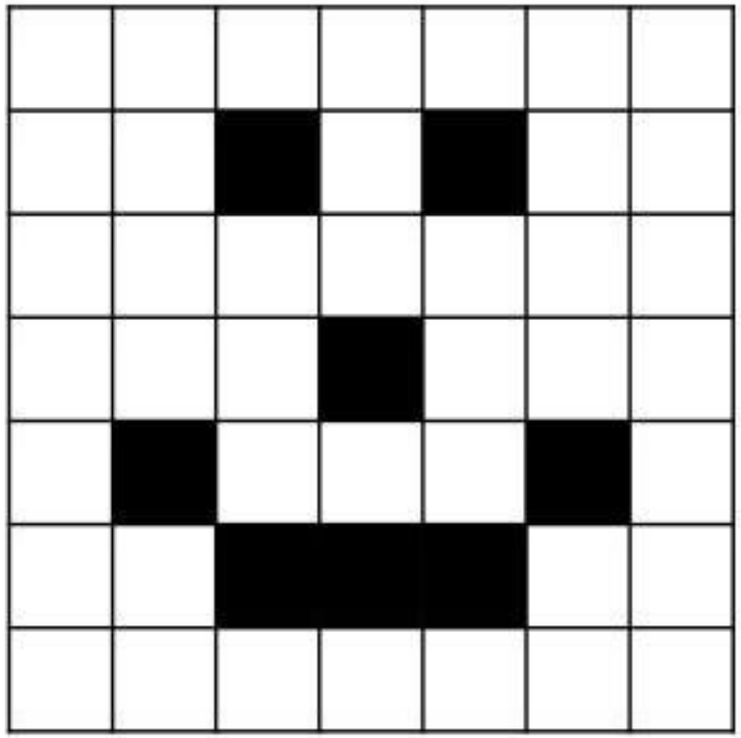
Convolutional Neural Networks

© SuperDataScience

Convolutional Neural Networks



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

© SuperDataScience

Convolutional Neural Networks

STEP 1: Convolution

STEP 2: Max Pooling

STEP 3: Flattening

STEP 4: Full Connection

© SuperDataScience

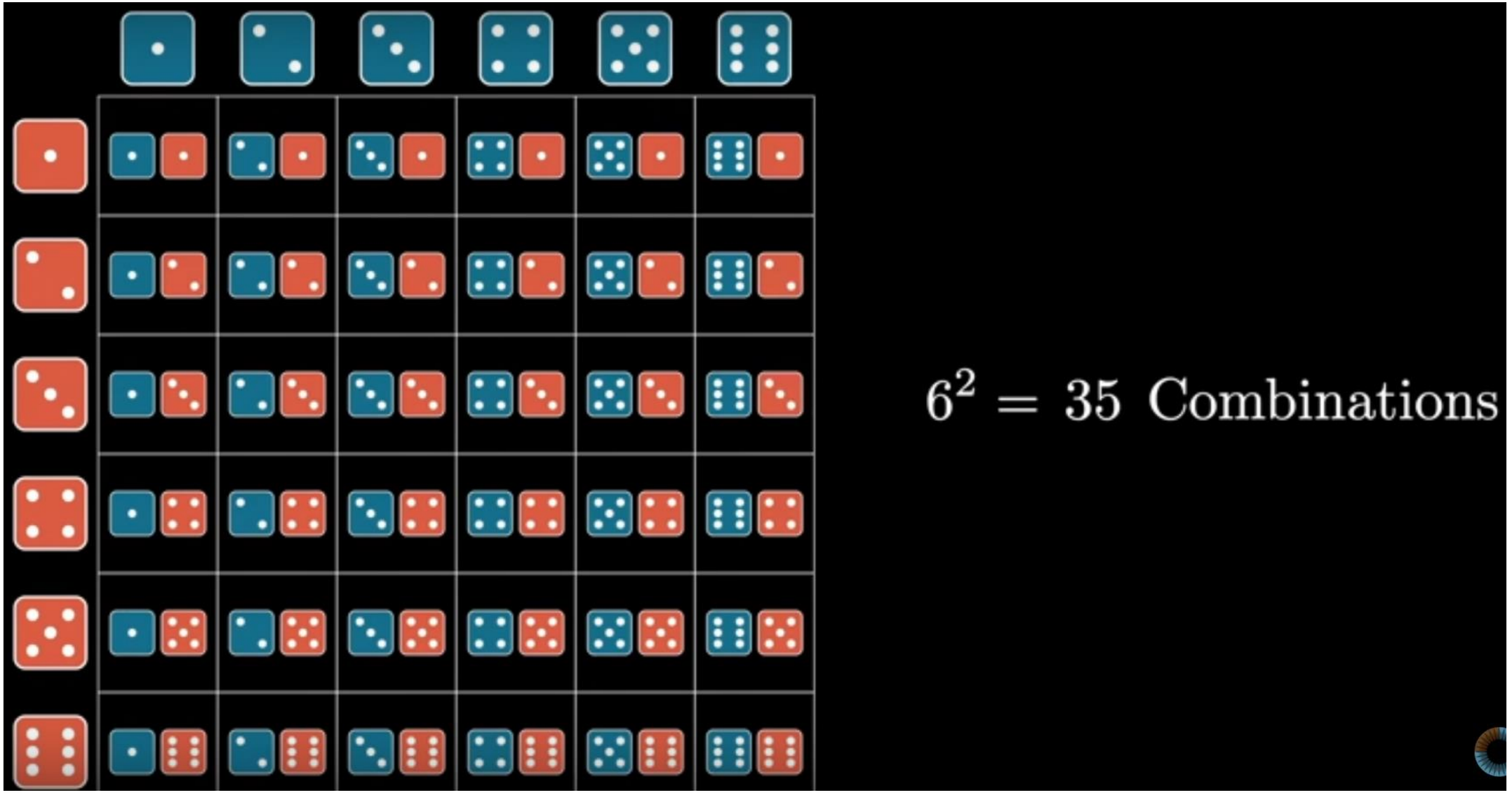
Step 1 – Convolution

© SuperDataScience

Simple Way of Convolution

A = [1,2,3,4]
B = [5,6,7,8]

A*B = [5,16,34,60,61,52,32]


$6^2 = 35$ Combinations

$P(\square + \square = 2) = a_1 \cdot b_1$

$P(\square + \square = 3) = a_1 \cdot b_2 + a_2 \cdot b_1$

$P(\square + \square = 4) = a_1 \cdot b_3 + a_2 \cdot b_2 + a_3 \cdot b_1$

$P(\square + \square = 5) = a_1 \cdot b_4 + a_2 \cdot b_3 + a_3 \cdot b_2 + a_4 \cdot b_1$

$P(\square + \square = 6) = a_1 \cdot b_5 + a_2 \cdot b_4 + a_3 \cdot b_3 + a_4 \cdot b_2 + a_5 \cdot b_1$

$P(\square + \square = 7) = a_1 \cdot b_6 + a_2 \cdot b_5 + a_3 \cdot b_4 + a_4 \cdot b_3 + a_5 \cdot b_2 + a_6 \cdot b_1$

$P(\square + \square = 8) = a_2 \cdot b_6 + a_3 \cdot b_5 + a_4 \cdot b_4 + a_5 \cdot b_3 + a_6 \cdot b_2$

$P(\square + \square = 9) = a_3 \cdot b_6 + a_4 \cdot b_5 + a_5 \cdot b_4 + a_6 \cdot b_3$

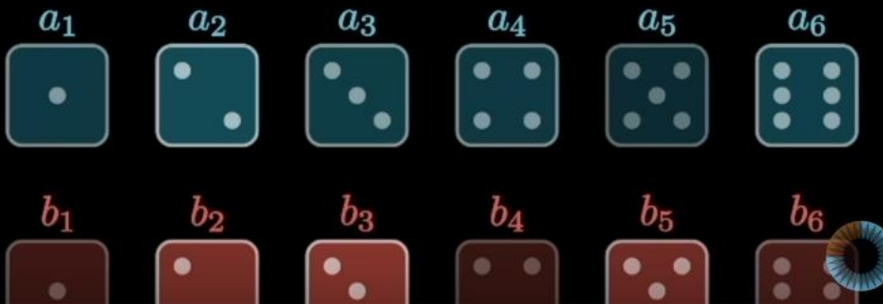
$P(\square + \square = 10) = a_4 \cdot b_6 + a_5 \cdot b_5 + a_6 \cdot b_4$

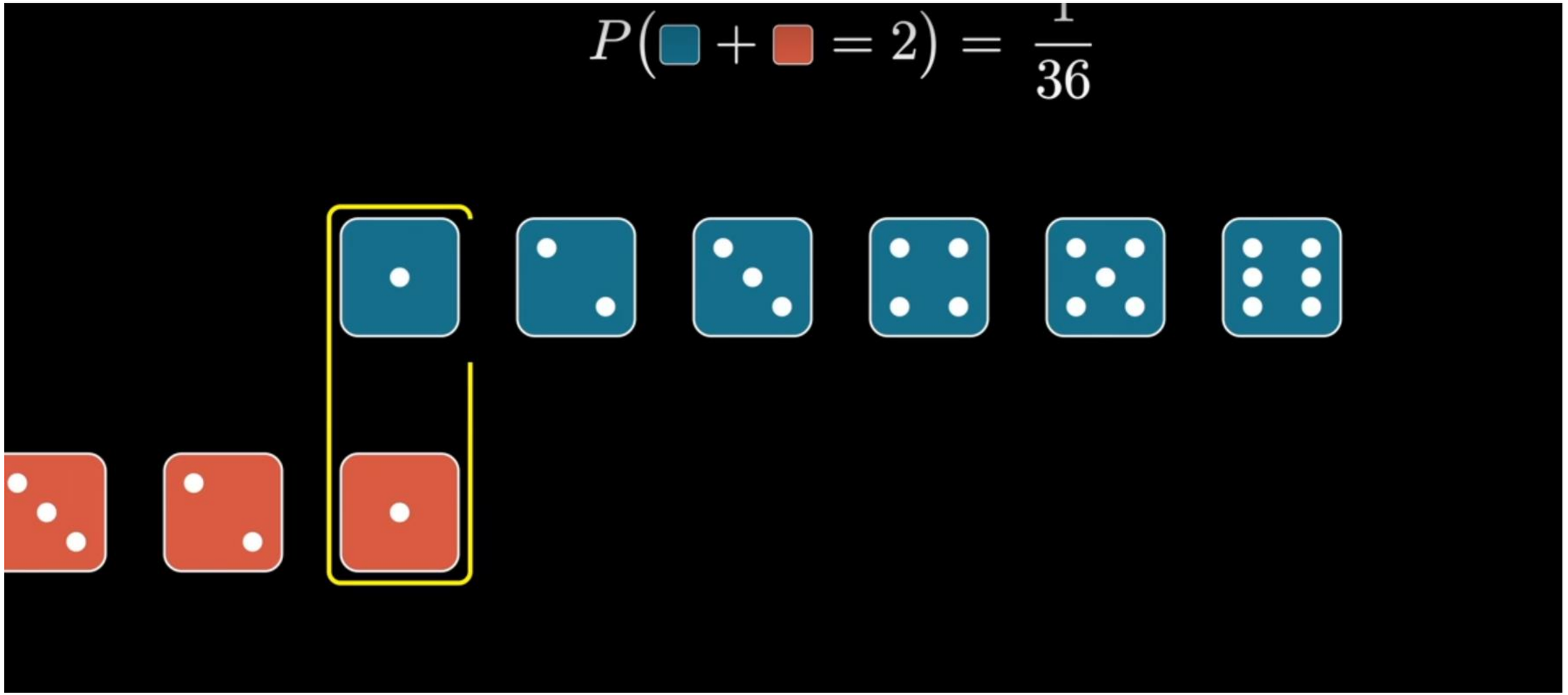
$P(\square + \square = 11) = a_5 \cdot b_6 + a_6 \cdot b_5$

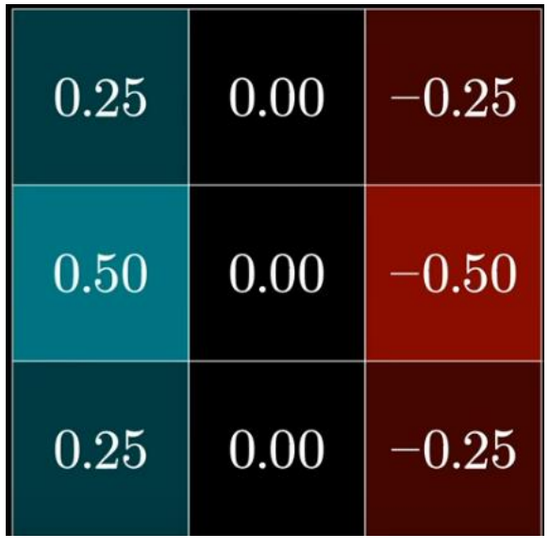
$P(\square + \square = 12) = a_6 \cdot b_6$

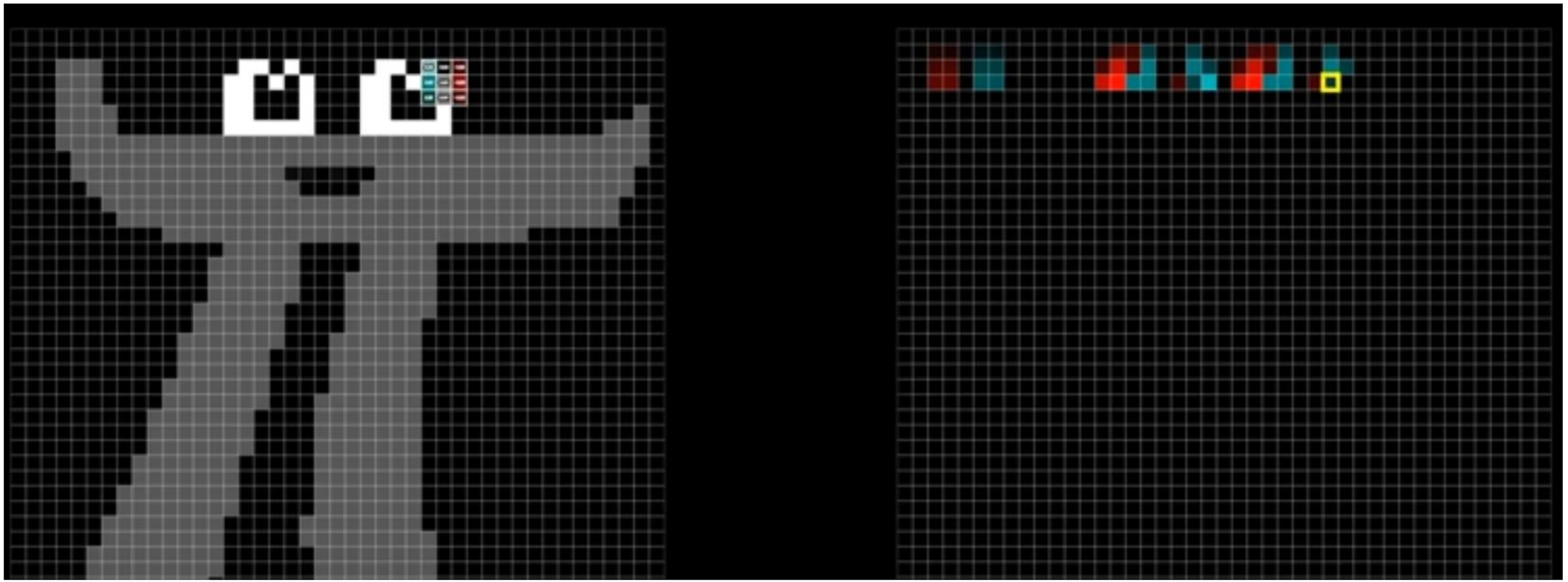
Convolution of (a_i) and (b_i)

$$(a * b)_n = \sum_{\substack{i,j \\ i+j=n}} a_i \cdot b_j$$

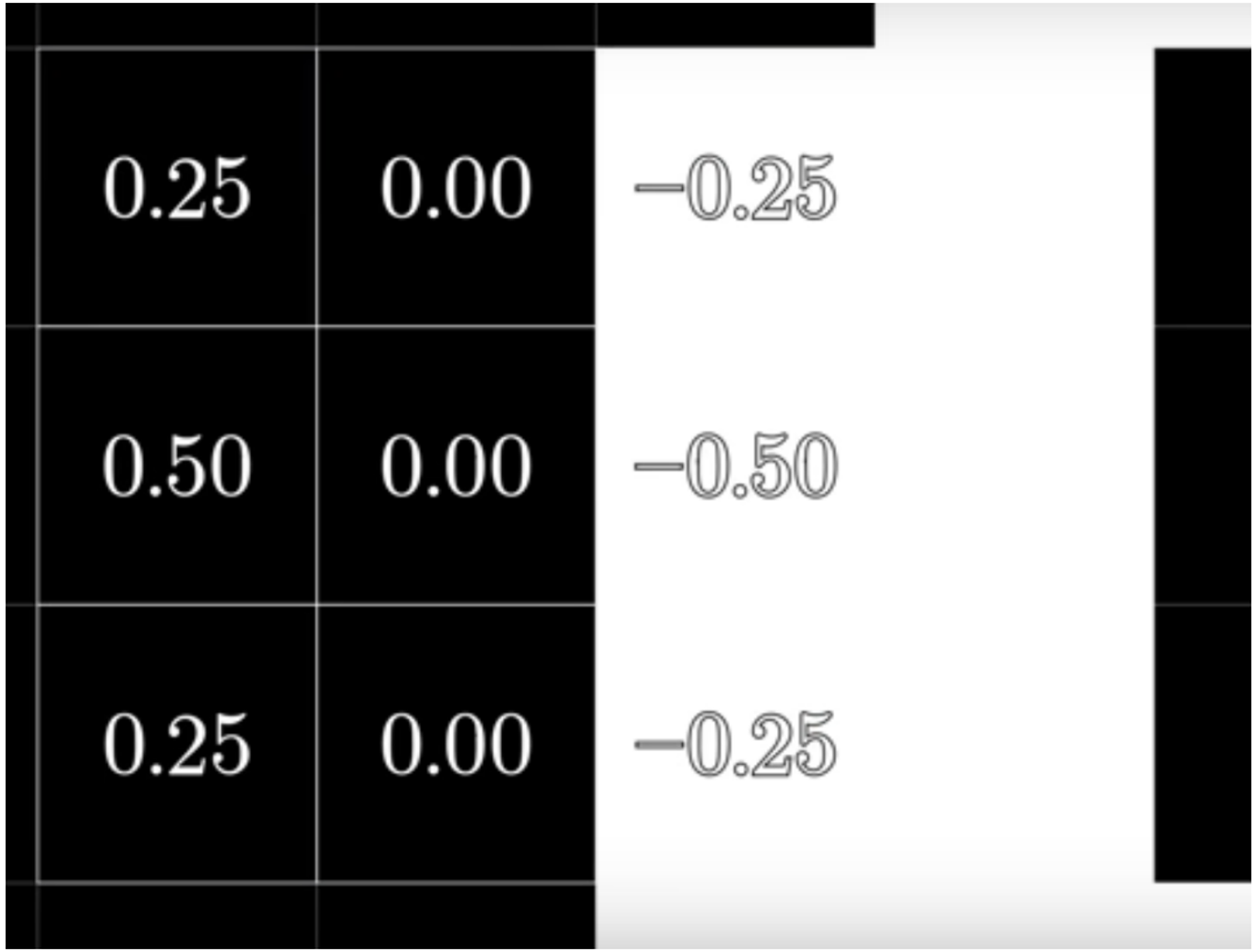



$P(\square + \square = 2) = \frac{1}{36}$

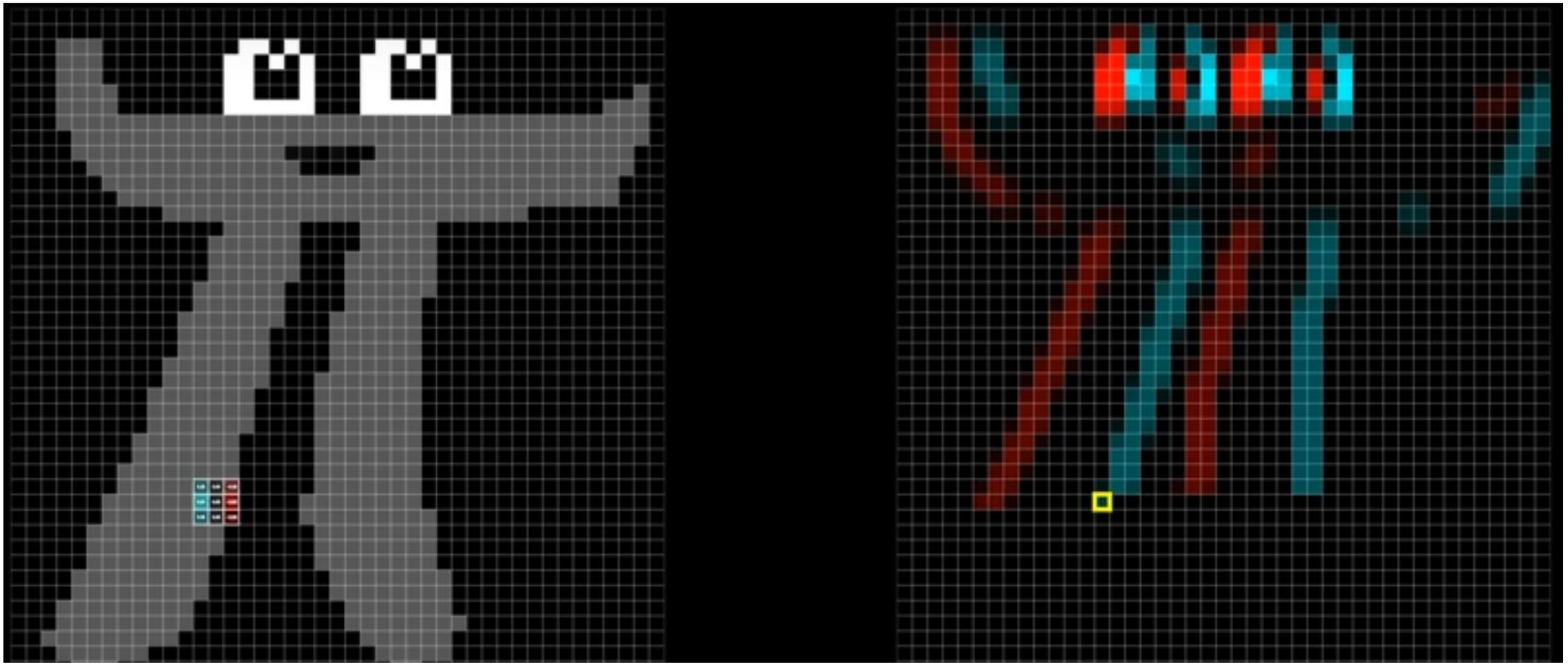

0.25
0.00
−0.25
0.50
0.00
−0.50
0.25
0.00
−0.25



CNN



CNN



CNN

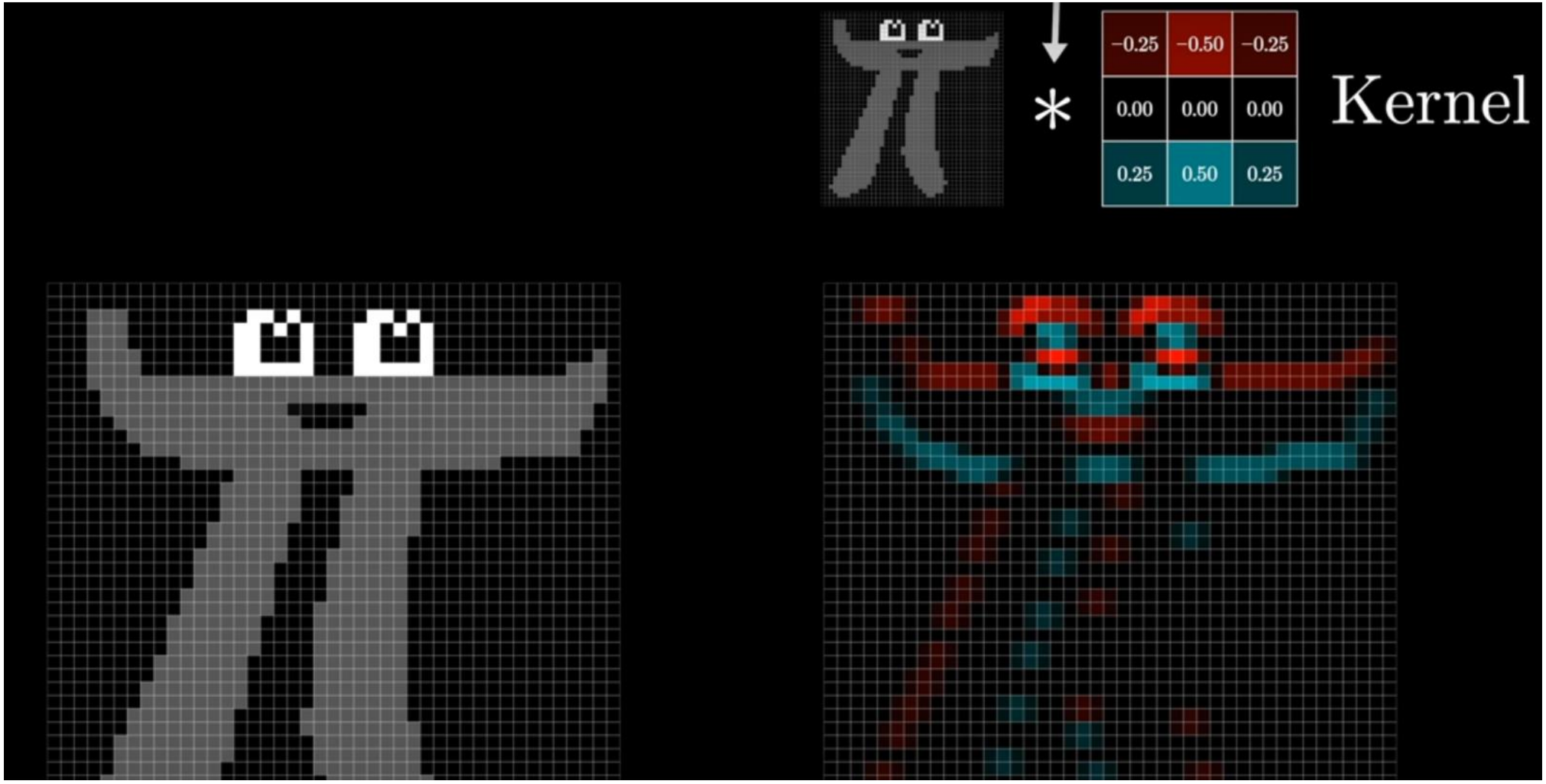
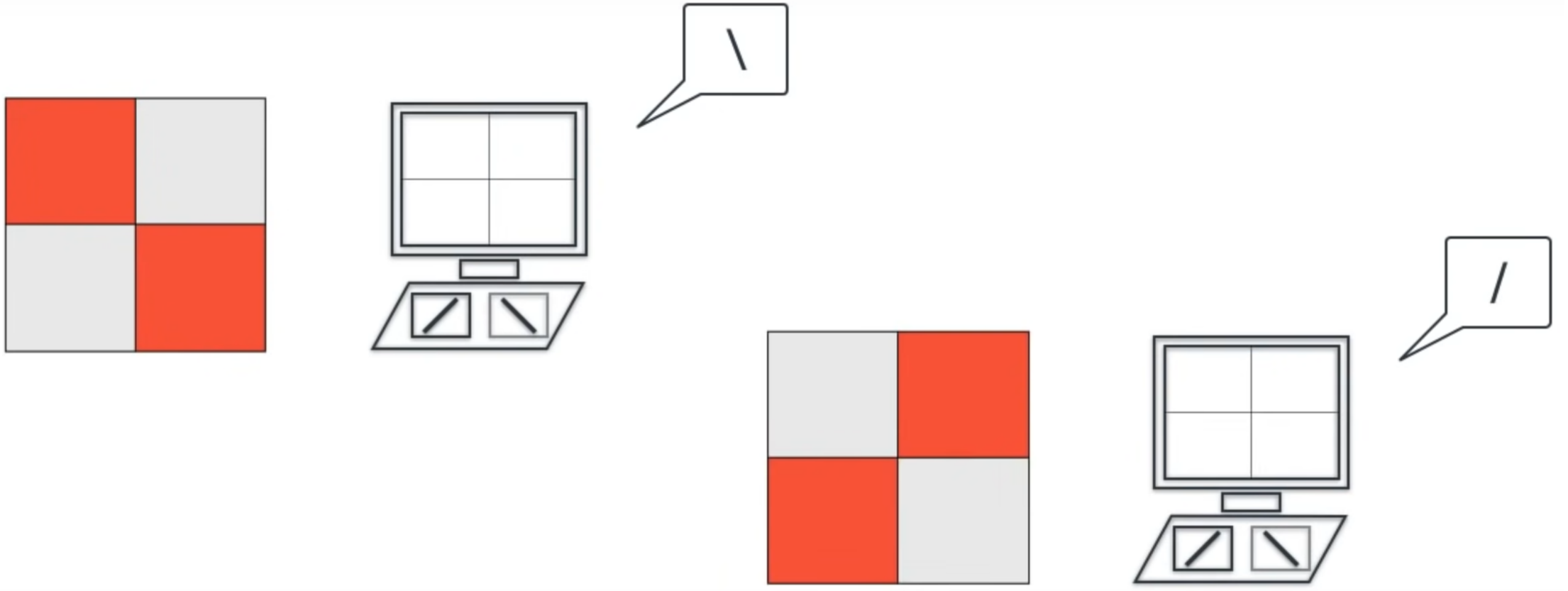
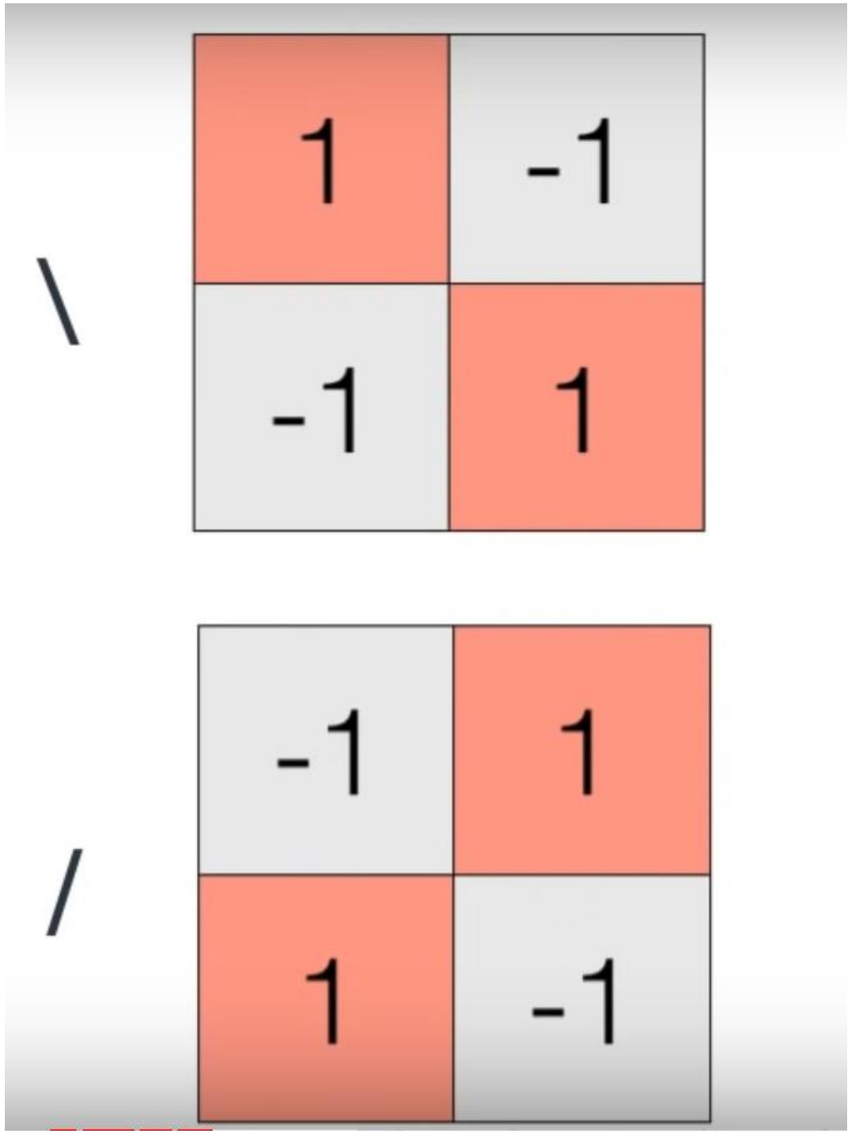
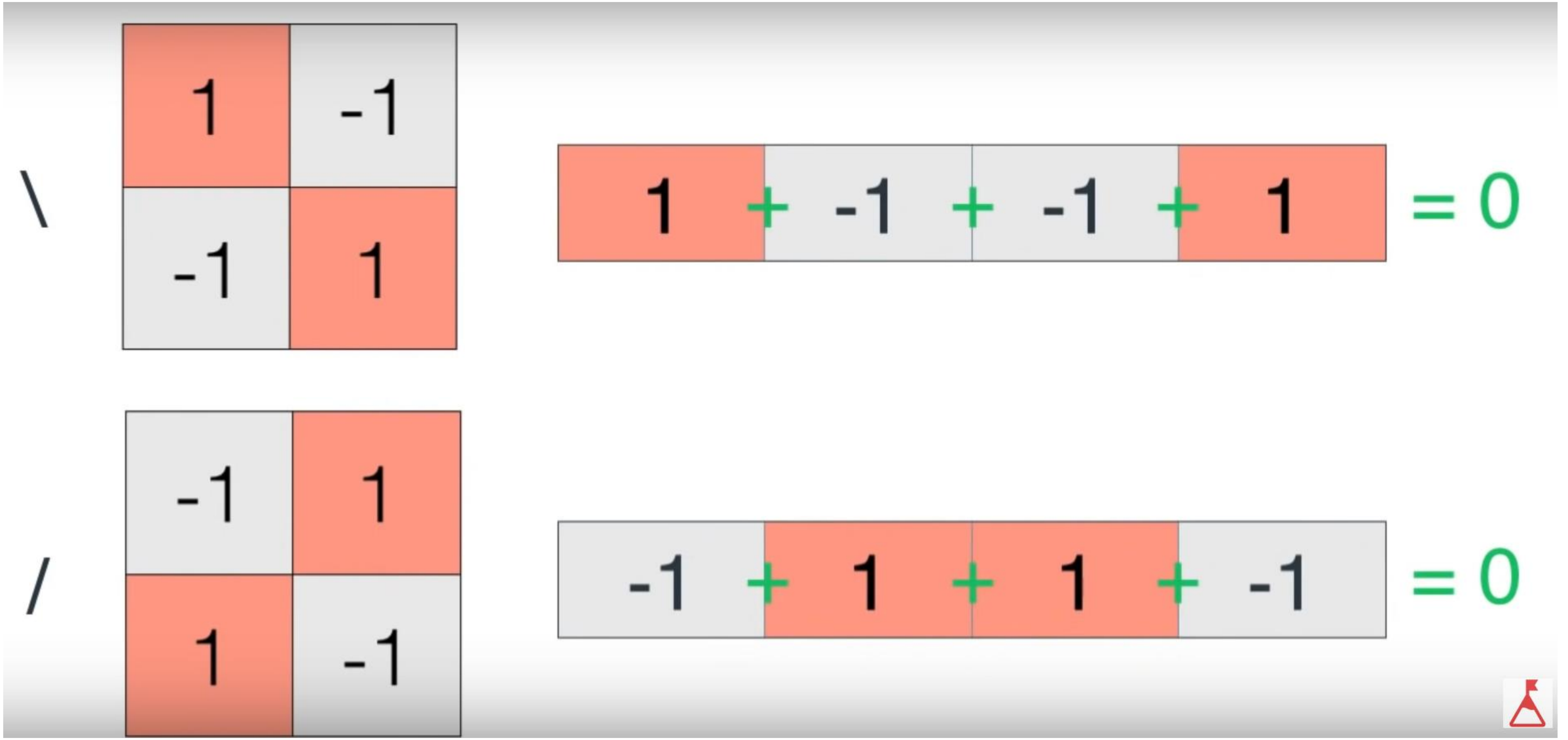
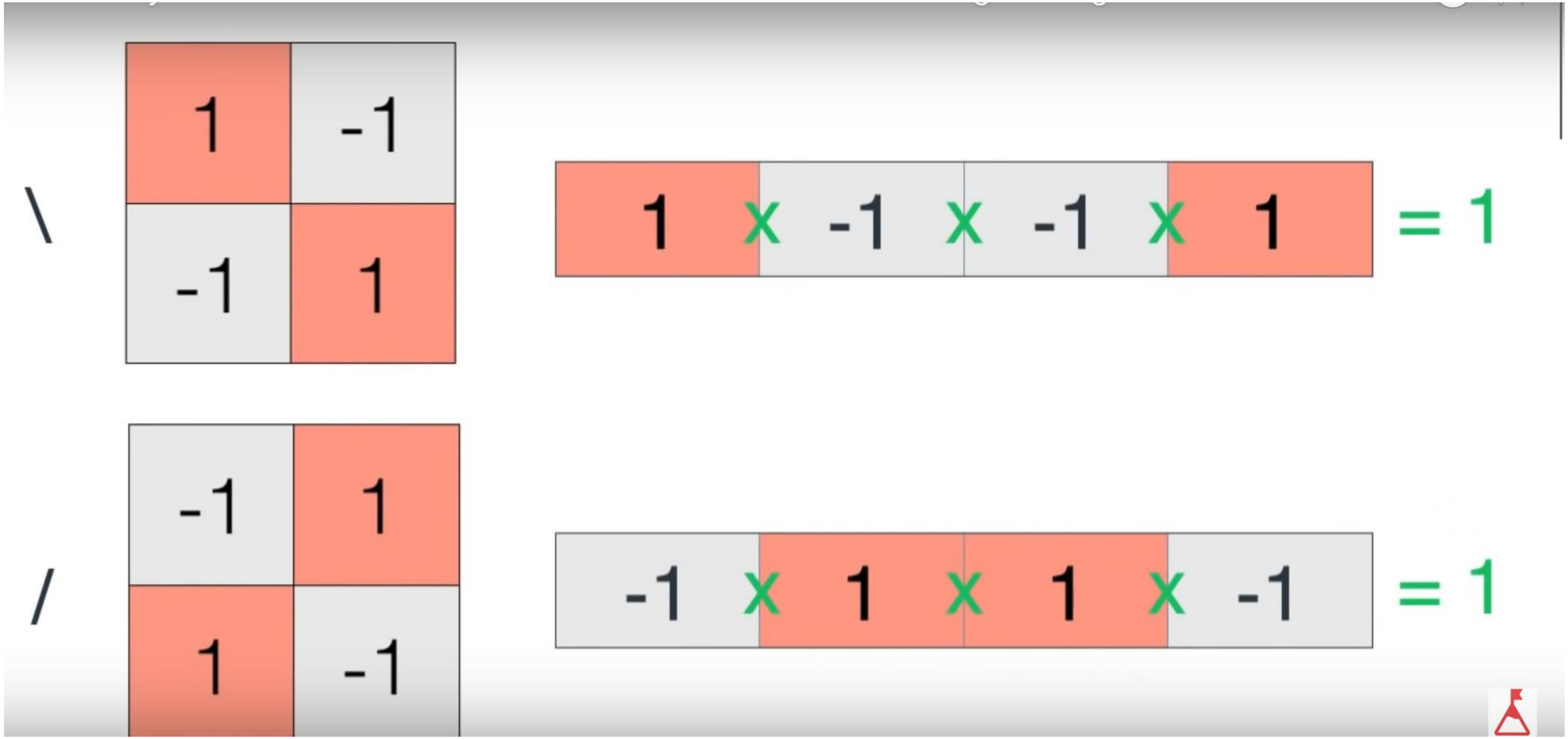

−0.25
−0.50
−0.25
0.00
0.00
0.00
0.25
0.50
0.25
Kernel

Image recognition software




1
-1
\
-1
1
-1
1
/
1
-1


\
1
-1
-1
1
1 + -1 + -1 + 1 = 0
/
-1
1
1
-1
-1 + 1 + 1 + -1 = 0


\
1
-1
-1
1
1 x -1 x -1 x 1 = 1
/
-1
1
1
-1
-1 x 1 x 1 x -1 = 1

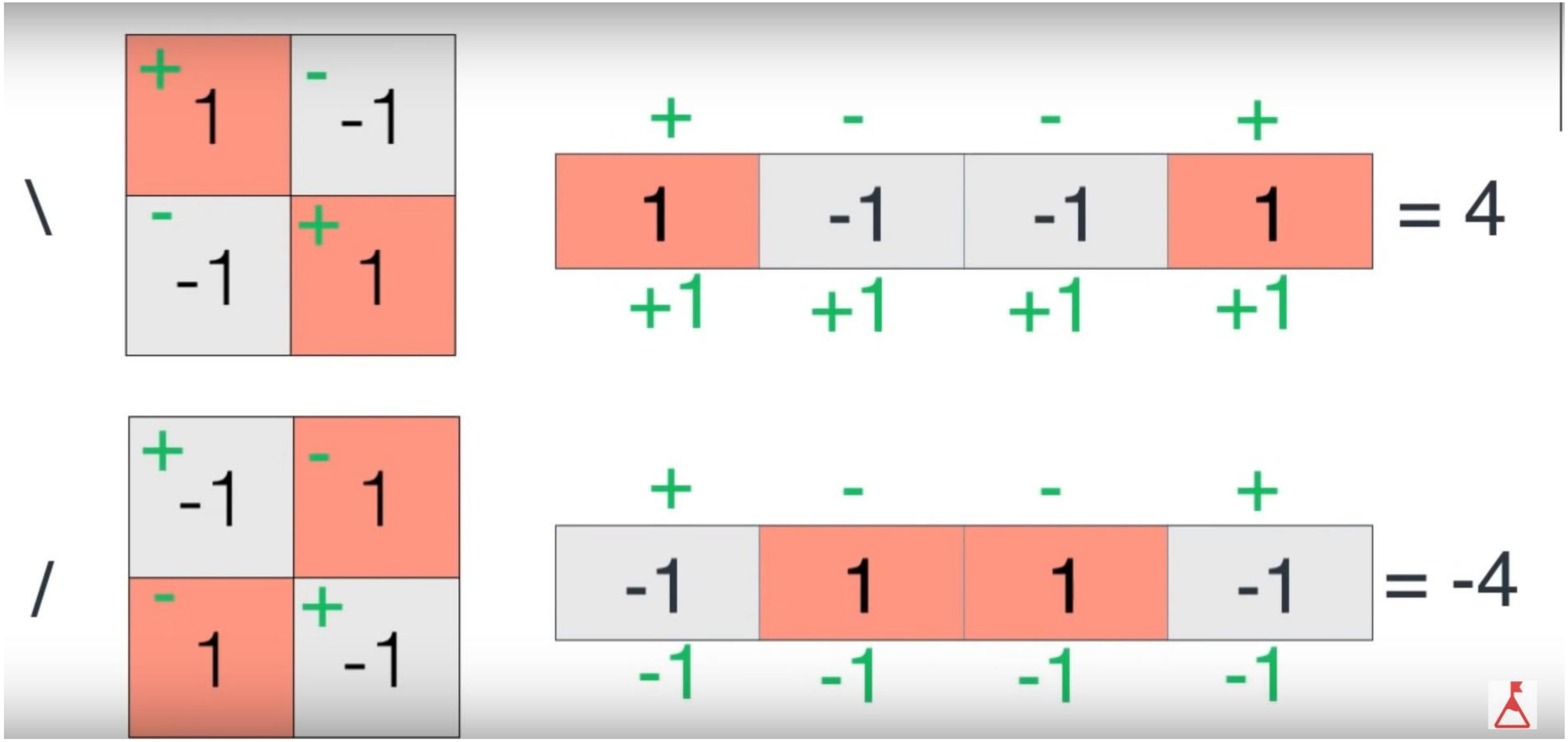
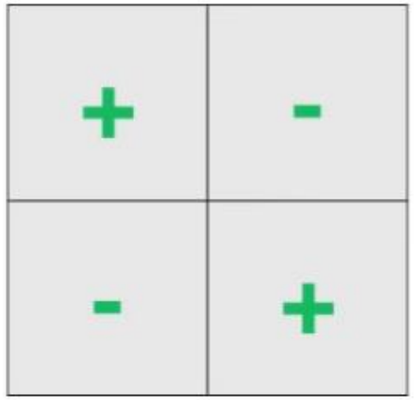
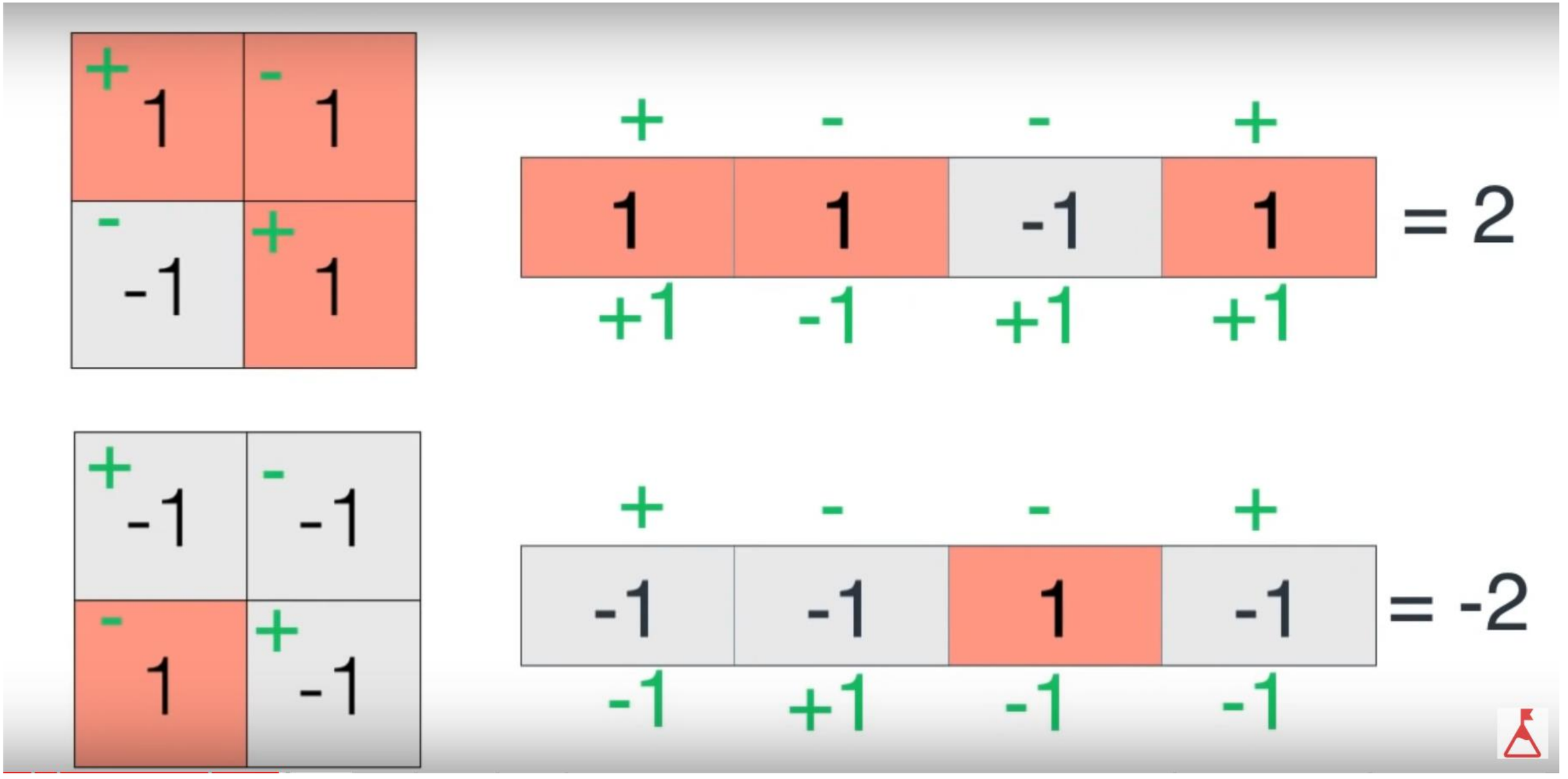

\
+ 1
- -1
- -1
+ 1
+ - - +
1 -1 -1 1
= 4
+1 +1 +1 +1
/
+ -1
- 1
- 1
+ -1
+ - - +
-1 1 1 -1
= -4
-1 -1 -1 -1

Image Recognition Classifier



If positive, “\”

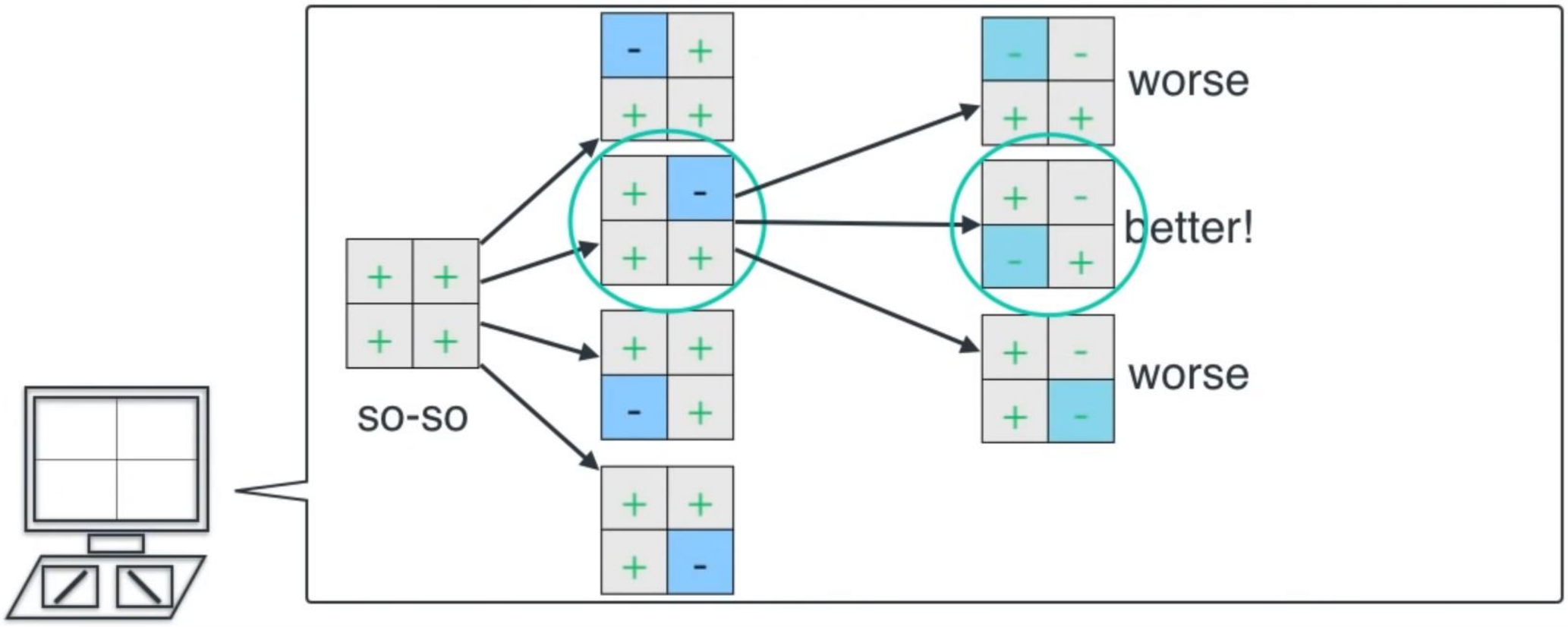
If negative, “/”


+ 1
- 1
- -1
+ 1
+ - - +
1 1 -1 1
= 2
+1 -1 +1 +1
+ -1
- -1
- 1
+ -1
+ - - +
-1 -1 1 -1
= -2
-1 +1 -1 -1

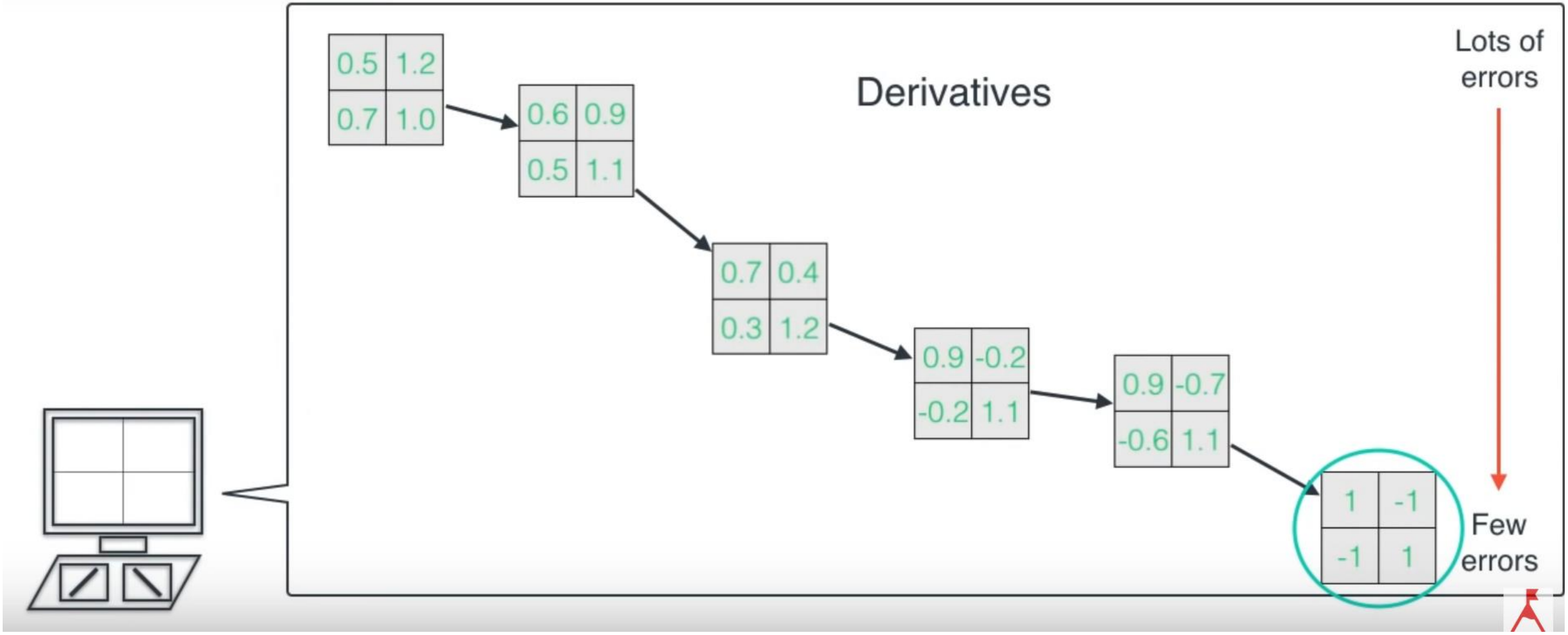
Artificial Intelligence

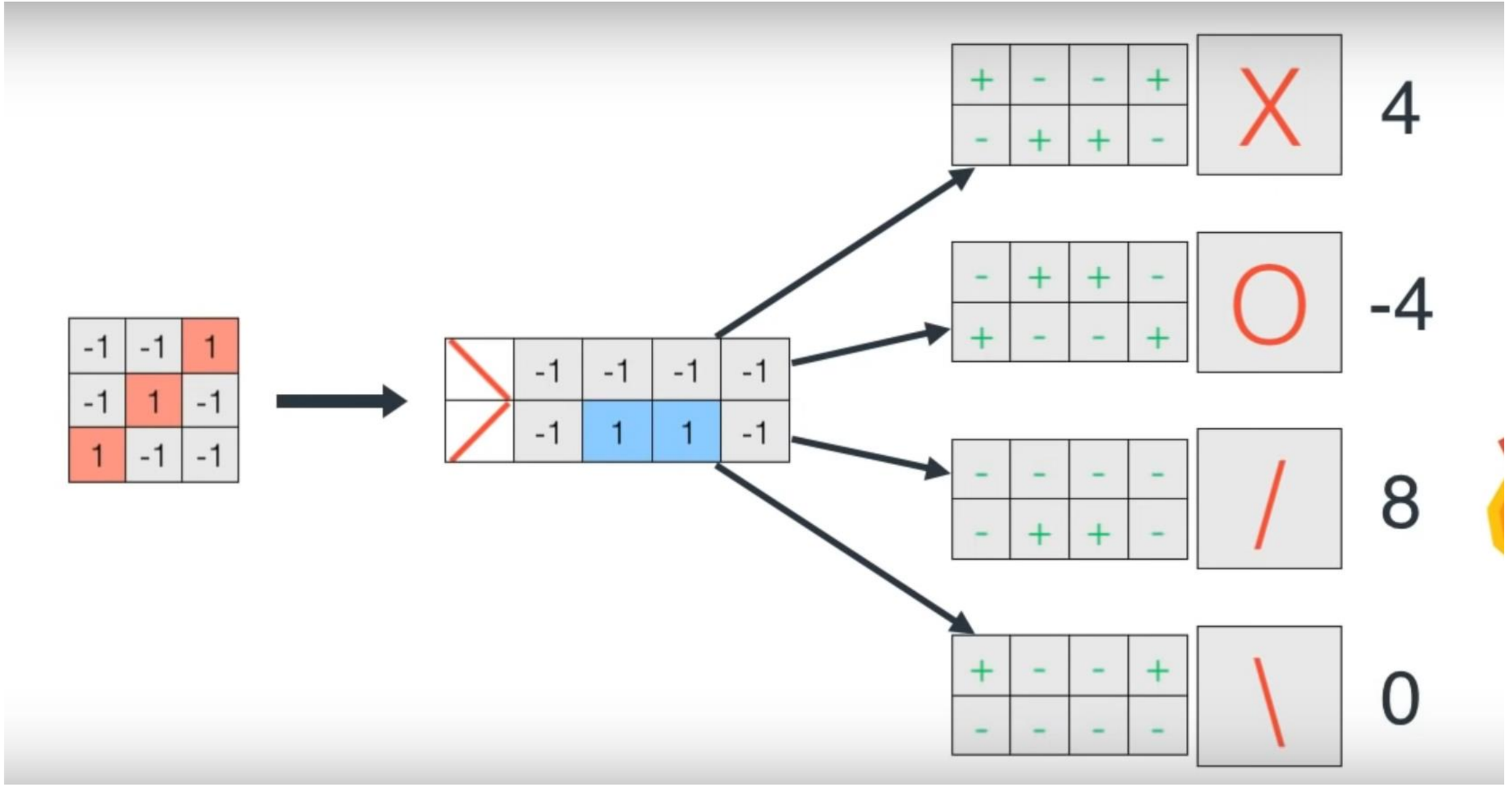
16 choices

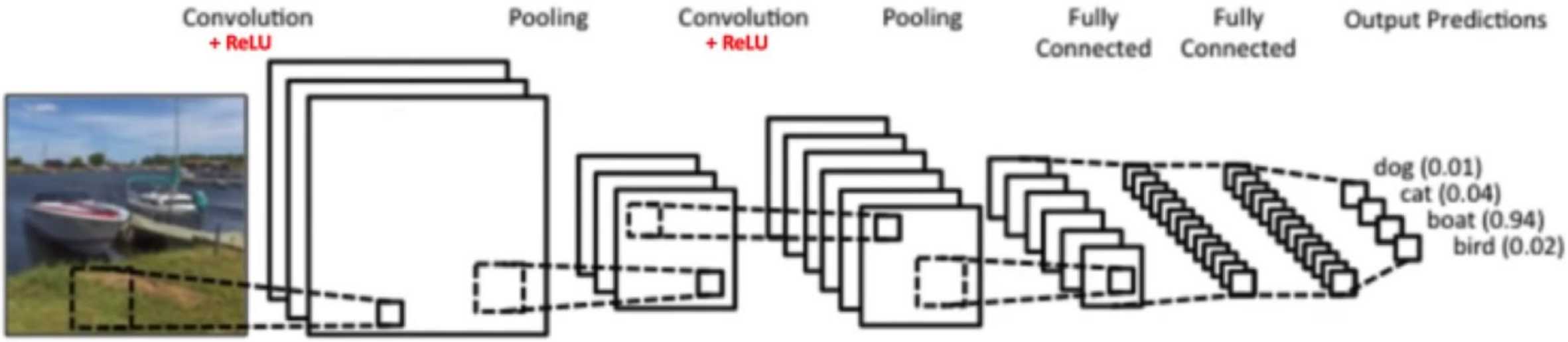
Artificial Intelligence



Gradient Descent




-1 -1 1
-1 1 -1
1 -1 -1
-1 -1 -1 -1
-1 1 1 -1
+ - - +
- + + -
X
4
- + + -
+ - - +
O
-4
- - - -
- + + -
/
8
+ - - +
- - - -
\
0


Convolution
+ ReLU
Pooling
Convolution
+ ReLU
Pooling
Fully
Connected
Fully
Connected
Output Predictions
dog (0.01)
cat (0.04)
boat (0.94)
bird (0.02)

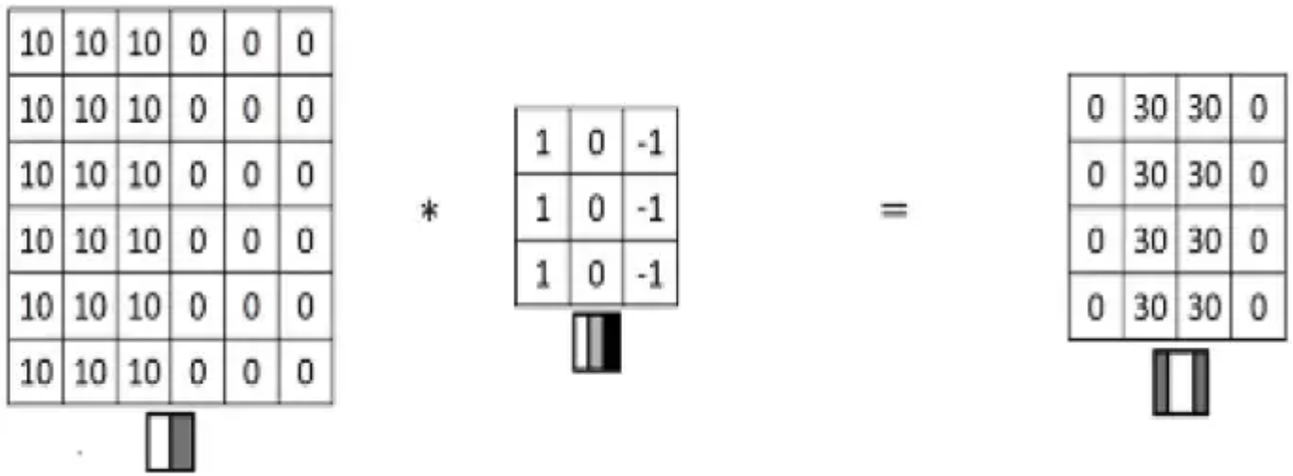


Image (m) = 6*6 matrix
Kernel (n) = 3 *3 matrix
Output image = (m-n+1) *(m-n+1)
=(6-3+1) * (6-3+1)
= 4 * 4

Issues with convolution

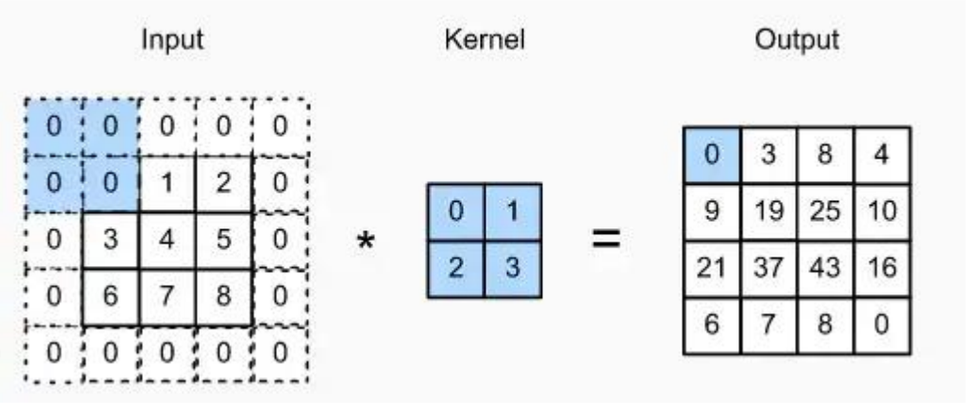
- Image gets shrink
- Touching the edges of the images will be very less comparatively , overlapping occurs in the middle.
- Corner features are not used much in the output

0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

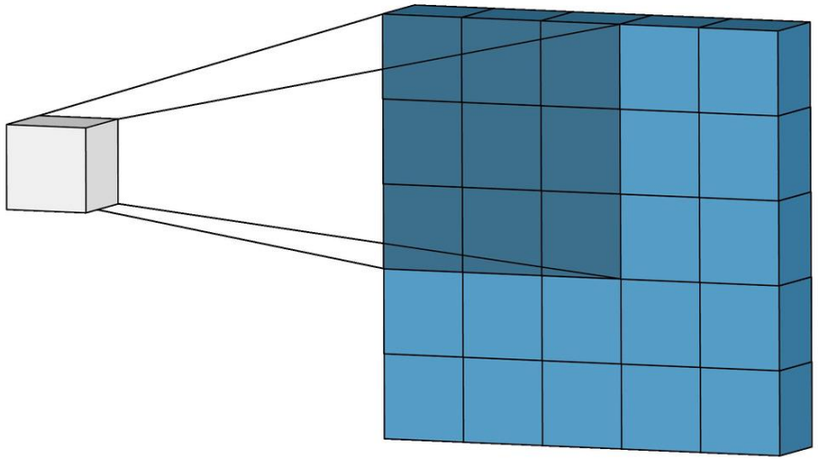
- Solution to issues in convolution

- To solve these issues padding is used
- (n + 2p — f + 1) * (n + 2p — f + 1)
- f-> filter
- p->padding size
- n-> image

3+2-2+1 = 4



Animation of cnn convolution

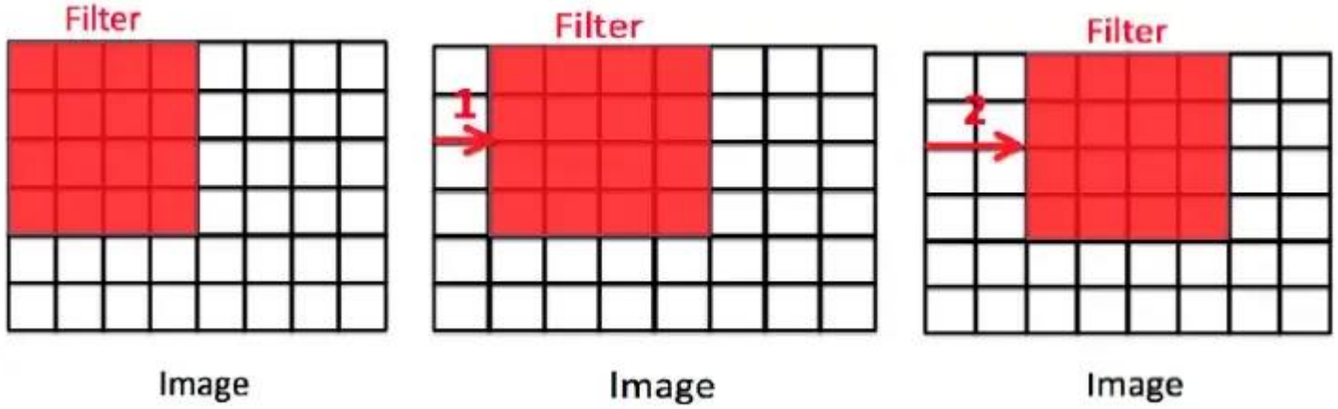


ReLu Activation function

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.0	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.0	-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	1.77

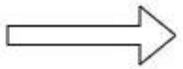
Examples of striding and pooling(average)



4	3	1	5
1	3	4	8
4	5	4	3
6	5	9	4

Avg([4, 3, 1, 3]) = 2.75

4	3	1	5
1	3	4	8
4	5	4	3
6	5	9	4



2.8	4.5
5.3	5.0