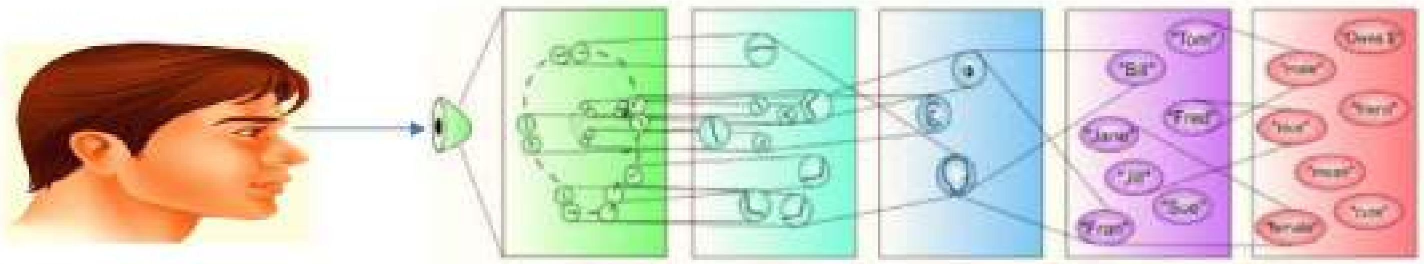
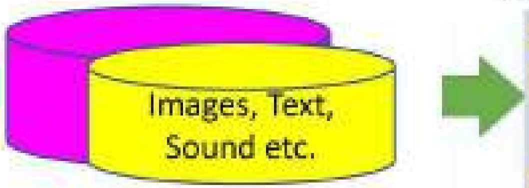
Convolutional Neural Networks

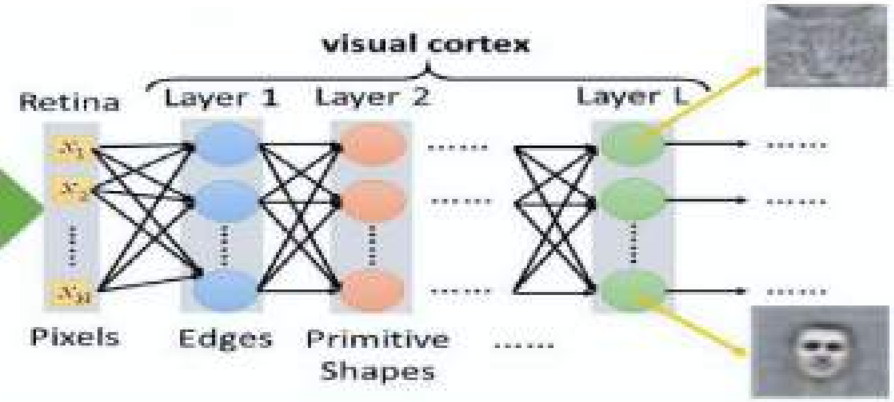
Outline of Presentation

- Image and Computers
- Features Extraction
- Manual Feature Extraction
- Manual feature Representation
- Feature Extraction Issues
- Fully Connected Neural Network
- Feature Extraction and Convolution
- Feature Matching
- Convolution Operation
- Convolution Layer and Feature Maps
- Convolutional Neural Network: Spatial View
- Pooling and Activation Function (ReLU)
- CNNs Applications

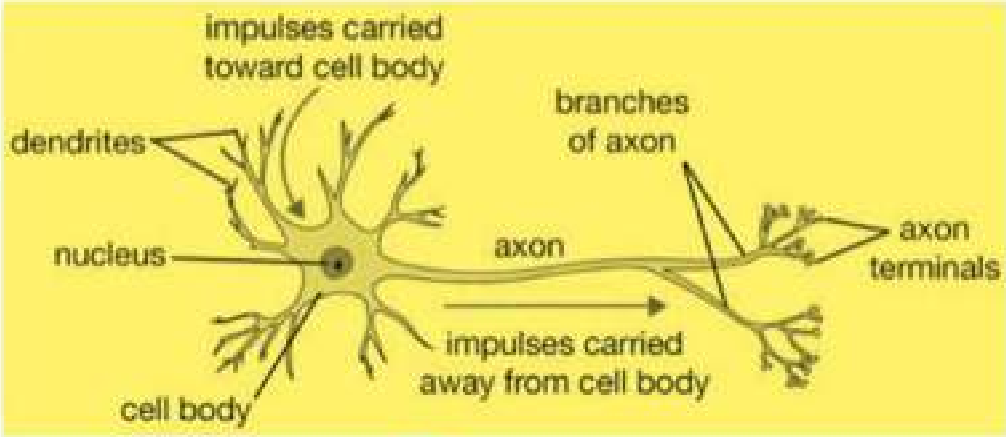
Visual Cortex and Deep Neural Network (DNN)



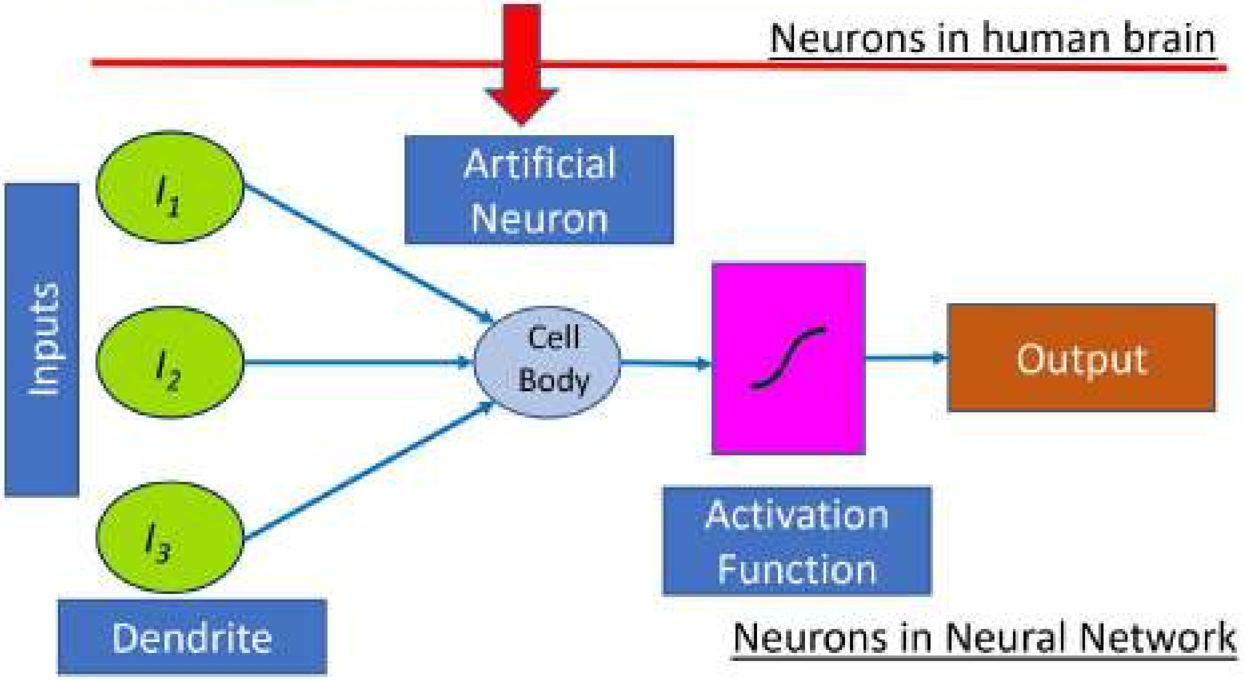




Visual Cortex and Deep Neural Network (DNN)



Neurons in human brain

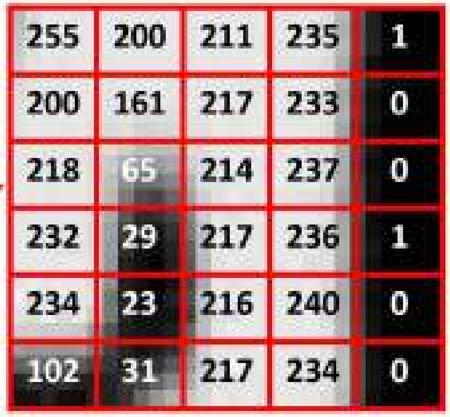


Neurons in Neural Network

Comparison of Biological neural network and Artificial Neural Network

Biological Neural Network	Artificial Neural Network
Stimulus	Input
Receptor	Input Layer
Neural Network	Processing Layer
Neuron	Processing Element
Dendrite	Addition Function
Synapse	Weight
Cell Body	Transfer/ Activation Function
Axon	Artificial Neural Output

Images are Just a Matrix of Numbers



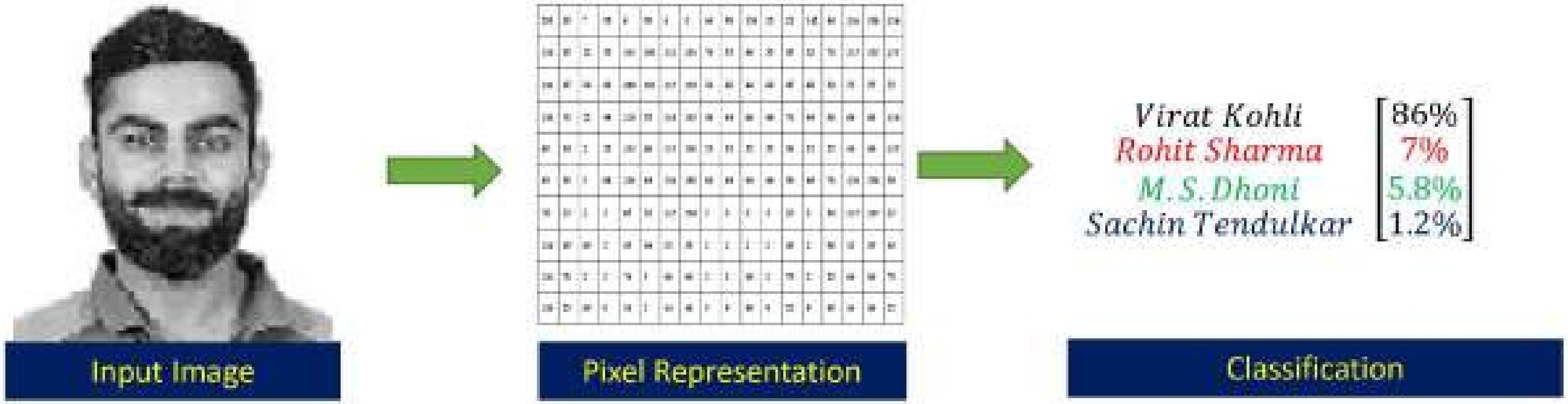
255	200	211	235	1
200	161	217	233	0
218	65	214	237	0
232	29	217	236	0
234	23	216	240	0
102	31	217	234	0

Computer Interpretation

- For a grayscale images, the pixel value is a single number that represents the brightness of the pixel. The most common pixel format is the byte image, where this number is stored as an 8-bit integer giving a range of possible values from 0 to 255.
- Similarly for color images, each level is represented by the range of decimal numbers from 0 to 255 (256 levels for each color), equivalent to the range of binary numbers from 00000000 to 11111111, or hexadecimal 00 to FF. The total number of available colors is 256 x 256 x 256, or 16,777,216 possible color.

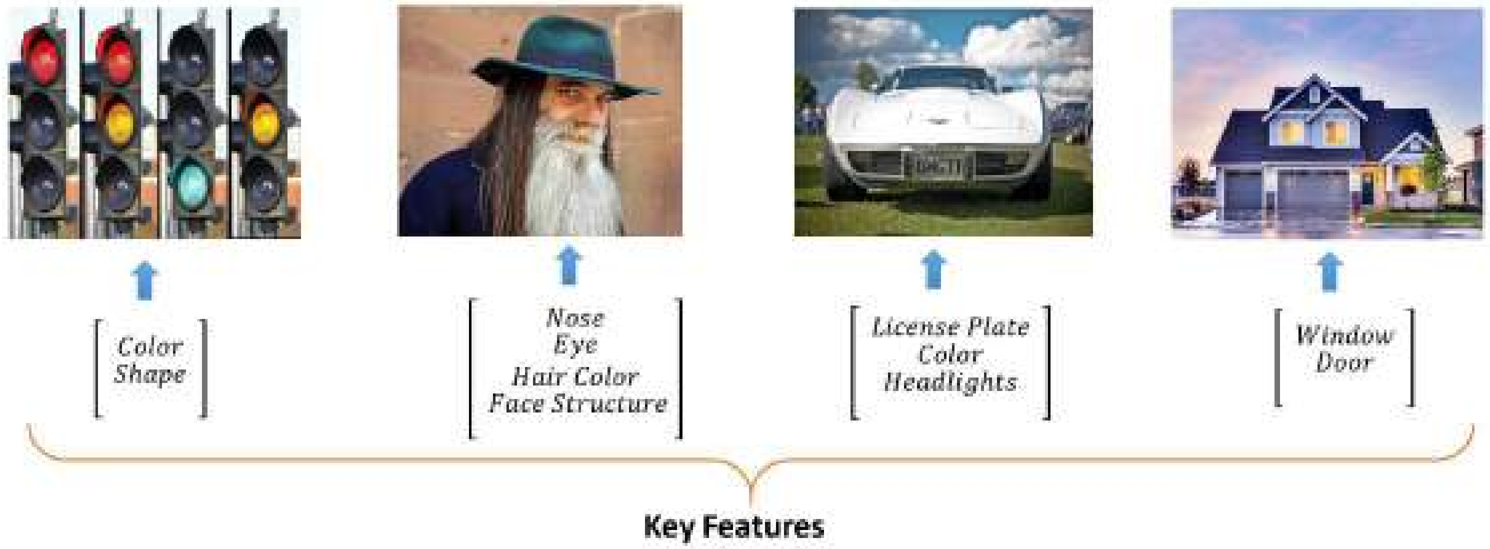
Computer Vision Task

- ***Regression*** - is about predicting a quantity. It is the process of finding a model or function for distinguishing the data into continuous real values. Example: Price estimation of houses.
- ***Classification***- is about predicting a label. It is the process of finding or discovering a model or function which helps in separating the data into multiple categorical classes i.e. discrete values.



High Level Feature Detection

Low-level features are minor details of the image, like lines or dots, that can be picked up by convolutional filter (for really low-level things) or SIFT or HOG (for more abstract things like edges). High-level features are built on top of low-level features to detect objects and larger shapes in the image.

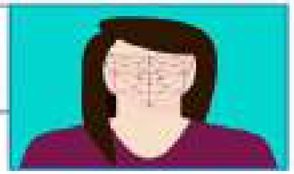


Manual Feature Extraction

1. *Domain Knowledge*

2. Specify Features

3. Detect Features for Classification



Manual Feature Extraction: Issues

Occlusion

Cluttered Background

Intra-class Diversity

Object Deformation

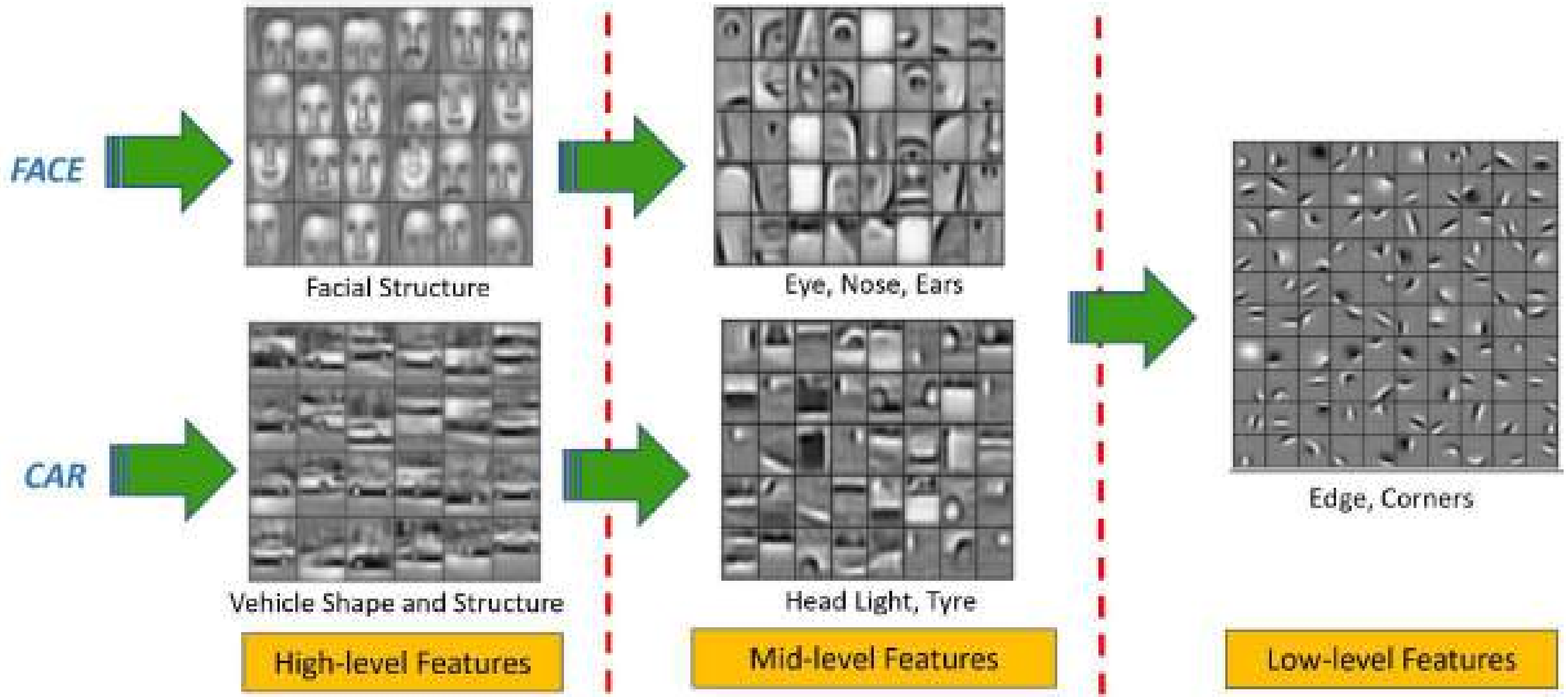
Scale Variation

Illumination Variation

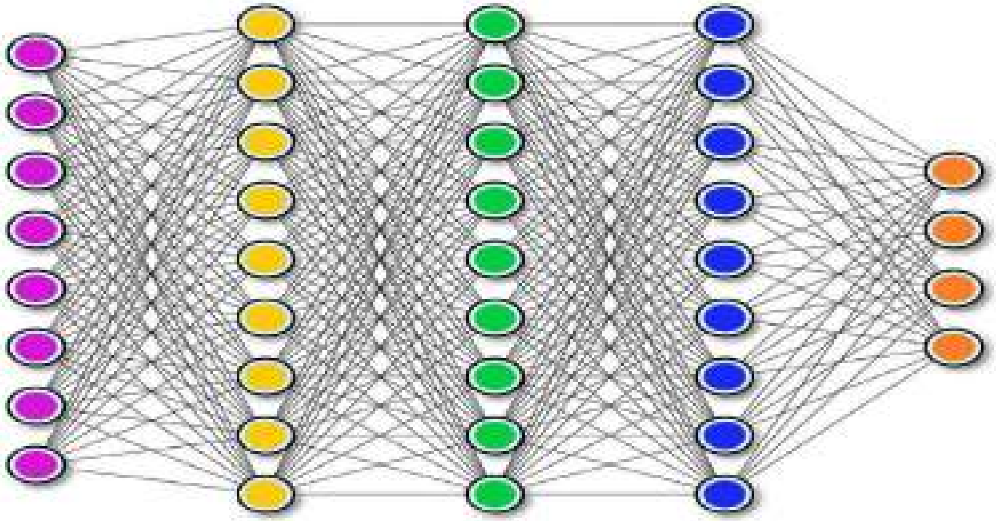
Change in Viewing Angles

Image Deformity

Feature Representation

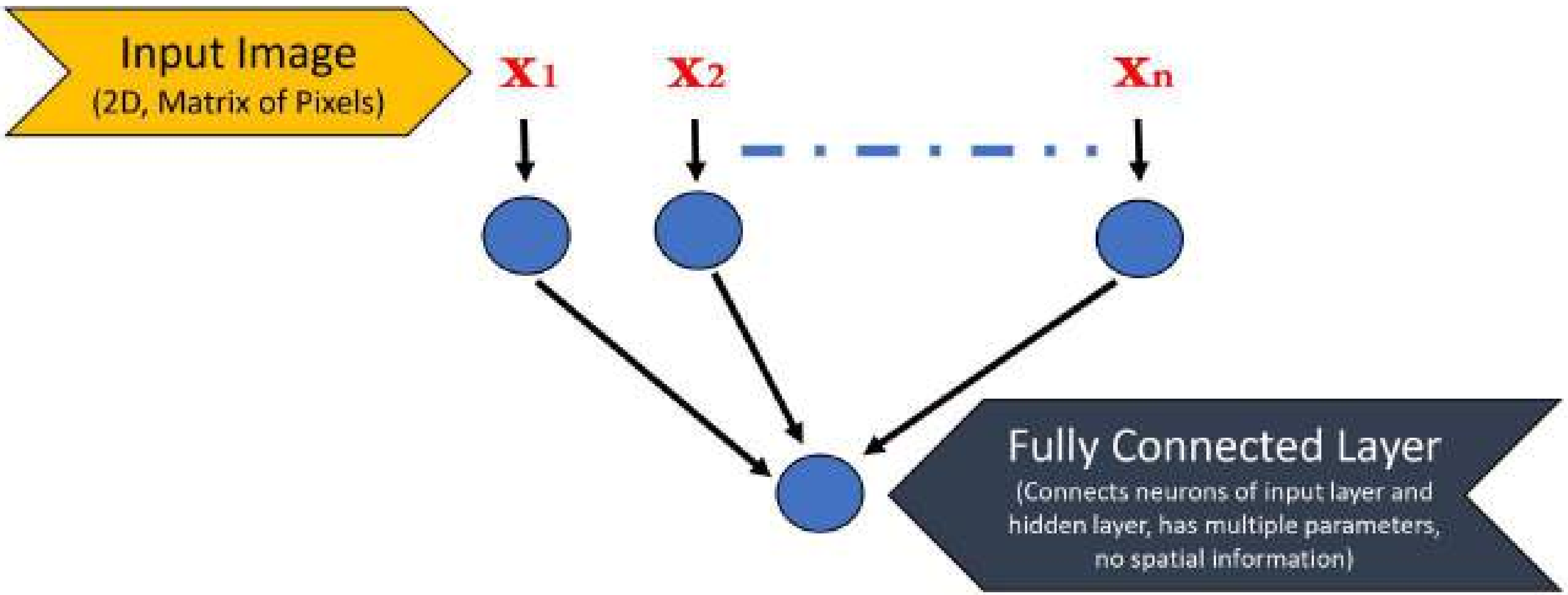


Fully Connected Neural Network

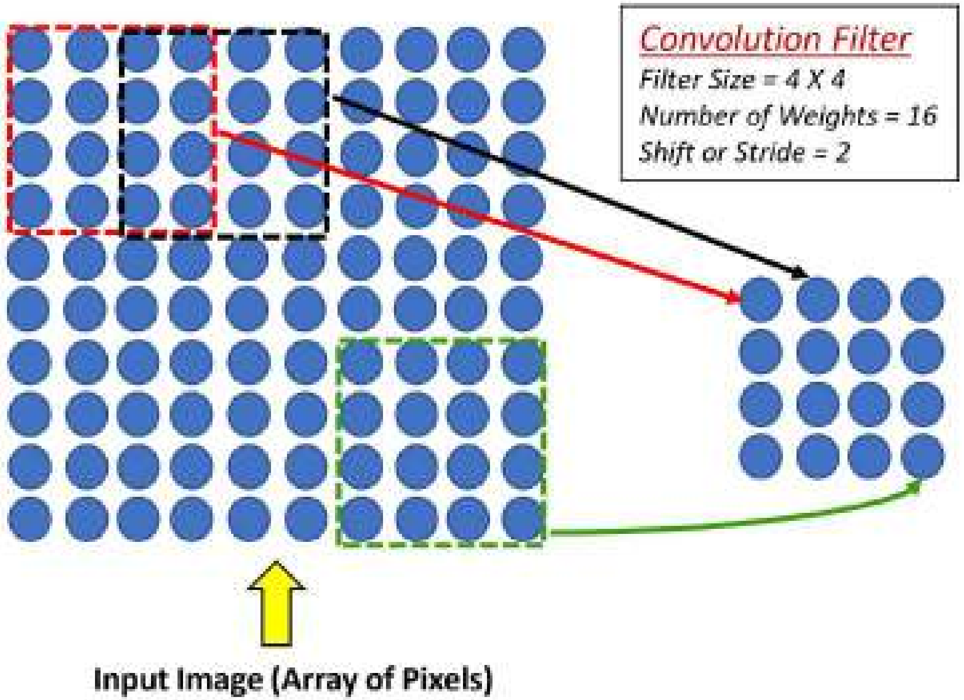


Fully connected neural networks (FCNNs) are a type of artificial neural network where the architecture is such that all the nodes, or neurons, in one layer are connected to the neurons in the next layer.

Fully Connected Neural Network



Feature Extraction using Convolution



Connect Input Layer patches to neurons of hidden layer/subsequent layer with sliding window approach.

Step 1:

Extract Set of Local Features by applying filters (set of weights)

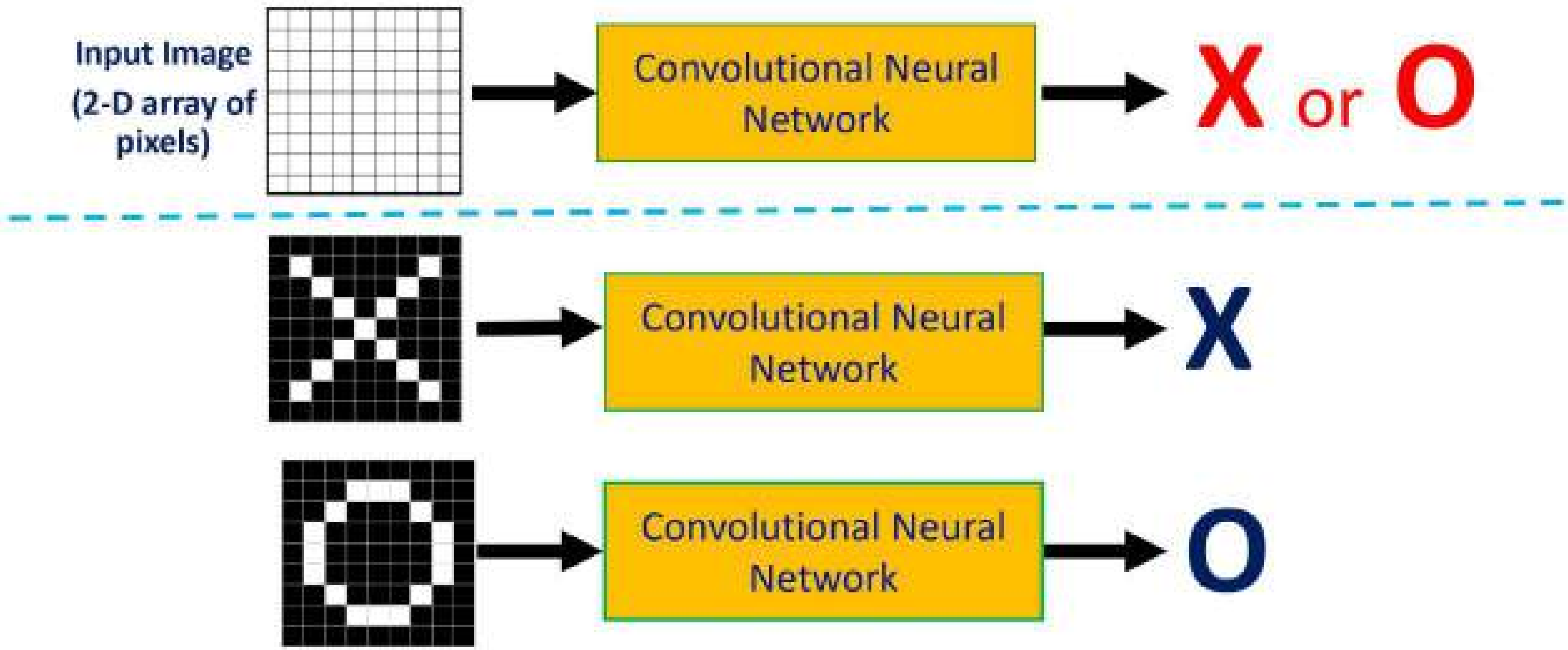
Step 2:

Apply Multiple Filters for extraction of different features

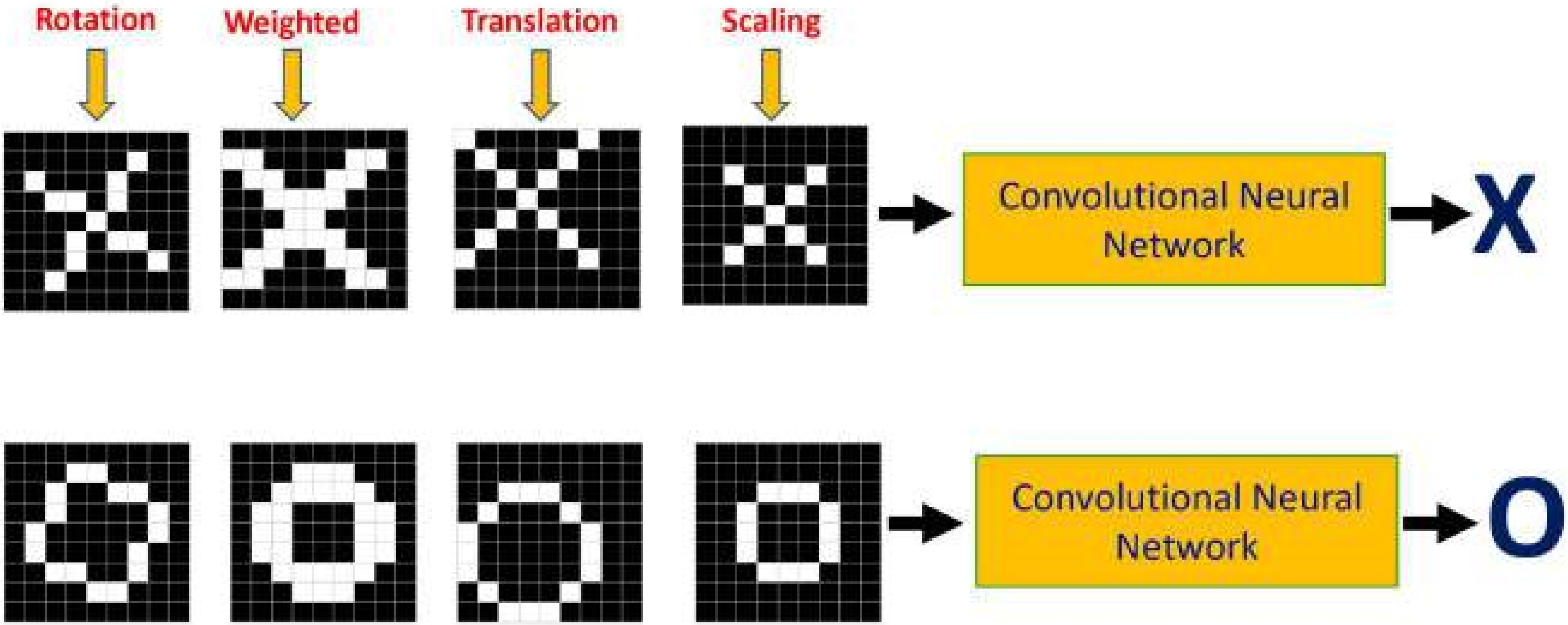
Step 3:

Spatial Sharing of parameters for each filter

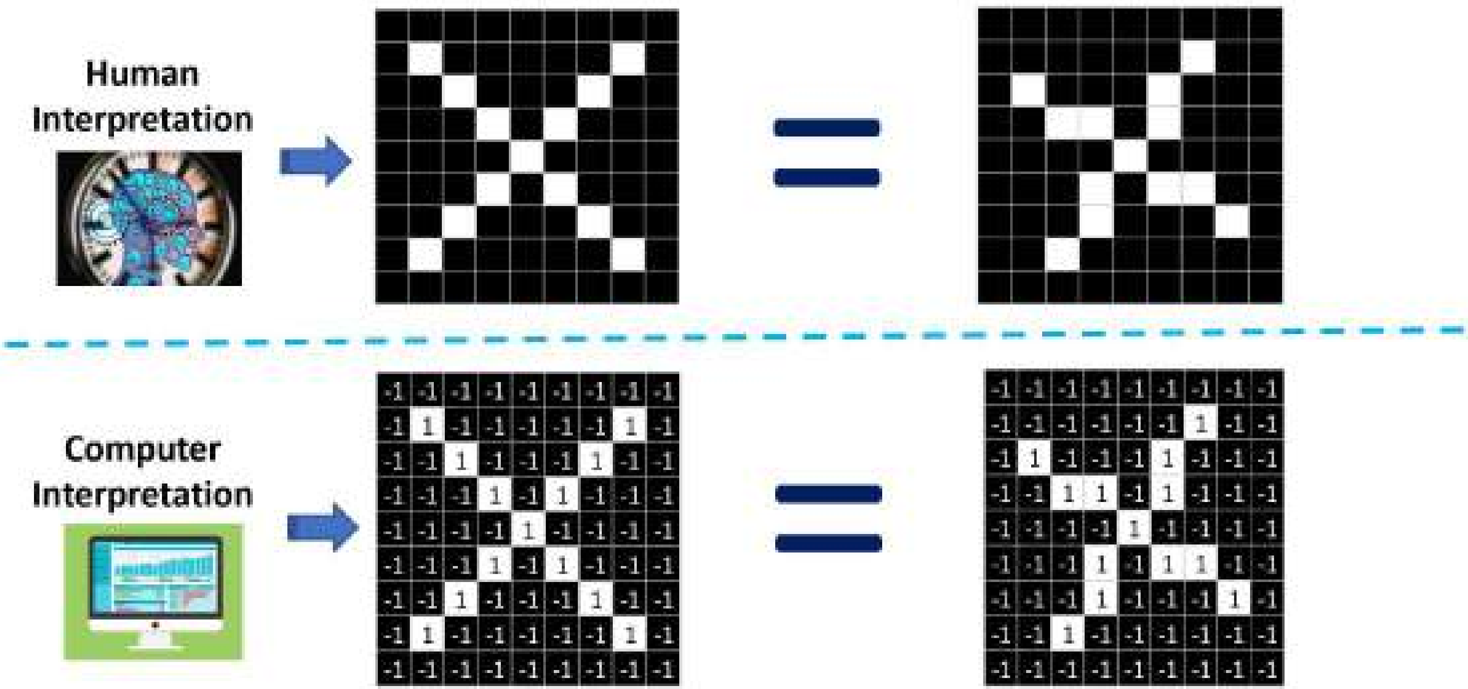
Convolutional Neural Network (Feature Extraction and Convolution)



Challenging Cases

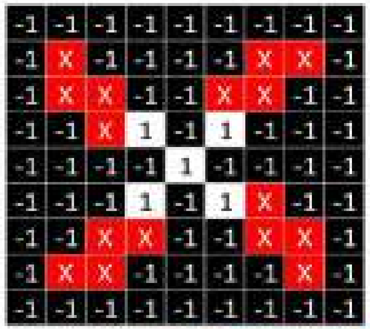


Computer and Human Interpretation

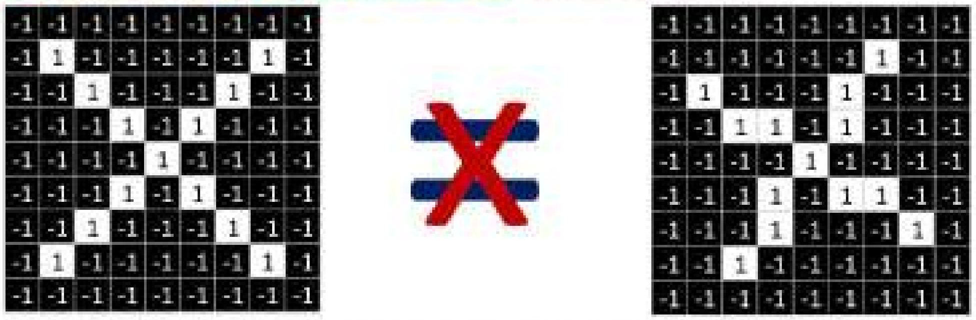


Computer Interpretation

Pixel wise Matching

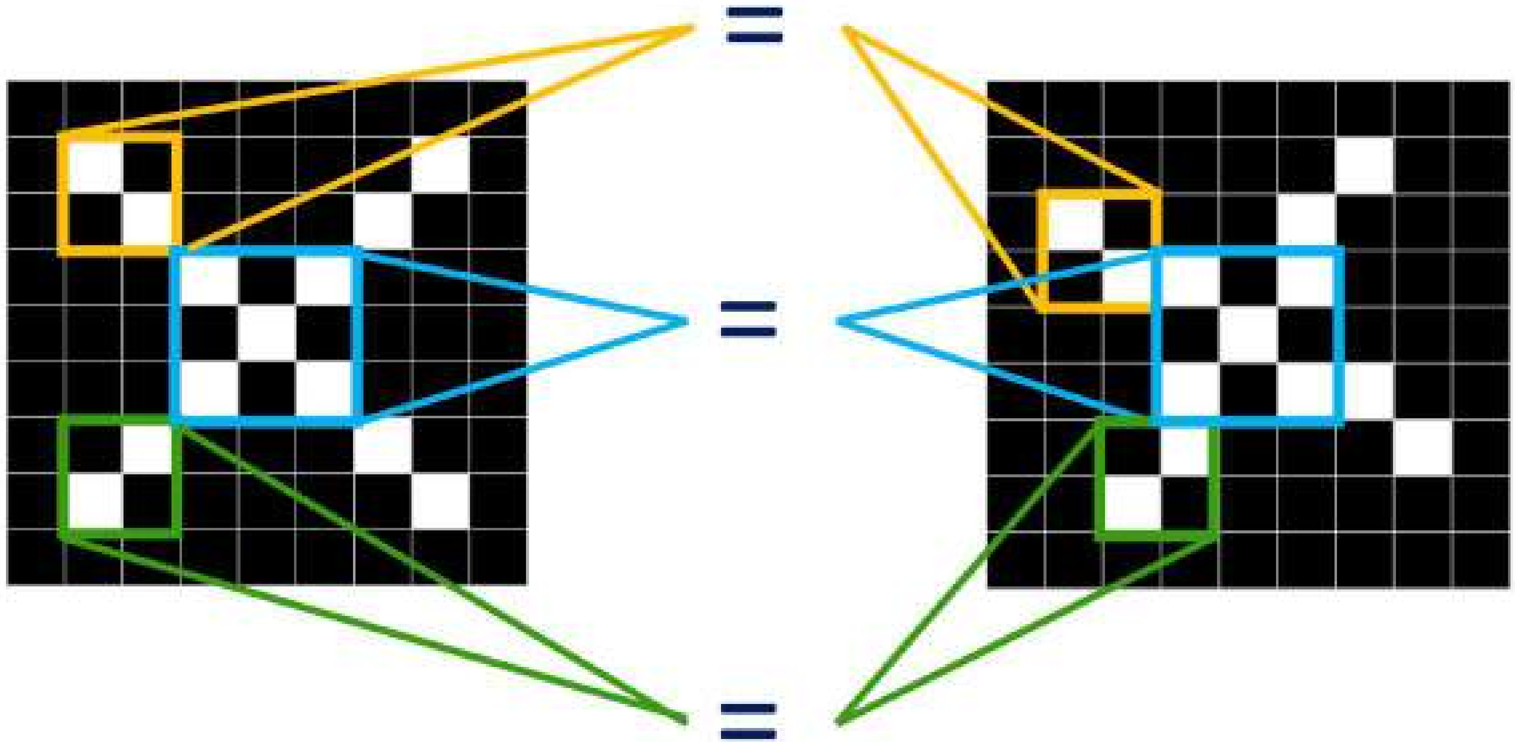


Decision

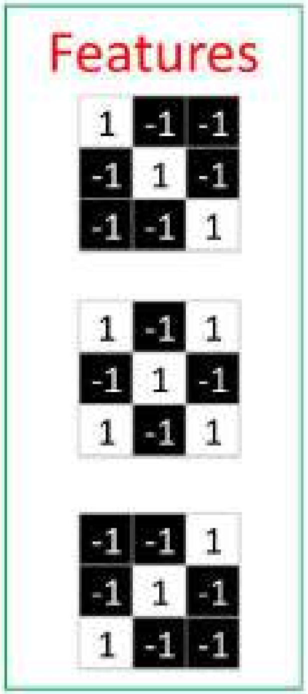


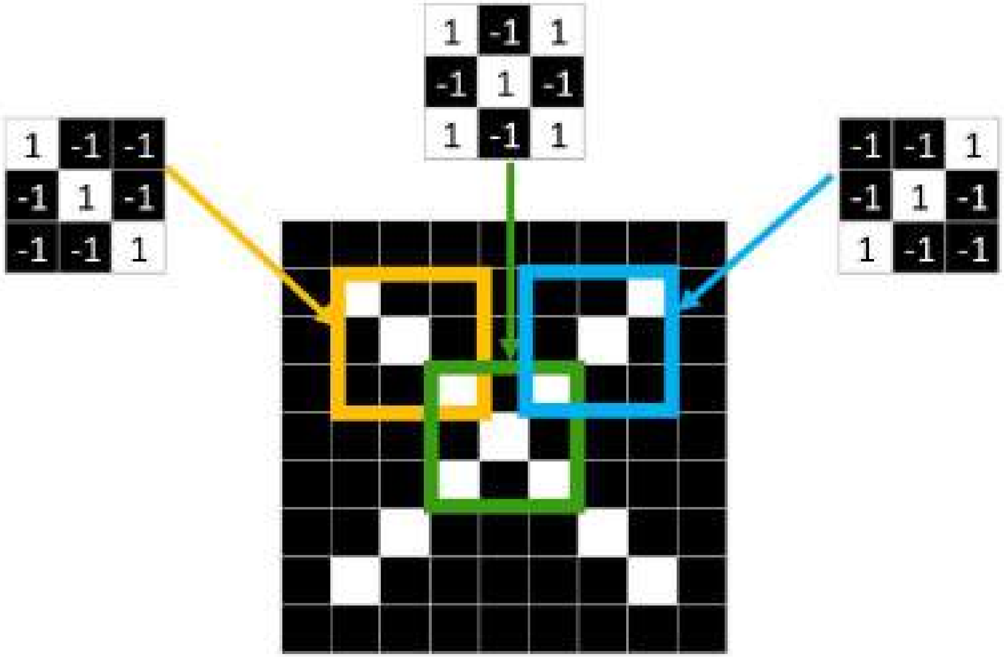
Computers are Literal

Feature matching for symbol ‘X’

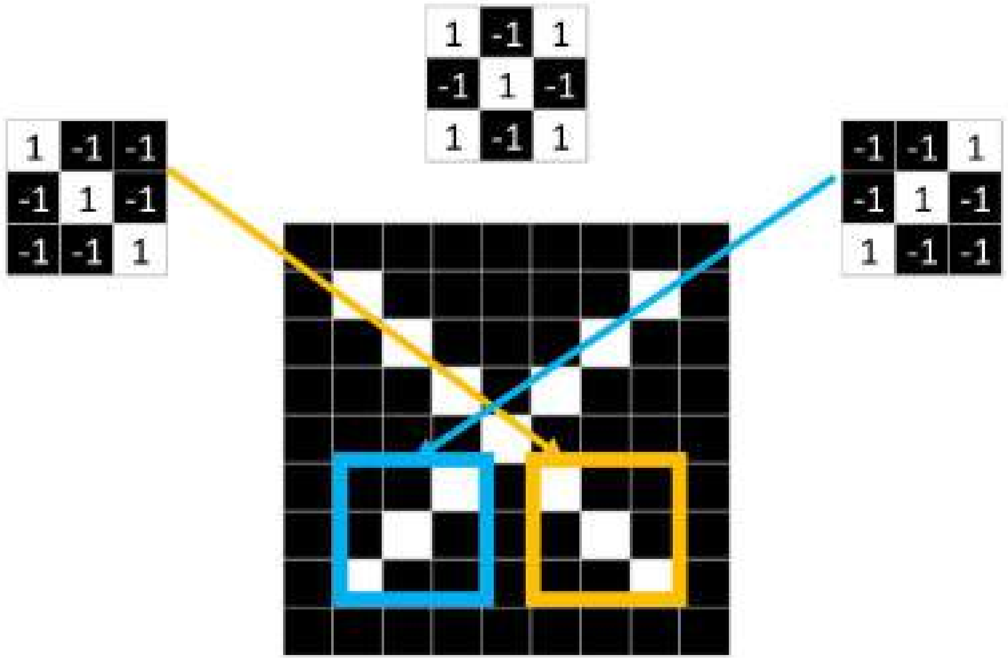


Piece Matching of Features

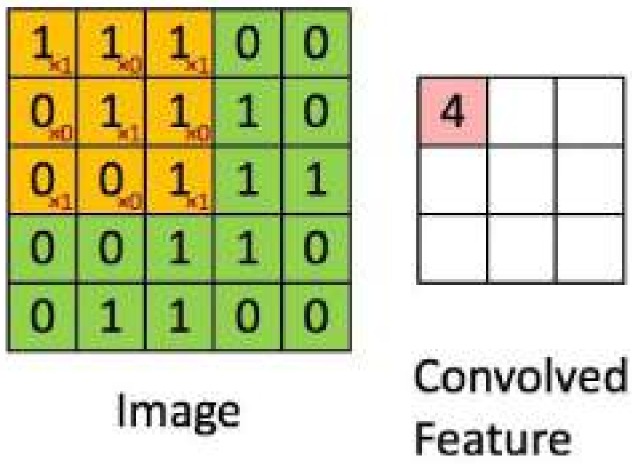




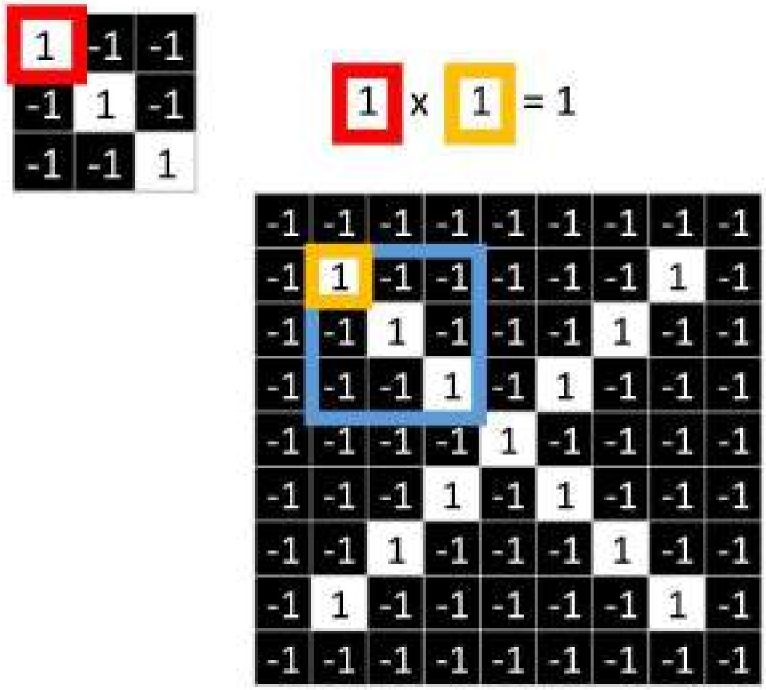
Piece Matching of Features



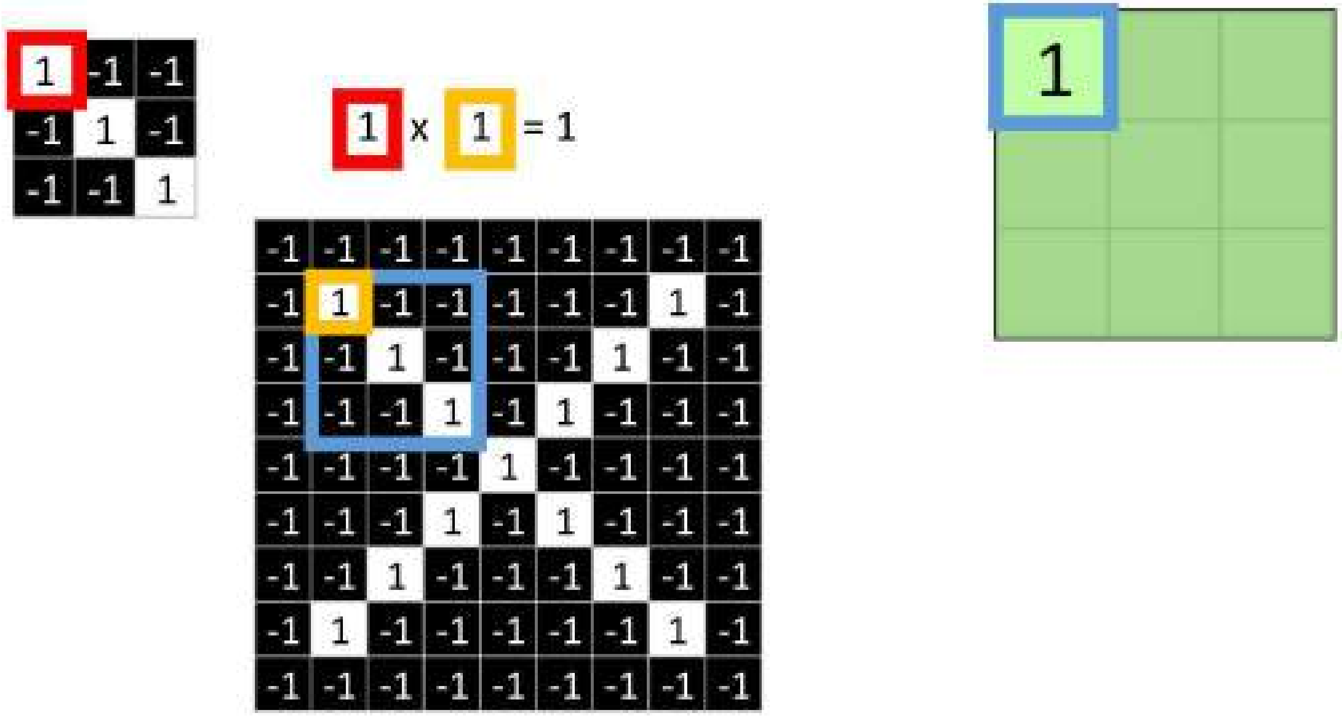
Convolution Operation



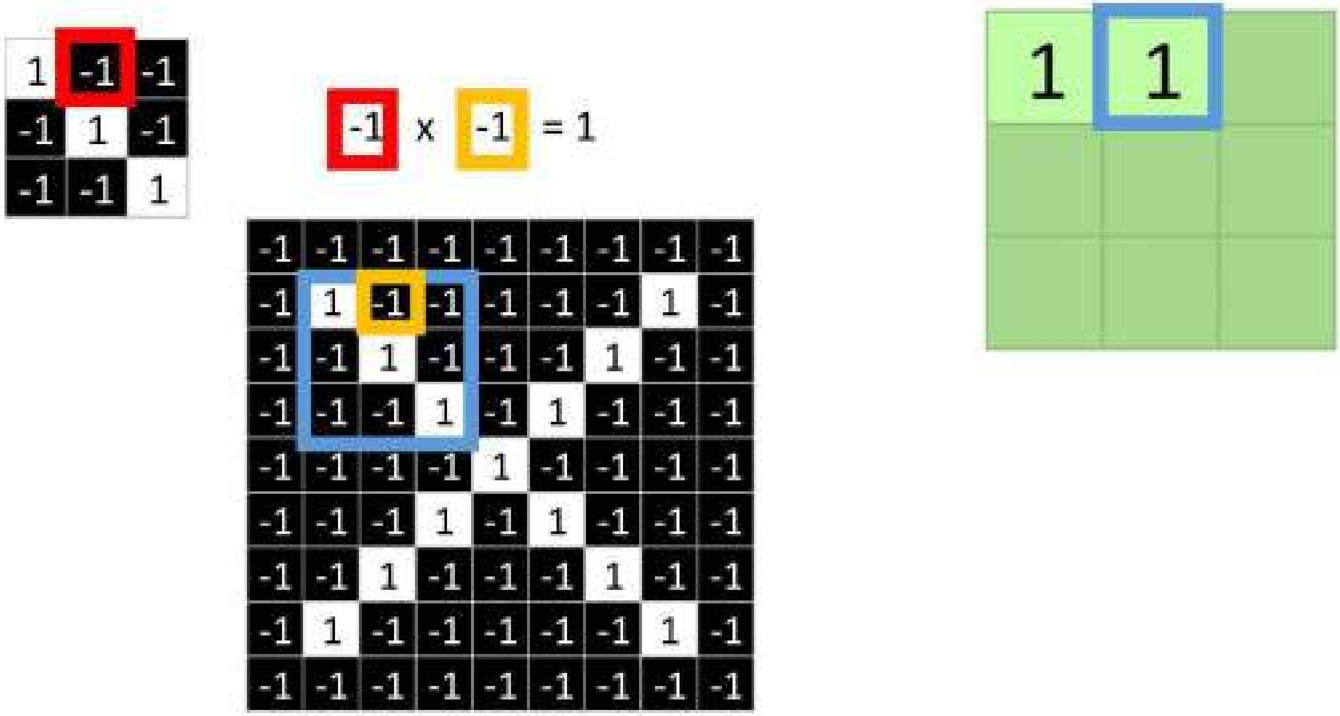
Convolution Operation



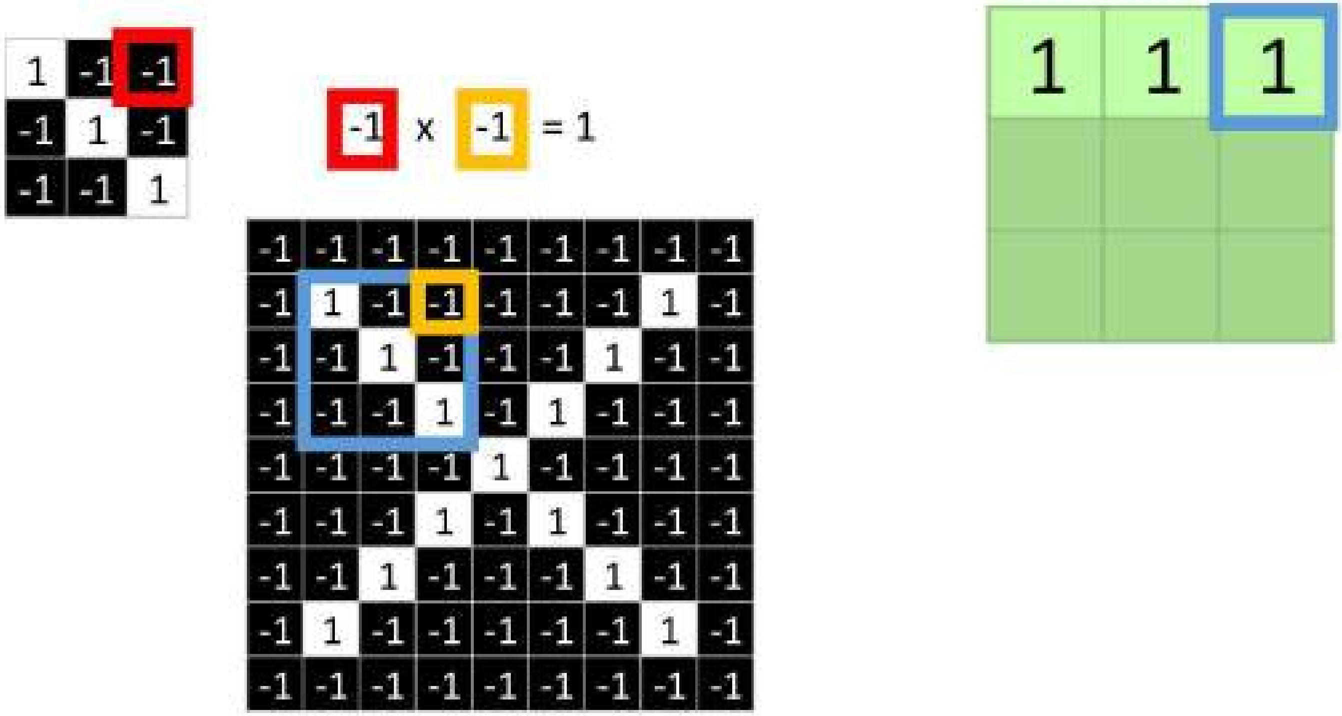
Convolution Operation



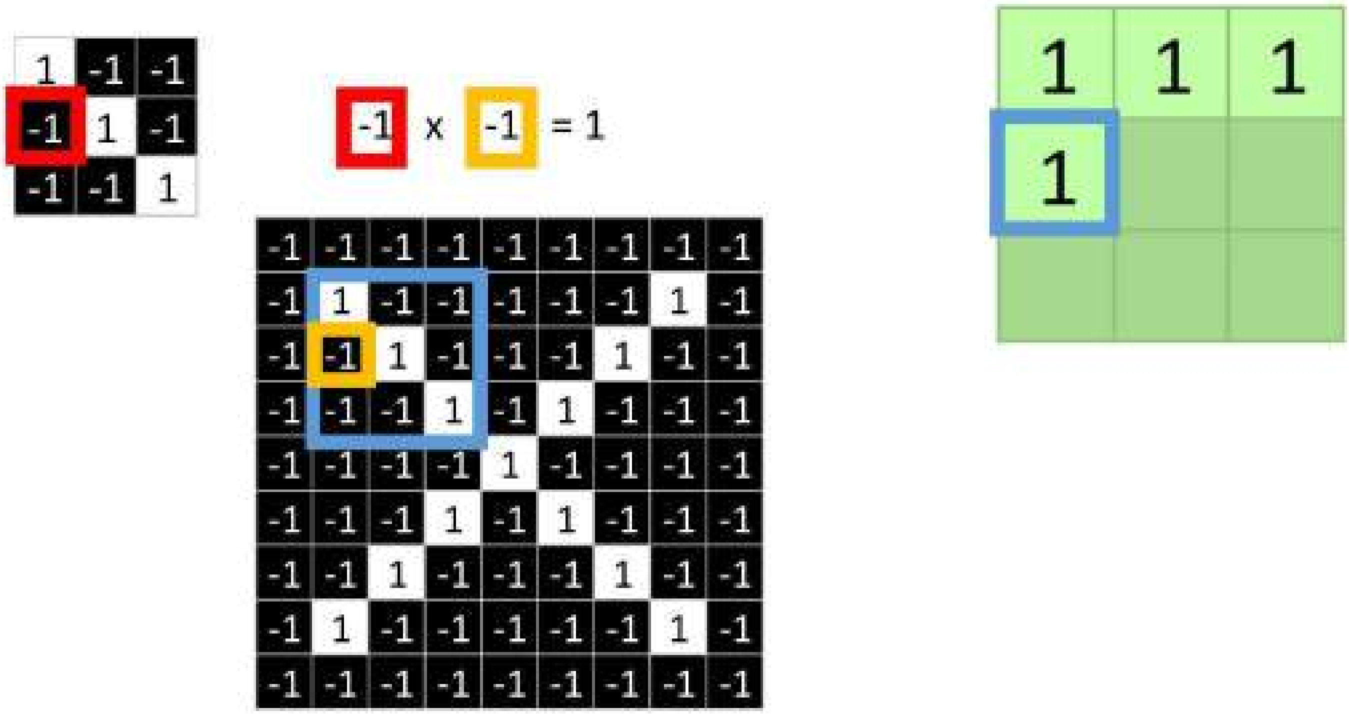
Convolution Operation



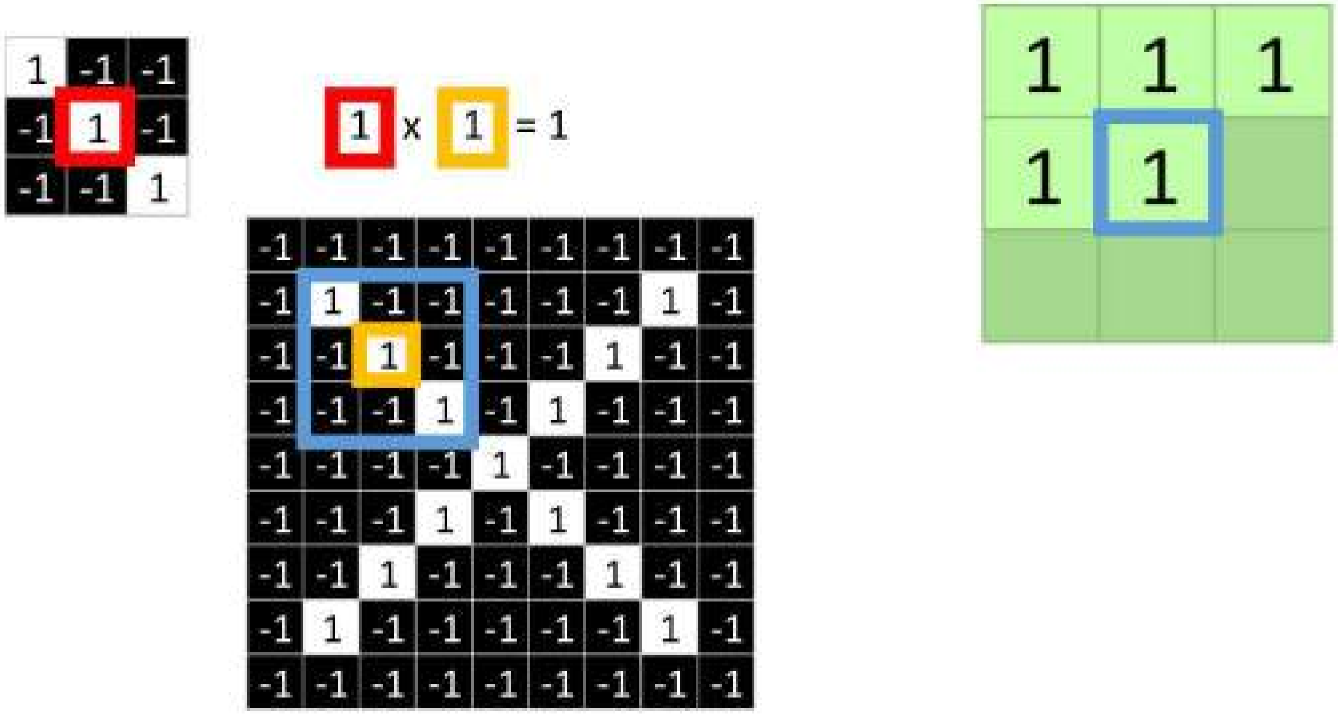
Convolution Operation



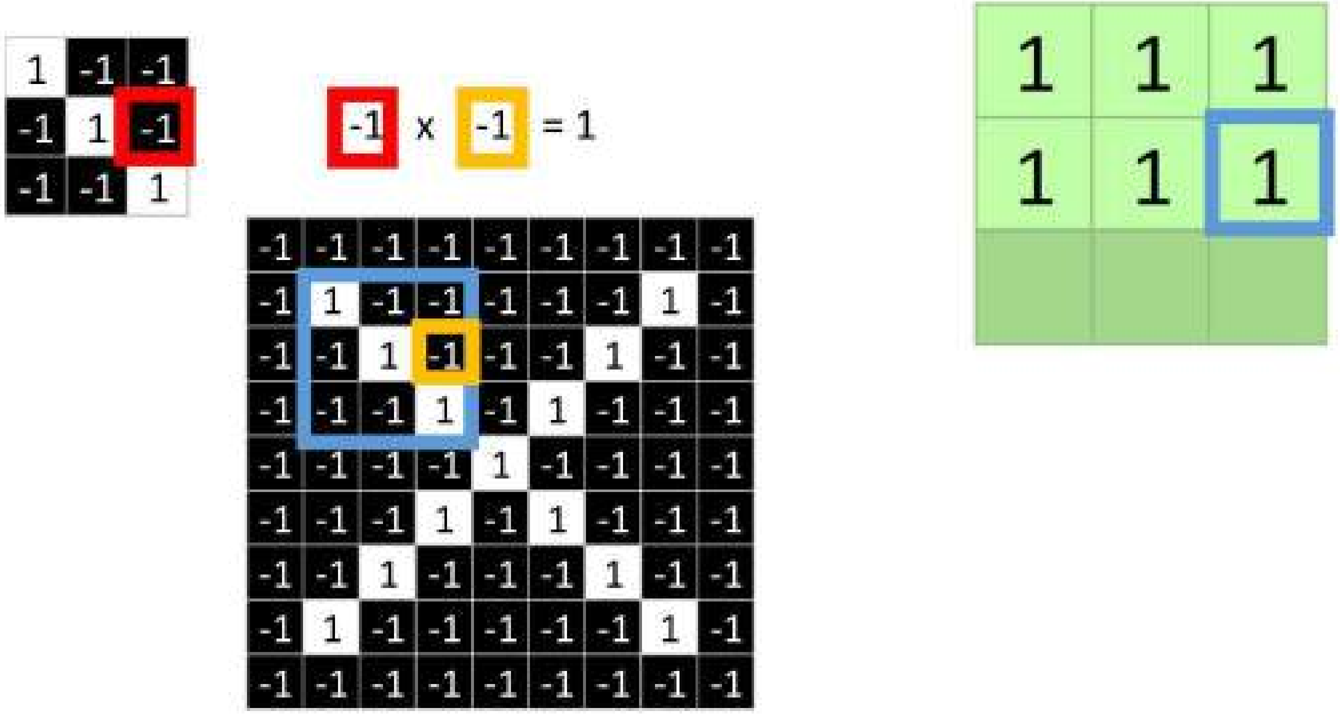
Convolution Operation



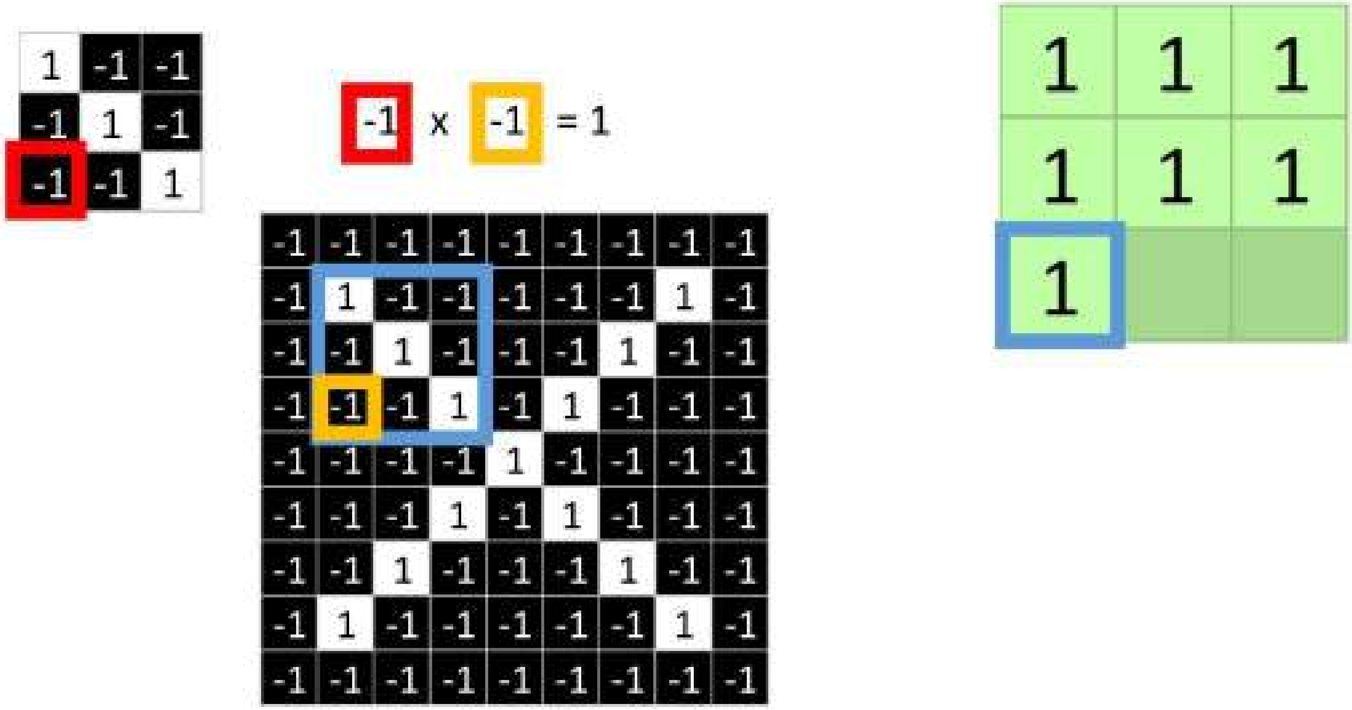
Convolution Operation



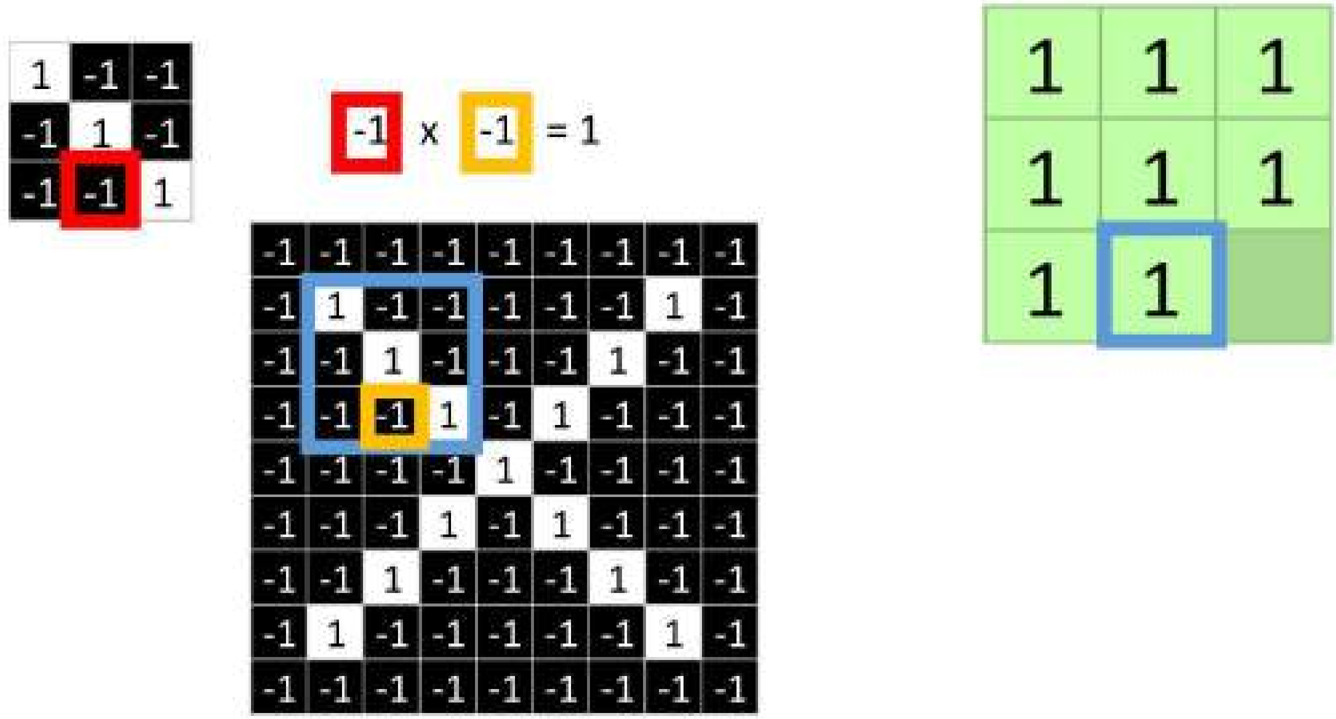
Convolution Operation



Convolution Operation



Convolution Operation



Convolution Operation

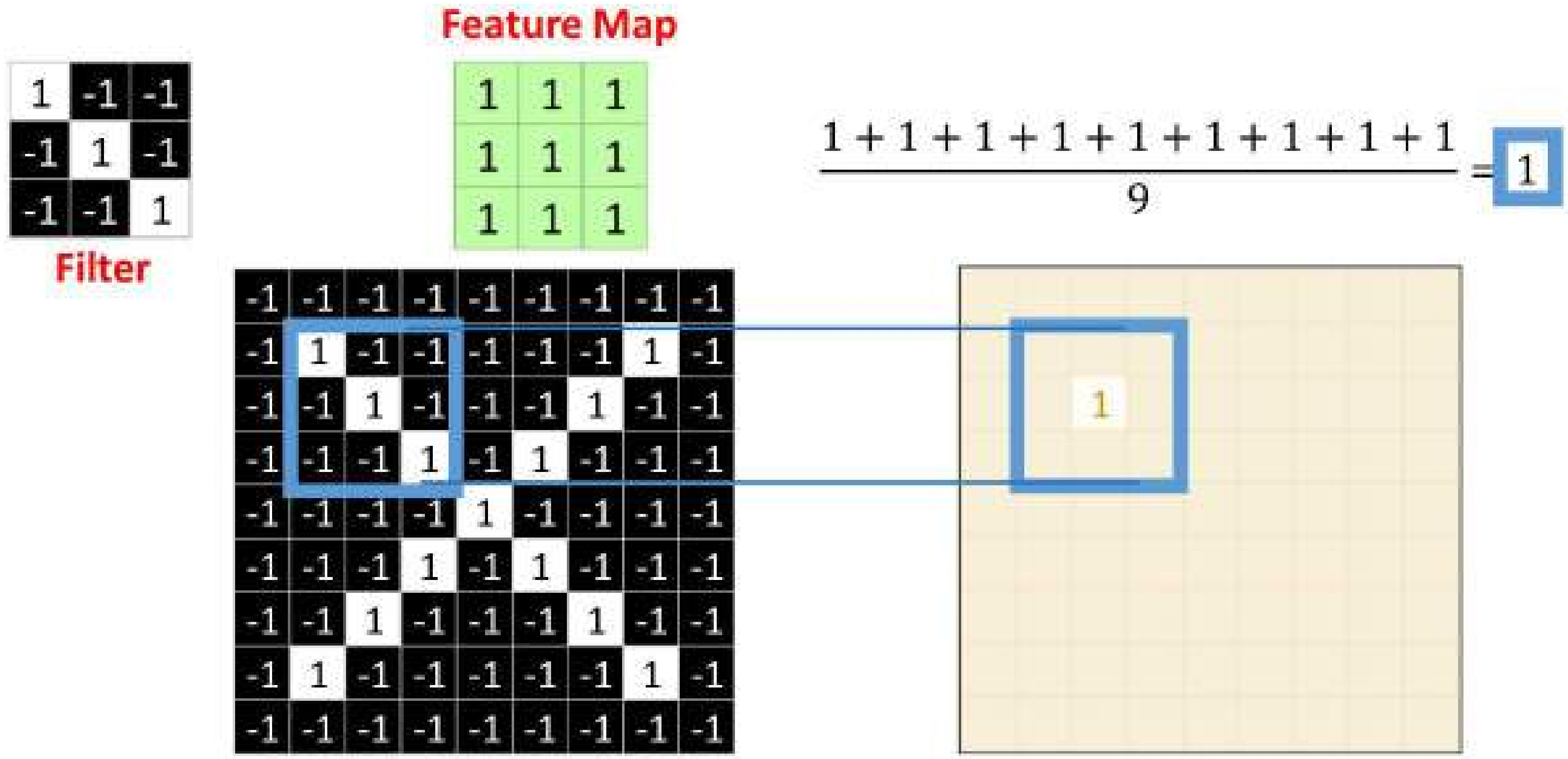
1	-1	-1
-1	1	-1
-1	-1	$\boxed{1}$

$\boxed{1} \times \boxed{1} = 1$

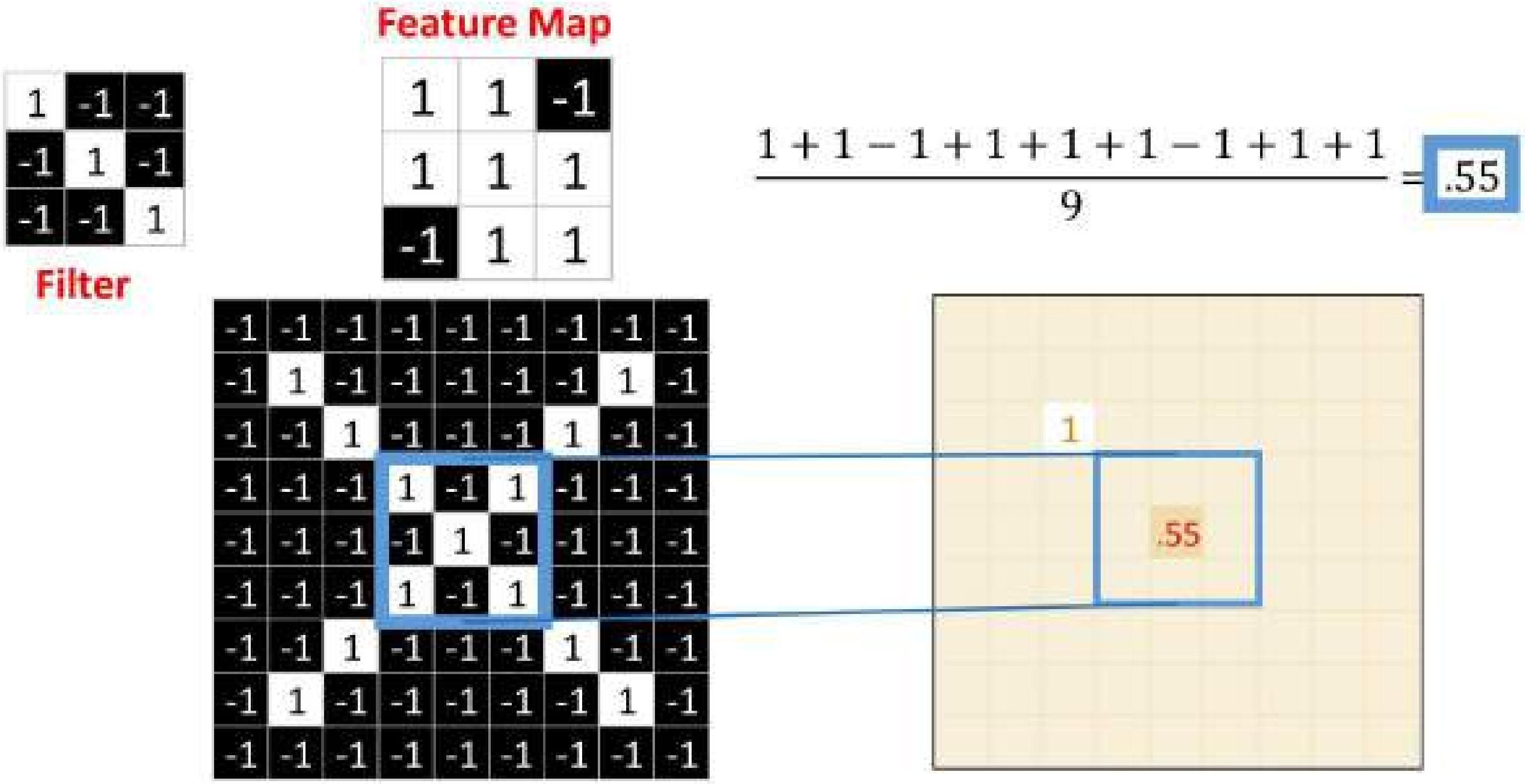
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	$\boxed{1}$	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1	1	1
1	1	1
1	1	$\boxed{1}$

Convolution Operation



Convolution Operation

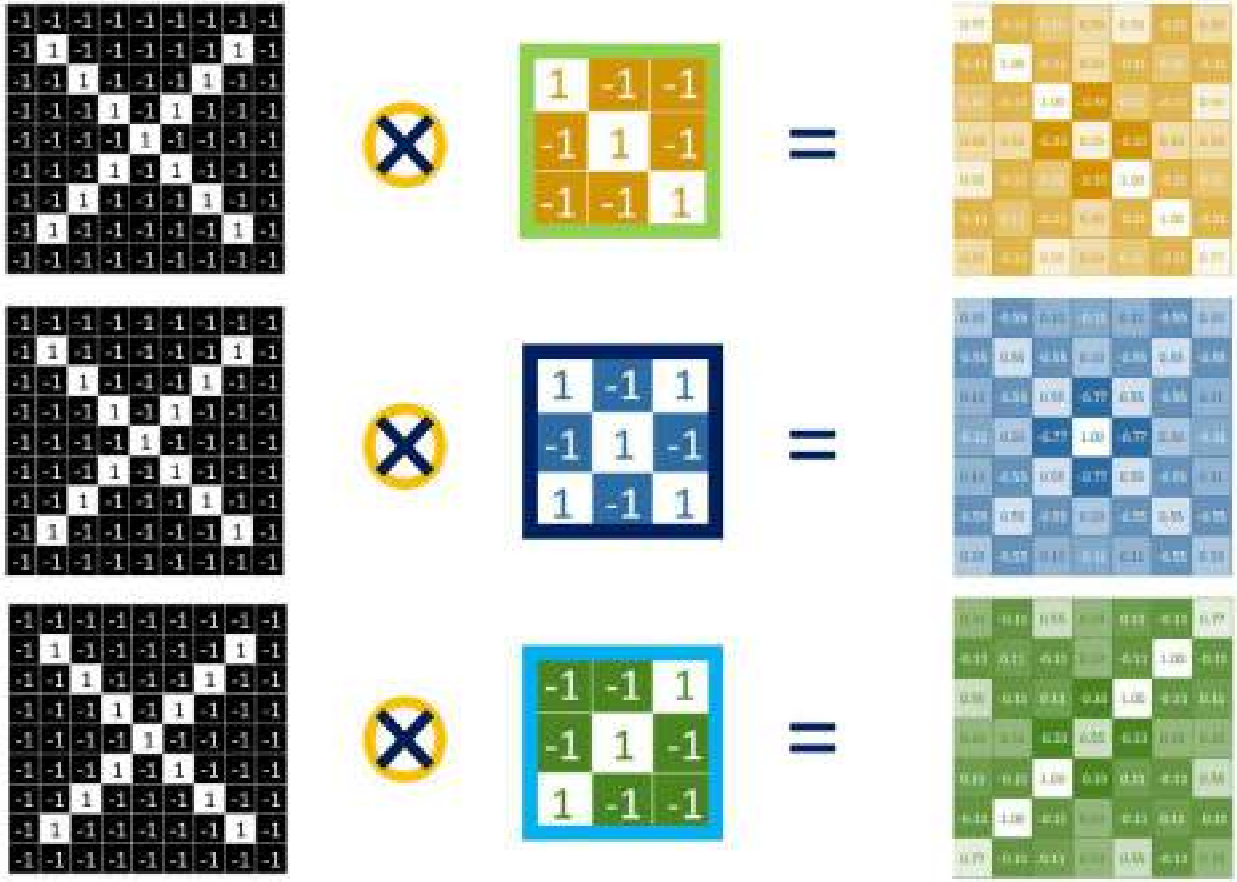


$$\frac{1+1-1+1+1+1-1+1+1}{9} = \boxed{.55}$$

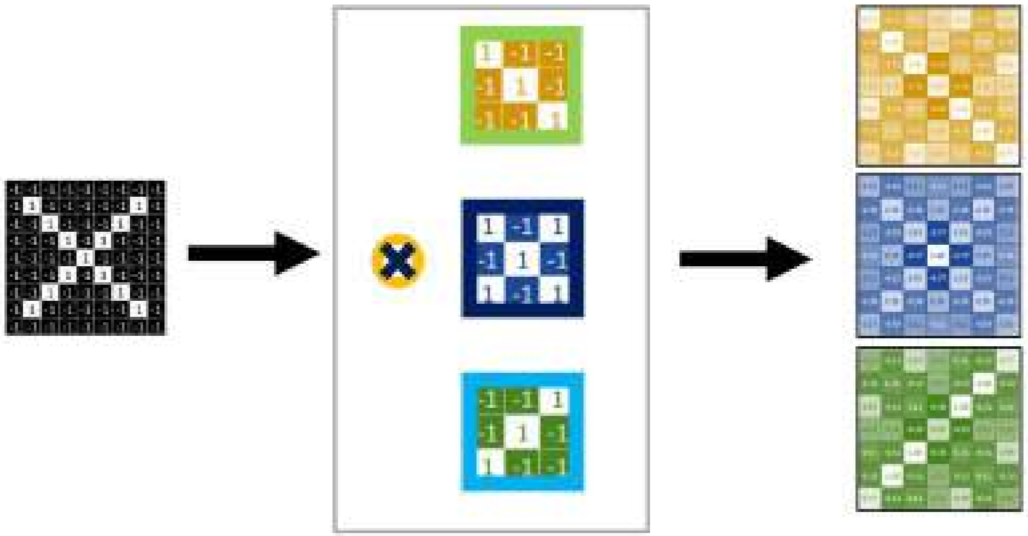
Convolution Operation

$$
\begin{bmatrix}
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\
-1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\
-1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\
-1 & -1 & -1 & -1 & 1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 \\
-1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\
-1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1
\end{bmatrix}
\otimes
\begin{bmatrix}
1 & -1 & -1 \\
-1 & 1 & -1 \\
-1 & -1 & 1
\end{bmatrix}
=
\begin{bmatrix}
0.77 & -0.11 & 0.11 & 0.33 & 0.55 & -0.11 & 0.33 \\
-0.11 & 1.00 & -0.11 & 0.33 & -0.11 & 0.11 & -0.11 \\
0.11 & -0.11 & 1.00 & -0.33 & 0.11 & -0.11 & 0.55 \\
0.33 & 0.33 & -0.33 & 0.55 & -0.33 & 0.33 & 0.33 \\
0.55 & -0.11 & 0.11 & -0.33 & 1.00 & -0.11 & 0.11 \\
-0.11 & 0.11 & -0.11 & 0.33 & -0.11 & 1.00 & -0.11 \\
0.33 & -0.11 & 0.55 & 0.33 & 0.11 & -0.11 & 0.77
\end{bmatrix}
$$

Convolution Operation



Convolutional layer



Convolutional layer

Feature Maps on an Image

Original Image

Sharpen

Edge Detected

Kernels and their effects on an Image

Original Image

0	0	0
0	1	0
0	0	0

Identity Kernel

Output Image – Same as Original

Kernels and their effects on an Image

Original Image

0.0625	0.125	0.0625
0.125	0.25	0.125
0.0625	0.125	0.0625

Blur

Output Image

Kernels and their effects on an Image

Original Image

1	0	-1
2	0	-2
1	0	-1

Left Sobel

Output Image

Sobel kernels are used to show *only* the differences in adjacent pixel values in a particular direction

Kernels and their effects on an Image

Original Image

-1	0	1
-2	0	2
-1	0	1

Right Sobel

Output Image

Sobel kernels are used to show *only* the differences in adjacent pixel values in a particular direction

Kernels and their effects on an Image

-1	-2	-1
0	0	0
1	2	1

Bottom Sobel

Original Image

Output Image

Sobel kernels are used to show *only* the differences in adjacent pixel values in a particular direction

Kernels and their effects on an Image

Original Image

1	2	1
0	0	0
-1	-2	-1

Top Sobel

Output Image

Sobel kernels are used to show *only* the differences in adjacent pixel values in a particular direction

Kernels and their effects on an Image

Original Image

-2	-1	0
-1	1	1
0	1	2

Emboss

Output Image

The **emboss** kernel (similar to the **Sobel** kernel and sometimes referred to mean the same) givens the illusion of depth by emphasizing the differences of pixels in a given direction. In this case, in a direction along a line from the top left to the bottom right.

Kernels and their effects on an Image

-1	-1	-1
-1	8	-1
-1	-1	-1

Outline

Original Image

Output Image

An **outline** kernel (also called an "edge" kernel) is used to highlight large differences in pixel values. A pixel next to neighbor pixels with close to the same intensity will appear black in the new image while one next to neighbor pixels that differ strongly will appear white.

Kernels and their effects on an Image

Original Image

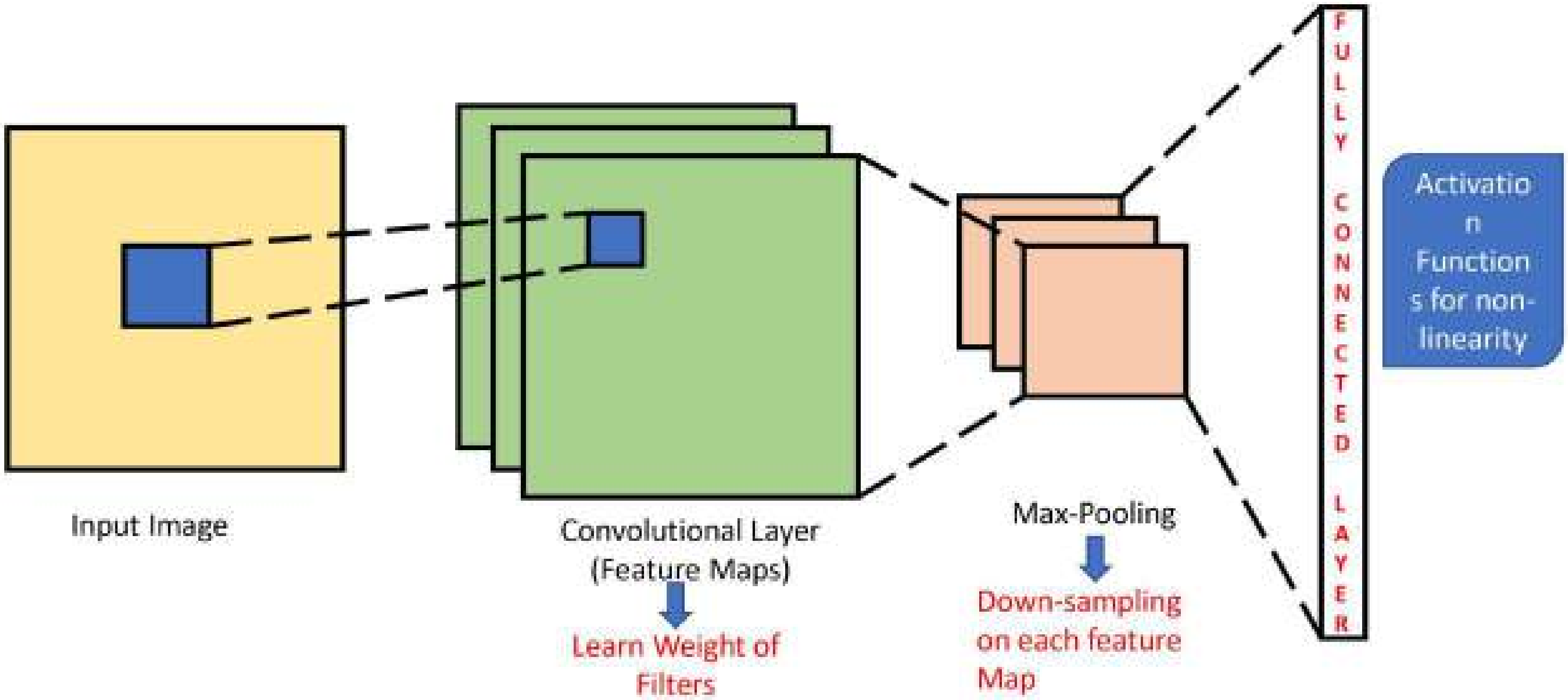
0	-1	0
-1	5	-1
0	-1	0

Sharpen

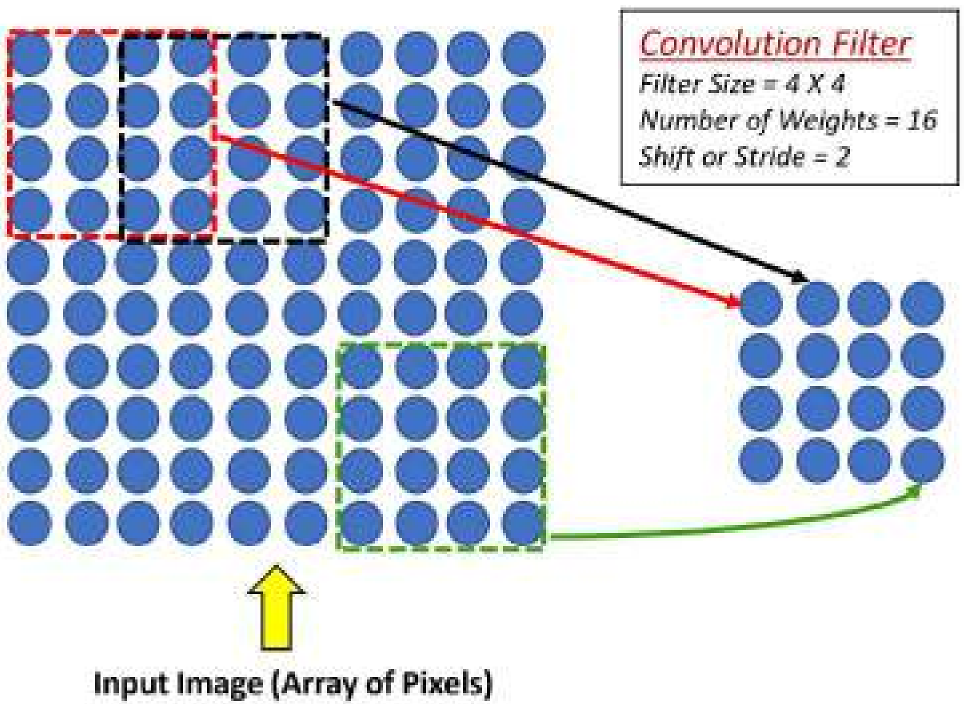
Output Image

The **sharpen** kernel emphasizes differences in adjacent pixel values. This makes the image look more vivid.

CONVOLUTIONAL NEURAL NETWORK--- CLASSIFICATION



Feature Extraction using Spatial Structure



Connect Input Layer patches to neurons of hidden layer/subsequent layer with sliding window approach.

Step 1:

Take inputs from the window (set of weights)

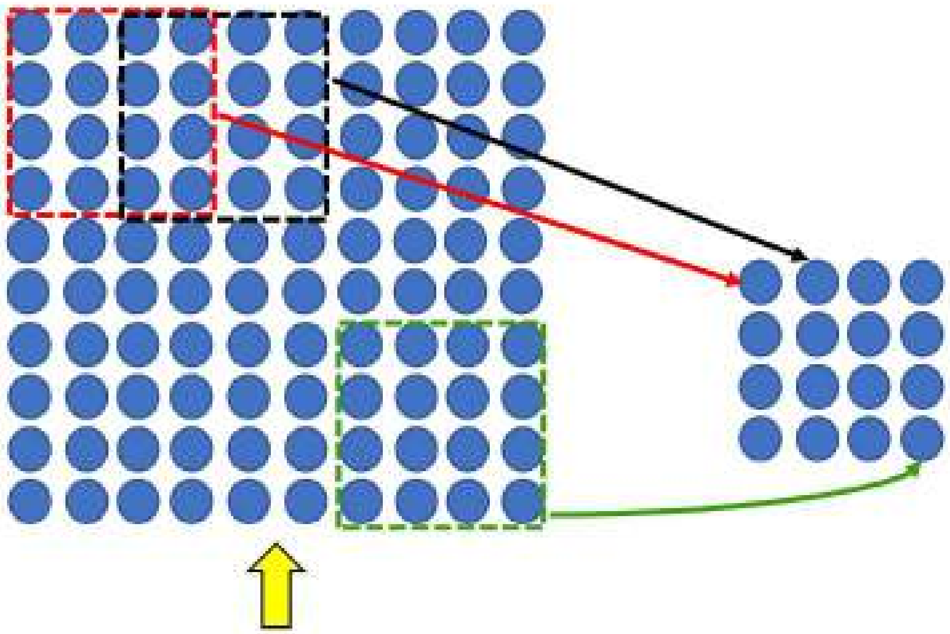
Step 2:

Calculation of weighted sum

Step 3:

Apply Bias

Feature Extraction using Spatial Structure



Input Image (Array of Pixels)

Step 1:

Take inputs from the window (set of weights)

Step 2:

Calculation of weighted sum

Step 3:

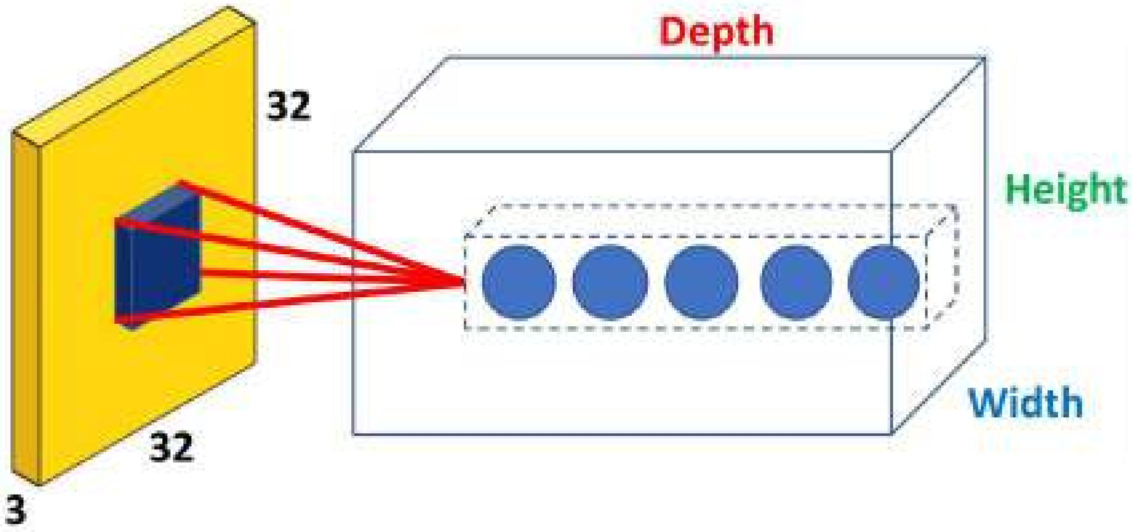
Apply Bias

For (p, q) neuron in hidden layer

$$\sum_{i=1}^{4}\sum_{j=1}^{4} w_{ij}x_{i+p,j+q} + b$$

Where, w_{ij} = weights in matrix

Convolutional Neural Network--- Spatial View



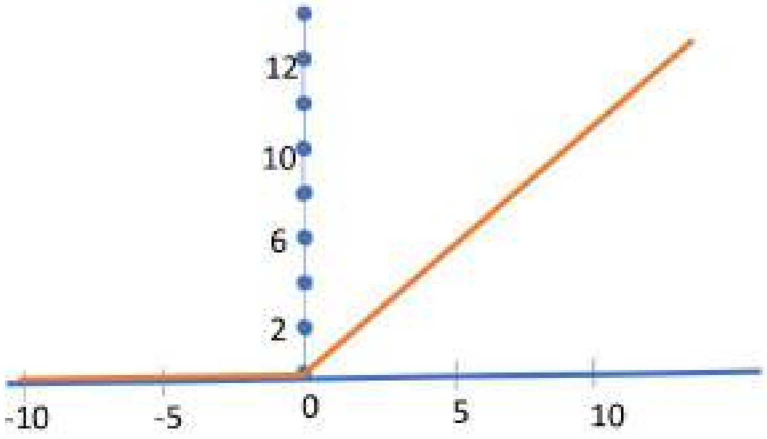
Dimensions of Layer

H * W * D

H (height) and W (width) are spatial dimensions whereas D (depth) is number of filters

Stride = Step size of filter, Receptive Field = Location of connected path in an input image

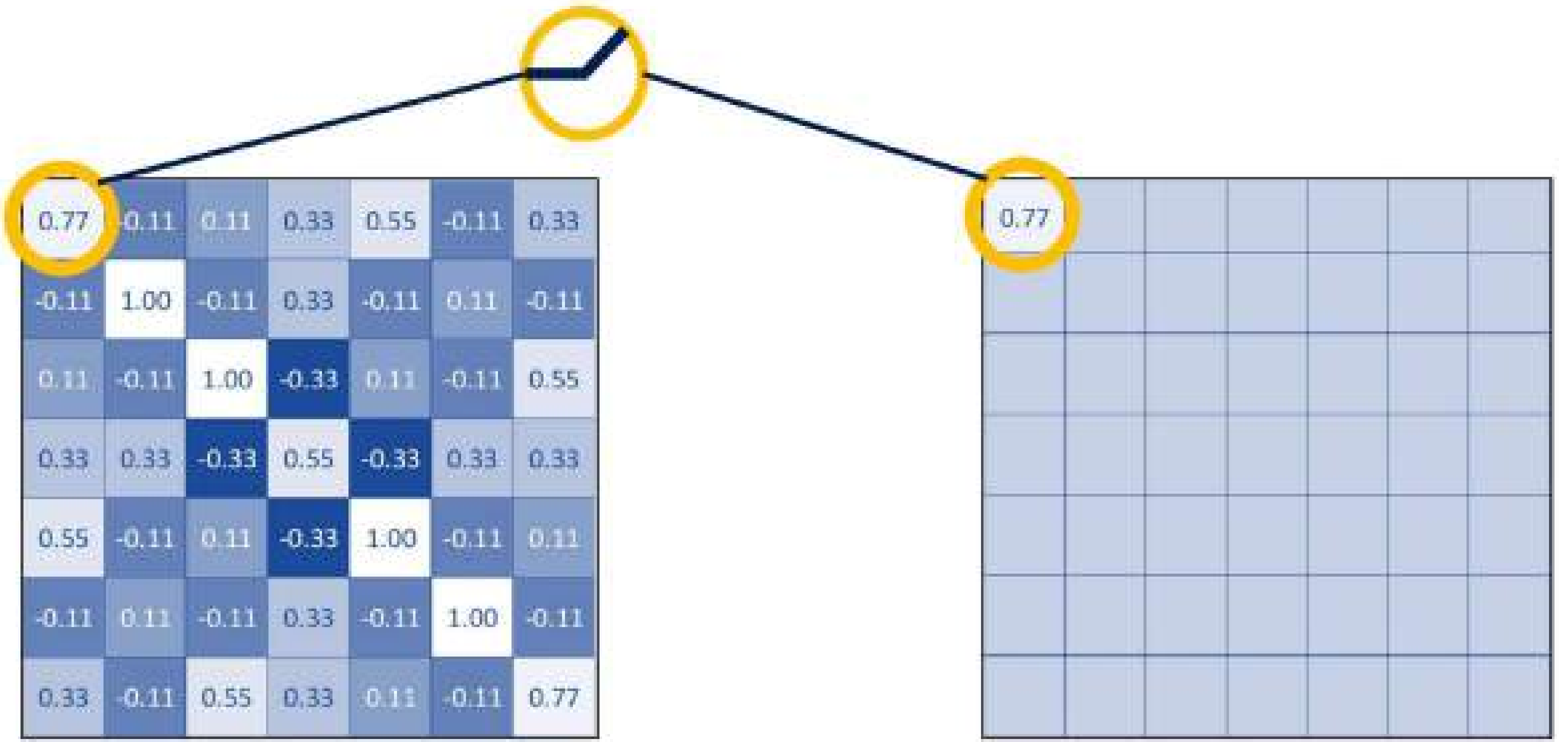
Non-Linearity in Convolutional Neural Network



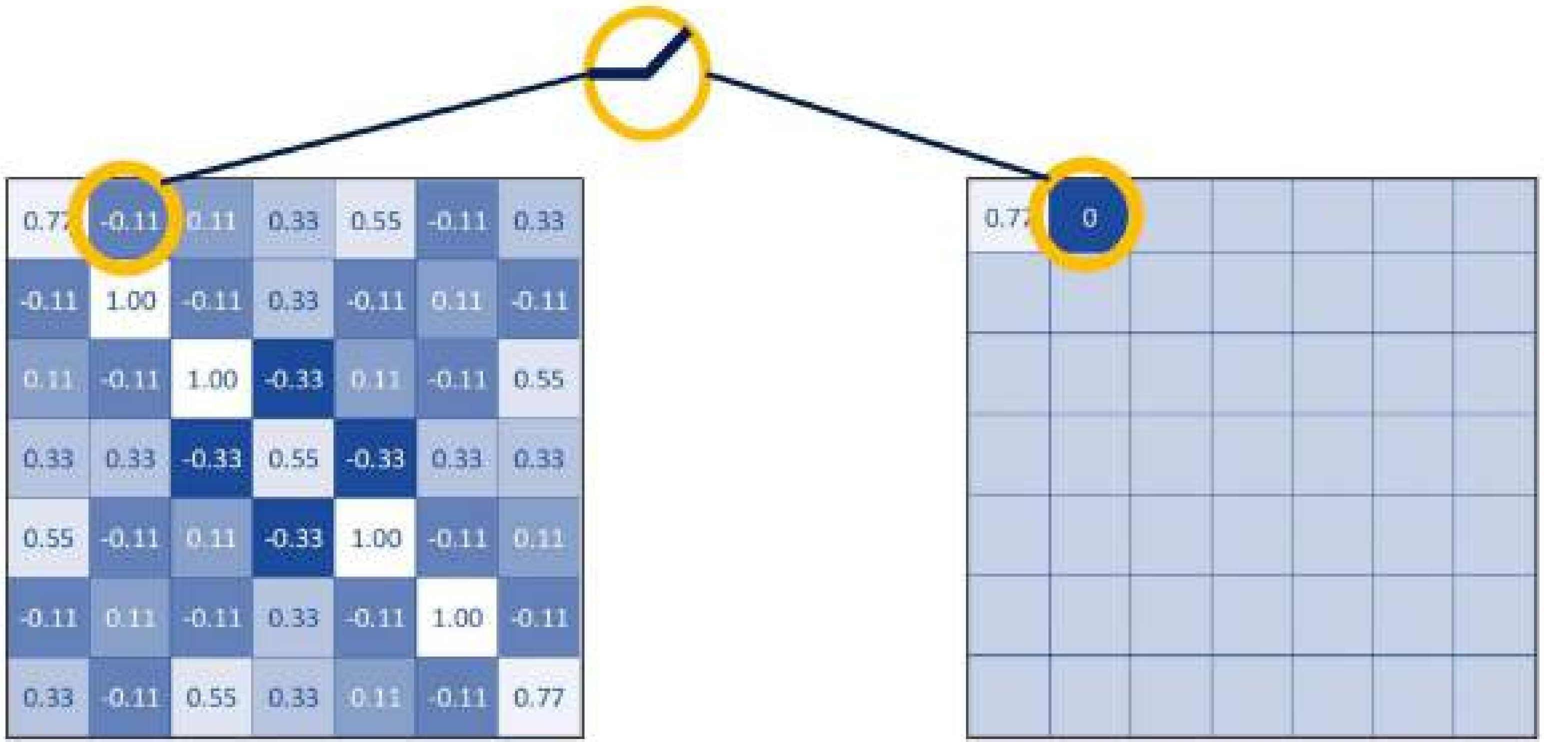
Applied after every convolutional layer. The rectified linear activation (ReLU) function is a simple calculation that returns the value provided as input directly, or the value 0.0 if the input is 0.0 or less.

$$\boxed{g(x) = \max(0, x)}$$

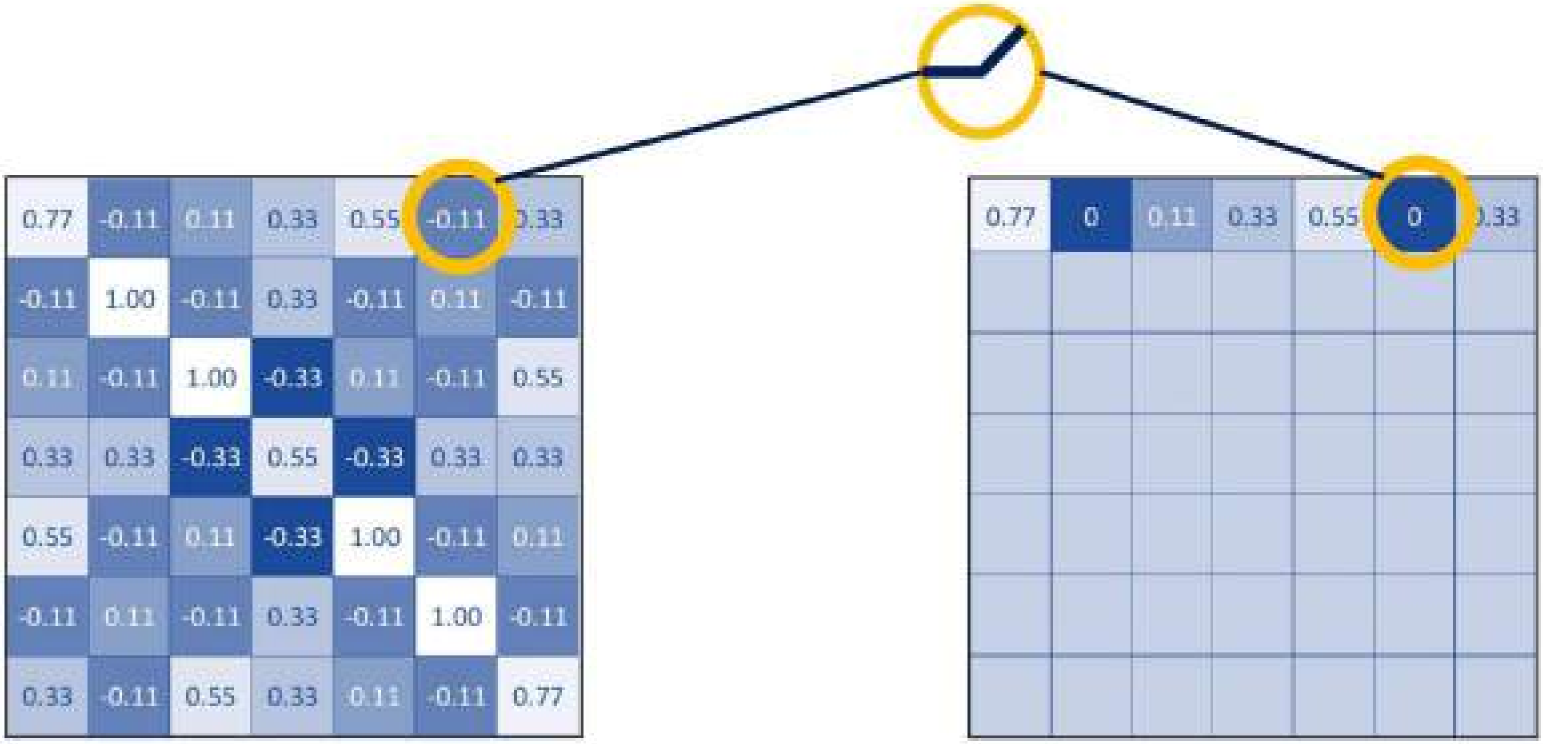
Rectified Linear Units (ReLUs)



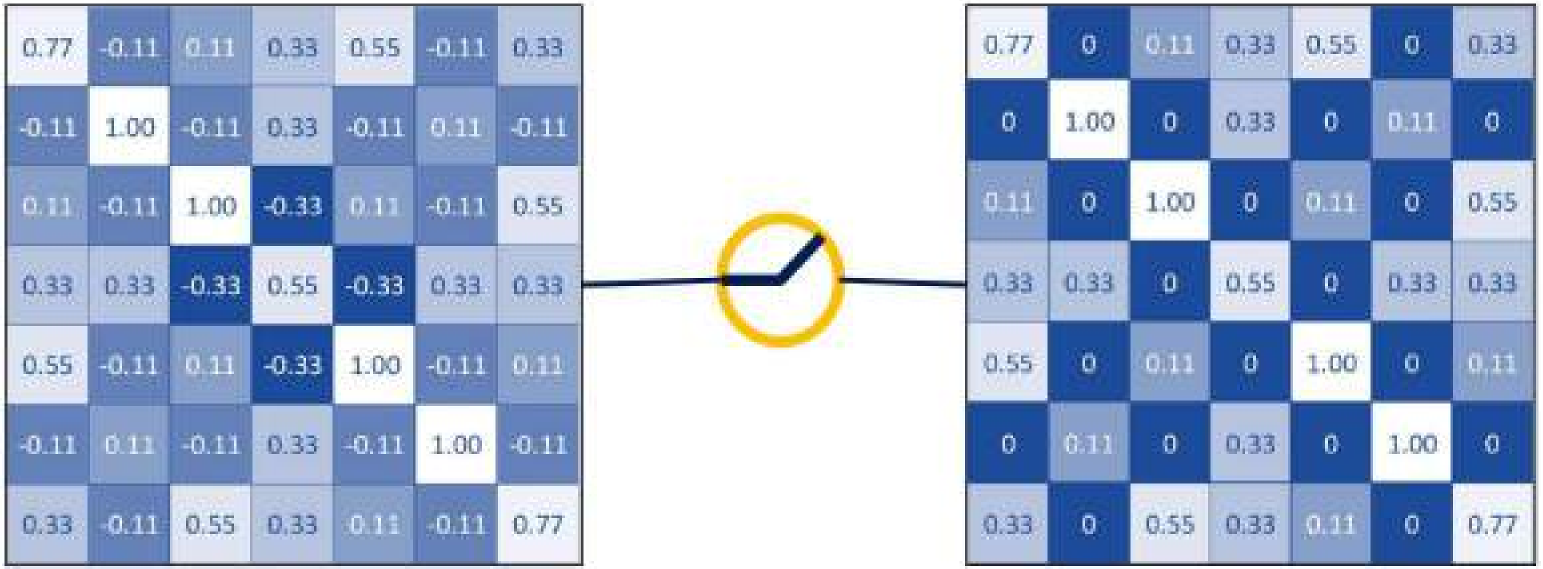
Rectified Linear Units (ReLUs)



Rectified Linear Units (ReLUs)



Rectified Linear Units (ReLUs)



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

ReLU layer

A stack of images becomes a stack of images with no negative values.

POOLING

- *Dimensionality Reduction*
- *Preserve Spatial Invariance*

STEPS

1. Pick a window size (usually 2 or 3).
2. Pick a stride (usually 2).
3. Walk your window across your filtered images.
4. From each window, take the maximum value.

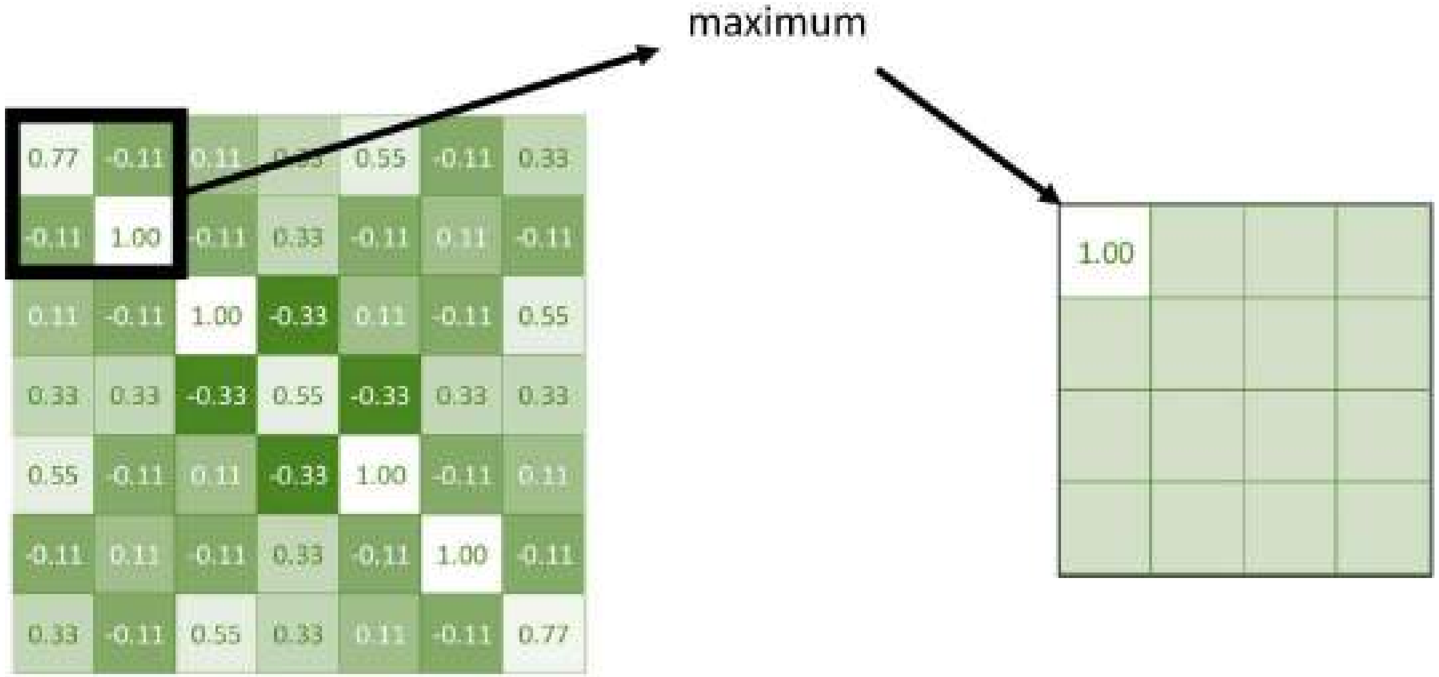
Pooling: Shrinking the Image stack

1. Pick a window size (usually 2 or 3).
2. Pick a stride (usually 2).
3. Walk your window across your filtered images.
4. From each window, take the maximum value.

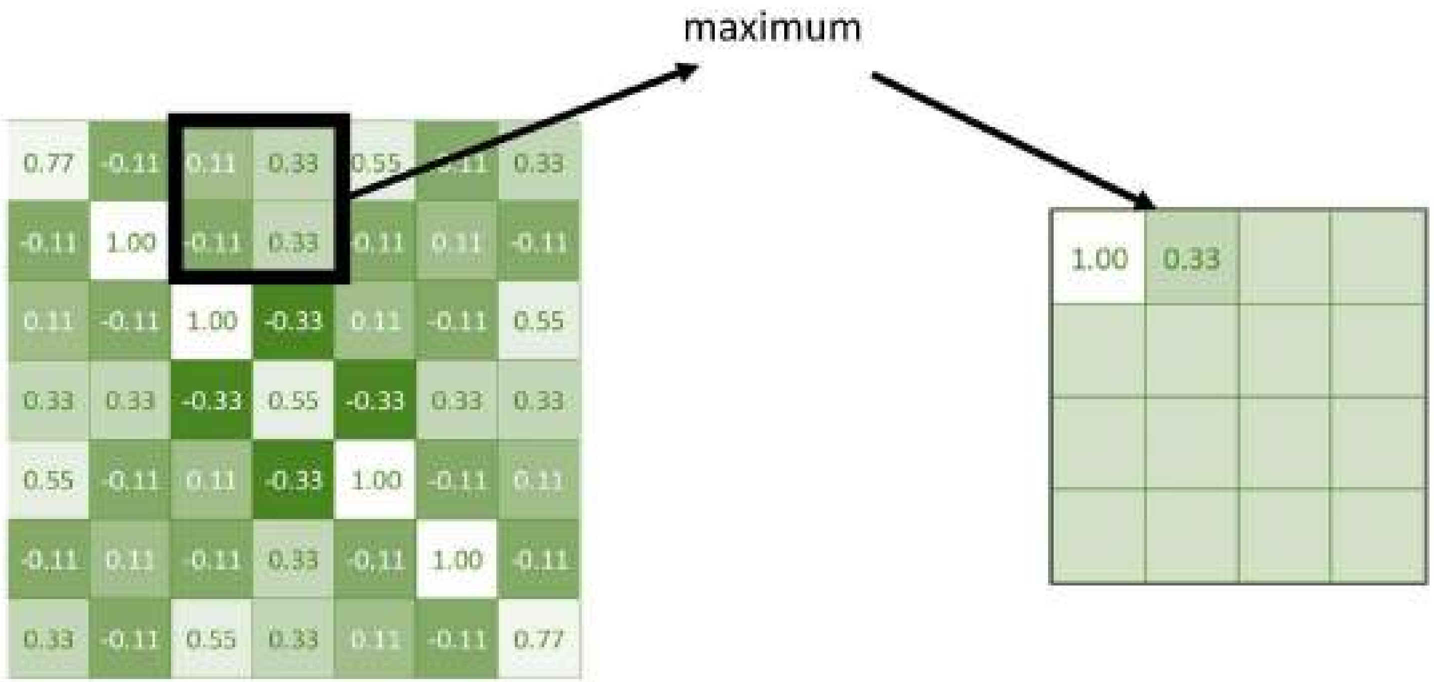
The three types of pooling operations are:

- **Max pooling:** The maximum pixel value of the batch is selected.
- **Min pooling:** The minimum pixel value of the batch is selected.
- **Average pooling:** The average value of all the pixels in the batch is selected.

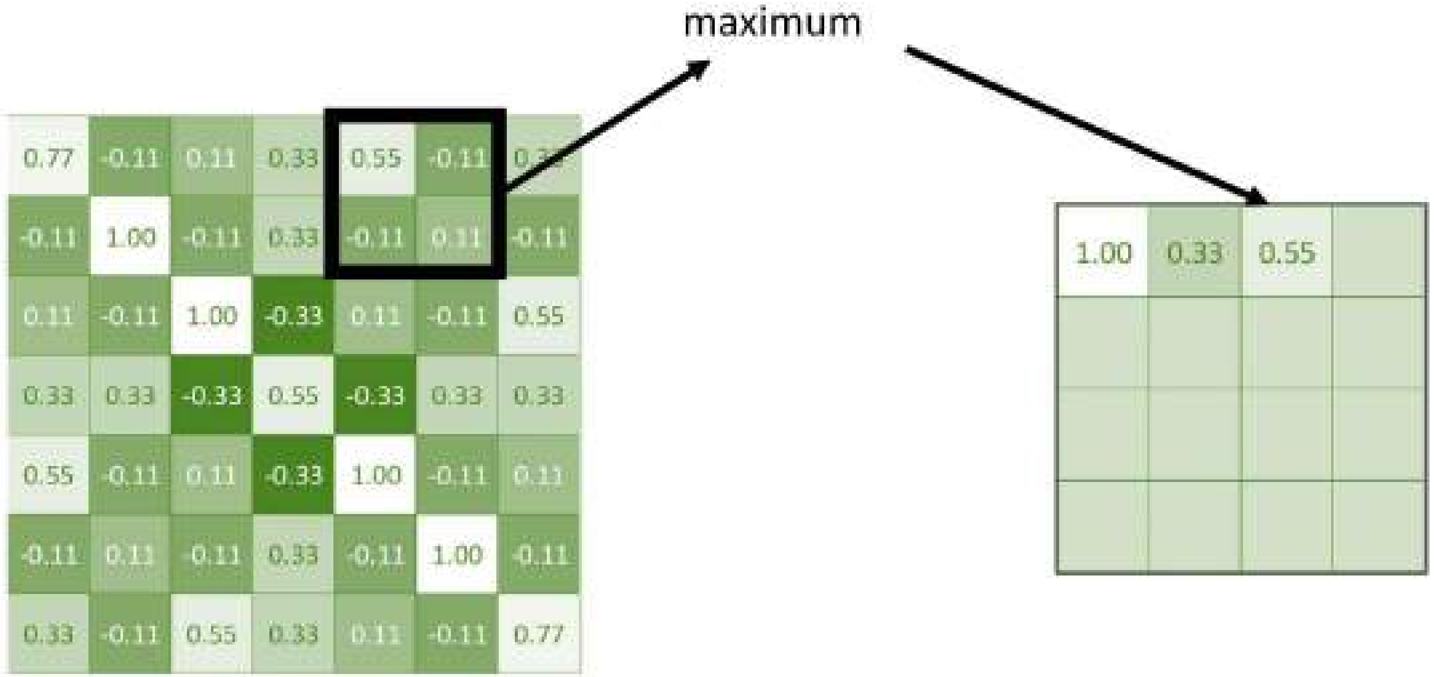
Max-Pooling


maximum
0.77 -0.11 0.11 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
1.00

Max-Pooling


maximum
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
1.00 0.33

Max-Pooling


maximum
0.77 -0.11 0.11 0.33 0.55 -0.11
-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
1.00 0.33 0.55

Max-Pooling

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

maximum

1.00	0.33	0.55	0.33

Max-Pooling

maximum

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

1.00	0.33	0.55	0.33
0.33			

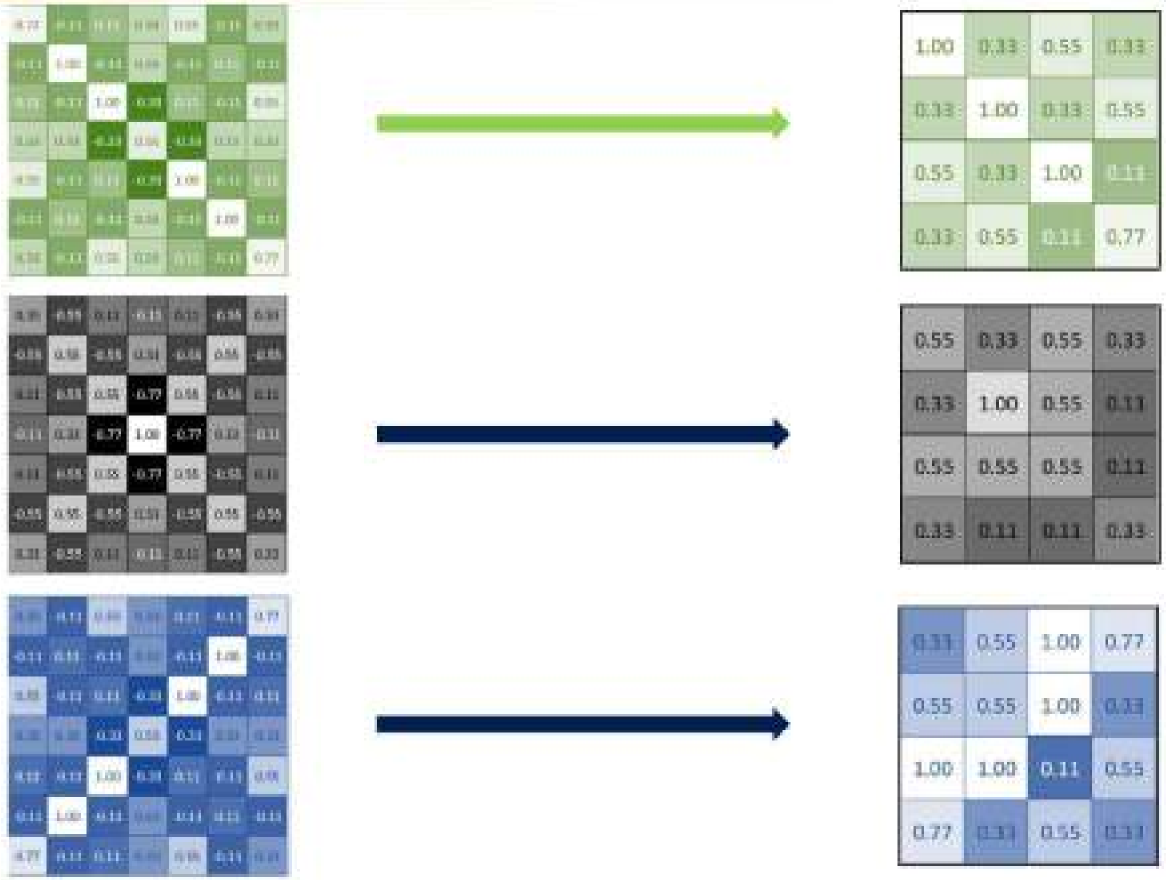
Max-Pooling

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

Max Pooling

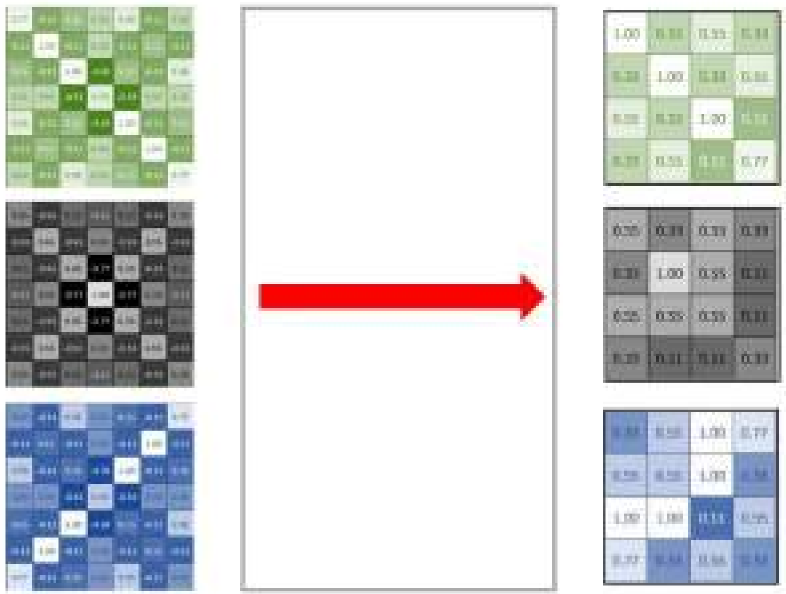
1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

Max-Pooling

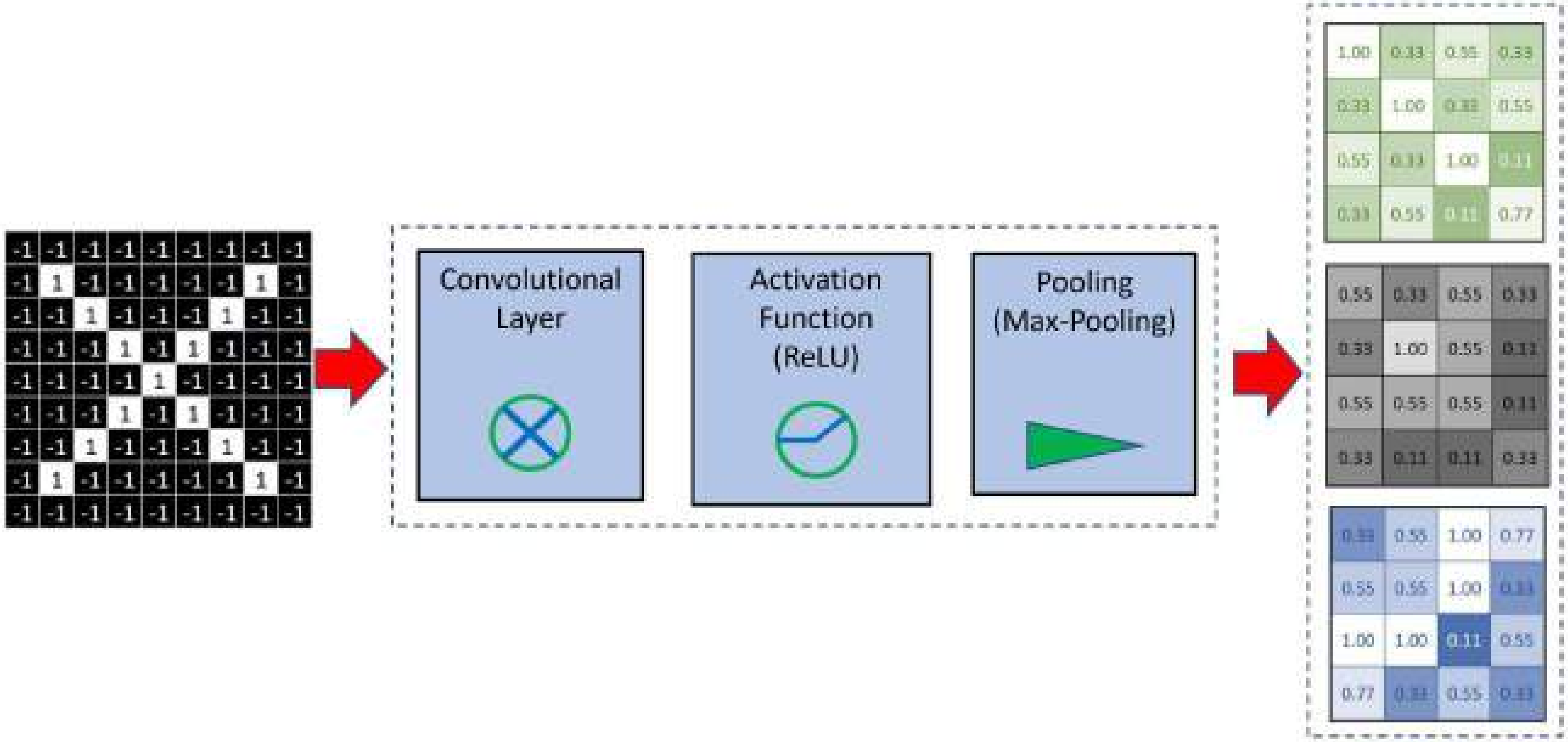

1.00 0.33 0.55 0.33
0.33 1.00 0.33 0.55
0.55 0.33 1.00 0.11
0.33 0.55 0.11 0.77
0.55 0.33 0.55 0.33
0.33 1.00 0.55 0.11
0.55 0.55 0.55 0.11
0.33 0.11 0.11 0.33
0.33 0.55 1.00 0.77
0.55 0.55 1.00 0.33
1.00 1.00 0.11 0.55
0.77 0.33 0.55 0.33

Pooling layer

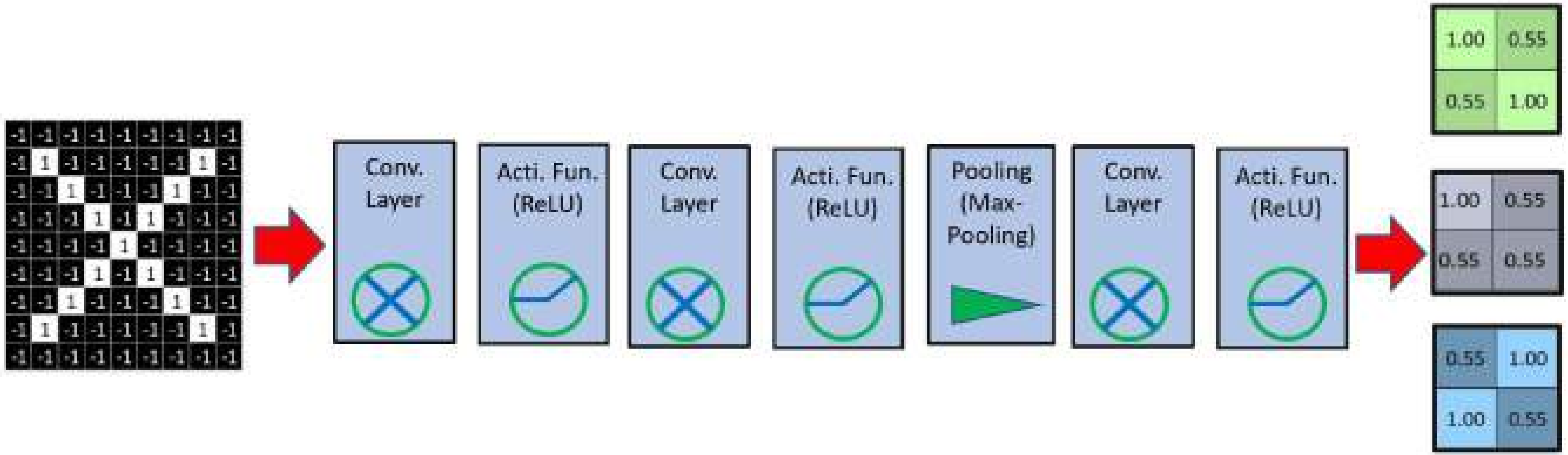
A stack of images becomes a stack of smaller images.



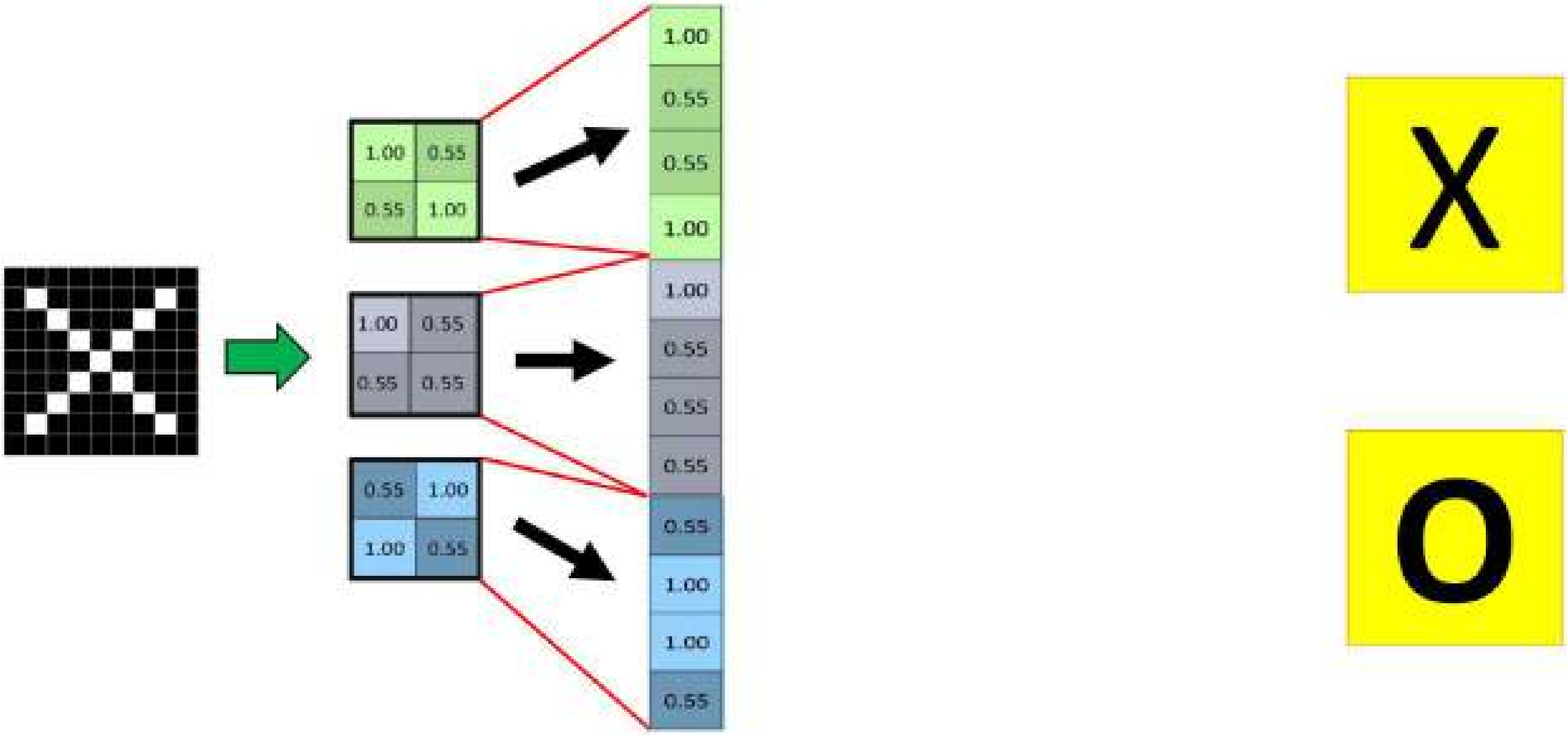
Stacking of Layers



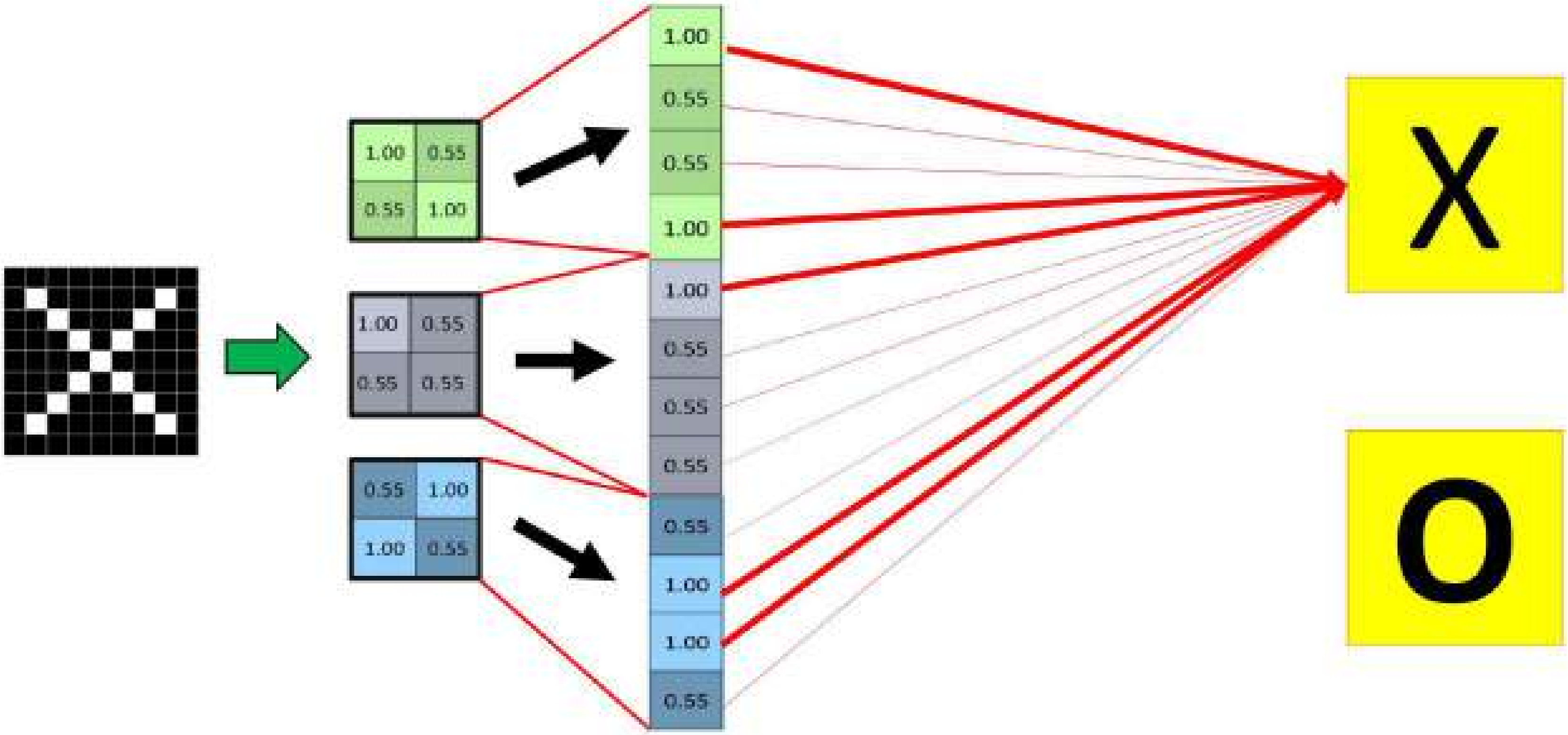
Multiple Stacking of Layers



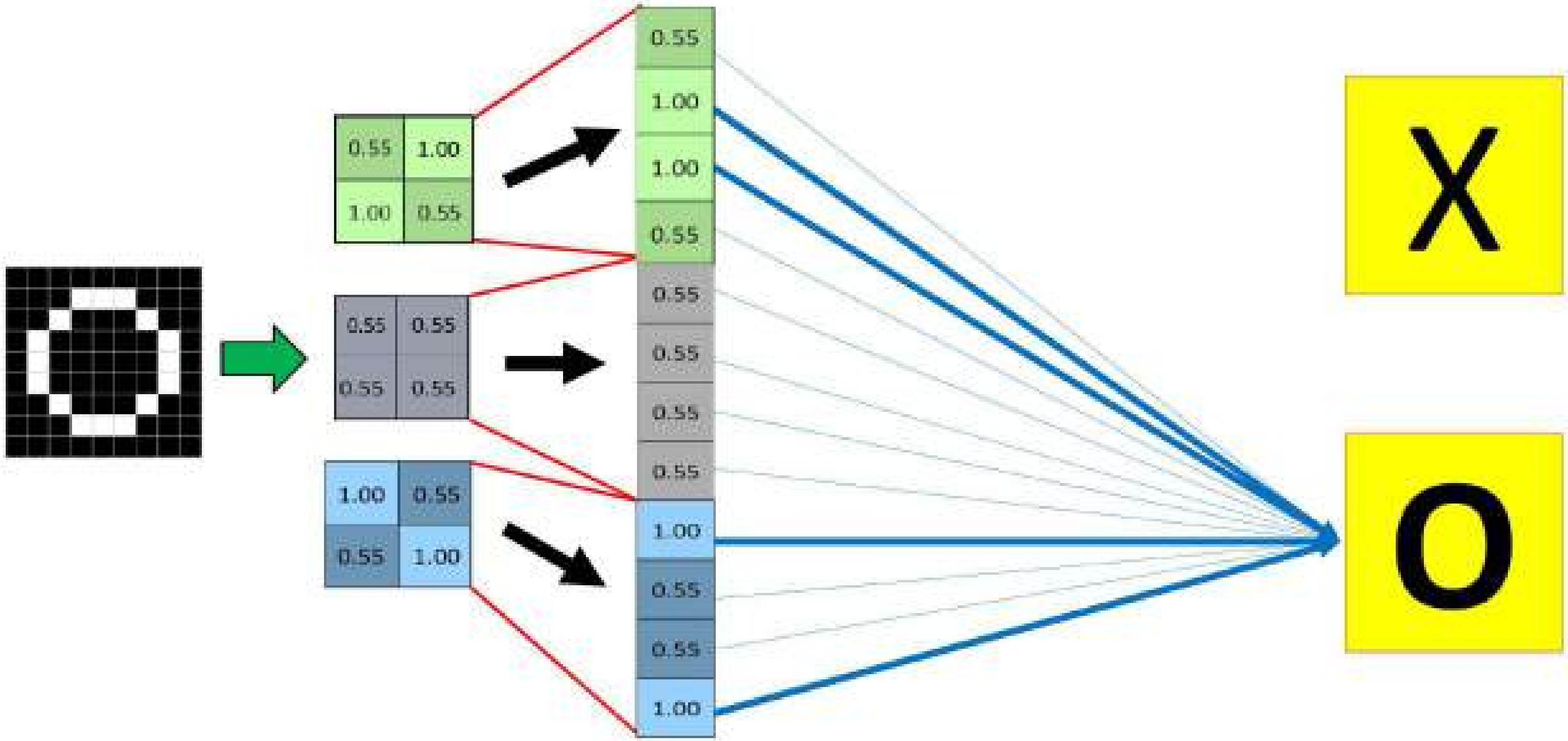
Fully Connected Layer (Training Phase)



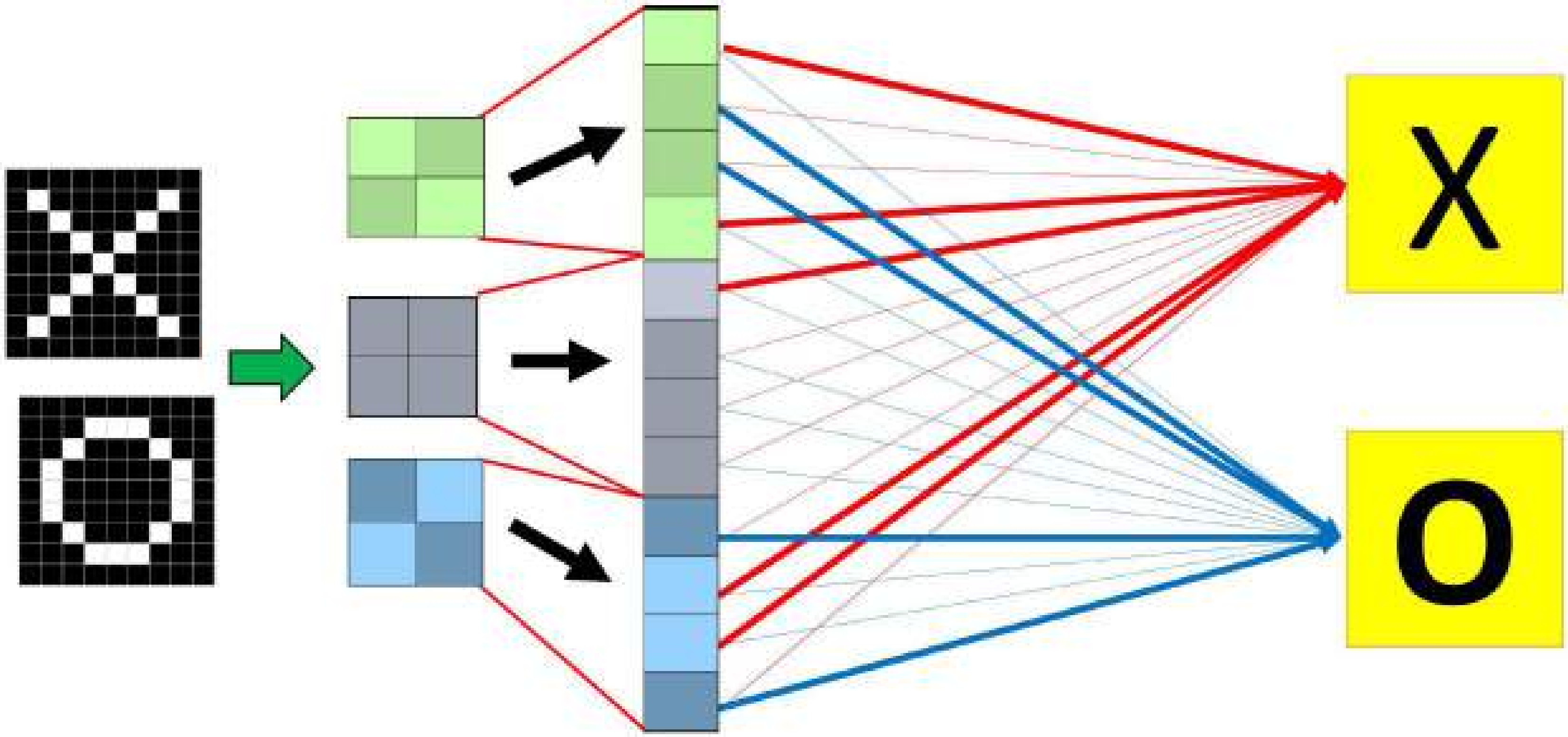
Fully Connected Layer (Training Phase)



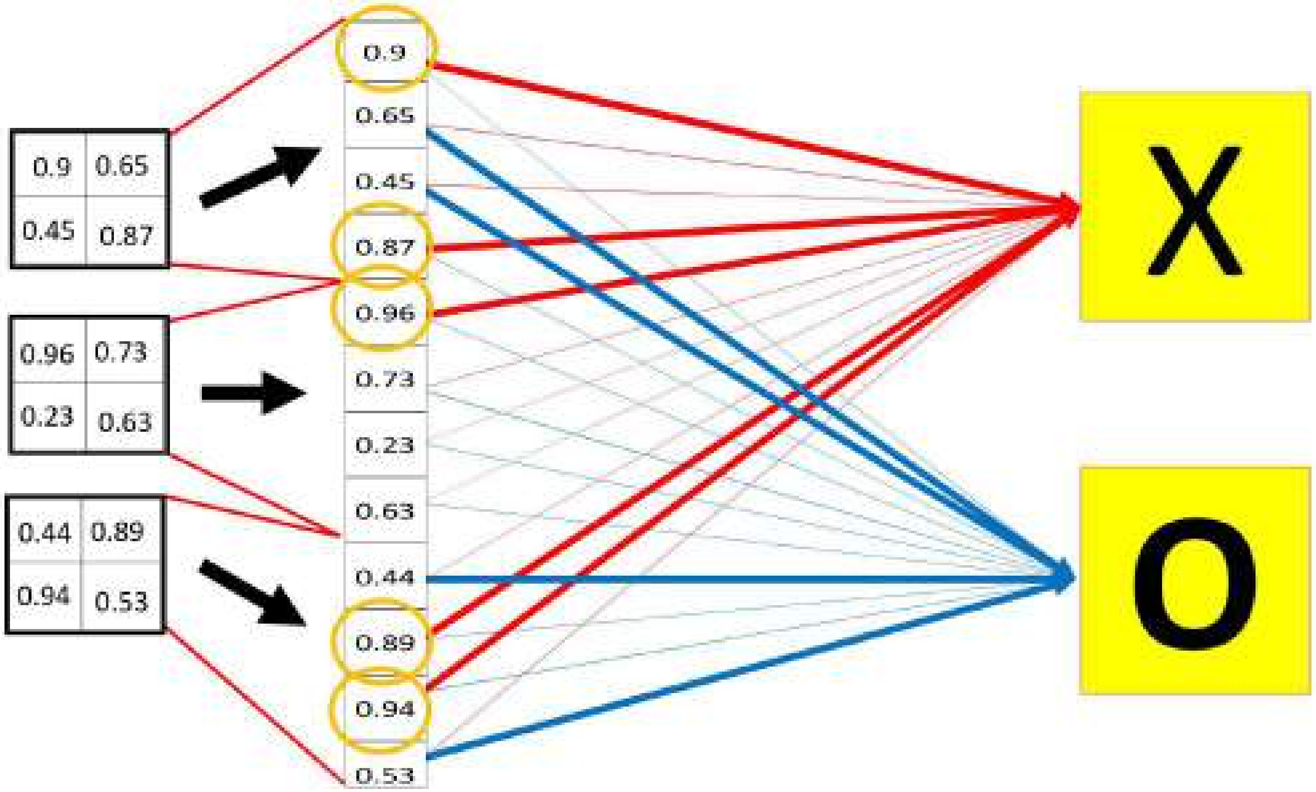
Fully Connected Layer (Training Phase)



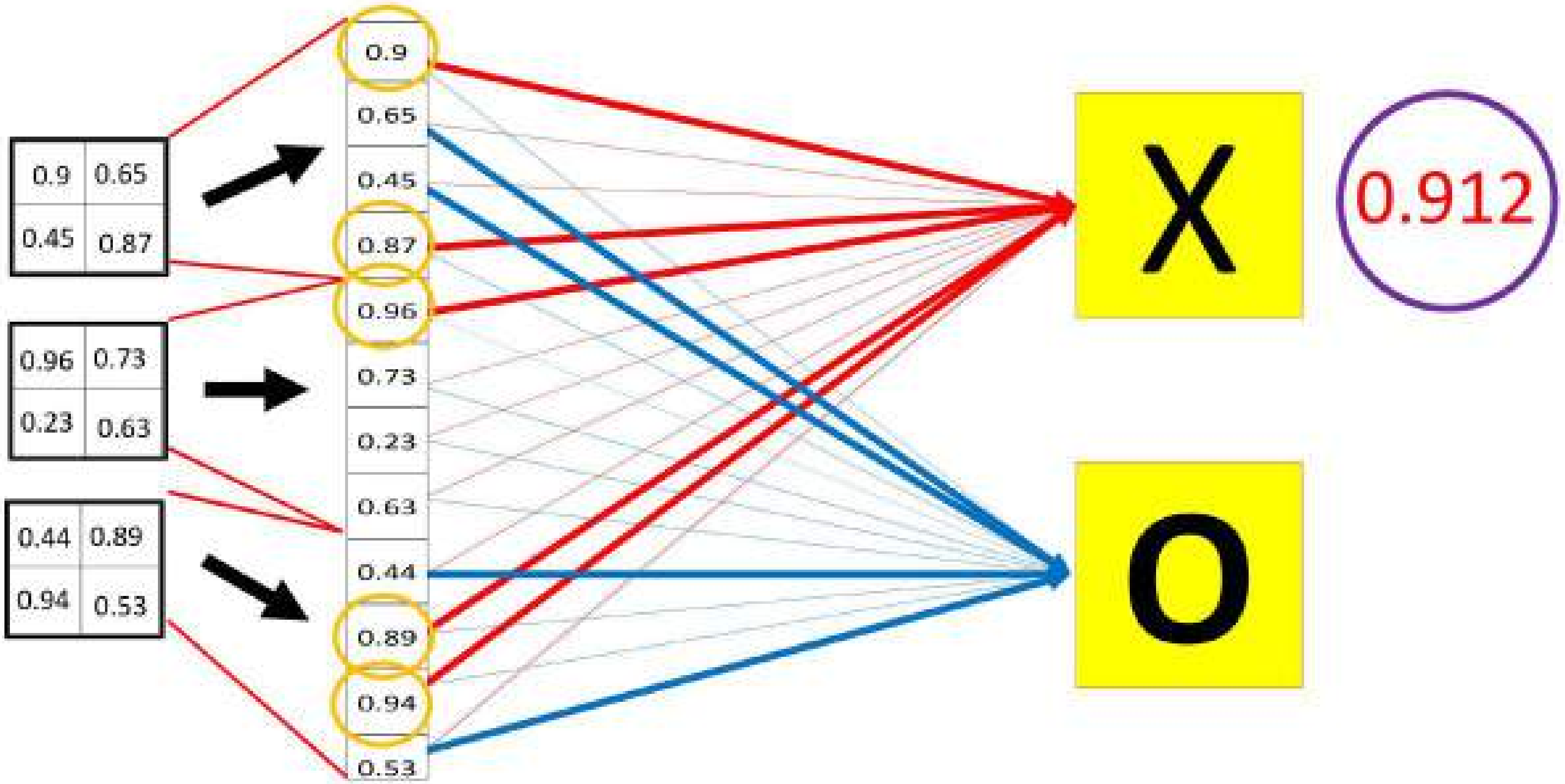
Fully Connected Layer (Training Phase)



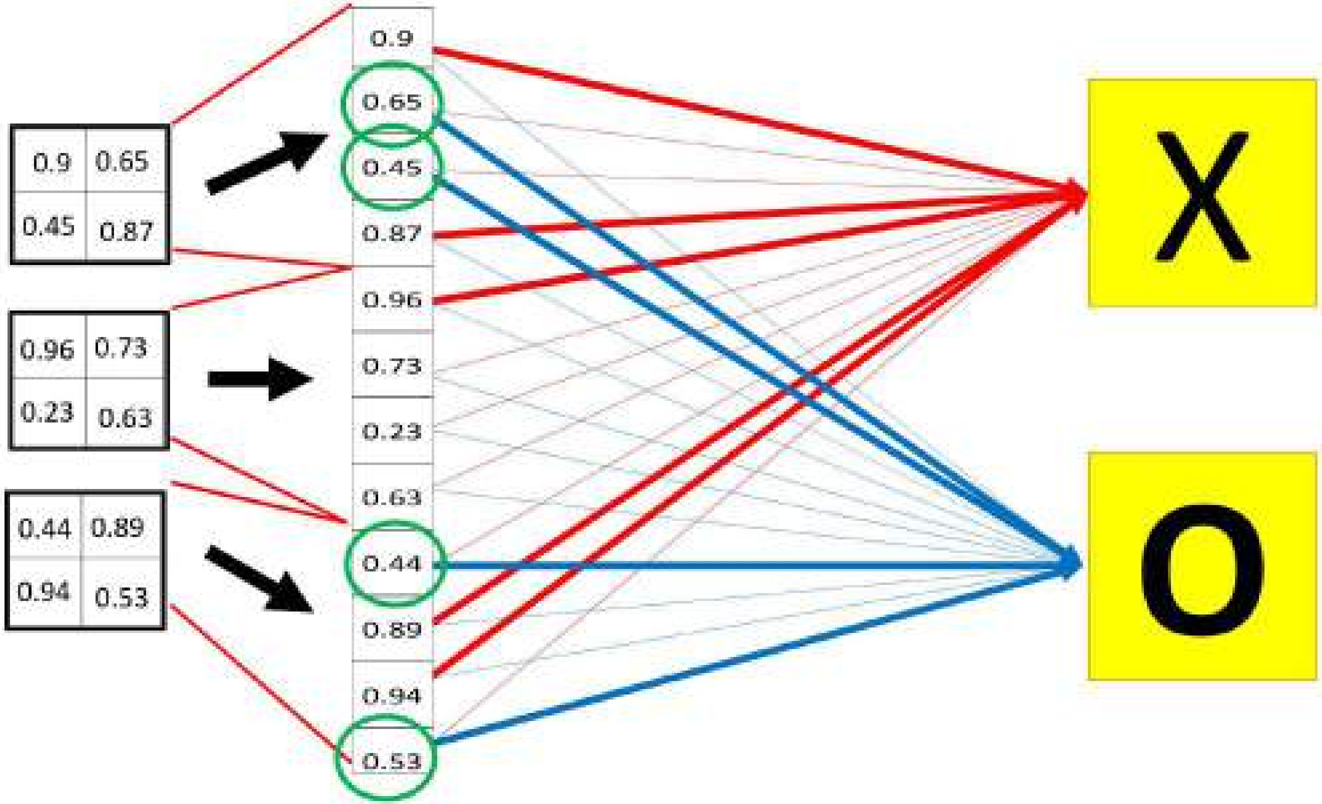
Fully Connected Layer (Testing Phase)



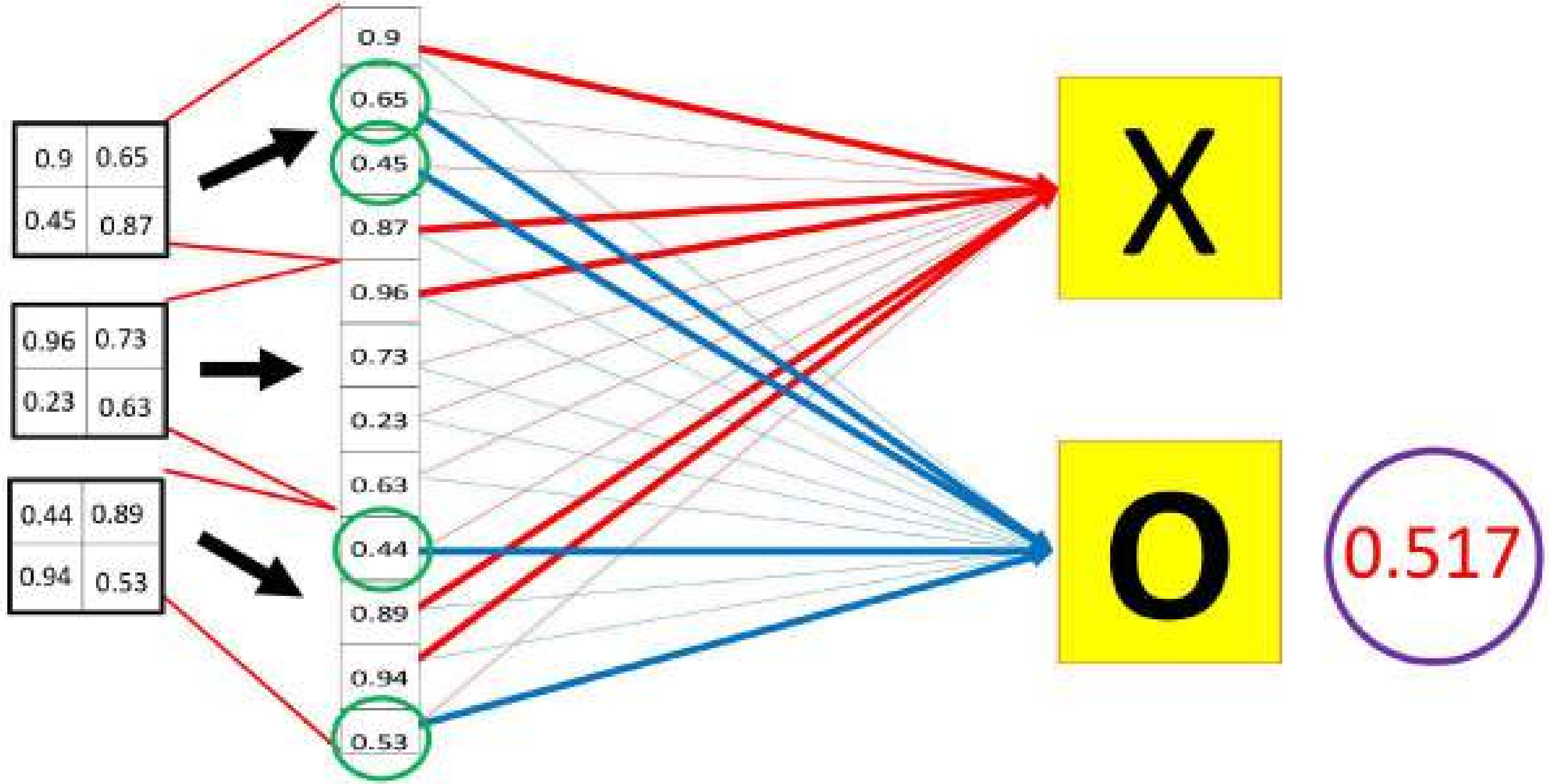
Fully Connected Layer (Testing Phase)



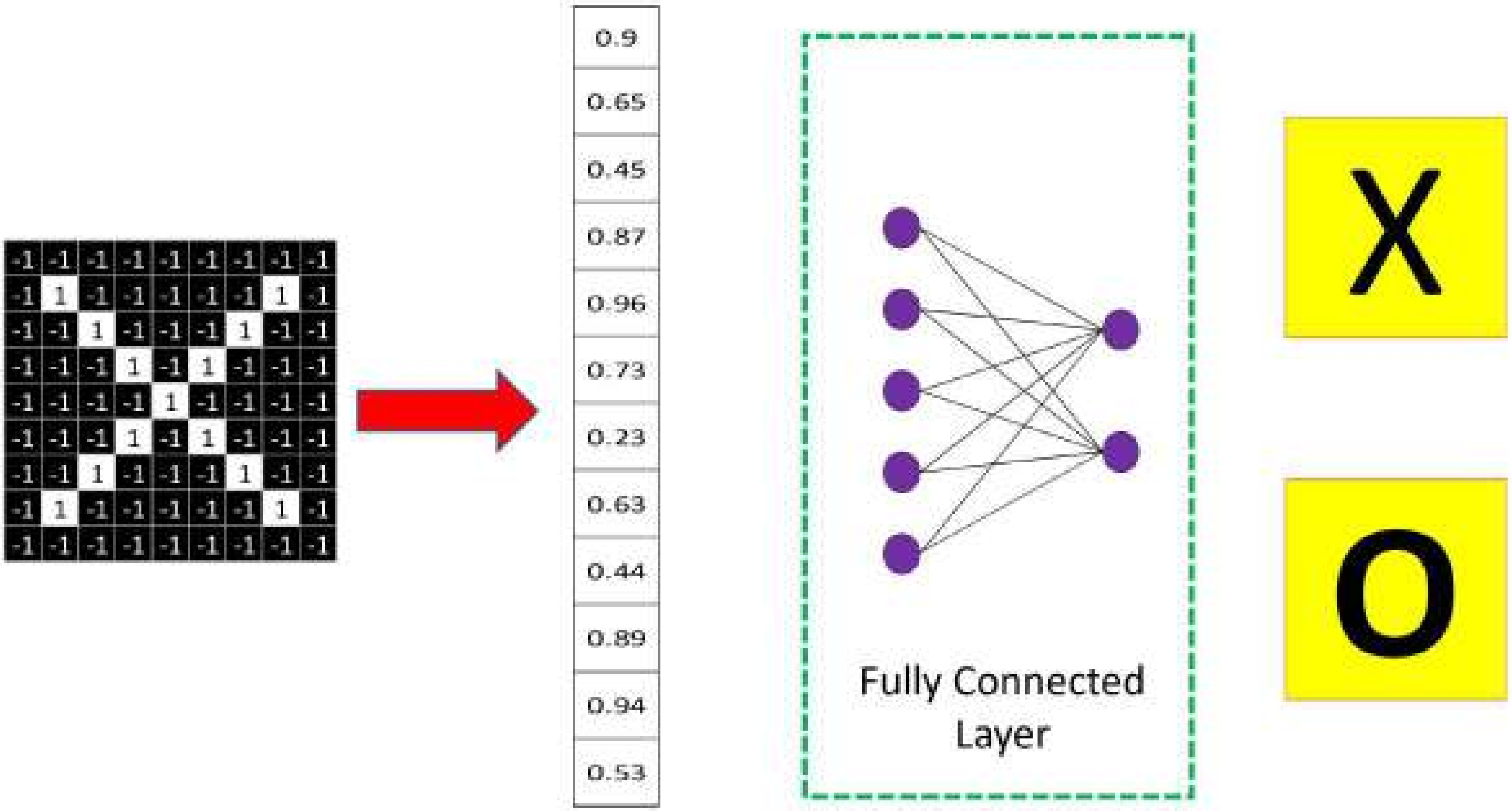
Fully Connected Layer (Testing Phase)



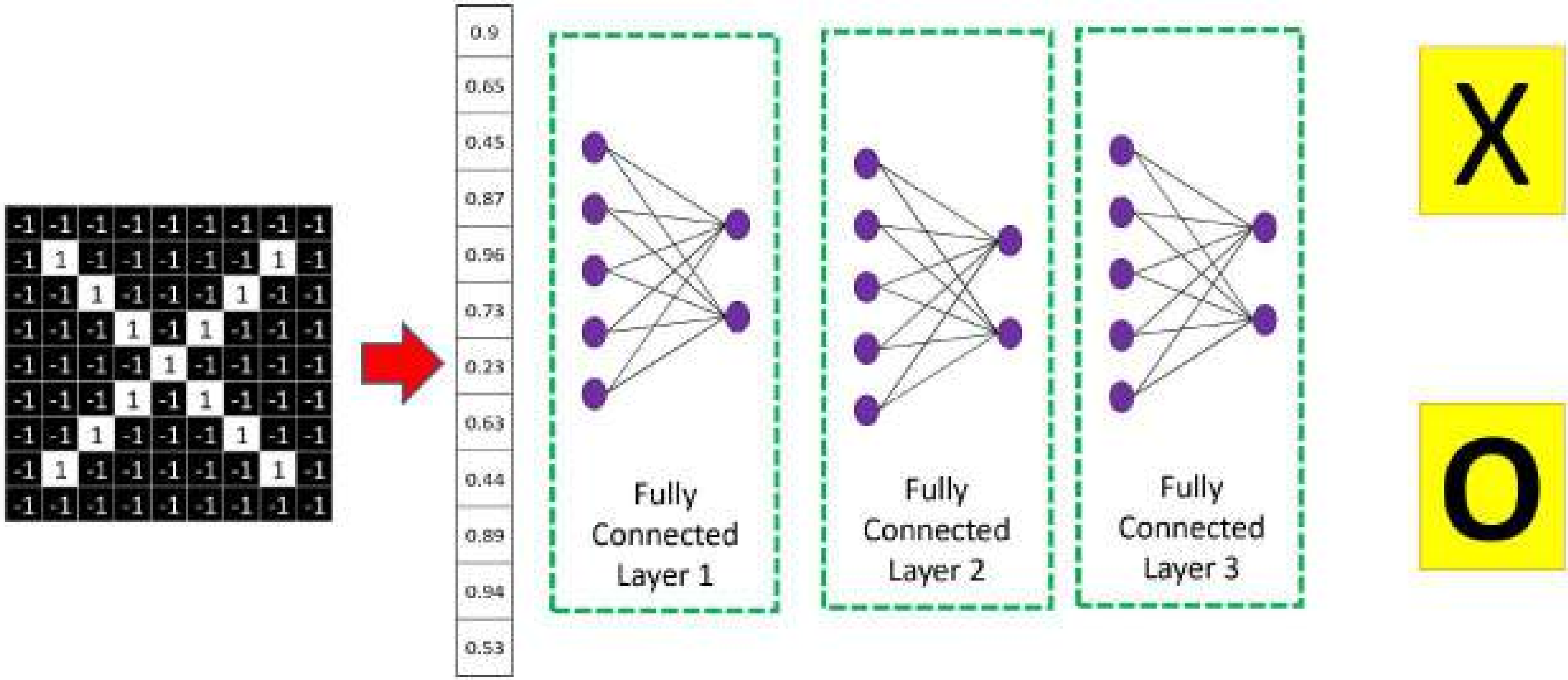
Fully Connected Layer (Testing Phase)



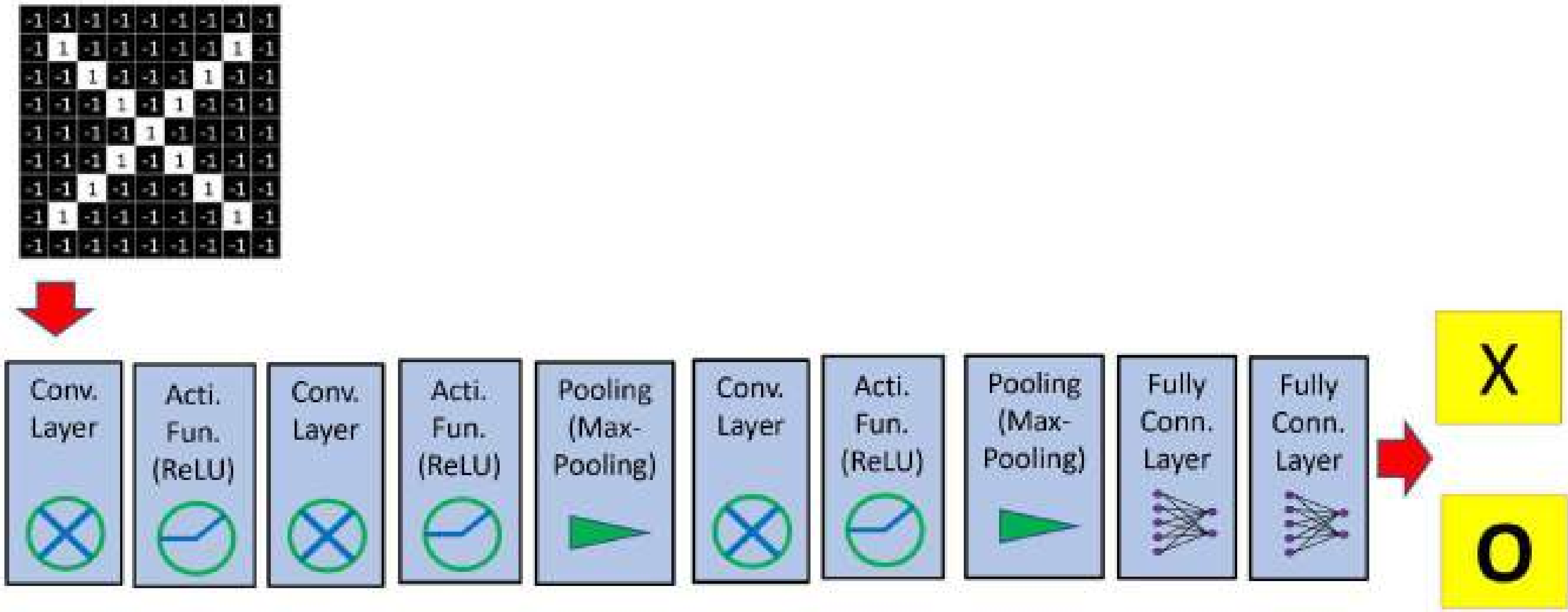
Fully Connected Layer



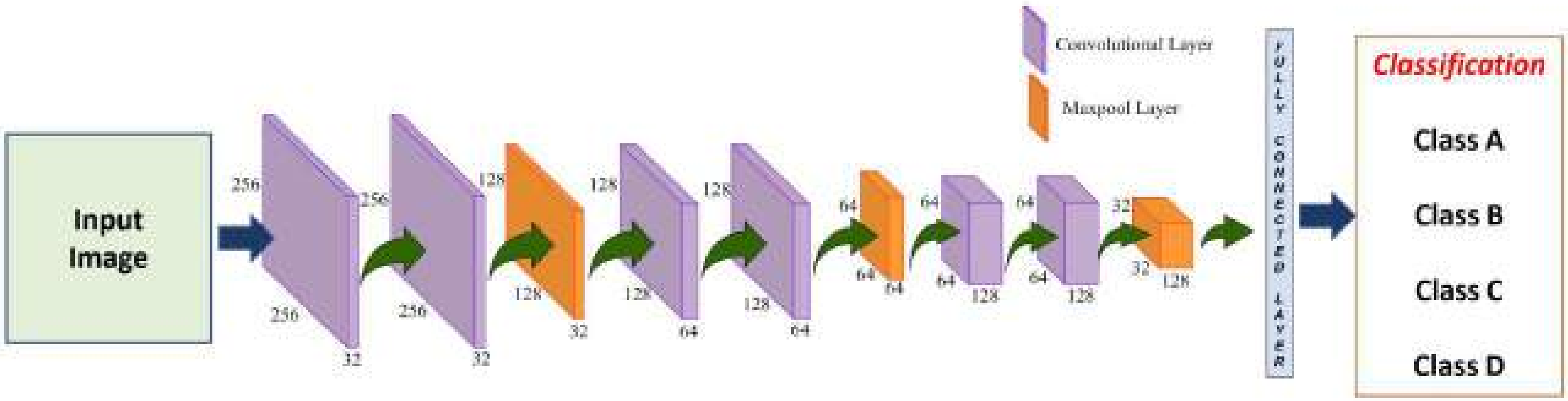
Multiple Stacking of Fully Connected Layers



Stacking of Multiple Layers



Convolutional Neural Network- Classification



- Convolutional layer and Pooling help to extract high level features of input
- Fully connected layer used extracted high level features for classification of input image in different classes
- Output also include the class probability of the image

Image Augmentation

To reduce the problem of overfitting, various augmentation variants can be applied to particular image of the training dataset.

Original Image

90° left rotation

90° right rotation

180° Horizontal rotation

Horizontal Flip

Vertical Flip

Increased Brightness

Addition of noise

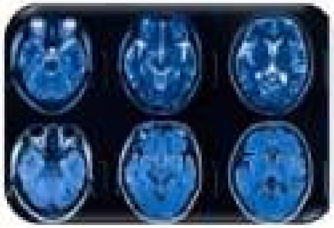
Low contrast

Background Removal

Applications of CNNs

Face Recognition

Analyzing Documents



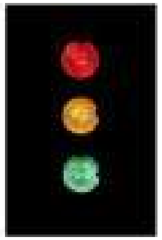
Medical Imaging

Applications of CNNs

Automatic Navigation

G.P. S.

Image Recognition and Decision making



Applications of CNNs

Image Recognition *(Prediction of objects in different classes)*