COMPUTATIONAL ASTROPHYSICS

Observatorio
Astronómico
Nacional

Computational Astrophysics

14. Convolution

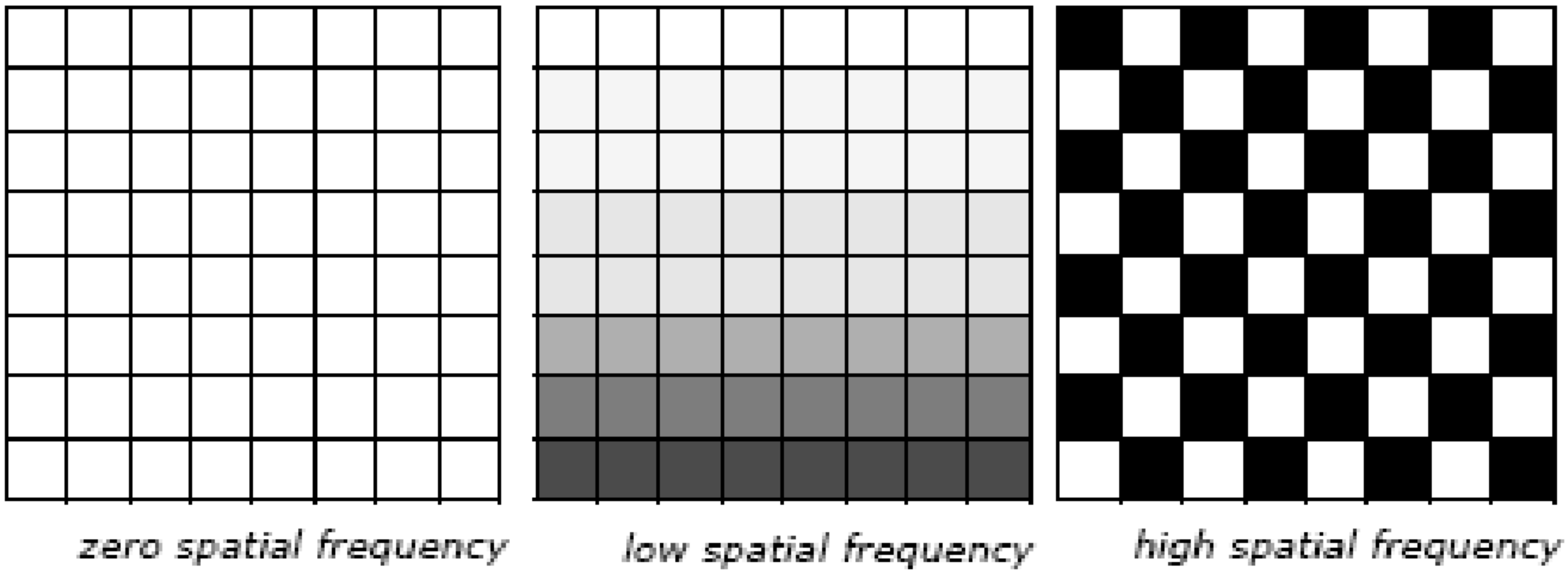
Eduard Larrañaga
Observatorio Astronómico Nacional
Universidad Nacional de Colombia

What is convolution?

What is convolution?

- Convolution is a general purpose filter effect for images.
- It is implemented as a matrix applied to an image and a mathematical operation comprised of integers.
- It works by determining the value of a central pixel by adding the weighted values of all its neighbors together.
- The output is a new modified (filtered) image.

Spatial Frequencies



Convolution filtering modifies the spatial frequency of an image

Why convolution?

- Smooth
- Sharpen
- Intensify
- Enhance

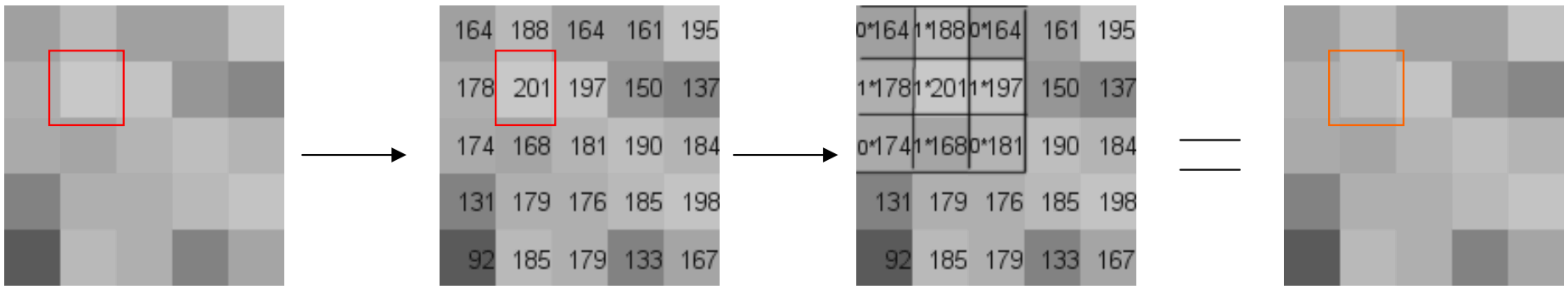
The Process of Image Convolution

Kernel: is a (usually) small matrix of numbers that is used in image convolution.

The size of a kernel is arbitrary but 3x3 is often used.

Differently sized kernels containing different patterns of numbers produce different results under convolution.

0	1	0
1	1	1
0	1	0



Original image

Image with color values placed over it

Image with 3x3 kernel placed over it

Output image

164	188	164
178	201	197
174	168	181

Color values

×

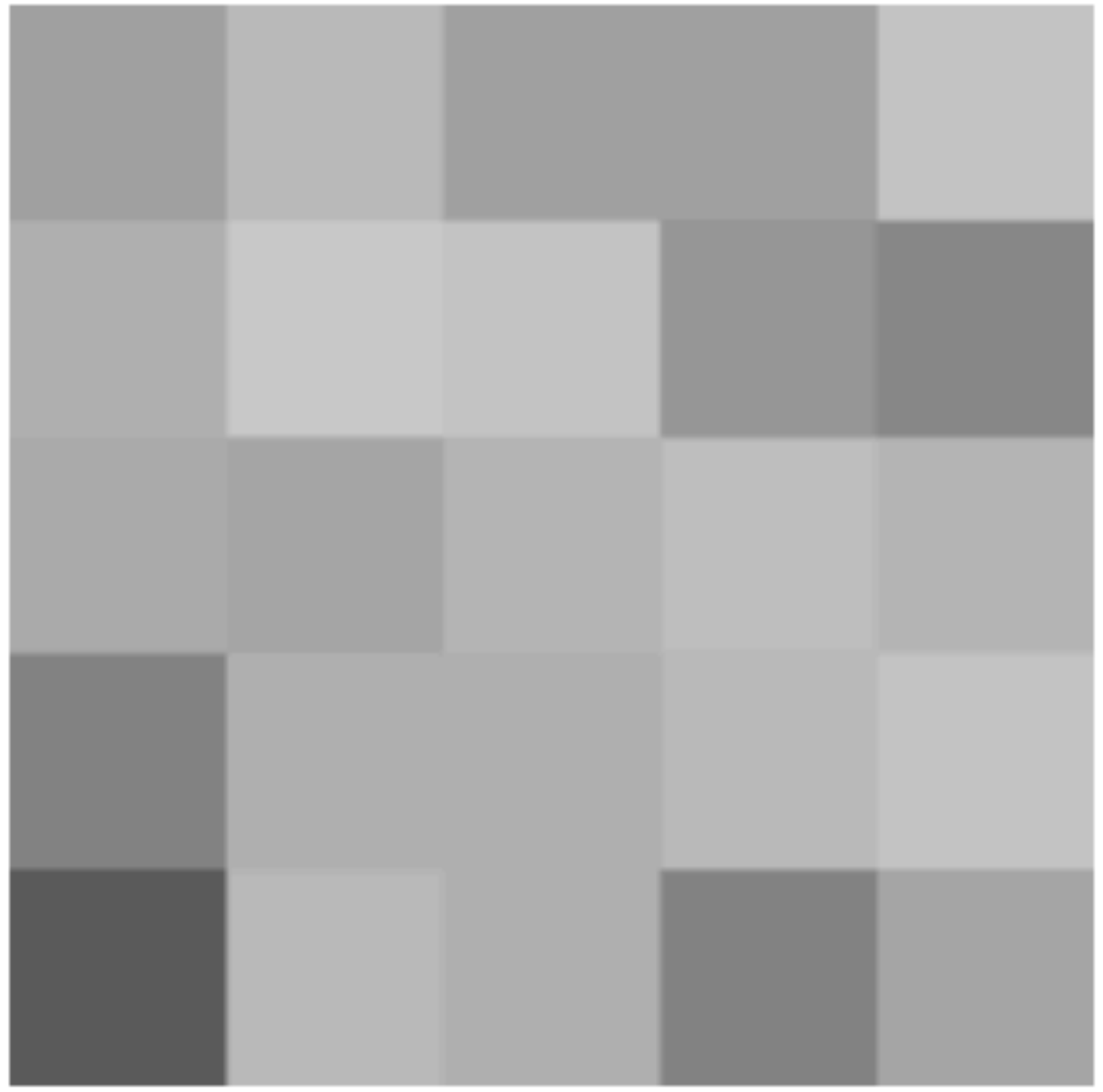
0	1	0
1	1	1
0	1	0

Kernel

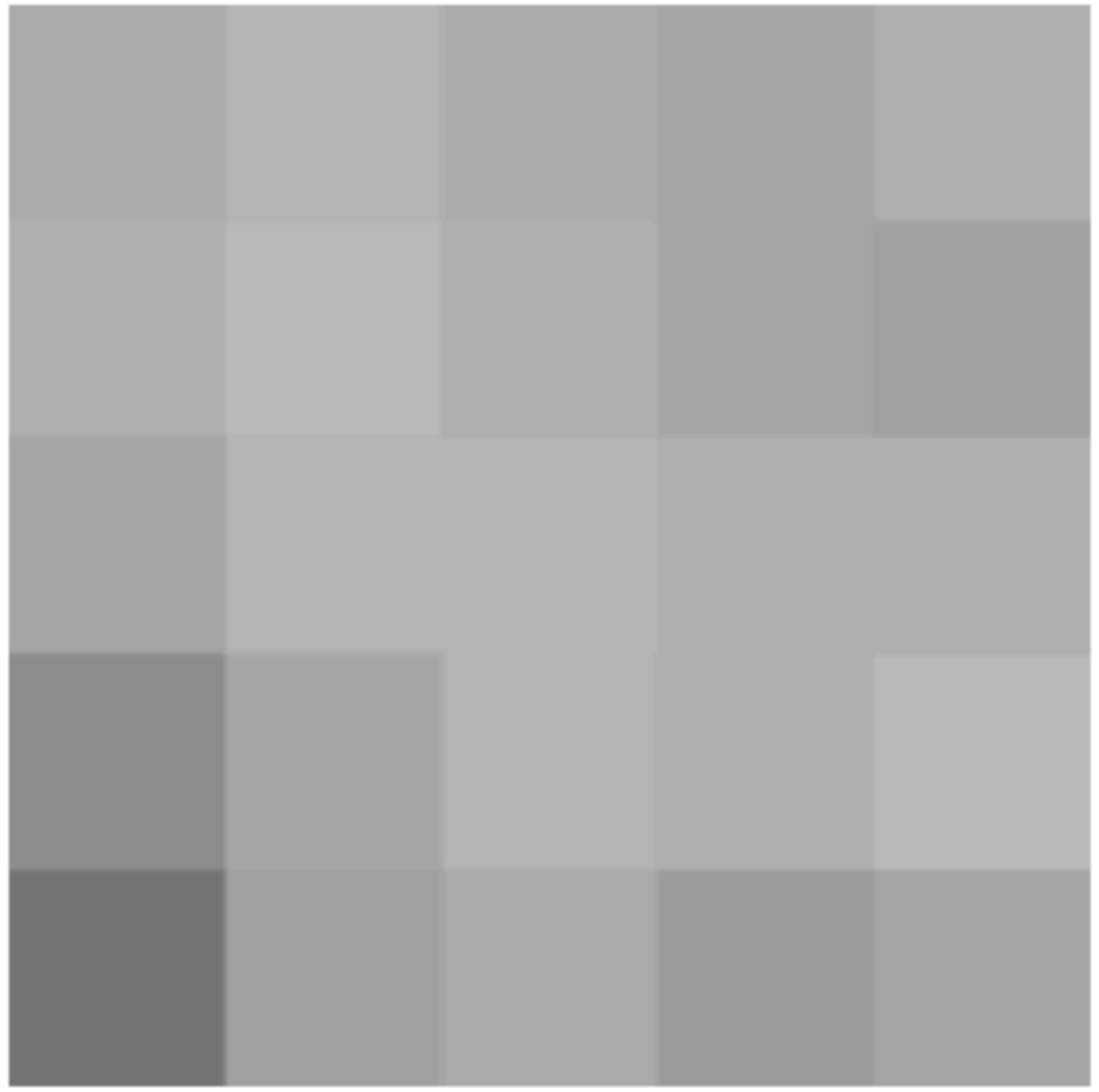
Divided by the sum of the kernel

932\5 = new pixel color

- Original Image



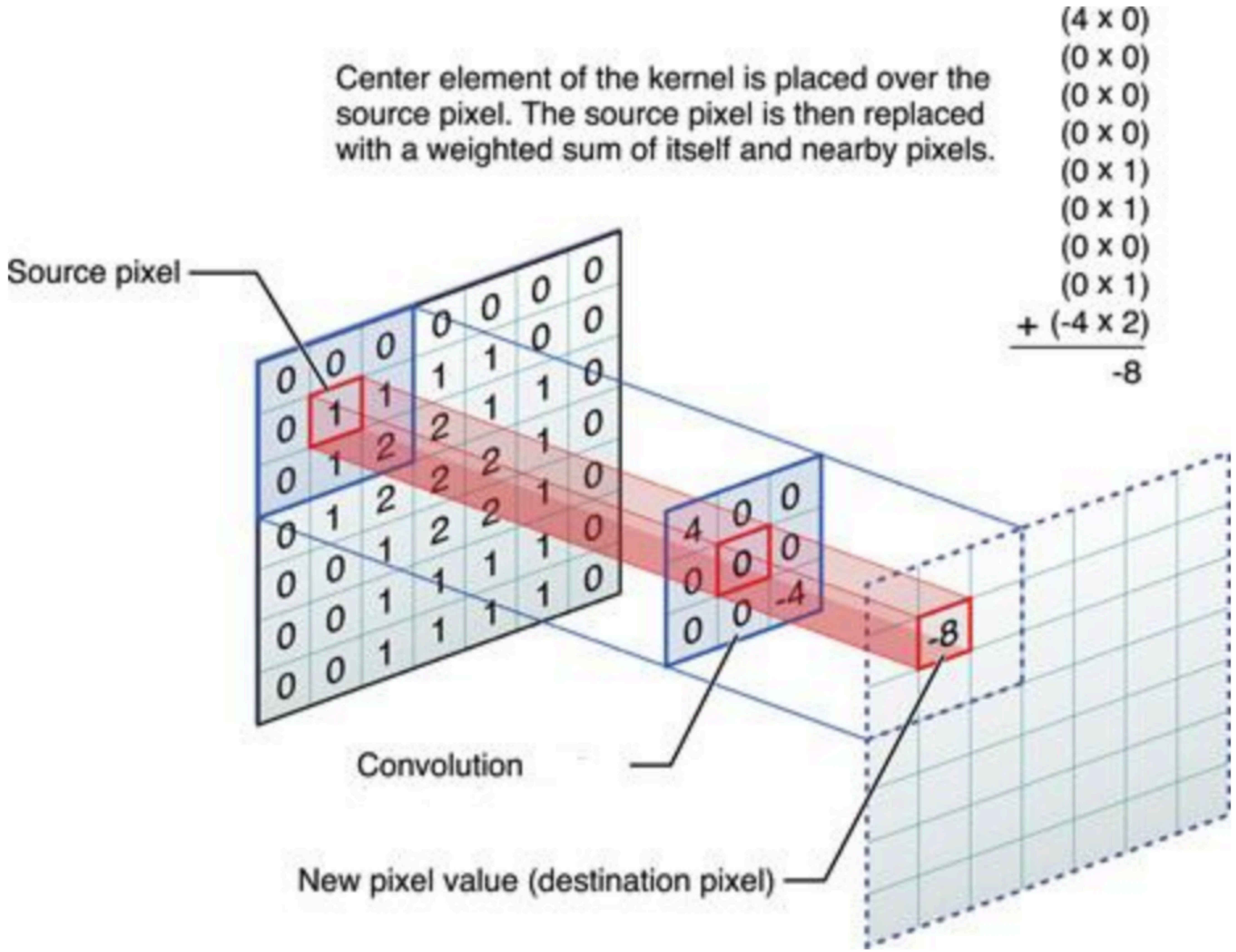
- Smoothed modified image

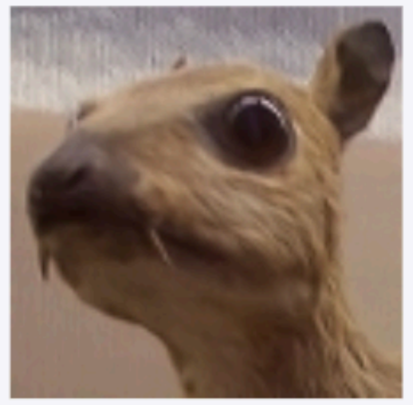
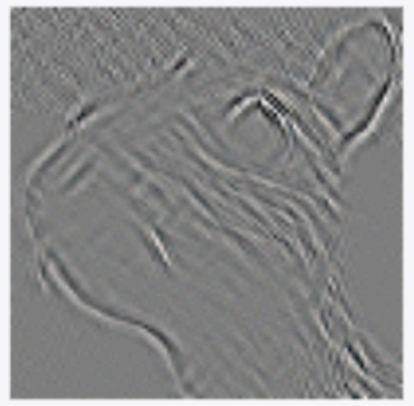
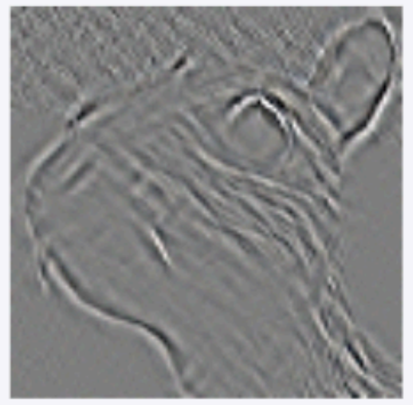
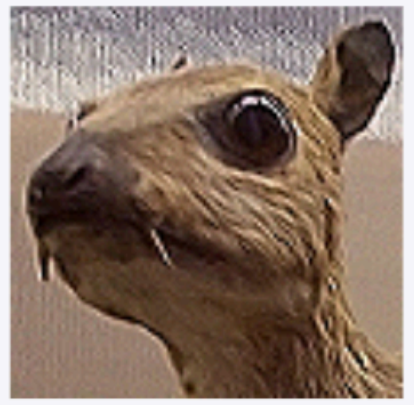


Convolution

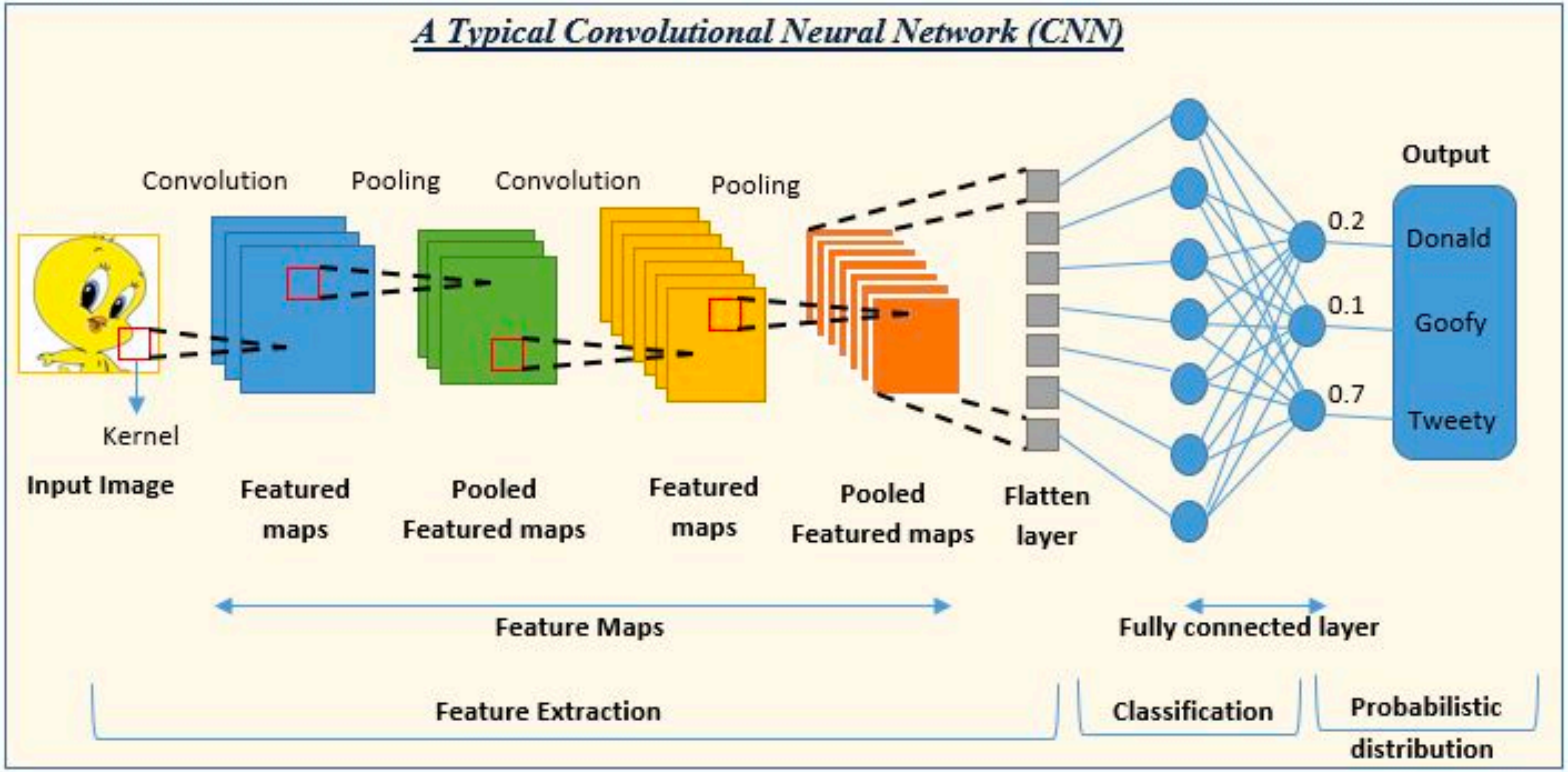
$$g = \frac{\sum_{i=1}^{q} \sum_{j=1}^{q} w_{ij} d_{ij}}{\sum_{i=1}^{q} \sum_{j=1}^{q} w_{ij}}$$

- d_{ij} : Data value of the pixel [i,j]
- w_{ij} : Coefficient (weight) in the Kernel
- q : dimension of the Kernel (usually 3)


Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.
(4 x 0)
(0 x 0)
(0 x 0)
(0 x 0)
(0 x 1)
(0 x 1)
(0 x 0)
(0 x 1)
+ (-4 x 2)
-8
Source pixel
Convolution
New pixel value (destination pixel)

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Ridge detection	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 4 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3 (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

A Convolutional Neural Network



Conv2D

Conv2D

2D convolution layer.

How many filters to apply to the image?

Kernel size

```
tf.keras.layers.Conv2D(
    filters,
    kernel_size,
    strides=(1, 1),
    padding='valid',
    data_format=None,
    dilation_rate=(1, 1),
    groups=1,
    activation=None,
    use_bias=True,
    kernel_initializer='glorot_uniform',
    bias_initializer='zeros',
    kernel_regularizer=None,
    bias_regularizer=None,
    activity_regularizer=None,
    kernel_constraint=None,
    bias_constraint=None,
    **kwargs
)
```

To obtain filtered images with the same size of the original:

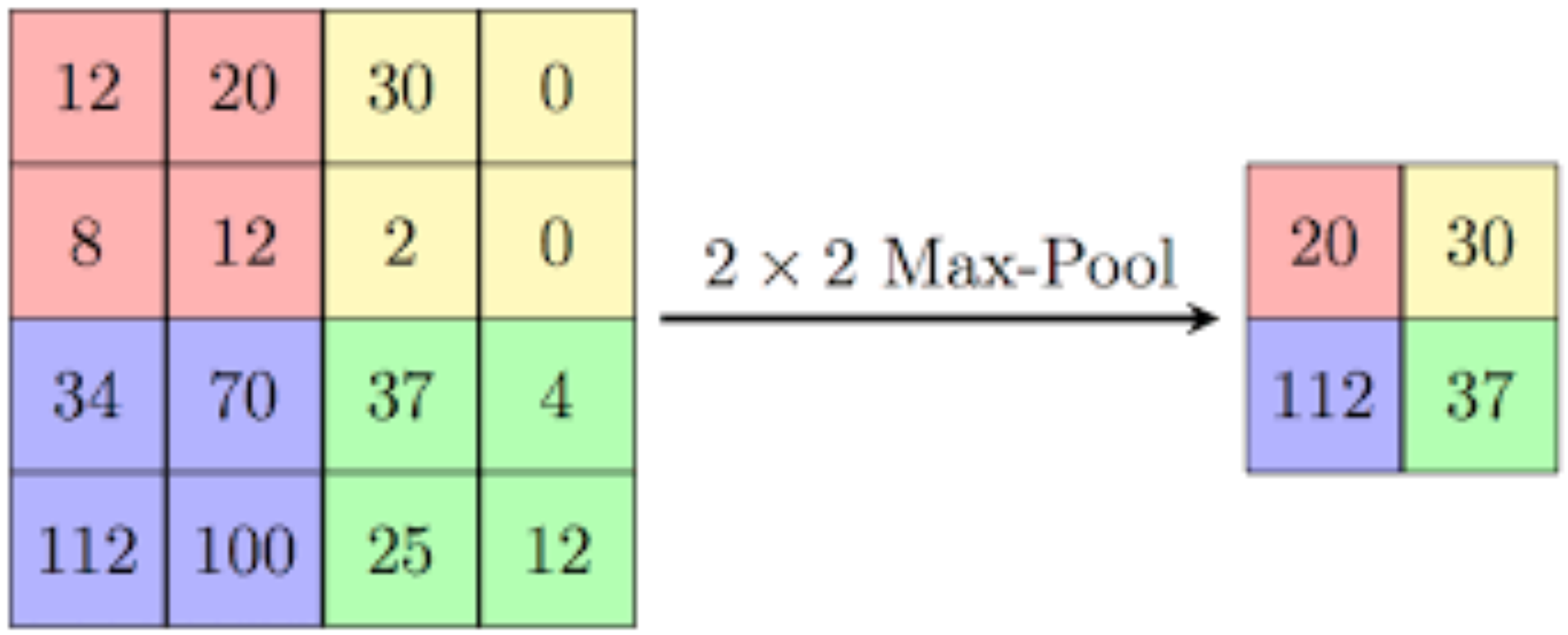
padding = “same”
strides = 1

If the size is changing with the convolution, you must be aware of the size I order to put other hidden layers!

MaxPooling2D

MaxPooling2D

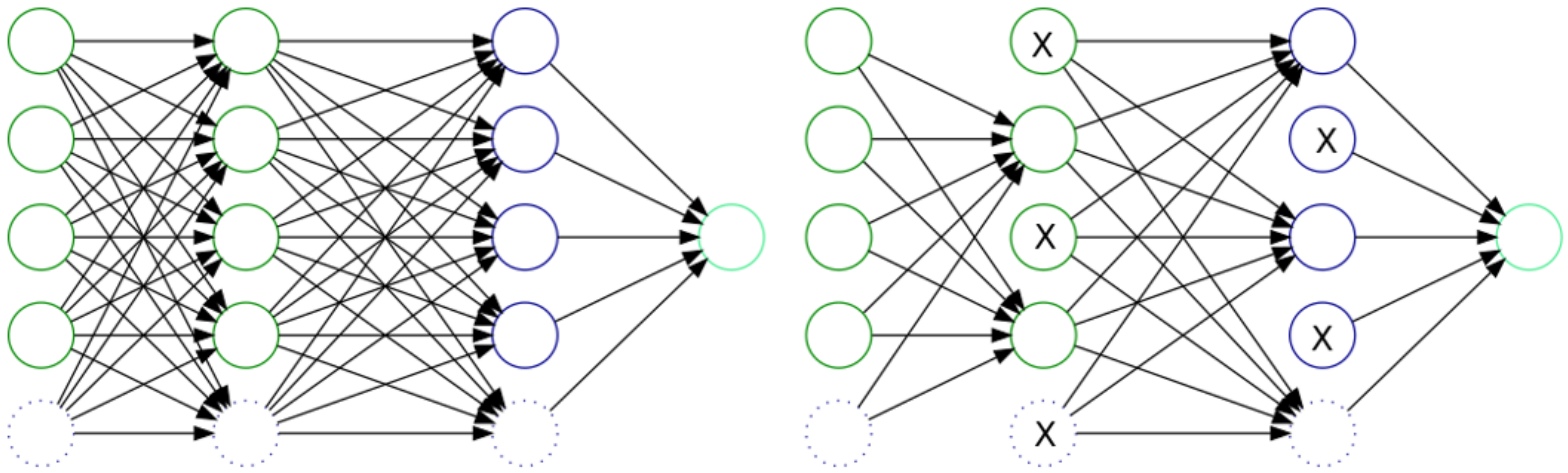
Downsamples the input along its spatial dimensions by taking the maximum value over an input window (poolsize) for each channel of the input.



Dropout

Dropout

The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Inputs not set to 0 are scaled up by 1/(1 - rate) such that the sum over all inputs is unchanged.



Without dropout

With dropout

@astronomiaOAN

AstronomiaOAN

@astronomiaOAN