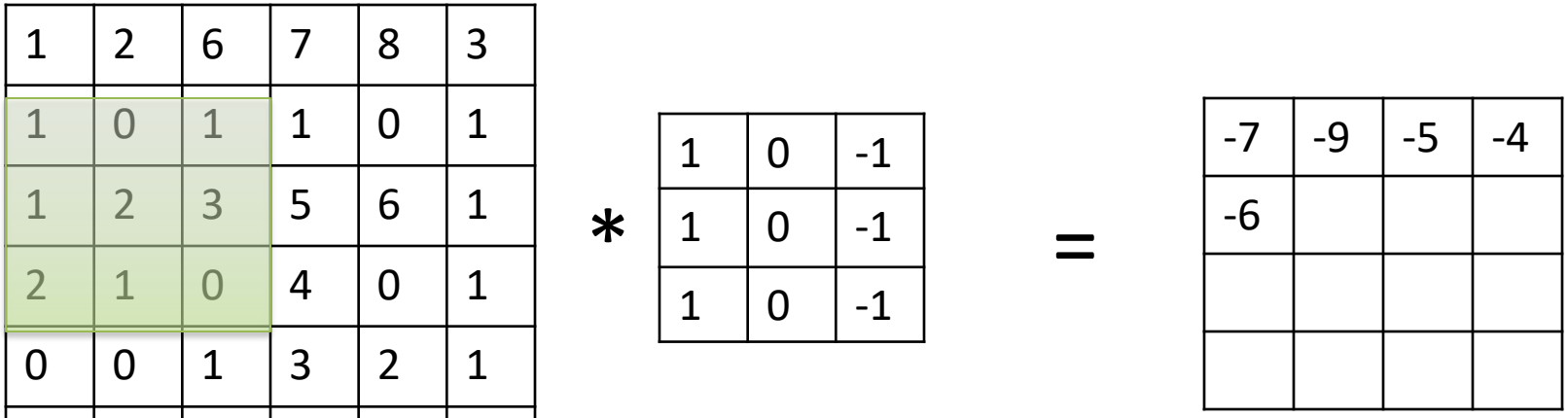
=

-7	-9		

2*1 +6*0+7*-1 +0*1+1*0+1*-1+2*1+3*0+5*-1 =


1 2 6 7 8 3
1 0 1 1 0 1
1 2 3 5 6 1
2 1 0 4 0 1
0 0 1 3 2 1
1 2 2 2 1 1
*
1 0 -1
1 0 -1
1 0 -1
=
-7 -9 -5


1 2 6 7 8 3
1 0 1 1 0 1
1 2 3 5 6 1
2 1 0 4 0 1
0 0 1 3 2 1
1 2 2 2 1 1
*
1 0 -1
1 0 -1
1 0 -1
=
-7 -9 -5 -4


1 2 6 7 8 3
1 0 1 1 0 1
1 2 3 5 6 1
2 1 0 4 0 1
0 0 1 3 2 1
1 2 2 2 1 1
*
1 0 -1
1 0 -1
1 0 -1
=
-7 -9 -5 -4
-6

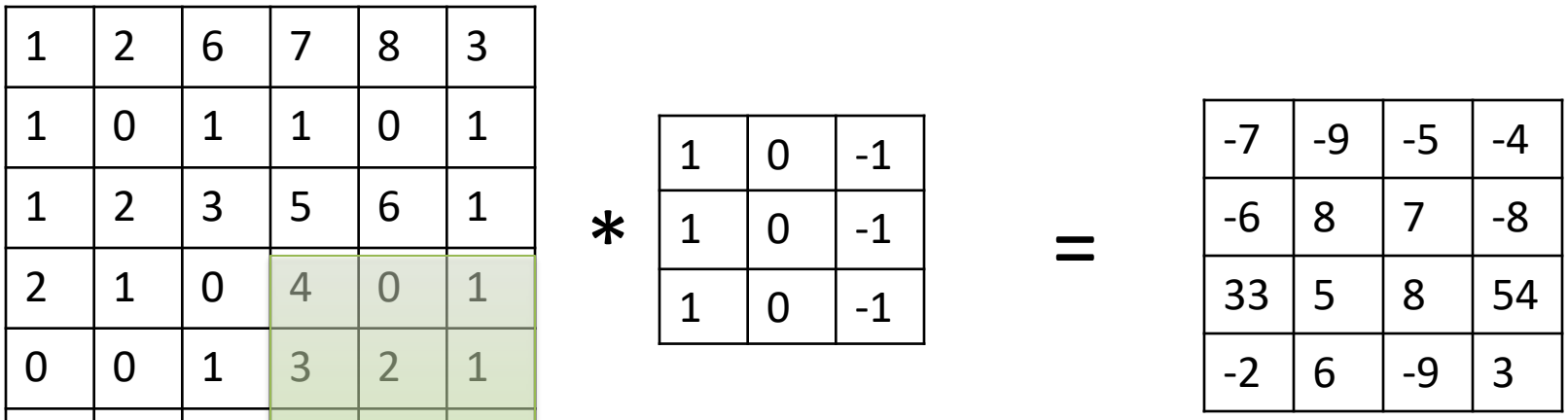
1	2	6	7	8	3
1	0	1	1	0	1
1	2	3	5	6	1
2	1	0	4	0	1
0	0	1	3	2	1
1	2	2	2	1	1

*

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

=

-7	-9	-5	-4
-6	8		

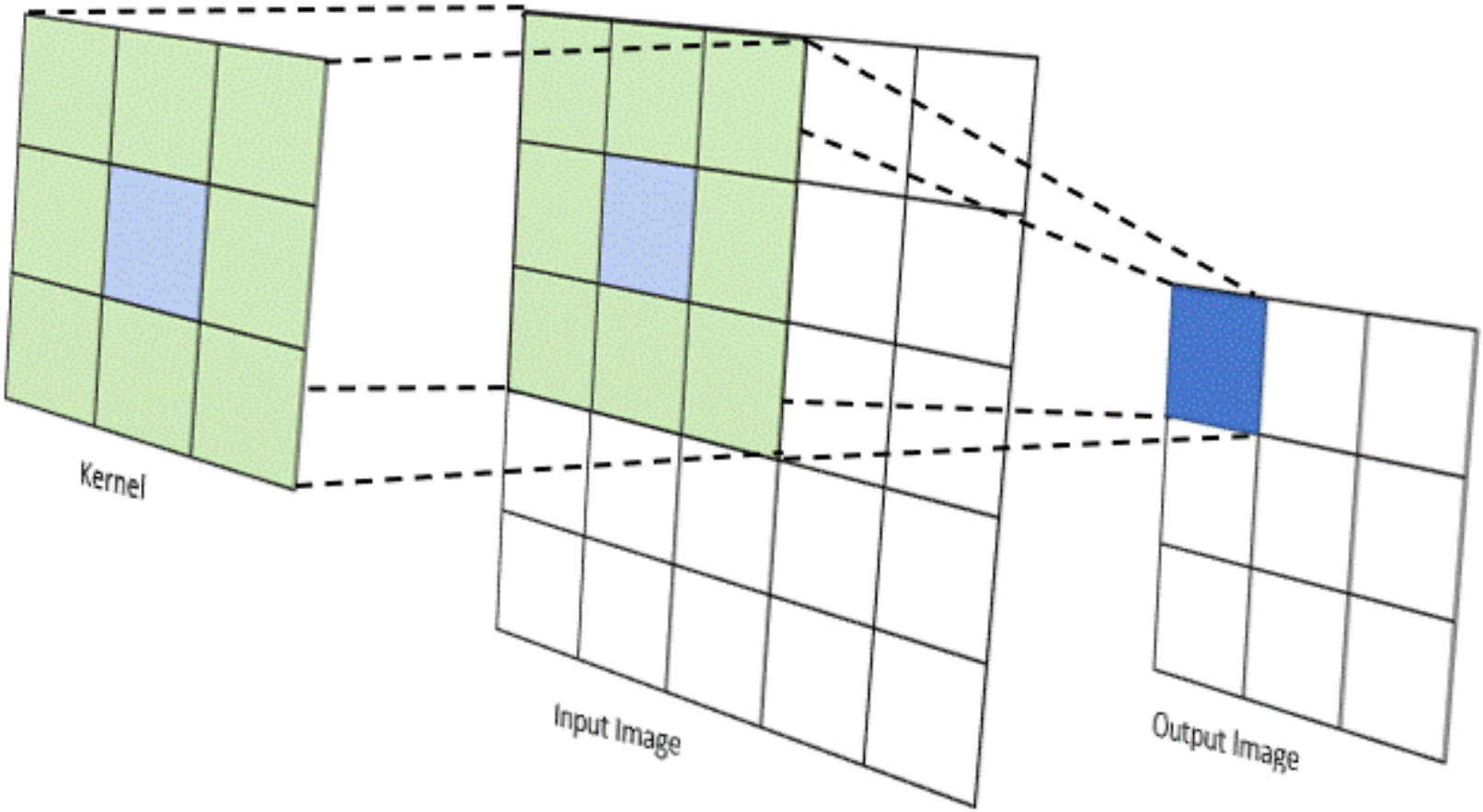


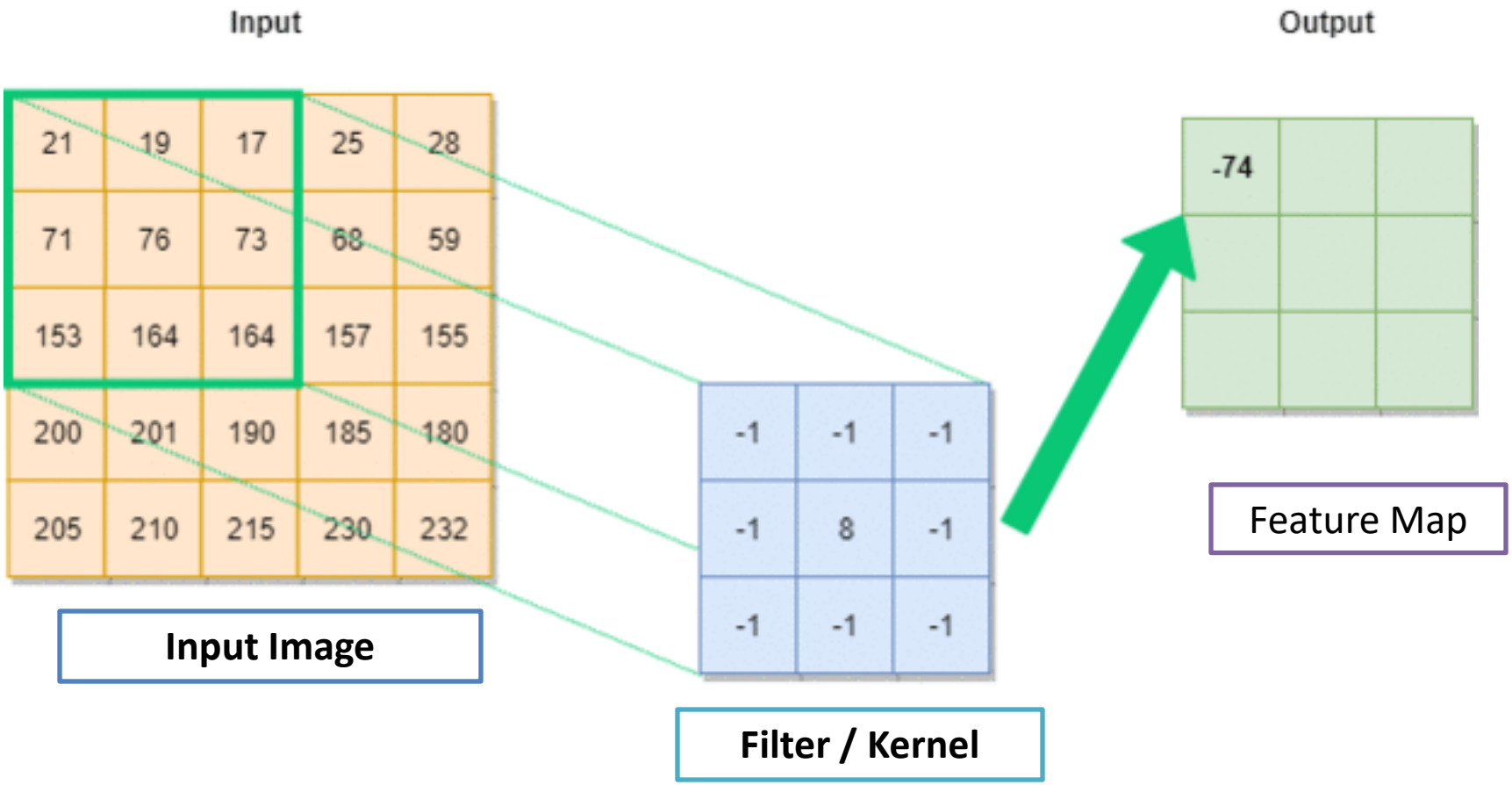
Output = N-f+1

Where : N = input dimension

f= Filter dimension

Convolution




Input
21 19 17 25 28
71 76 73 68 59
153 164 164 157 155
200 201 190 185 180
205 210 215 230 232
Input Image
-1 -1 -1
-1 8 -1
-1 -1 -1
Filter / Kernel
Output
-74
Feature Map

Convolution Example

- https://deeplizard.com/resource/pavq7noze2

Edge Detector Example

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
255	255	255	255	255	255
255	255	255	255	255	255
255	255	255	255	255	255

*

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

=

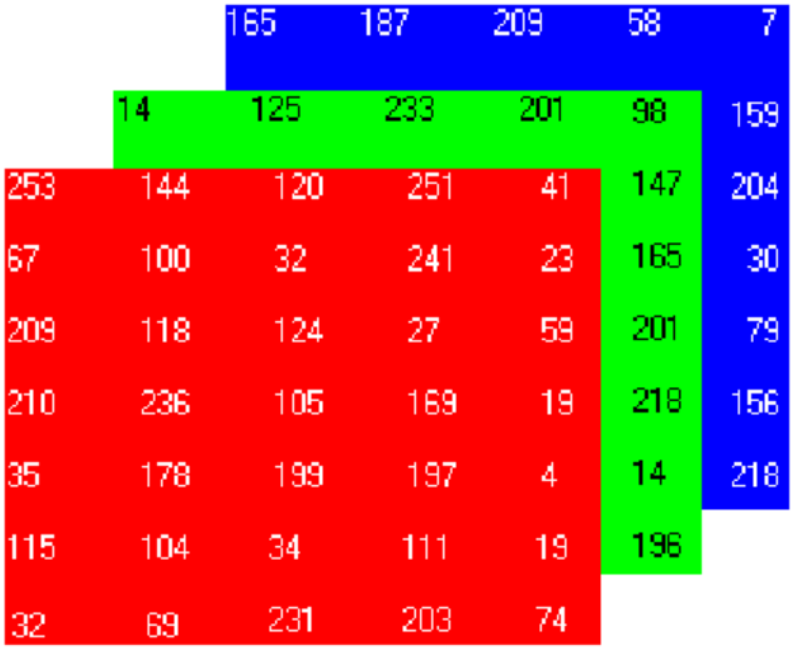
Color Images

- To store an image on a computer, the image is first broken down into tiny elements called **PIXELS.** The **smallest** element in a picture or image is called Pixel(in short of **Pic**tu

- If your image resolution is 1020 x 800(width x height), the total number of pixels is 816,000. re **El**ement = **Pixel**).

- A colored image is composed of multiple colors and all colors can be generated from three (red, green and blue) colors.
-

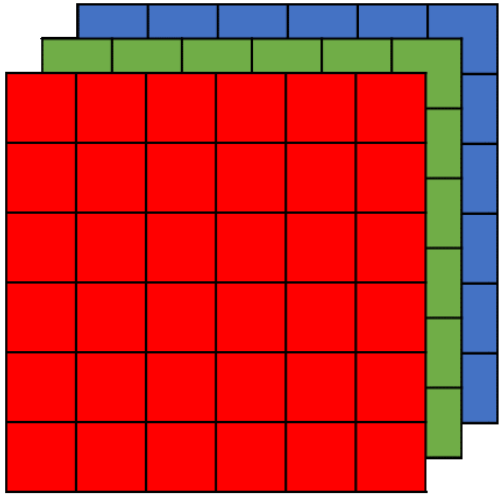
N X M X3



- where N is the number of pixels across the height, M is the number of pixels across the width, and 3 represents the number of channels

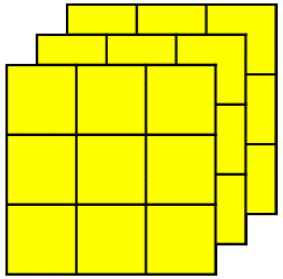
- each colored image is a unique composition of these three colors or 3 channels – Red, Green, and Blue.

CONVOLUTION



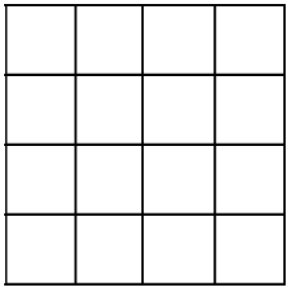
$6 \times 6 \times 3$

$*$

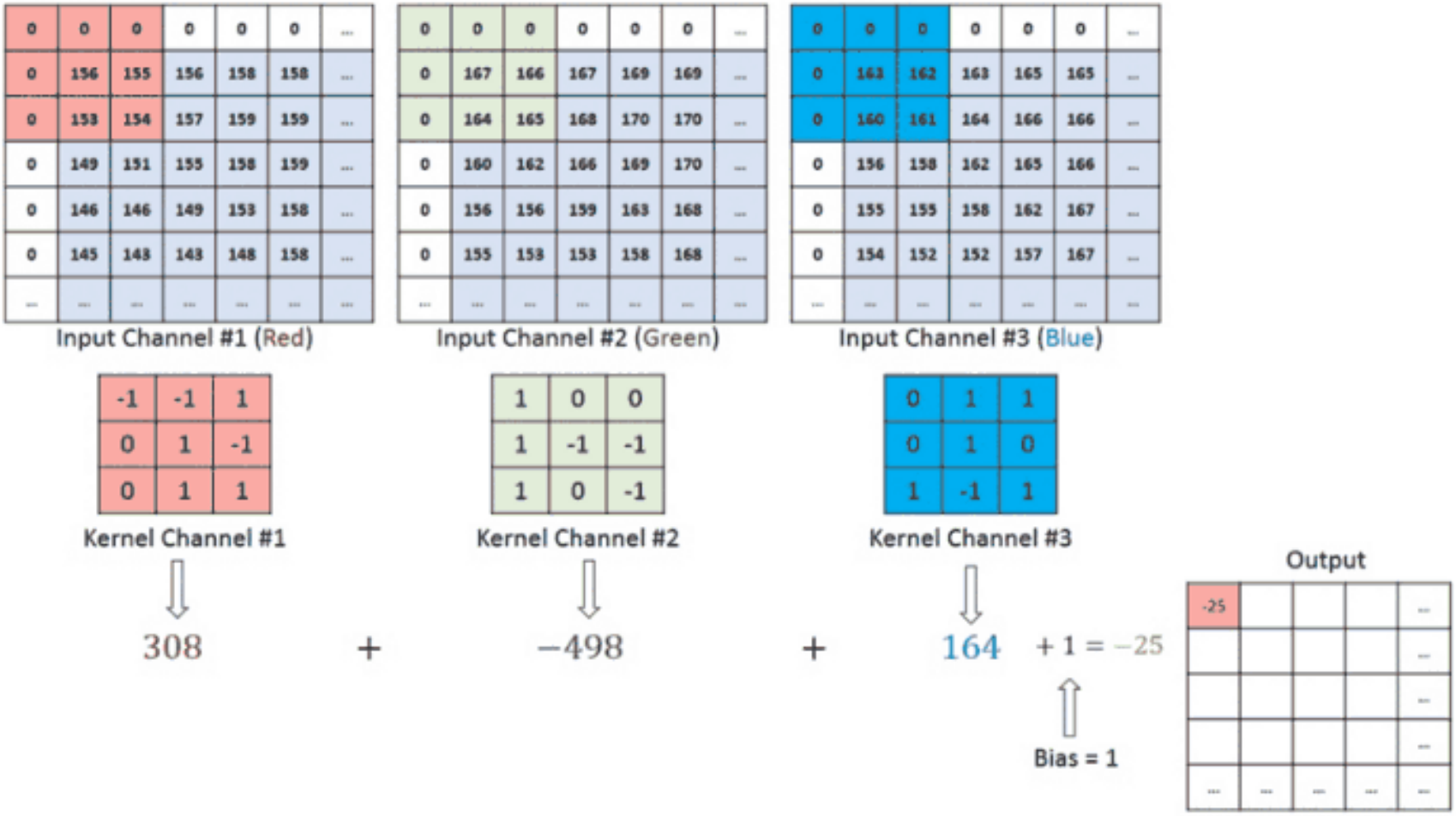


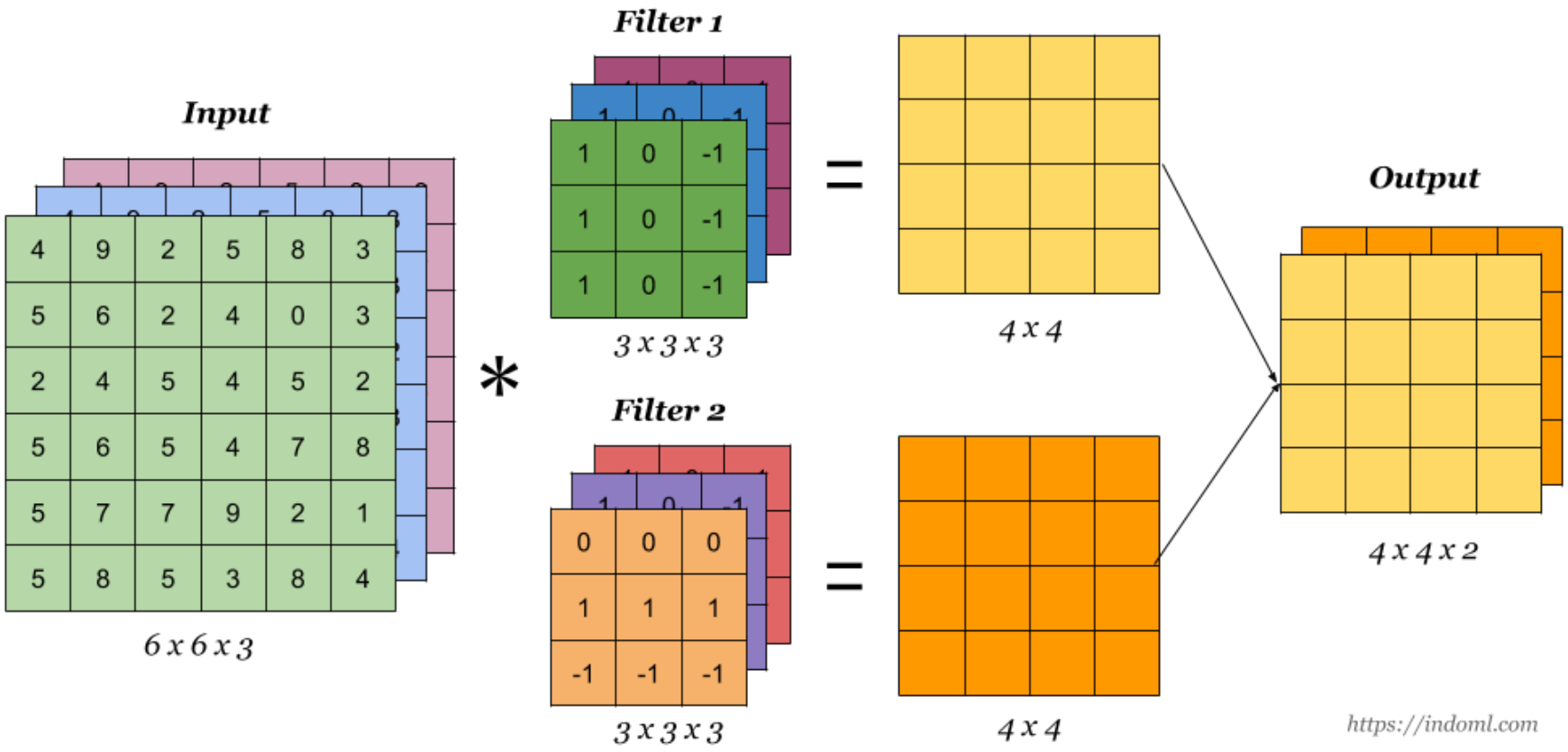
$3 \times 3 \times 3$

$=$



4×4


0 0 0 0 0 0 ...
0 156 155 156 158 158 ...
0 153 154 157 159 159 ...
0 149 151 155 158 159 ...
0 146 146 149 153 158 ...
0 145 148 143 148 158 ...
... ...
Input Channel #1 (Red)
0 0 0 0 0 0 ...
0 167 166 167 169 169 ...
0 164 165 168 170 170 ...
0 160 162 166 169 170 ...
0 156 156 159 163 168 ...
0 155 153 153 158 168 ...
... ...
Input Channel #2 (Green)
0 0 0 0 0 0 ...
0 163 162 163 165 165 ...
0 160 161 164 166 166 ...
0 156 158 162 165 166 ...
0 155 155 158 162 167 ...
0 154 152 152 157 167 ...
... ...
Input Channel #3 (Blue)
-1 -1 1
0 1 -1
0 1 1
Kernel Channel #1
1 0 0
1 -1 -1
1 0 -1
Kernel Channel #2
0 1 1
0 1 0
1 -1 1
Kernel Channel #3
308
+
−498
+
164
+ 1 = −25
Bias = 1
Output
-25 ... ...
... ...
... ...
... ...
... ...


Input
4 9 2 5 8 3
5 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
6 x 6 x 3
*
Filter 1
1 0 -1
1 0 -1
1 0 -1
3 x 3 x 3
=
4 x 4
Filter 2
0 0 0
1 1 1
-1 -1 -1
3 x 3 x 3
=
4 x 4
Output
4 x 4 x 2
https://indoml.com