



2D Convolution (1)

Diagram showing the first input image $I_1(x,y)$ and its corresponding output $O_1(x,y)$. The input image is a 4x4 grid with values 100, 100, 0, 0 in the first two columns and 0, 0 in the last two columns. The output image is a 4x4 grid with values 0, -100, -100, 0 in the first two columns and 0, 0 in the last two columns.

	x	$I_1(x,y)$			
y	1	100	100	0	0
y	2	100	100	0	0
y	3	100	100	0	0
y	4	100	100	0	0

Diagram showing the kernel $k(x,y)$ used for the first convolution operation. The kernel is a 3x3 grid with values -1, 0, 1 in the first column and 0, 0 in the last two columns.

	x	$k(x,y)$		
y	1	-1	0	1
y	2	-1	0	1
y	3	-1	0	1

Diagram showing the second input image $I_2(x,y)$ and its corresponding output $O_2(x,y)$. The input image is a 4x4 grid with values 100, 100, 100, 0 in the first row and 100, 100, 0, 0 in the remaining rows. The output image is a 4x4 grid with values 0, 0, -100, -100 in the first row and 0, -100, -100, 0 in the remaining rows.

	x	$I_2(x,y)$			
y	1	100	100	100	0
y	2	100	100	0	0
y	3	100	100	0	0
y	4	100	100	0	0

Diagram showing the kernel $k(x,y)$ used for the second convolution operation. The kernel is a 3x3 grid with values -1, 0, 1 in the first column and 0, 0 in the last two columns.

	x	$k(x,y)$		
y	1	-1	0	1
y	2	-1	0	1
y	3	-1	0	1

Diagram showing the second output image $O_2(x,y)$. The output image is a 4x4 grid with values 0, 0, -100, -100 in the first row and 0, -100, -100, 0 in the remaining rows.

	x	$O_2(x,y)$			
y	1	0	0	-100	-100
y	2	0	-100	-100	0
y	3	0	-100	-100	0
y	4	0	-100	-100	0



2D Convolution (2)

$I_3(x,y)$

	x			
y				
	100	100	100	0
	100	100	100	0
	100	100	0	0
	100	100	0	0

$k(x,y)$

-1	0	1
-1	0	1
-1	0	1

$O_3(x,y)$

	x			
y				
	0	0	-100	-100
	0	0	-100	-100
	0	-100	-100	0
	0	-100	-100	0

$I_4(x,y)$

	x			
y				
	100	100	100	0
	100	100	100	0
	100	100	100	0
	100	100	0	0

$k(x,y)$

-1	0	1
-1	0	1
-1	0	1

$O_4(x,y)$

	x			
y				
	0	0	-100	-100
	0	0	-100	-100
	0	0	-100	-100
	0	-100	-100	0



2D Convolution (3)

Diagram showing the input image $I_5(x,y)$ for the first convolution step. The x and y axes are indicated by arrows.

	x	$I_5(x,y)$			
y		100	100	100	0
		100	100	100	0
		100	100	100	0
		100	100	100	0

Diagram showing the kernel $k(x,y)$ for the first convolution step.

	x	$k(x,y)$		
y		-1	0	1
		-1	0	1
		-1	0	1

Diagram showing the output image $O_5(x,y)$ for the first convolution step.

	x	$O_5(x,y)$			
y		0	0	-100	-100
		0	0	-100	-100
		0	0	-100	-100
		0	0	-100	-100

Diagram showing the input image $I_6(x,y)$ for the second convolution step. The x and y axes are indicated by arrows.

	x	$I_6(x,y)$			
y		0	100	100	0
		100	100	100	0
		100	100	100	0
		100	100	100	0

Diagram showing the kernel $k(x,y)$ for the second convolution step.

	x	$k(x,y)$		
y		-1	0	1
		-1	0	1
		-1	0	1

Diagram showing the output image $O_6(x,y)$ for the second convolution step.

	x	$O_6(x,y)$			
y		100	100	-100	-100
		0	0	-100	-100
		0	0	-100	-100
		0	0	-100	-100



2D Convolution (4)

$I_7(x,y)$

$k(x,y)$

-1	0	1
-1	0	1
-1	0	1

$O_7(x,y)$

$I_8(x,y)$

		0	100	100	0
		0	100	100	0
		100	100	100	0
		100	100	100	0

$k(x,y)$

-1	0	1
-1	0	1
-1	0	1

$O_8(x,y)$

	100	100	-100	-100
	100	100	-100	-100
	0	0	-100	-100
	0	0	-100	-100



2D Convolution (5)

$I_9(x,y)$

	0	100	100	0
	0	100	100	0
	0	100	100	0
y	100	100	100	0

$k(x,y)$

-1	0	1
-1	0	1
-1	0	1

$O_9(x,y)$

	100	100	-100	-100
	100	100	-100	-100
	100	100	-100	-100
	0	0	-100	-100

$I_{10}(x,y)$

	0	100	100	0
	0	100	100	0
	0	100	100	0
	0	100	100	0

$k(x,y)$

-1	0	1
-1	0	1
-1	0	1

$O_{10}(x,y)$

	100	100	-100	-100
	100	100	-100	-100
	100	100	-100	-100
	100	100	-100	-100



Kernel Coefficients to Neural Nets

One Image Plane

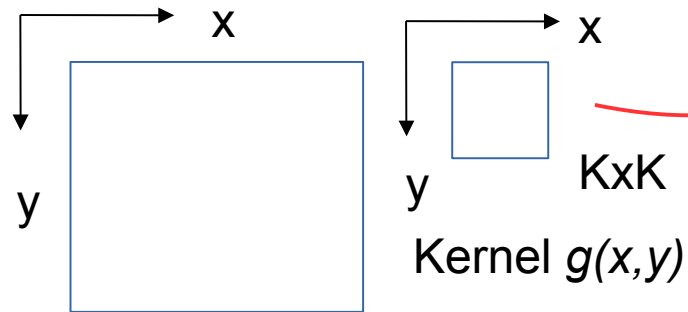
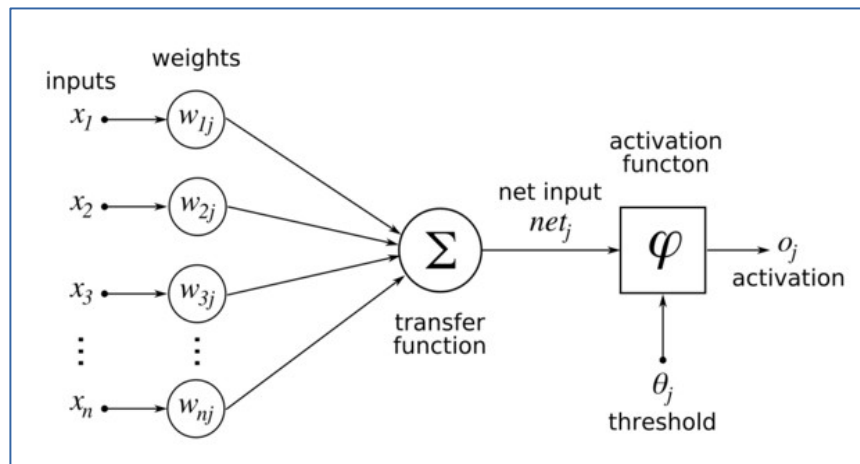


Image $f(x,y)$ (N-1,M-1)

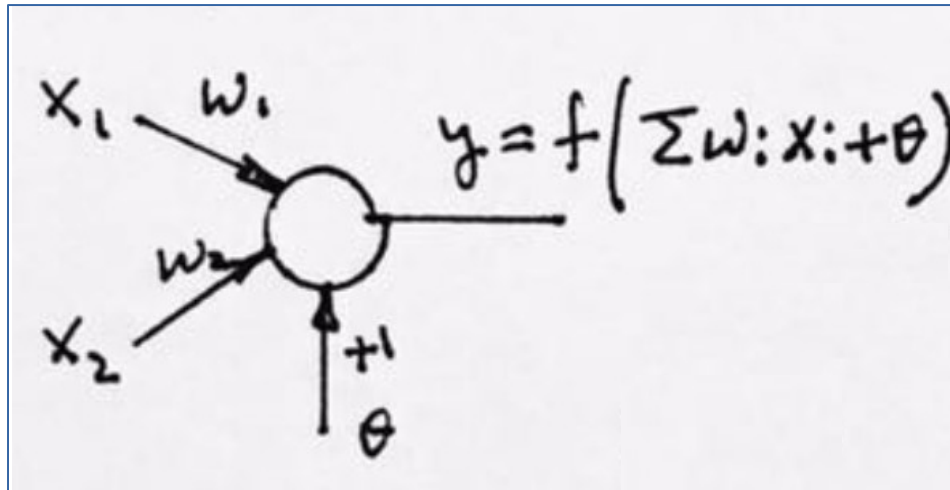
w_{11}	w_{12}	w_{13}
w_{21}	w_{22}	w_{23}
w_{31}	w_{32}	w_{33}

N layers



Input from image $I(x,y)$	weight
x_{11}	w_{11}
x_{12}	w_{12}
x_{13}	w_{13}
x_{21}	w_{21}
x_{22}	w_{22}
x_{23}	w_{23}
x_{31}	w_{31}
x_{32}	w_{32}
x_{33}	w_{33}

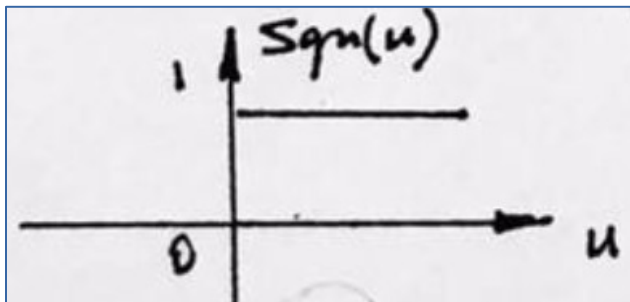
Map to Single Layer NN (1)



$$y = f(\sum w_i x_i + \theta) \quad \dots (1)$$

Where

$$y = f(\cdot) \text{ as } \text{sgn}(\cdot) \quad \dots (2)$$



Group 1, C1, as those from edge pixels

Group 2, C2, as those from non-edge pixels

Hence, from the following, we have

100	100	0	0
100	100	0	0
100	100	0	0
100	100	0	0

0	-100	-100	0
0	-100	-100	0
0	-100	-100	0
0	-100	-100	0

(1,1), $y = 0$; (1,2), $y=1$; (1,3), $y=1$; (1,4), $y=0$

(2,1), $y = 0$; (2,2), $y=1$; (2,3), $y=1$; (2,4), $y=0$

(3,1), $y = 0$; (3,2), $y=1$; (3,3), $y=1$; (3,4), $y=0$

(4,1), $y = 0$; (4,2), $y=1$; (4,3), $y=1$; (4,4), $y=0$

Continue till the last pair as



Map to Single Layer NN (2)

From the following,

0	100	100	0
0	100	100	0
0	100	100	0
0	100	100	0

100	100	-100	-100
100	100	-100	-100
100	100	-100	-100
100	100	-100	-100

(1,1), $y = 1$; (1,2), $y=1$; (1,3), $y=1$; (1,4), $y=1$
(2,1), $y = 1$; (2,2), $y=1$; (2,3), $y=1$; (2,4), $y=1$
(3,1), $y = 1$; (3,2), $y=1$; (3,3), $y=1$; (3,4), $y=1$
(4,1), $y = 1$; (4,2), $y=1$; (4,3), $y=1$; (4,4), $y=1$

The training algorithm:

$$w_i^+ = w_i^- + \eta (d - y) x_i$$

where

η is gain.
 d : desired the output.
 y : actual output.



Map to Single Layer NN (2)

From the following,

0	100	100	0
0	100	100	0
0	100	100	0
0	100	100	0

100	100	-100	-100
100	100	-100	-100
100	100	-100	-100
100	100	-100	-100

(1,1), y = 1; (1,2), y=1; (1,3), y=1; (1,4), y=1
 (2,1), y = 1; (2,2), y=1; (2,3), y=1; (2,4), y=1
 (3,1), y = 1; (3,2), y=1; (3,3), y=1; (3,4), y=1
 (4,1), y = 1; (4,2), y=1; (4,3), y=1; (4,4), y=1

The training algorithm:

$$w_i^+ = w_i^- + \eta (d - y) x_i$$

where

*η is gain.
 d : desired the output.
 y : actual output.*

So, we have

Input from image I(x,y)	weight	Output image O(x,y)
x_11	w_11	y_11
x_12	w_12	y_12
x_13	w_13	y_13
x_21	w_21	y_21
x_22	w_22	y_22
x_23	w_23	y_23
x_31	w_31	y_31
x_32	w_32	y_32
x_33	w_33	y_33

Now, train the NN find weights w_{ij}