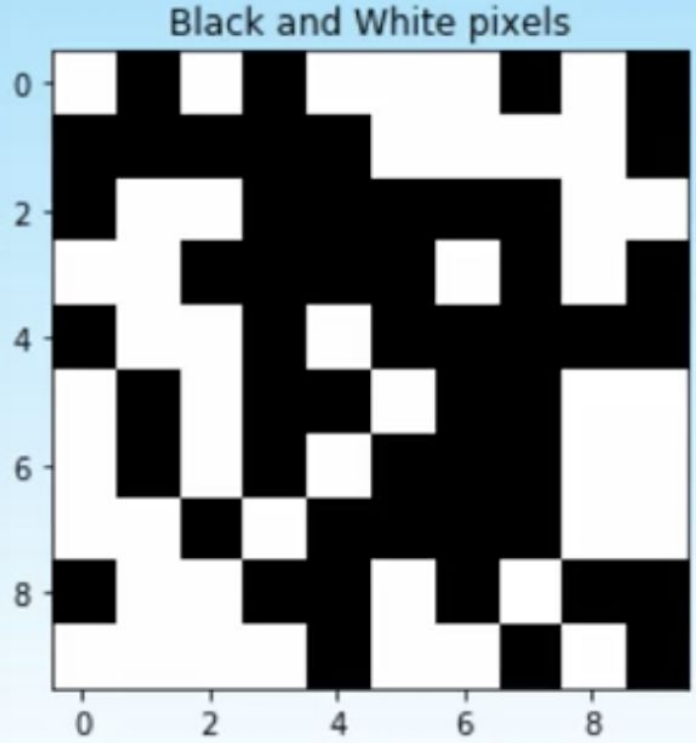
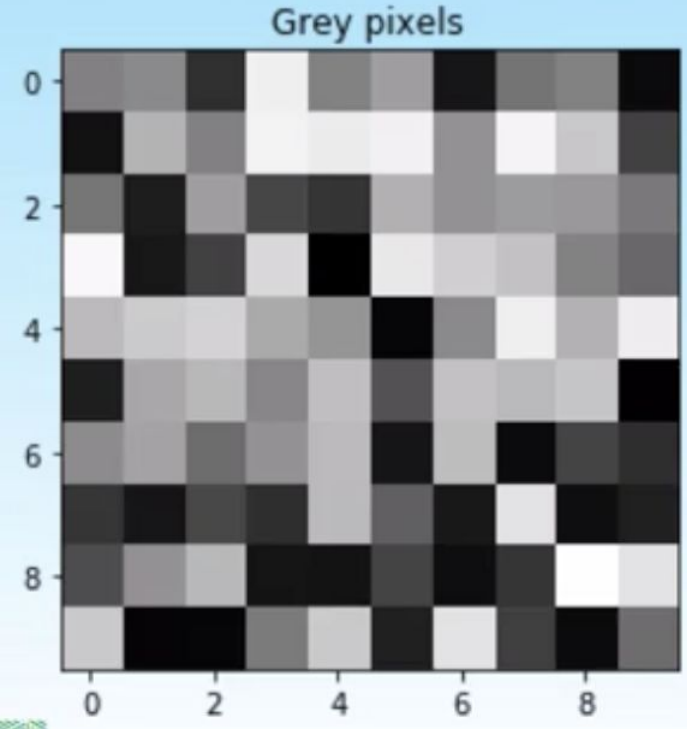
Convolutional Neural Networks

BLACK AND WHITE



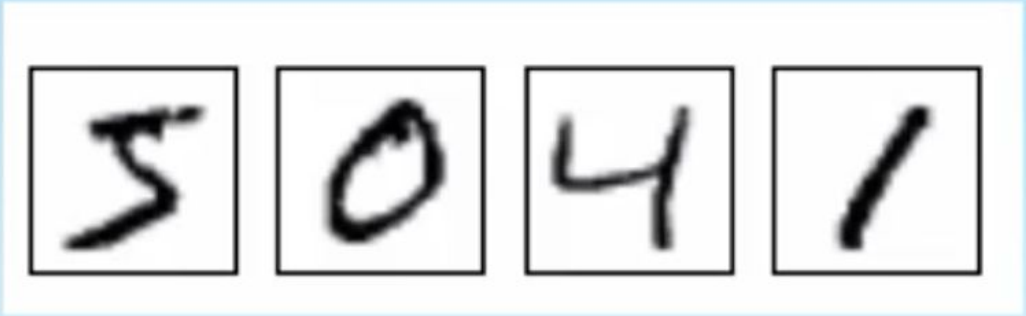

```
array([[1, 0, 1, 0, 1, 1, 1, 0, 1, 0],
       [0, 0, 0, 0, 0, 1, 1, 1, 1, 0],
       [0, 1, 1, 0, 0, 0, 0, 0, 1, 1],
       [1, 1, 0, 0, 0, 0, 1, 0, 1, 0],
       [0, 1, 1, 0, 1, 0, 0, 0, 0, 0],
       [1, 0, 1, 0, 0, 1, 0, 0, 1, 1],
       [1, 0, 1, 0, 1, 0, 0, 0, 1, 1],
       [1, 1, 0, 1, 0, 0, 0, 0, 1, 1],
       [0, 1, 1, 0, 0, 1, 0, 1, 0, 0],
       [1, 1, 1, 1, 0, 1, 1, 0, 1, 0]])
```

GRAYSCALE




```
array([[127, 136,  48, 234, 128, 155,  24, 114, 128,  12],
       [ 18, 177, 127, 239, 230, 236, 145, 240, 195,  65],
       [116,  29, 155,  70,  51, 174, 144, 153, 149, 120],
       [243,  26,  66, 213,   1, 227, 203, 191, 128, 105],
       [183, 199, 207, 167, 146,   7, 136, 234, 175, 232],
       [ 31, 166, 180, 133, 187,  82, 192, 182, 194,   2],
       [139, 161, 108, 144, 183,  23, 185,  12,  67,  47],
       [ 53,  24,  72,  46, 182,  94,  25, 221,  16,  34],
       [ 76, 144, 181,  22,  21,  67,  16,  53, 250, 222],
       [197,   6,   7, 122, 198,  34, 222,  64,  12, 108]])
```

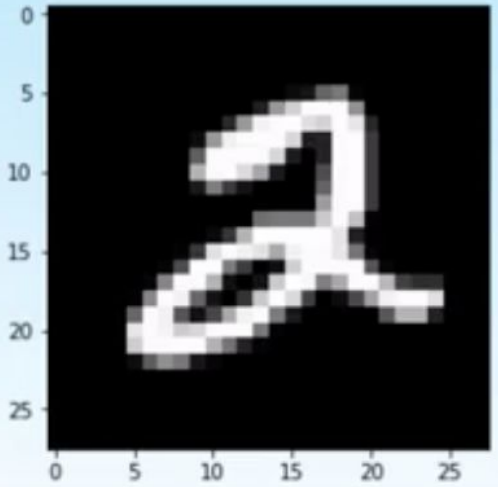
MNIST



5041

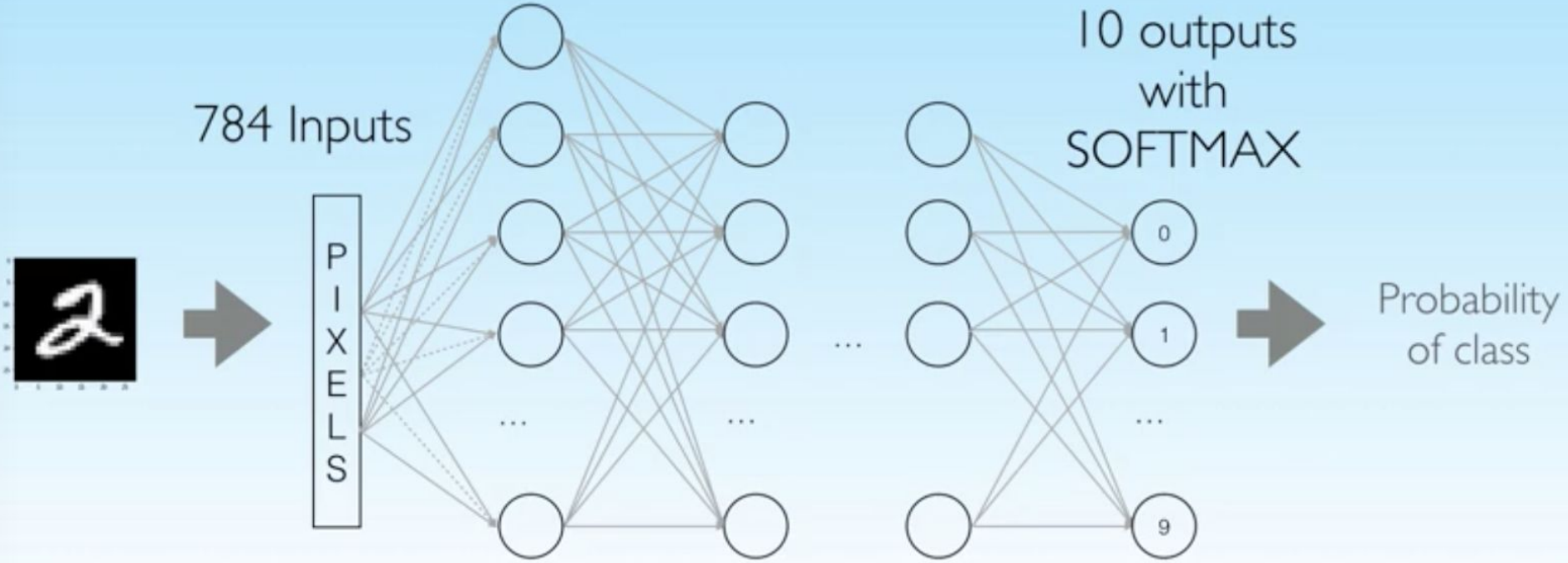
MNIST

.5 .5 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | | 0 0 0 0 0 .5 .5 | 0 0 0 0 0 0 0 0



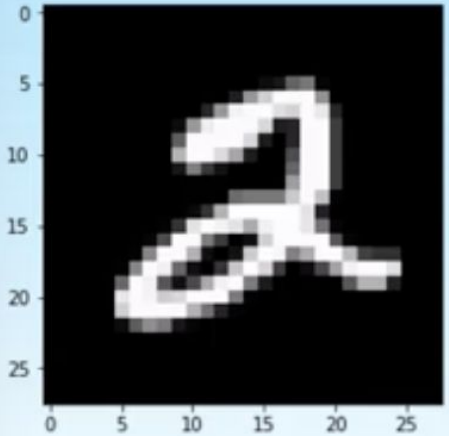
28 x 28 image => 784 input pixels array

FULLY CONNECTED



Fully connected NN

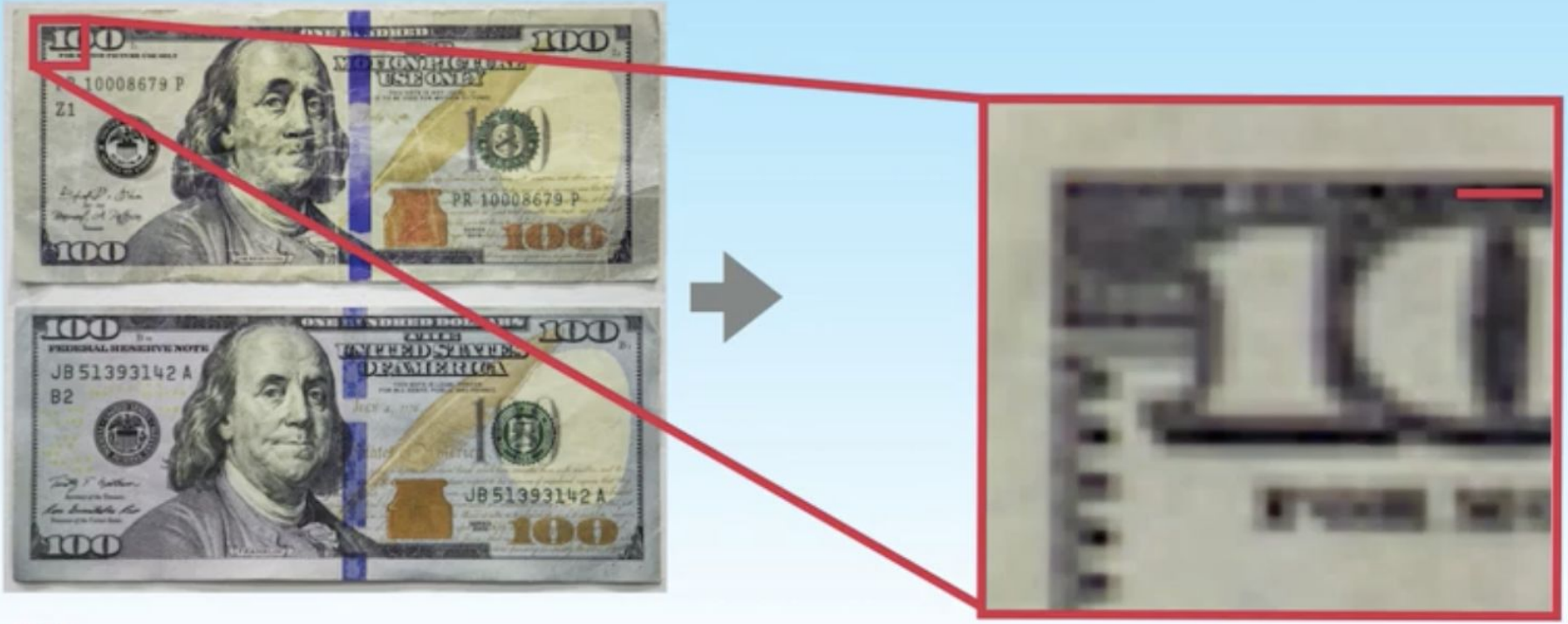
FEATURES



Feature Vector

- Fourier coefficients
- Wavelets
- Histogram of Oriented Gradients (HOG)
- Speeded Up Robust Features (SURF)
- Local Binary Patterns (LBP)
- Color histograms
- …

LOCAL PATTERNS



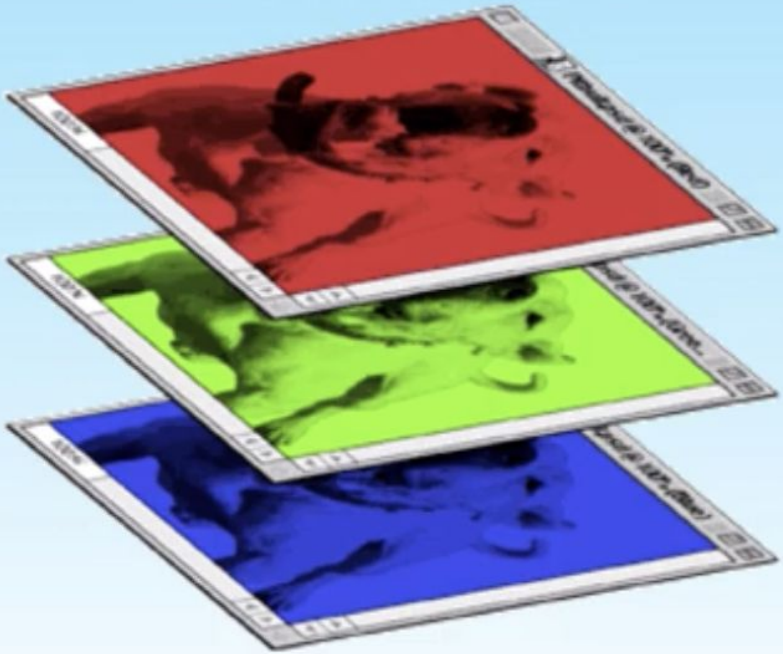
The importance of vision for humans

Image to Tensors

TENSORS

Order	Name	Example	Shape
0	Scalar	3	no shape
1	Vector	[4, 5, 0, 3, 1, 4, 5]	(7,)
2	Matrix	[[0, 1, 0], [5, 0, 2]]	(2, 3)
3	Tensor	[[[0, 1, 0, 5], [5, 0, 2, 6]], [[1, 2, 4, 4], [8, 3, 1, 9]]]	(2, 2, 4)

COLORED IMAGES



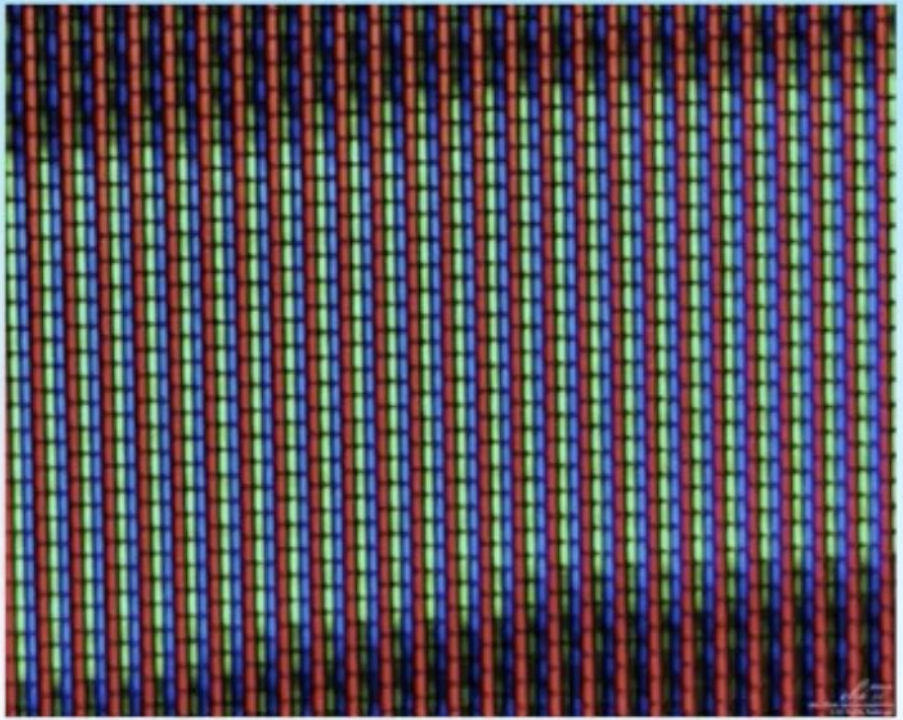
Red Channel

Green Channel

Blue Channel

(C, H, W)

COLORED IMAGES



(C, H, W)

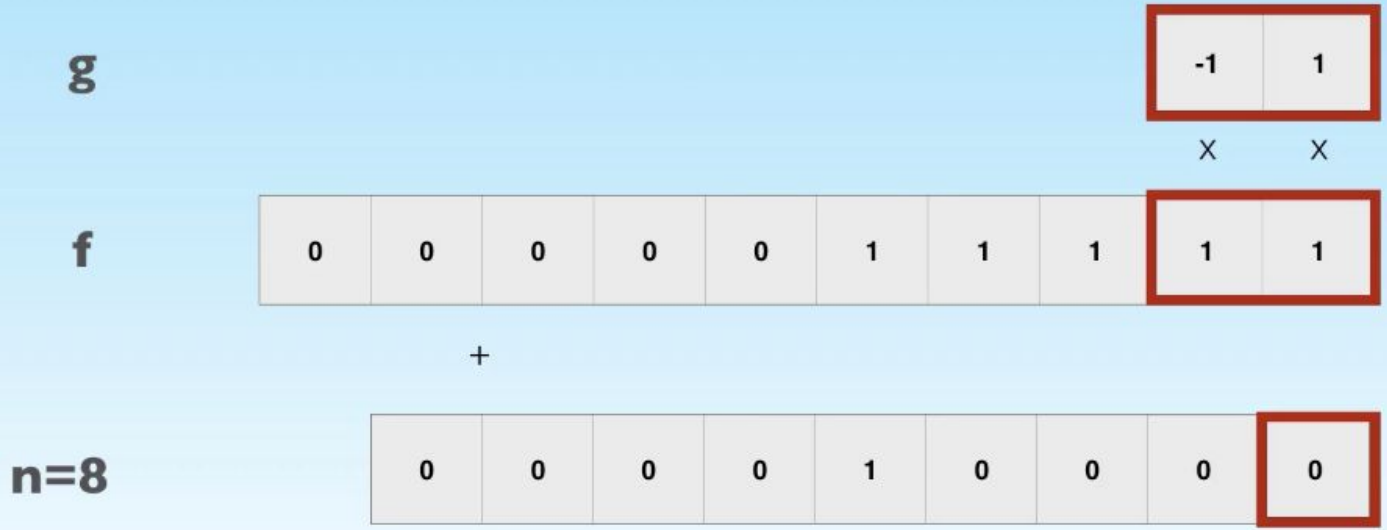
(H, W, C)

Code

Github - tensor_math.ipynb

DISCRETE CONVOLUTION

DISCRETE CONVOLUTION

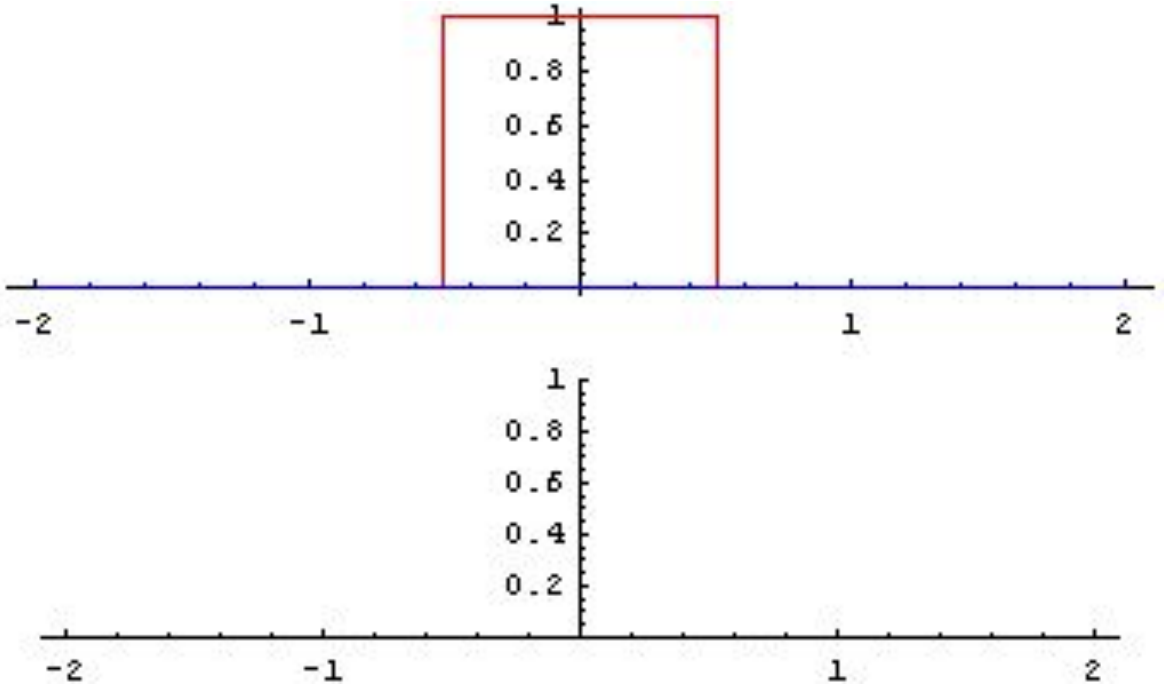


Definition

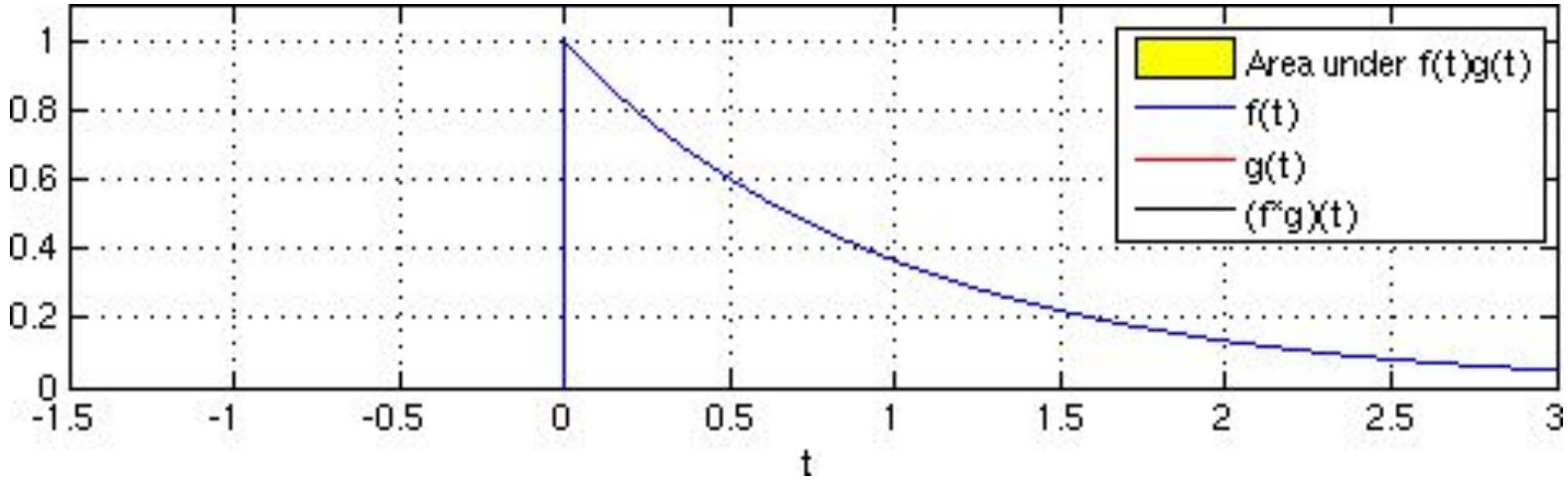
Convolution of f and g at position n is the sum of products between M and -M of f at n -m and g of m

$$(f * g)[n] = \sum_{m=-M}^{M} f[n-m]g[m]$$

Visualize



Another Example

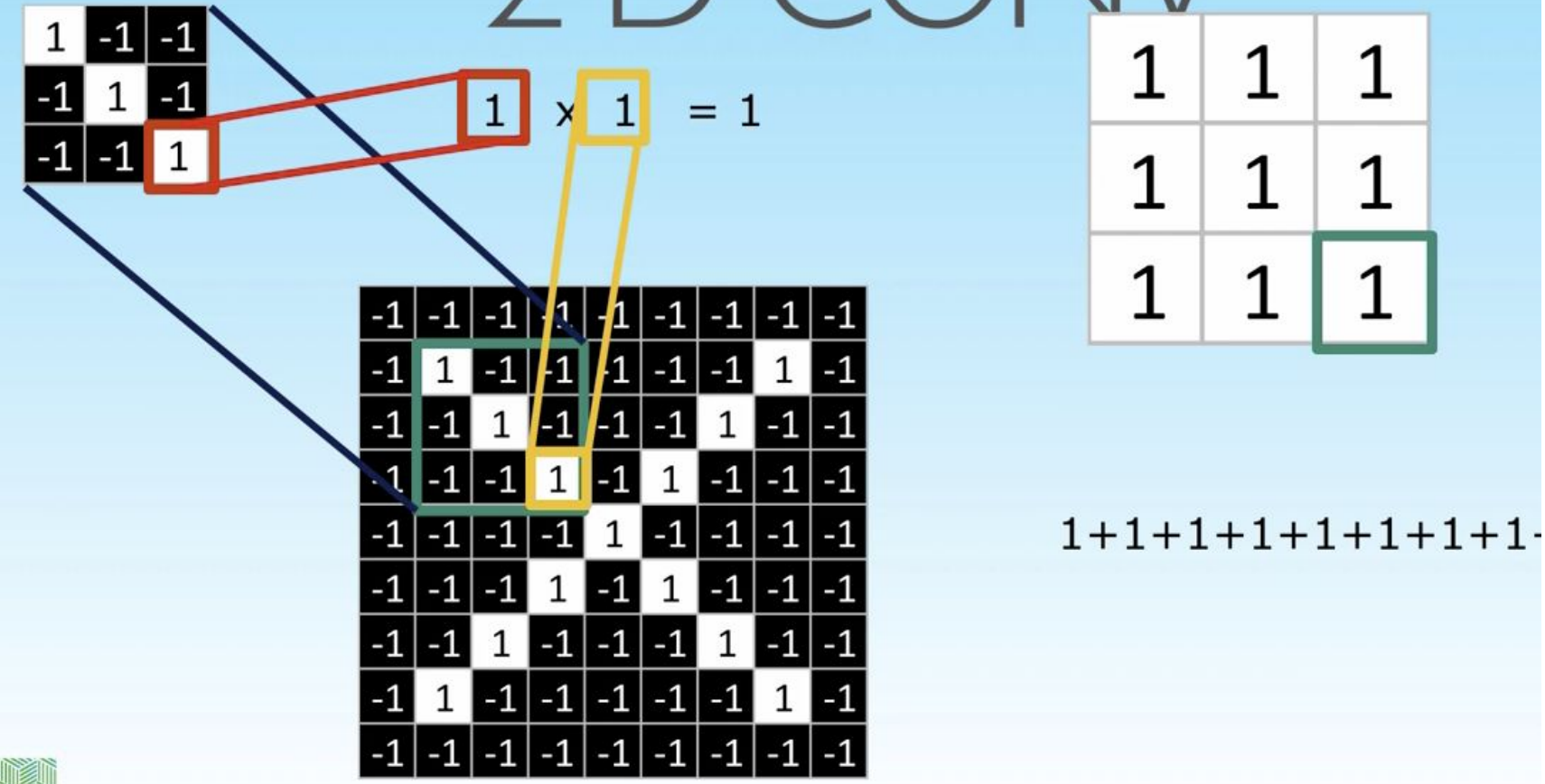


code

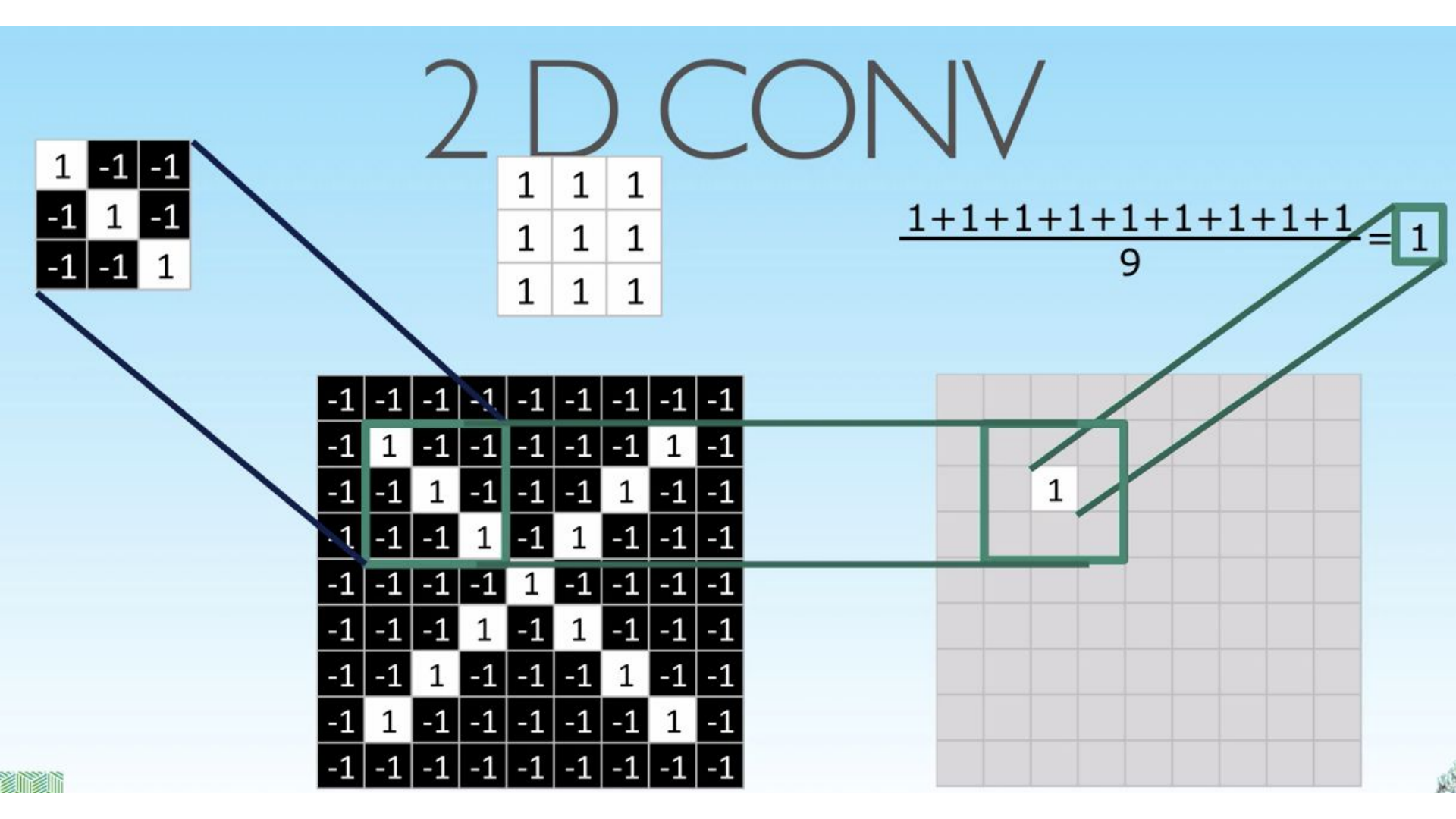
Github - convolutions.ipynb

2 D convolution

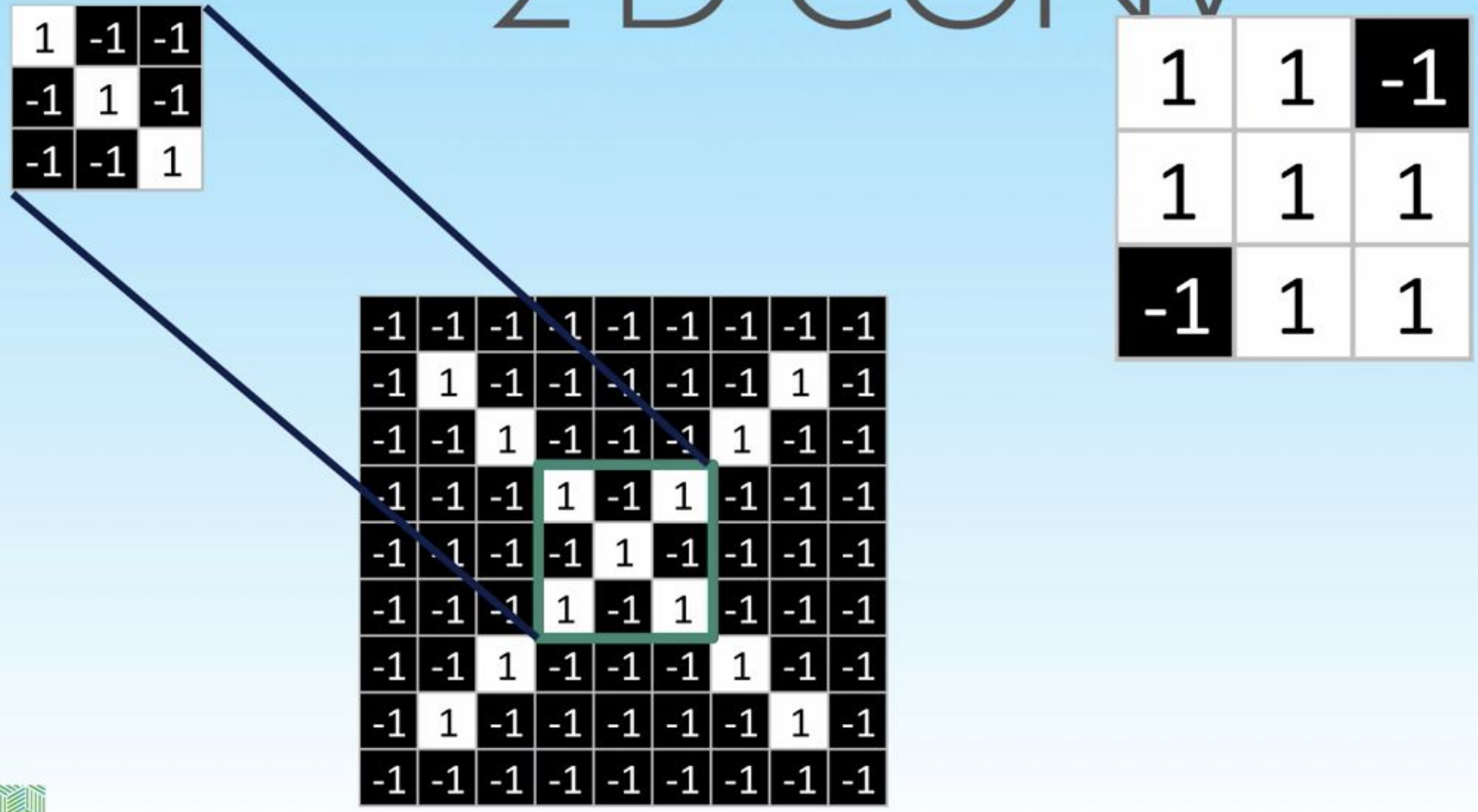
2 D CONV


1 -1 -1
-1 1 -1
-1 -1 1
1 x 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 -1 -1
1 1 1
1 1 1
1 1 1
1+1+1+1+1+1+1+1+1

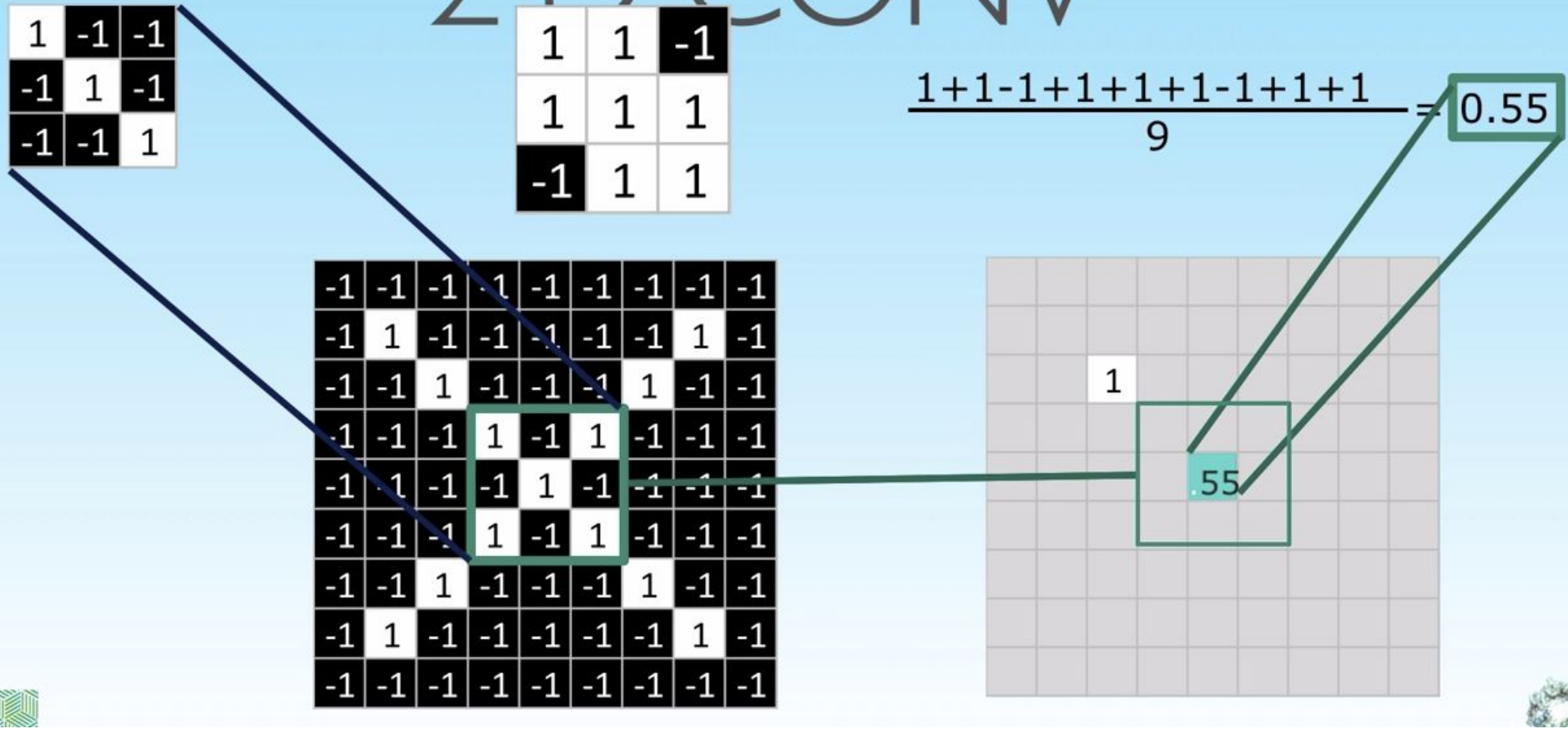
2 D CONV


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
9
= 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 -1
1

2 D CONV


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
-1 -1 -1 -1 -1 -1 -1 -1 -1
1 1 -1
1 1 1
-1 1 1

2D CONV



2 D CONV

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
-1	-1	-1	-1	-1	-1	-1	-1	-1

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

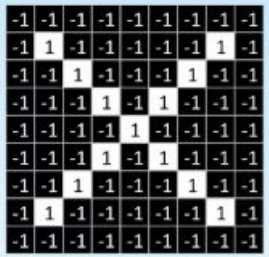
Code

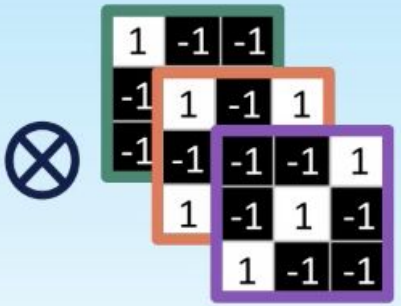
convolutions2d.ipynb

CNNs

- Convolutional Layer
- Convolution of tensors
- Strides and Padding

CONVOLUTION LAYER





Number of Filters

Output Channels

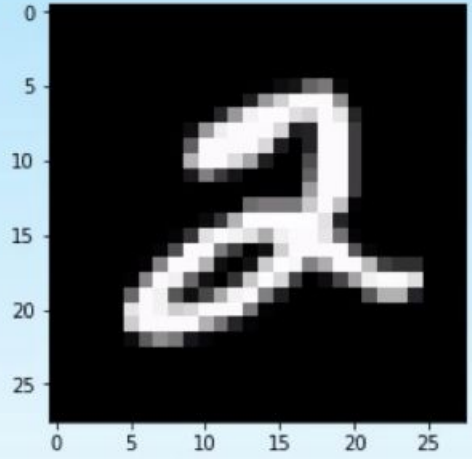
INPUT TENSOR

Input: order 4 tensor

(N, H, W, C)

N: Number of images
H: Height of image
W: Width of image
C: Number of color channels

MNIST training set



(60000, 28, 28, 1)

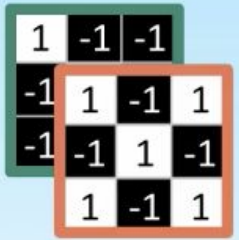
CONV LAYER TENSOR

CONV: order 4 tensor

$$(H_f, W_f, C_i, C_o)$$

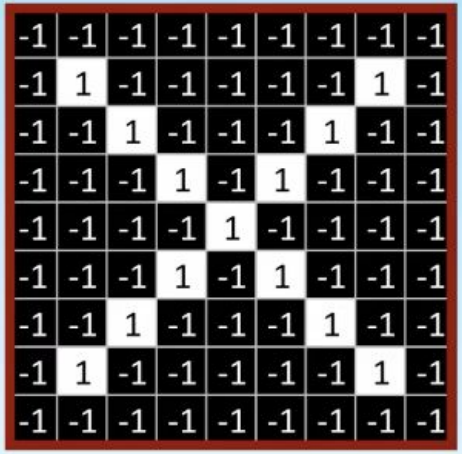
H_f: Height of filter patch
W_f: Width of filter patch
C_i: Channels in input
C_o: Channels in output (# filters)

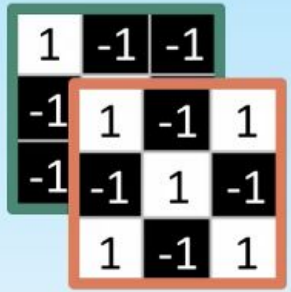
Example

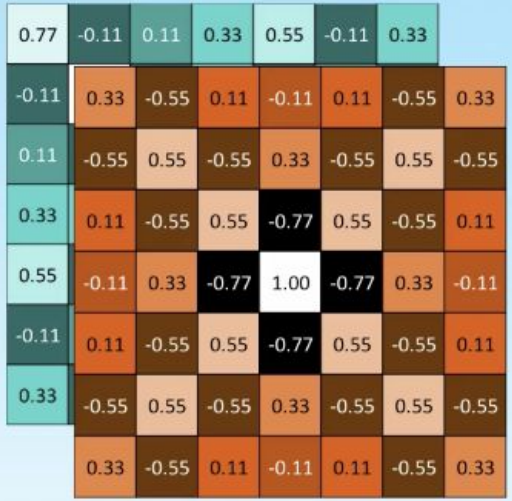


$$(3, 3, 1, 2)$$

CONV LAYER







Examples, H, W, inp channels

(-, 9, 9, 1)

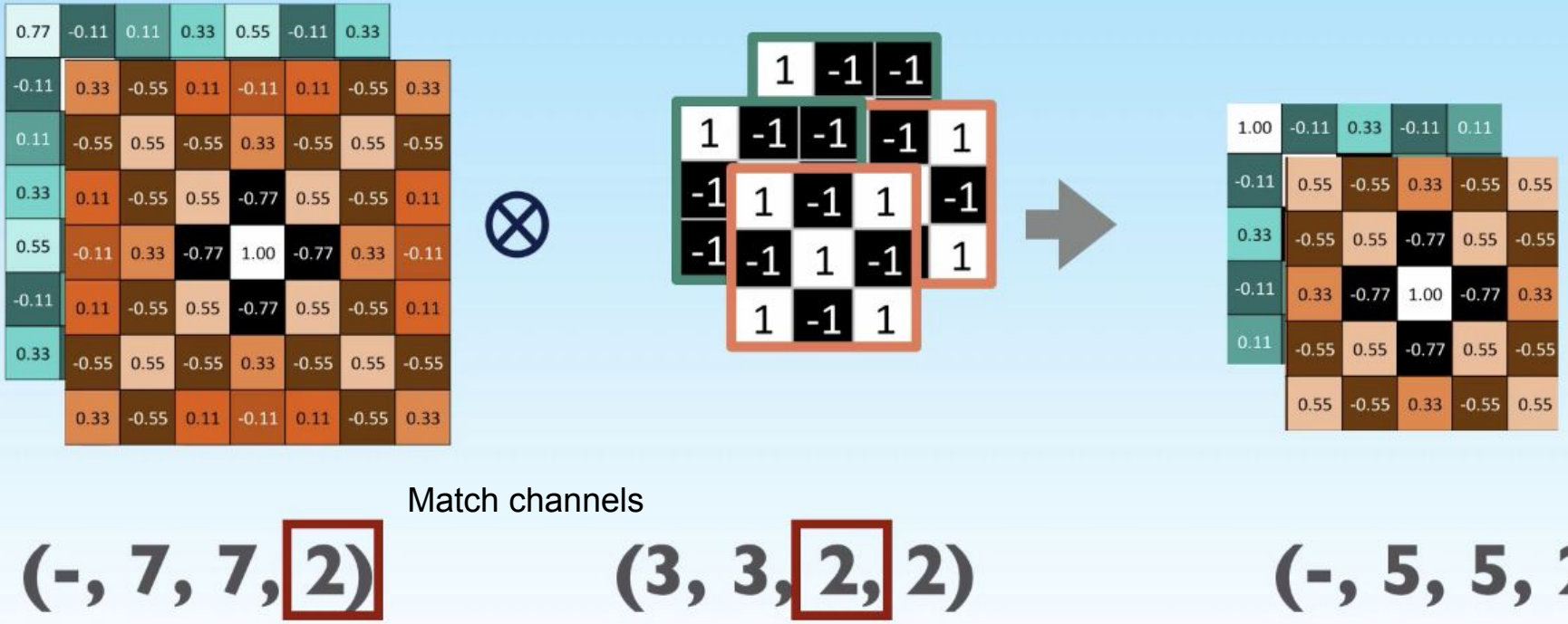
(3, 3, 1, 2)

H, W, inp channels , # filters, out channels

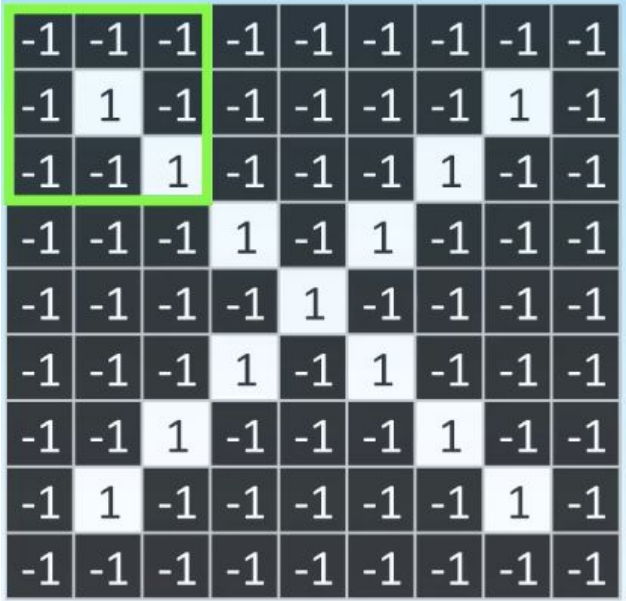
examples, H, W, out channels

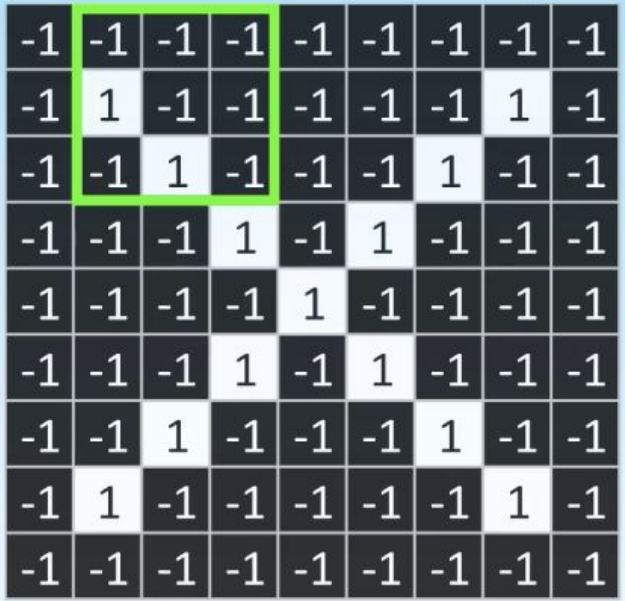
(-, 7, 7, 2)

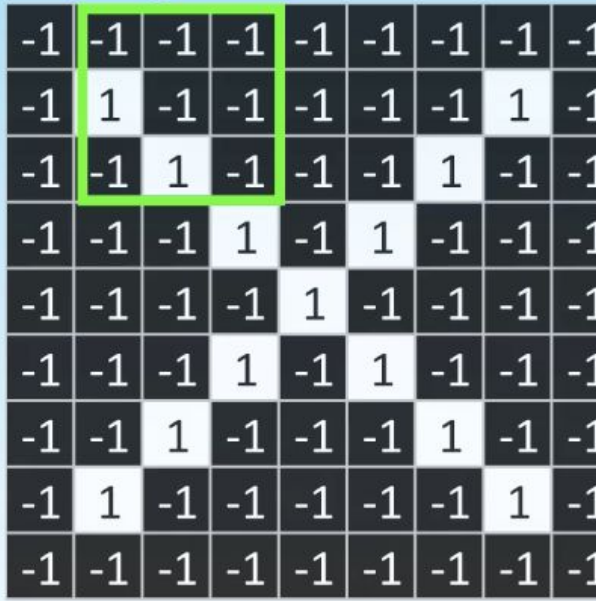
Since output is order 4, we can keep going



STRIDES: (1, 1)







Output image => 7 x 7

STRIDES: (2, 2)

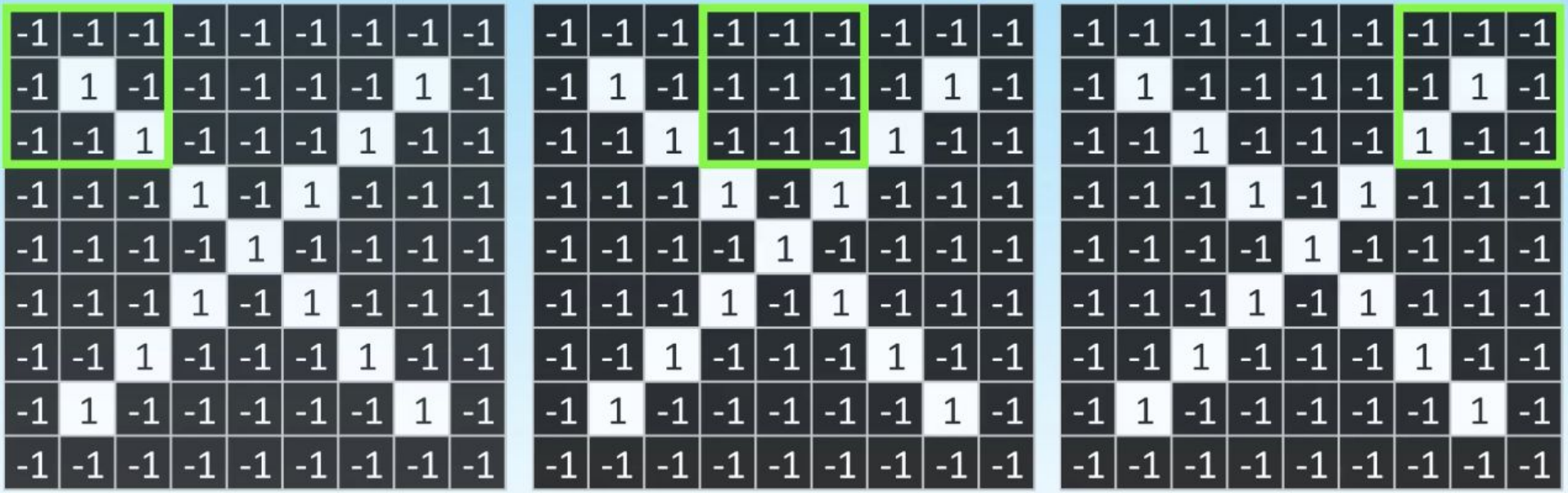
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

-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

-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

Output image => 4 x 4

STRIDES: (3, 3)



Output image => 3 x 3

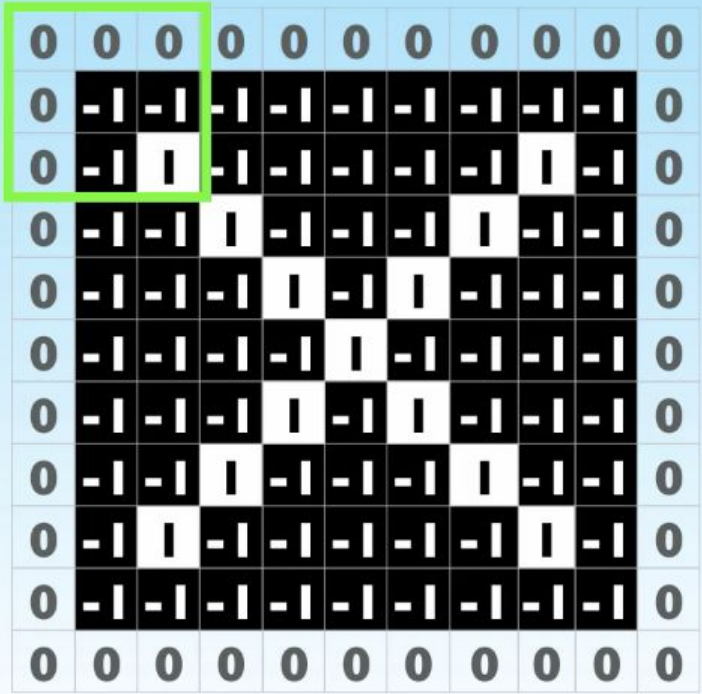
STRIDES: (3, 1)

-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	-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	-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	-1

PADDING



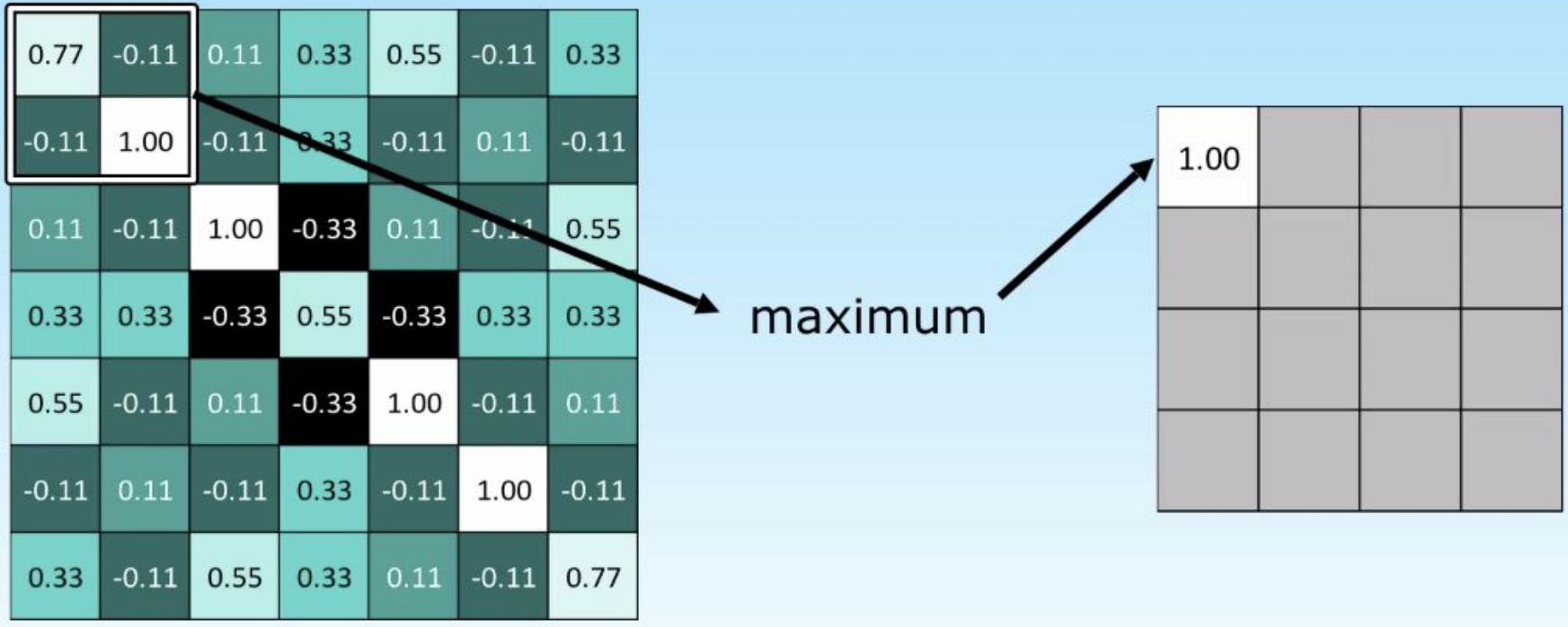
- Valid
- Same

Output image => 9 x 9

Code

convolutional_layer.ipynb

MAX POOLING

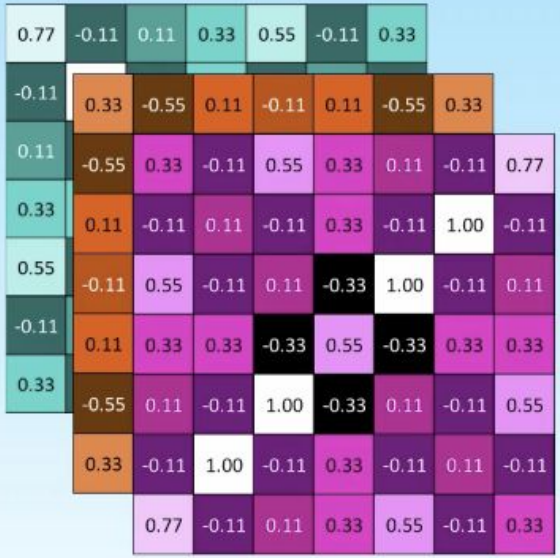

0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
maximum
1.00

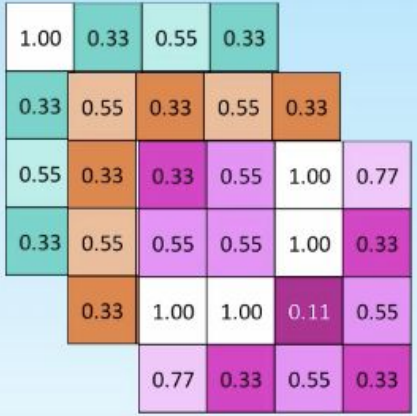
MAX POOLING

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

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

POOLING LAYER





(-, 7, 7, 3)

Operates on H and W

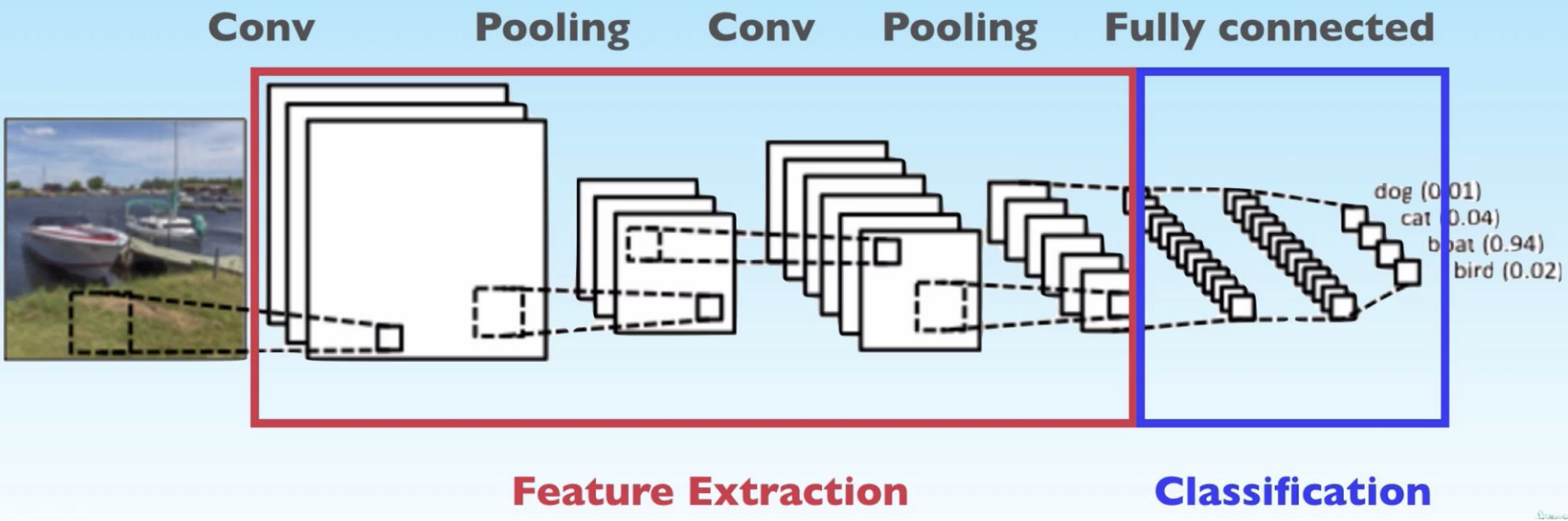
(-, 4, 4, 3)

Code

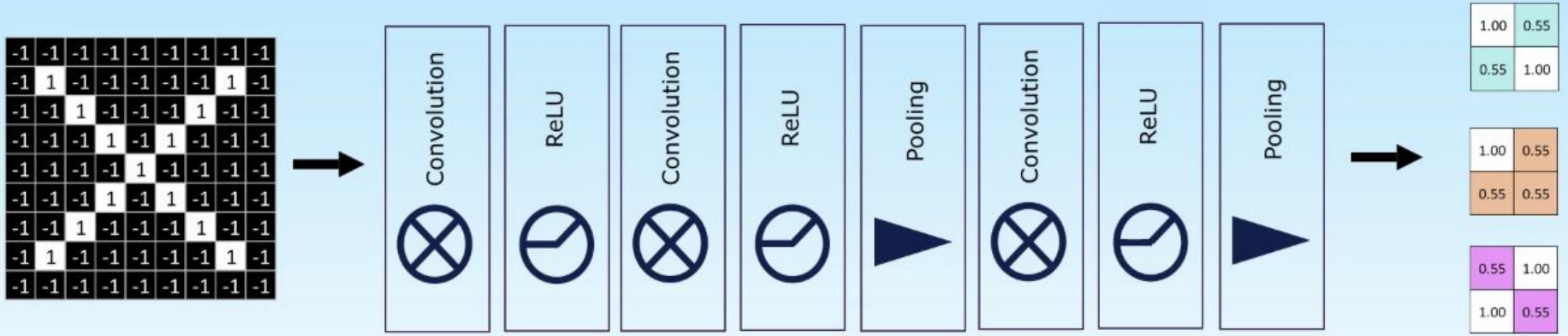
Pooling_layer.ipynb

CNN Finally!

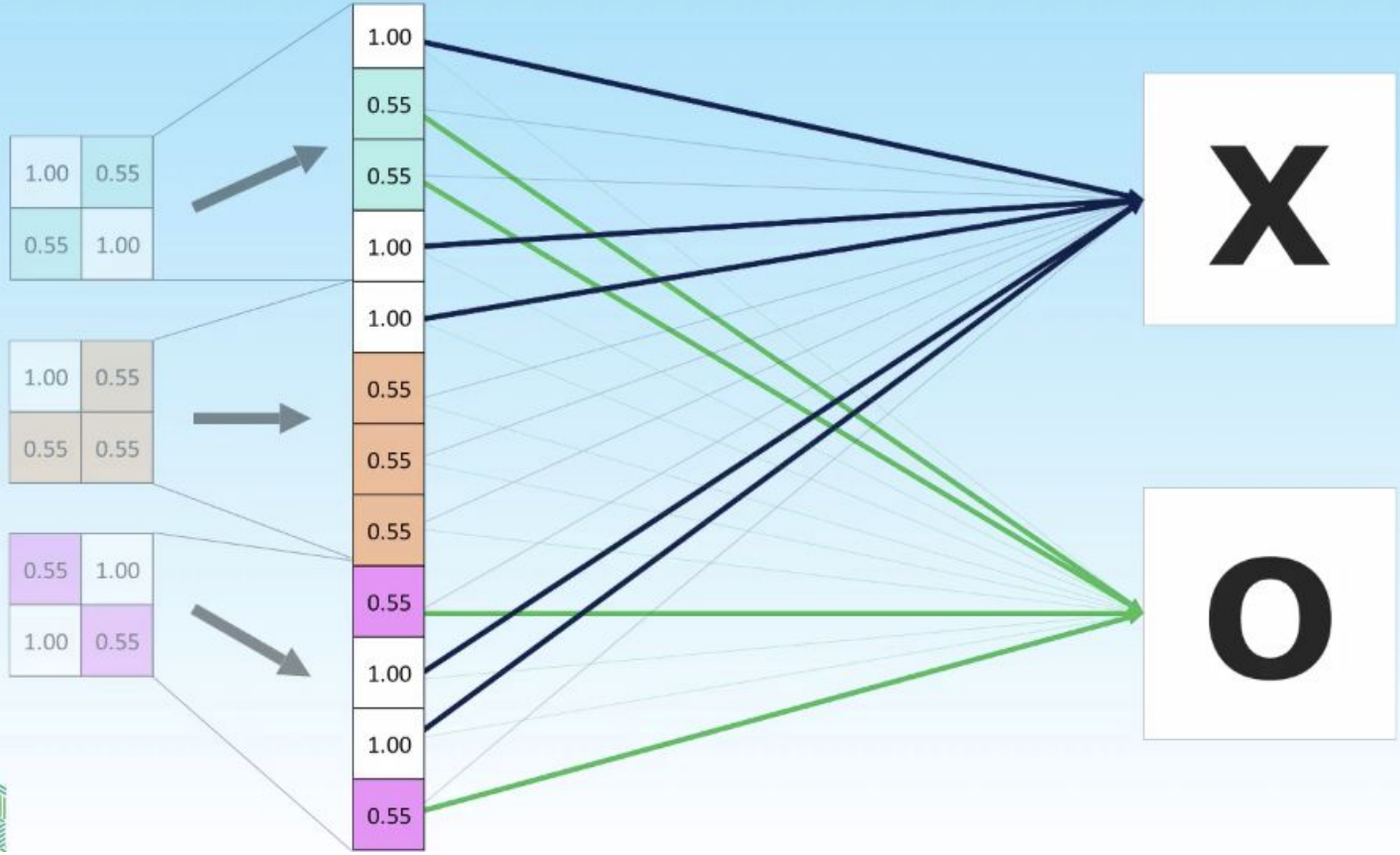
STACKING LAYERS

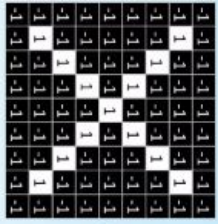


FEATURE EXTRACTION



FULLY CONNECTED LAYER





ALL TOGETHER

Code

CNN.ipynb