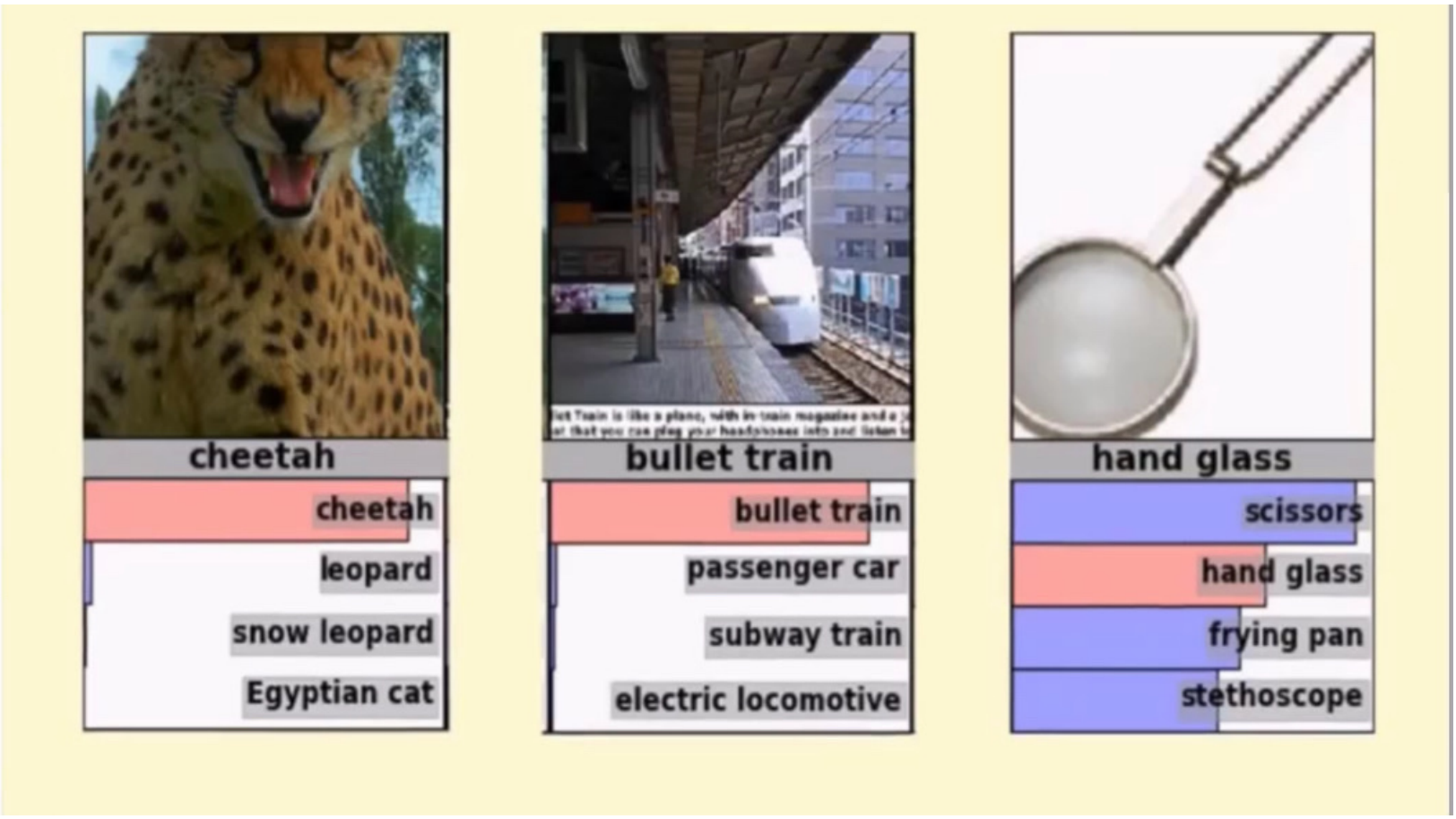
A Quick Introduction to Convolutional Neural Networks (CNNs)

what do you see?

Depending on what "distinctive features" your brain focuses on, you either find a young lady looking away or an older lady looking down.

⇒ Computers can also classify / label objects in an image by detecting their specific features.

Convolutional Neural Networks (CNNs) are a class of deep learning models that learn spatial features from grid-like data (like images) and are widely used for image classification, object detection, image segmentation, etc.



✓ ✓ ✗

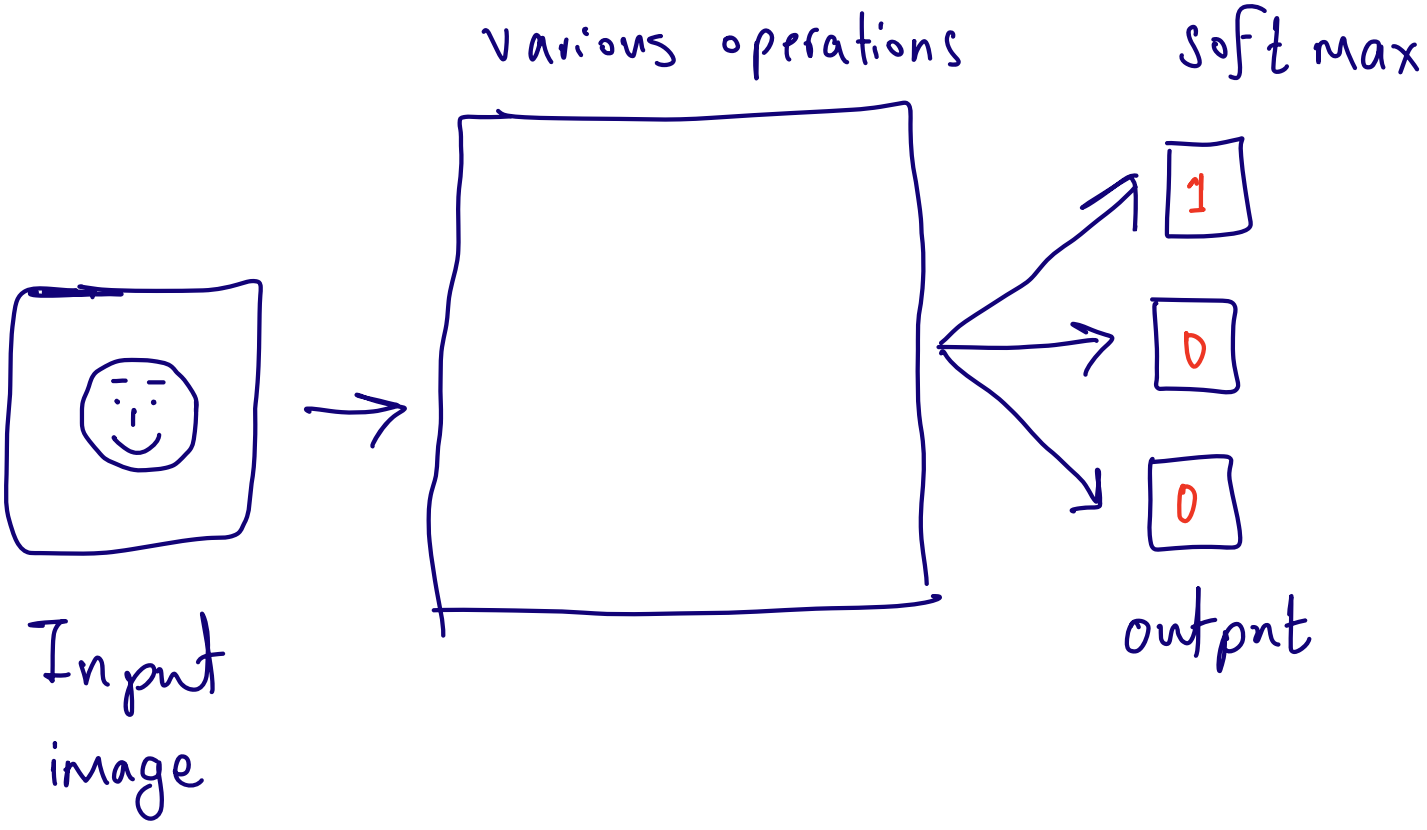
Source: A talk by Geoffrey Hinton

A very simple illustration:

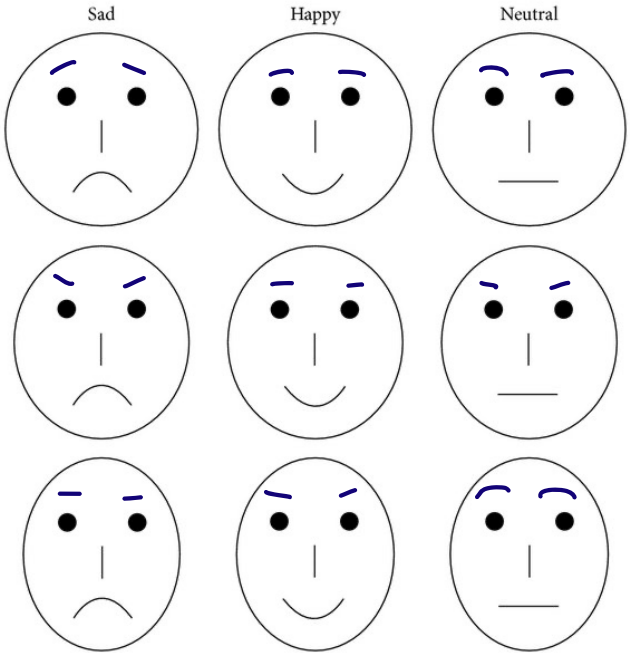
For instance let's assume that we want to predict if a picture shows a smiling face or a neutral face or a sad face:

Image classification :

happy → $\begin{pmatrix}1\\0\\0\end{pmatrix}$ neutral → $\begin{pmatrix}0\\1\\0\end{pmatrix}$ sad → $\begin{pmatrix}0\\0\\1\end{pmatrix}$



we train the CNN by giving it many happy, neutral and sad imges and minimizing the loss function (cross-entropy in this case)



Then, one can make predictions:

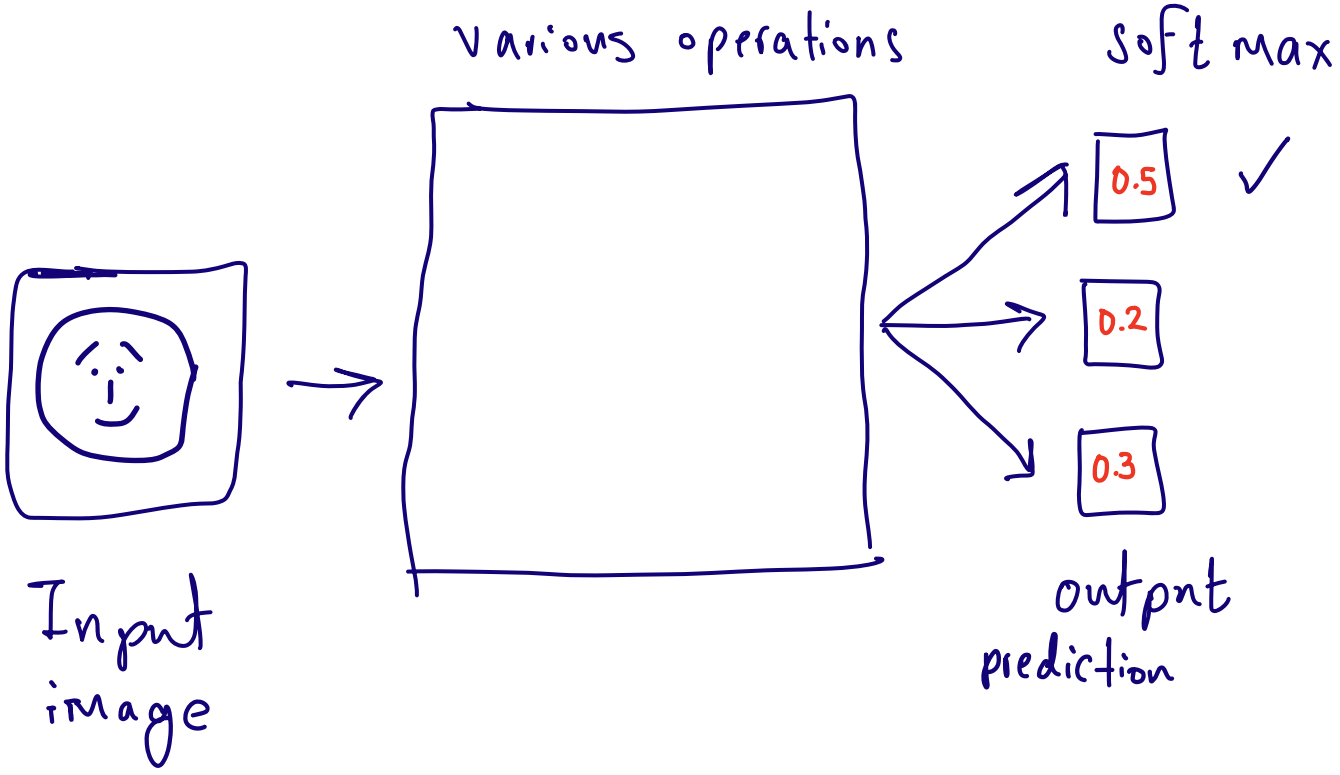
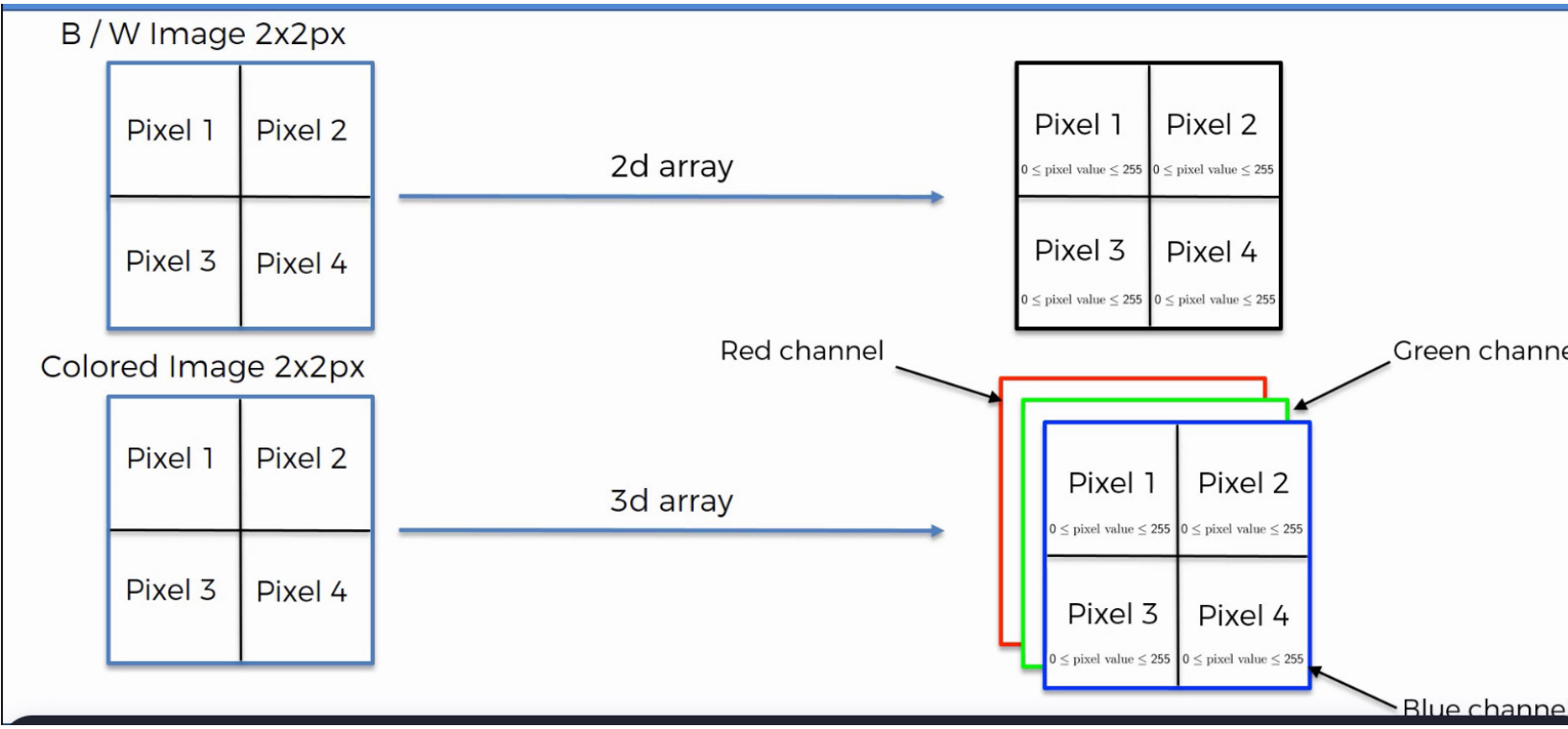


Image Data structure



Source: Super Data Science

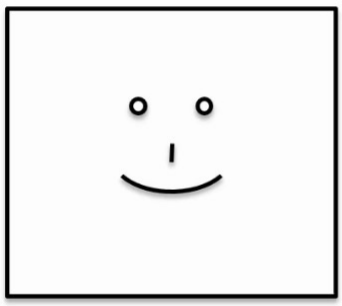
Images are grid-like structures, tiled with pixels.

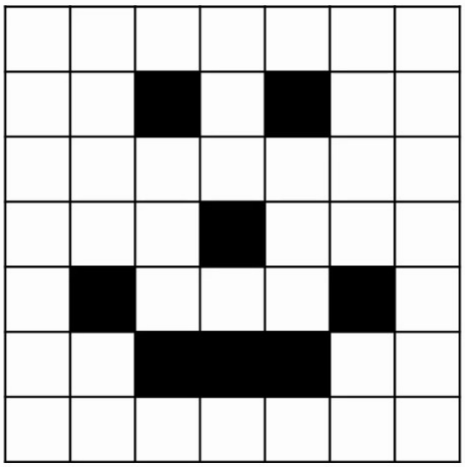
Each pixel is assigned 1 byte = 8 bits ⇒ pixel values range from 0 to 255 (2^8 possible values)

Colored images are a combination of Red, Green and Blue channels → Each pixel has three values: Pixel = (R, G, B)

Pure Red → (255, 0, 0)

How the data is fed into a CNN ?



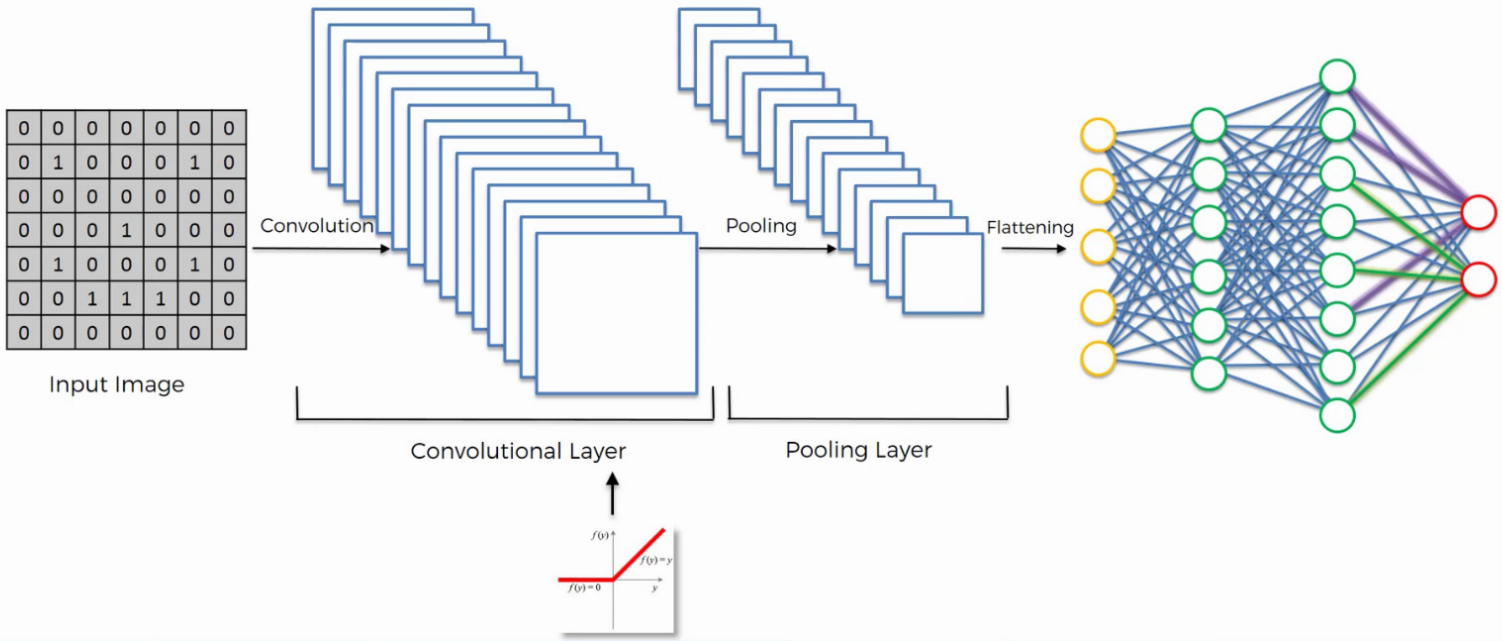


0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Source: Super Data Science

✯ Important: Usually pure black pixel is assigned a 0 and a pure white pixel is assigned 255 which after scaling by $\frac{1}{255}$ it becomes a 1. However, here, just for illustration purposes, we assign a values of 0 and 1 to pure white and pure black respectively.

A Simple CNN in a nutshell



Source: SuperDataScience

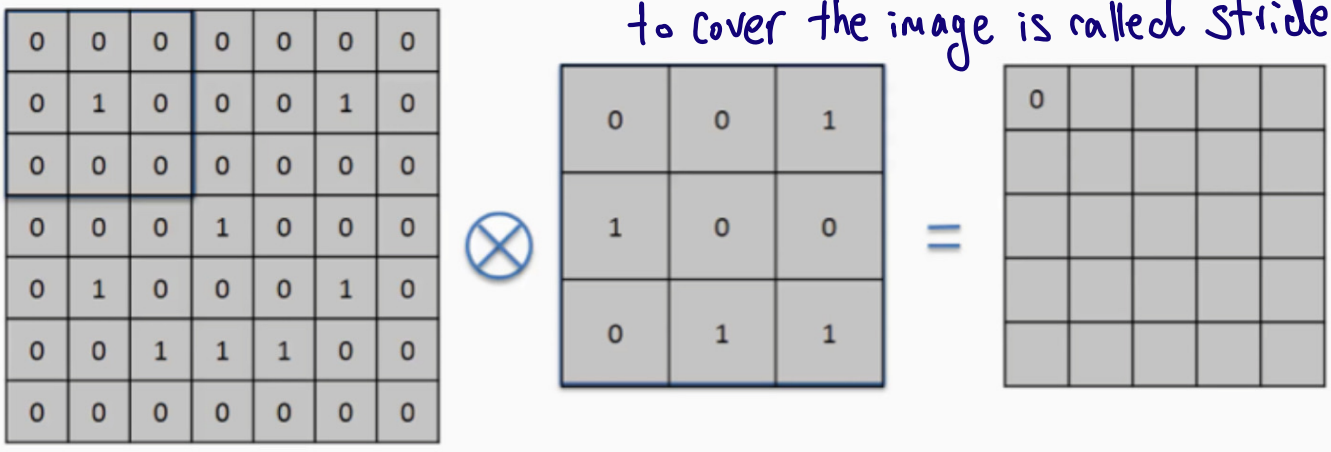
Input image → Convolution layer (ReLU) → Pooling layer

→ Flattening → Fully connected neural network

These can be additional Convolution and pooling layers before flattening

Convolution

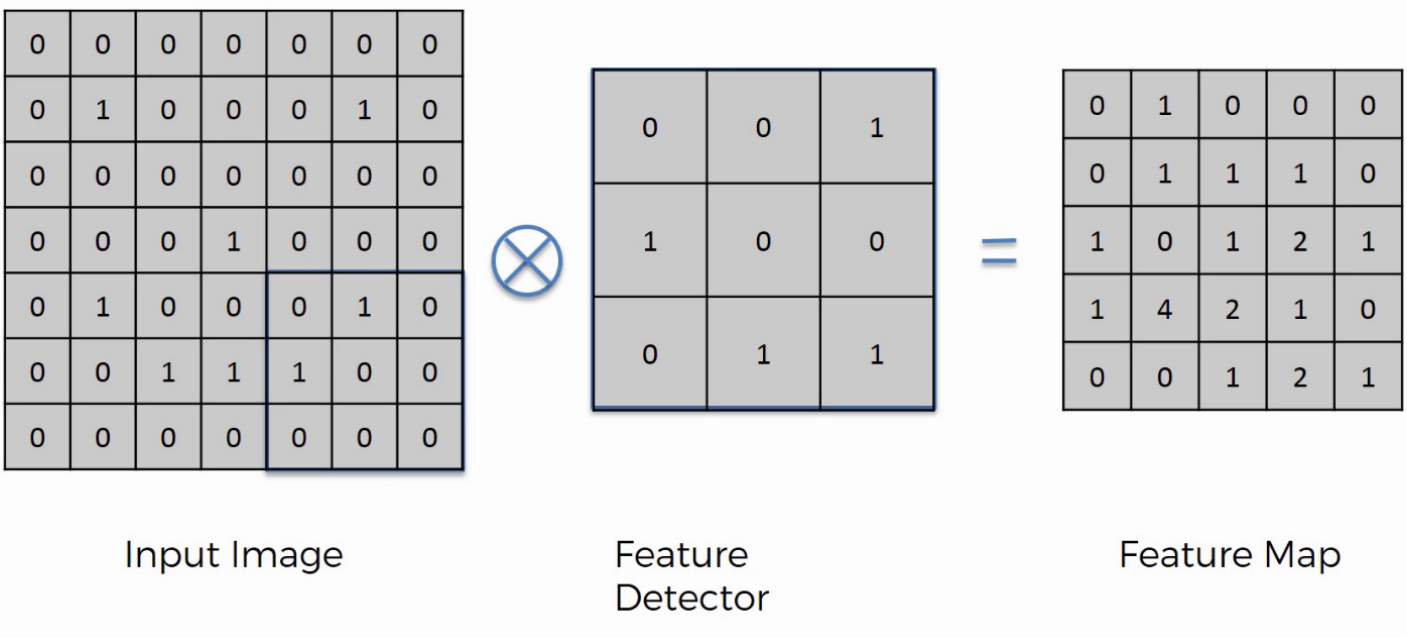
steps that the feature detector/filter takes to cover the image is called stride.



stride = 1 in this example

↳ Also called filter or kernel

Step 1 - Convolution



Source: Super Data Science

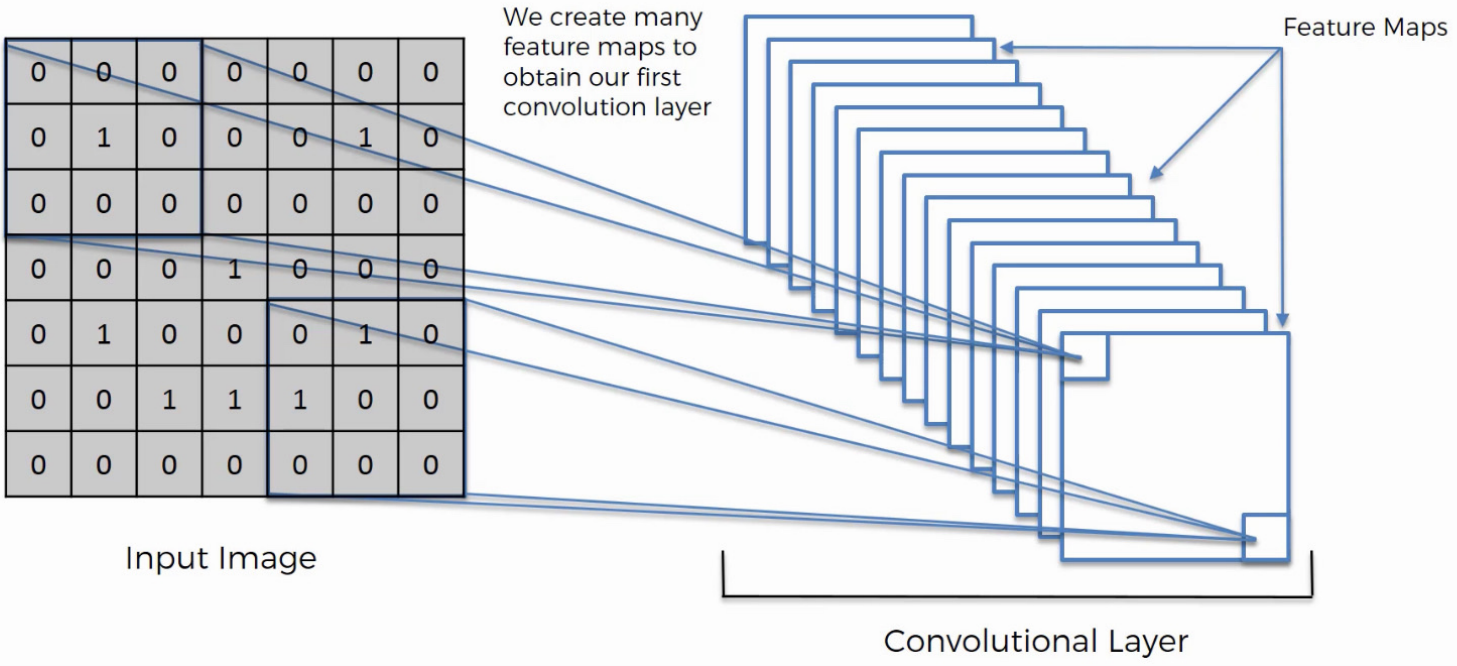
* Convolution reduces the image size and captures the distinctive features of an image.

* The size of the filter / kernel can be adjusted. Also the values of the stride (the steps that the filter takes to cover the entire image) can be modified.

* Filter "weights" (the values of the matrix) are randomy initialized and then their values gets updated after training to capture the most distinctive features of an image.

* In each feature map, the highest values are related to the detection of the specific feature that is presented by the corresponding filter (for example, the number 4 in the feature map shown in the previous page)

Step 1 - Convolution



Source: Super Data Science

☆ By creating the feature maps some information is lost, those are mostly the unimportant ones (the important features are captured)

☆ Feature maps preserve the spatial relation between pixels. In other words, the image doesn't get jumbled up.

Examples of filters applied to images

Sharpen:

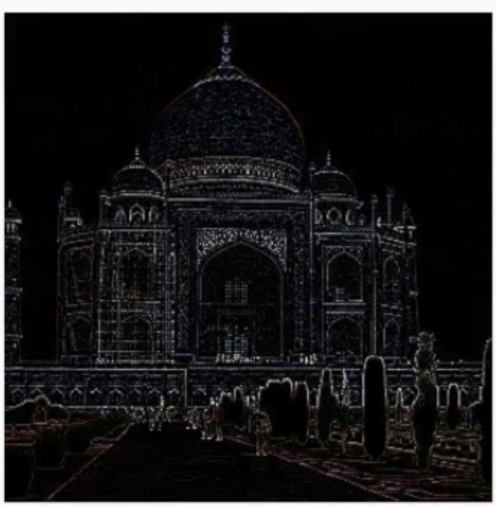
0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0

Blur:

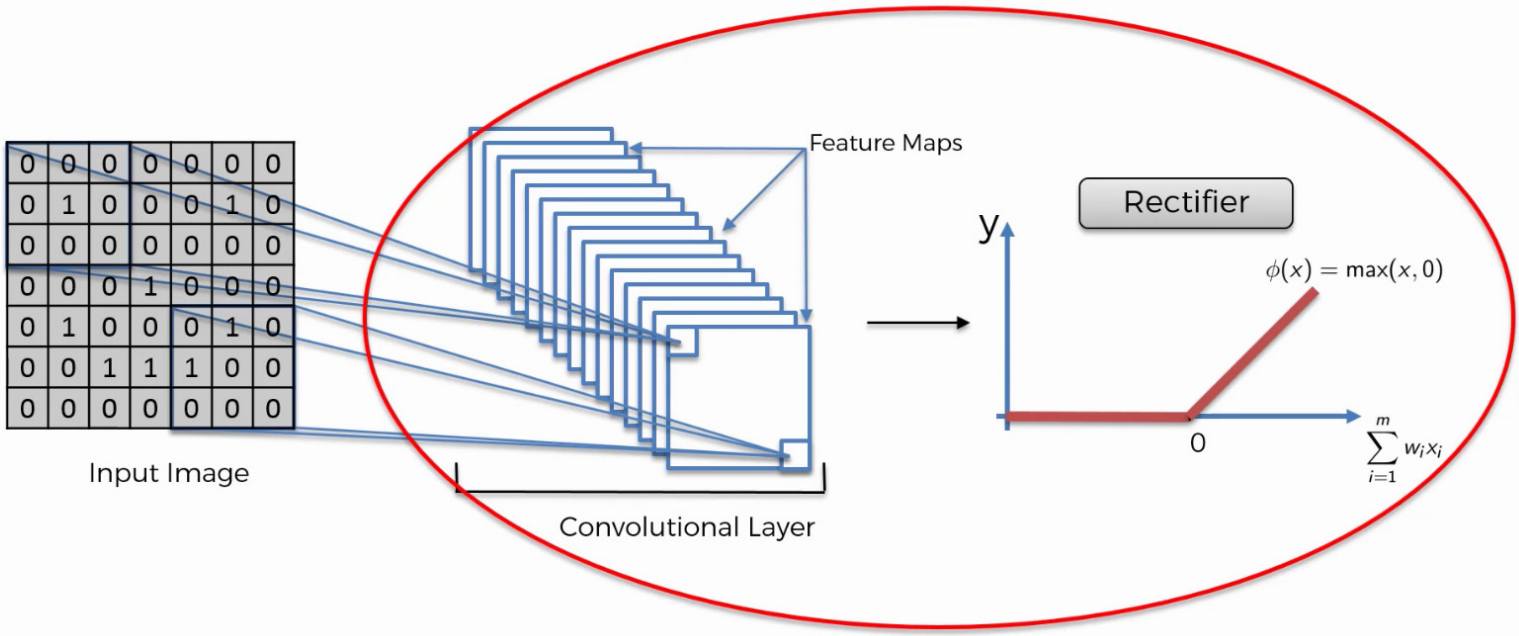
0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

Edge Detect:

	0	1	0	
	1	-4	1	
	0	1	0	



Step 1(B) – ReLU Layer



Source: SuperDataScience

Adding a ReLU activation function, adds non-linearity to filtered images

* There are other types of filters like leaky ReLU and others

An example of ReLU application to a filtered image

Step 1(B) – ReLU Layer

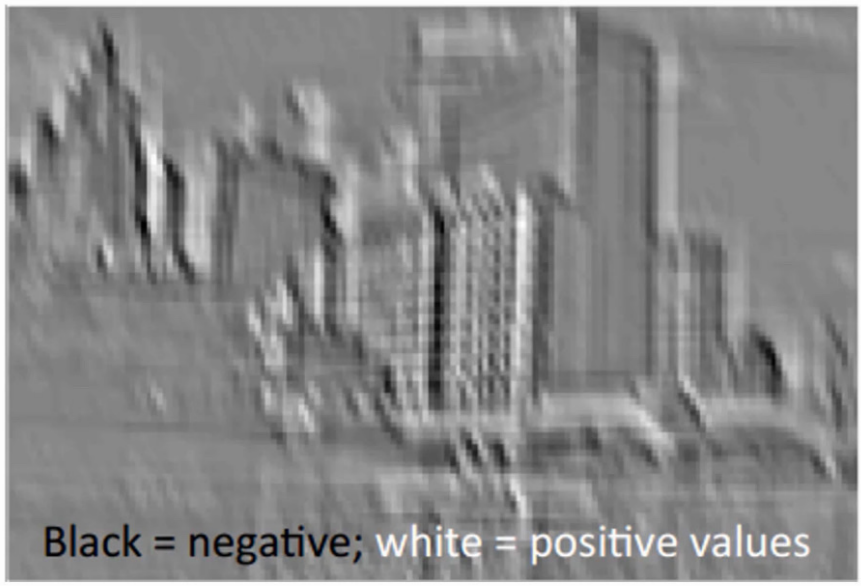


Image Source: http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf

Step 1(B) – ReLU Layer

Image Source: http://mlss.tuebingen.mpg.de/2015/slides/fergus/Fergus_1.pdf

Step 2 - Max Pooling

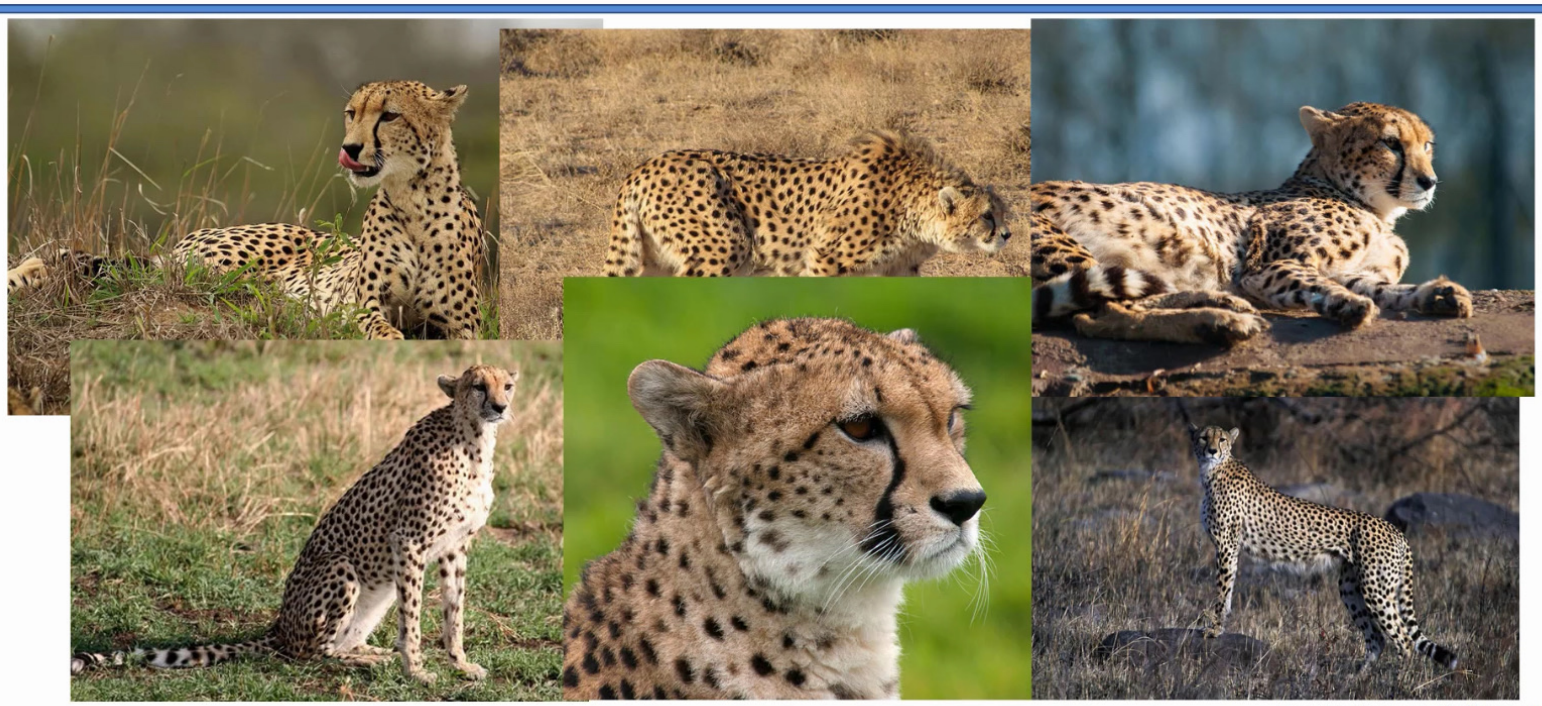
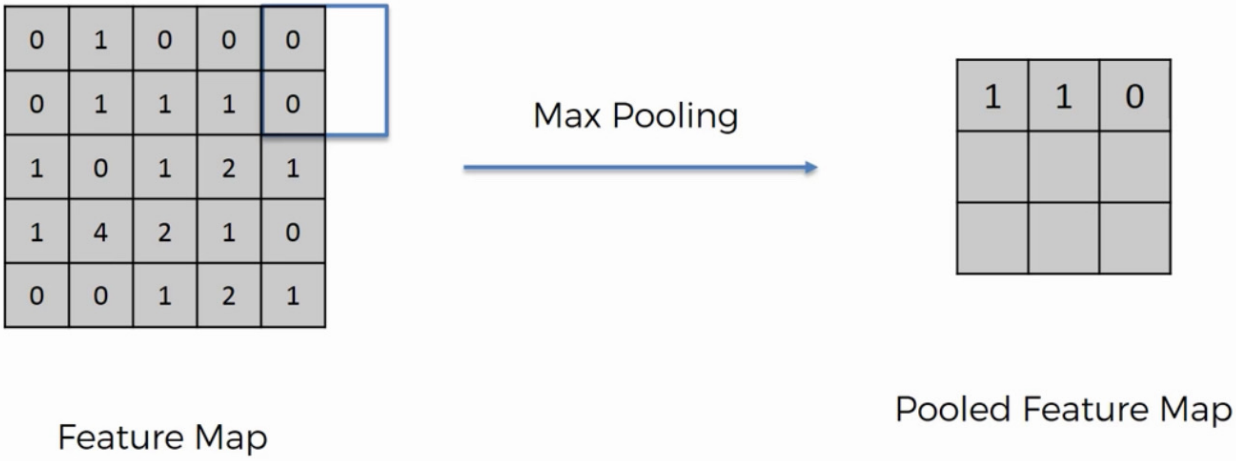


Image Source: Wikipedia

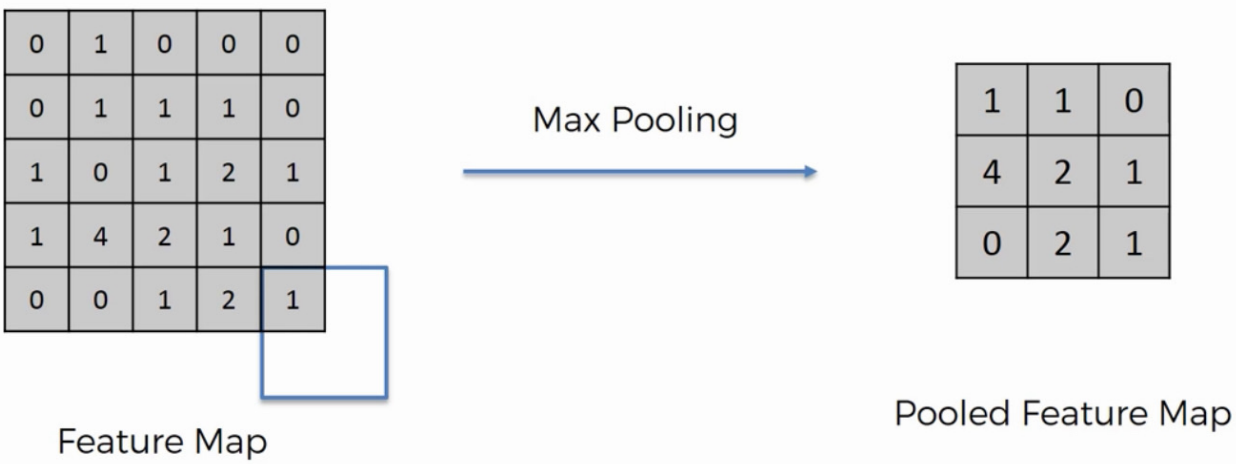
* Max pooling is an operation that ensures that our network can detect features even if they are distorted or at different distances, angles, bckgrounds, etc. This makes the network more flexible in detecting features.

Step 2 - Max Pooling

stride = 2



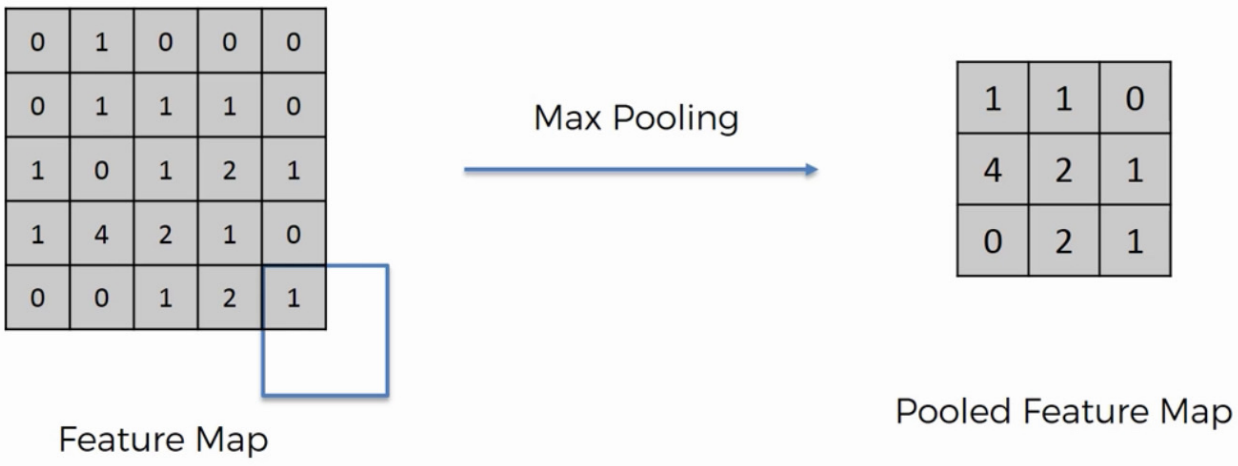
Step 2 - Max Pooling



This process is sometimes referred to as downsampling.

* Through max pooling, important features (max numbers) are preserved (and unimportant ones dropped).

Step 2 - Max Pooling

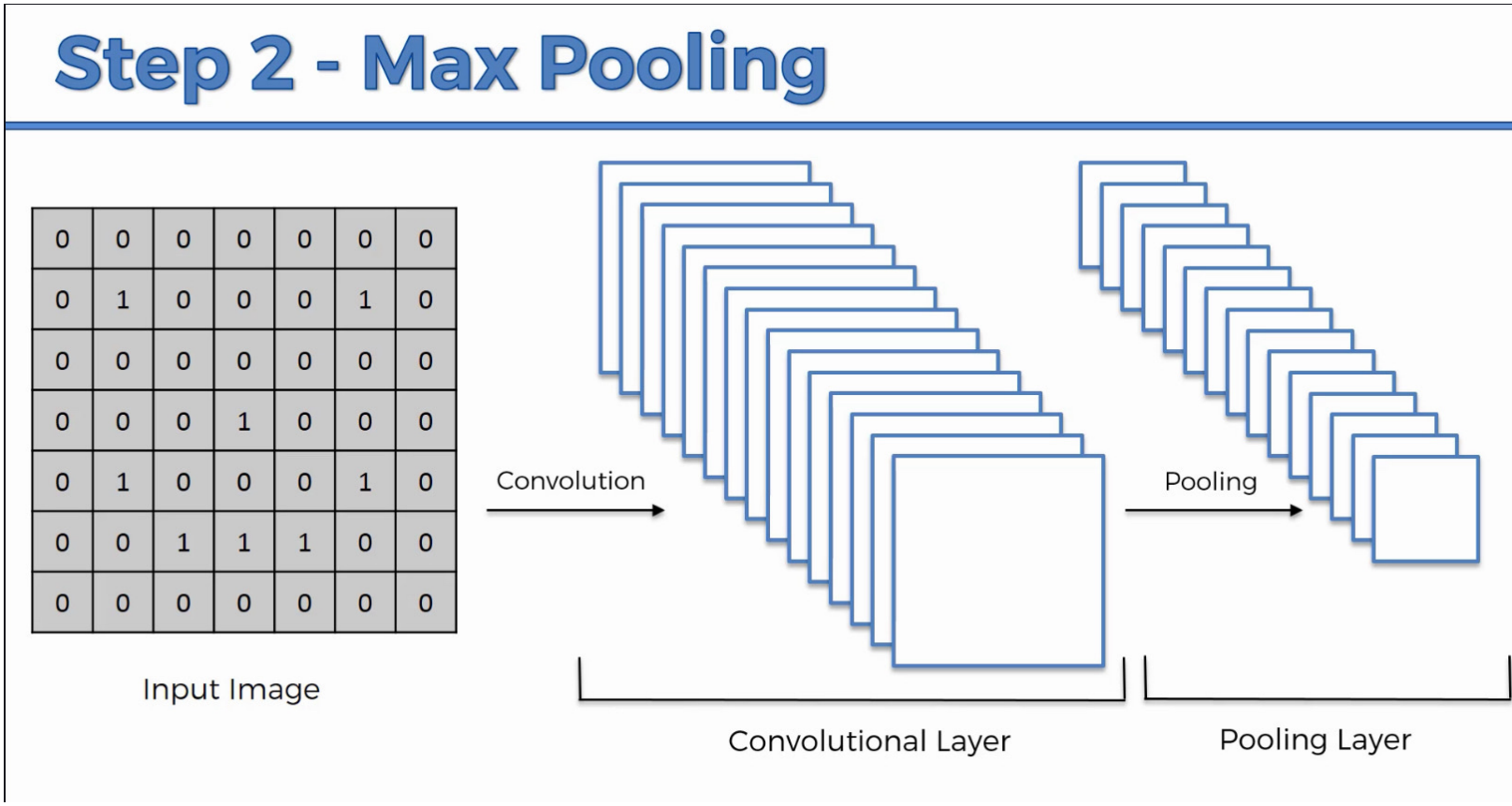


✯ The pooled map is also robust to distortions in the feature map. For instance, if that number 4 is moved up one cell (distortion), we get the same pooled map.

✯ By dropping the unimportant data, in addition to reducing the size of the data, it also helps avoiding overfitting as the network becomes less sensitive to unimportant features

✯ We can modify the size of the pooling matrix, stride, etc.

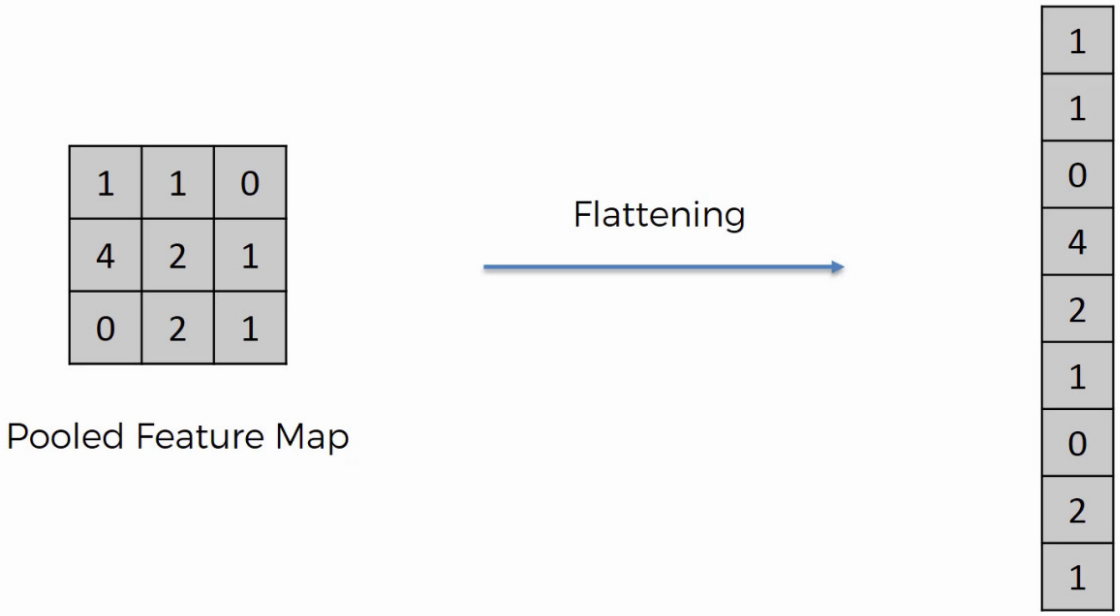
The pooling layer



* There are also other types of pooling such as mean/average pooling, sum pooling, sub-sampling, etc.

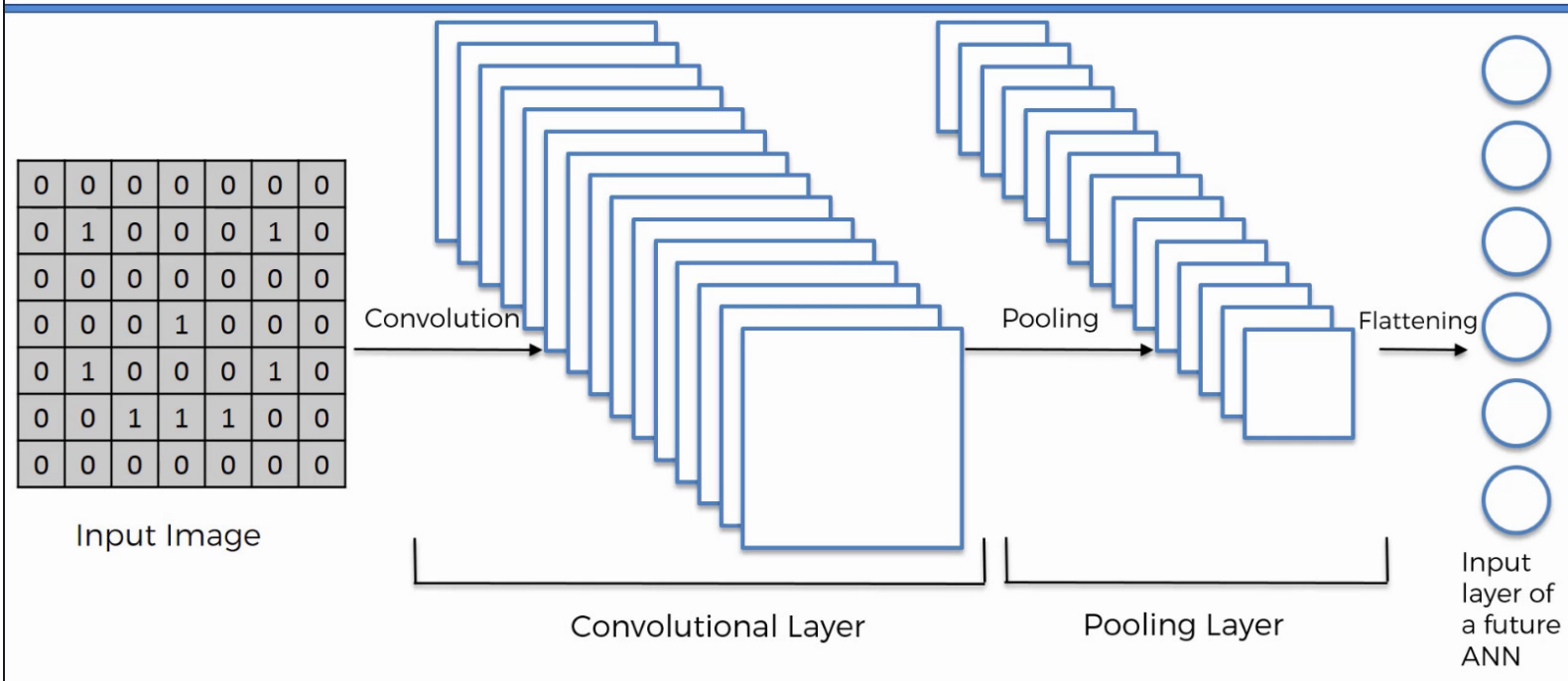
* Sometimes, even before sending the images into the network, people preprocess images by distorting them a bit (for instance by zooming, skewing, tilting, rotating, etc.) in order to avoid overfitting (by reducing the sensitivity of the network to the train images and making it more flexible)

Step 3 - Flattening

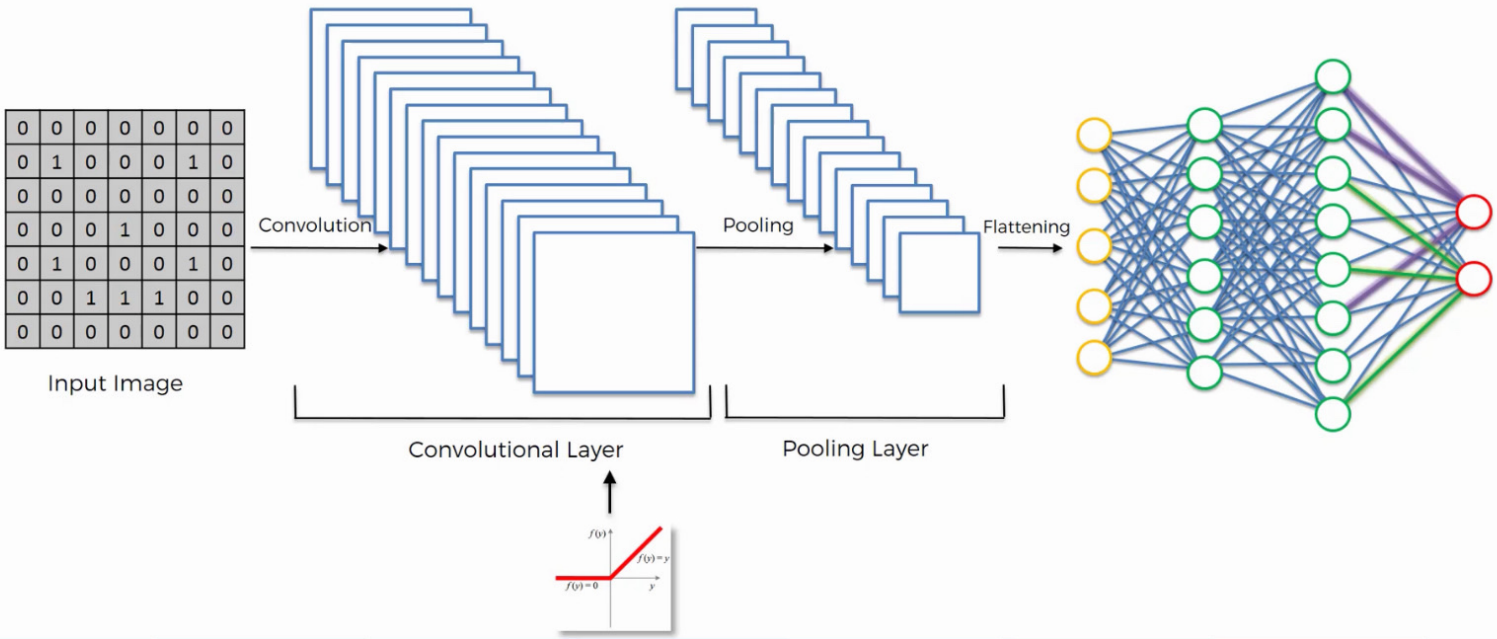


Here we prepare the data to be given as input to a fully connected neural network.

Step 3 - Flattening



Summary



Source: Super Data Science

★ There can be additional layers of convolution and pooling.

★ The network will be trained by defining an appropriate loss function (e.g. cross-entropy in case of classification problems).

★ To see how all of these work in action using an interactive environment, check the following link:

https://adamharley.com/nn_vis/cnn/2d.html

Some additional points:

* There are different architectures of CNNs, and some of them may even remove the fully connected layers or other parts.

* The classification problems can be binary or categorical and involve multiple nodes in the output layer.

* Apart from classification problem, there are CNN architectures that are used for tasks like image segmentation. They can assign labels or probabilities to each pixel (widely use in medical research, physics, cosmology, etc.)

* Search for architectures like U-net or V-net.