c_t h_t

Long Short-Term Memory (LSTM)

An LSTM is an RNN in which the recurrence conditions are changed.
Along with $h_t^{(k)}$ we have an additional hidden vector of p dimensions.
This is called as the cell state.
The cell state is a kind of long-term memory that retains at least a part of the information in earlier states by using a combination of partial forgetting and increment operations on the previous cell states.

element-wise operation

$$\begin{array}{l} \text{Input Gate} \\ \text{Forget Gate} \\ \text{Output gate} \\ \text{New Cell state (candidate)} \end{array} \begin{bmatrix} i \\ f \\ o \\ c \end{bmatrix} \begin{array}{l} p\ [0,1] \\ p\ [0,1] \\ p\ [0,1] \\ p\ [-1,1] \end{array} \begin{bmatrix} \text{sigmoid} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{bmatrix} \underbrace{W^{(k)} \begin{bmatrix} h_t^{(k-1)} \\ h_{t-1}^{(k)} \end{bmatrix}}_{} \; 2p$$

$4p$

$4p \times 2p$

element-wise multiplication $\odot$

Cell: $$c_t^{(k)} = \underbrace{f \odot c_{t-1}^{(k)}}_{\text{select forget}} + \underbrace{i \odot c}_{\text{selective addition}}$$

Selectively forget and add to long-term memory

hidden: $$h_t^{(k)} = \underbrace{o}_{\text{output Gate}} \odot \tanh\left(c_t^{(k)}\right)$$

selectively leak long-term memory to hidden state.

For the first layer $(k=1)$ $h_t^{(k-1)}$ should be replaced with x_t

$k>1$ $W^{(k)}$ is of size $4p \times 2p$

$k=1$ $W^{(1)}$ is of size $4p \times (p+d)$

h_t p

0.9 0.0 0.05 0.09

0.9 0.9 0.5 0.4

o c_t

Gated Recurrent Unit

Update Gate

Reset Gate

$$\begin{bmatrix} \underline{z} \\ \underline{r} \end{bmatrix} \begin{matrix} [0,1] \\ [0,1] \end{matrix} = \begin{bmatrix} \text{sigm} \\ \text{sigm} \end{bmatrix} \overset{2p \times 2p}{\underline{W}^{(k)}} \begin{bmatrix} h_t^{(k-1)} \\ h_{t-1}^{(k)} \end{bmatrix}$$

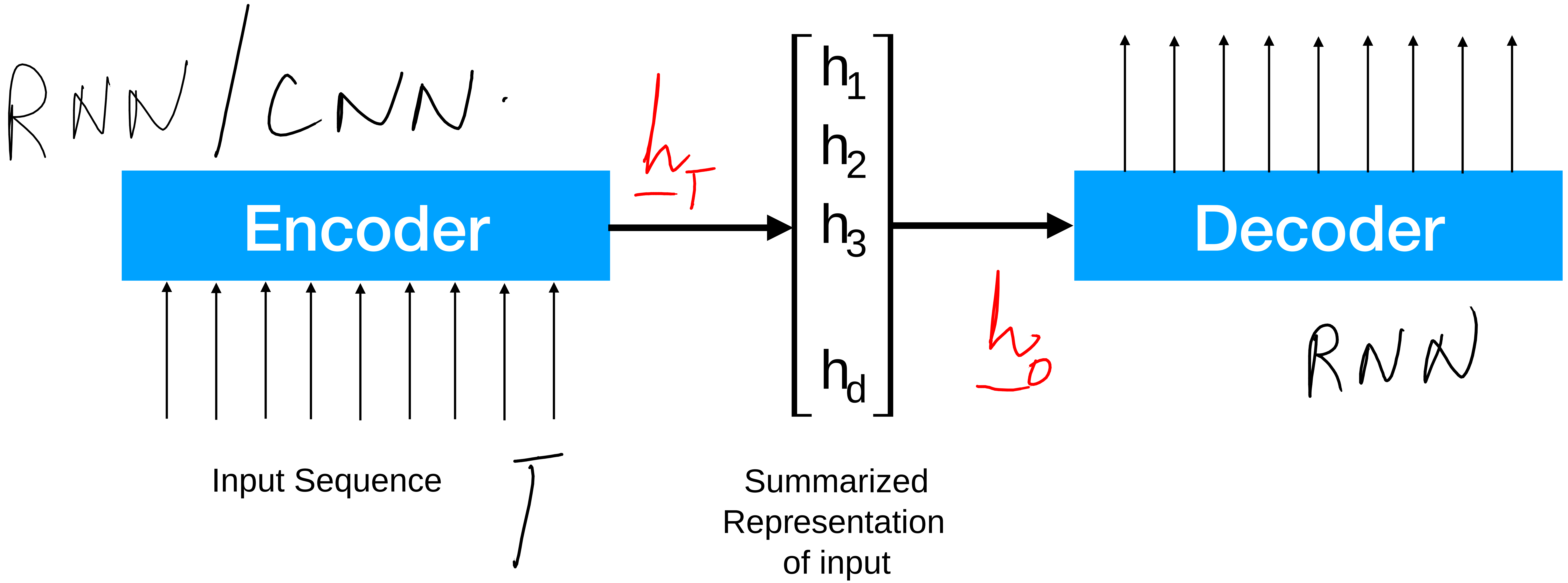
$$\underline{h}_t^{(k)} = \underbrace{\underline{z} \odot \underline{h}_{t-1}^{(k)}}_{\text{previous}} + \underbrace{(1-\underline{z}) \odot \underbrace{\tanh}_{\text{element-wise}} \overset{p \times 2p}{\underline{V}^{(k)}} \begin{bmatrix} \underline{h}_t^{(k-1)} \\ \underline{r} \odot h_{t-1}^{(k)} \end{bmatrix}}_{\text{new proposal}}$$

For $k=1$ $\underline{h}_t^{(k-1)}$ is to be replaced by $\underline{x}_t$.

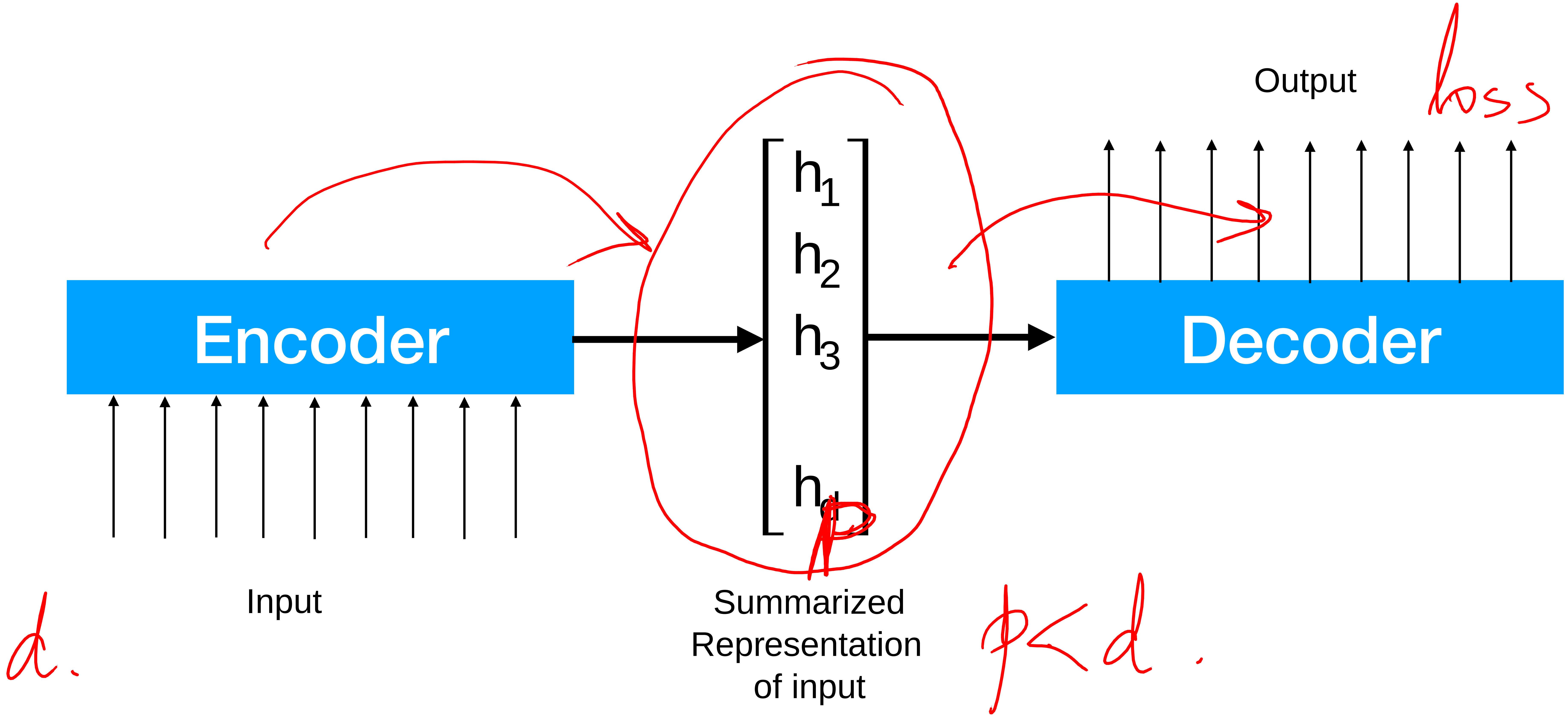
$\underline{W}^{(k)}$ and $\underline{V}^{(k)}$ are of sizes $2p \times 2p$ and $p \times 2p$, respectively

For $k=1$ $W^{(1)}$ and $V^{(1)}$ are of sizes $2p \times (p+d)$ and $p \times (p+d)$

Encoder-Decoder Architecture



Auto-Encoder Architecture



In an auto-encoder, the input and output sequences are the same

A simple filter

- This filter can detect horizontal lines

$$\begin{bmatrix} 1.0 & 1.0 & 1.0 & 1.0 \\ 1.0 & 1.0 & 1.0 & 1.0 \\ -1.0 & -1.0 & -1.0 & -1.0 \\ -1.0 & -1.0 & -1.0 & -1.0 \end{bmatrix}$$

Convolution Operation

$$V(x, y) \;=\; (I \cdot K)(x, y) \;=\sum_{m}\sum_{n}\; I(x+m, y+n)\;\; K(m, n)$$

2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40

Image

−1.0	−1.0	−1.0	−1.0
−1.0	−1.0	−1.0	−1.0
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0

Filter / Kernel

6	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	6	4
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	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	0	0
54	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	54	36
108	144	144	144	144	144	144	144	144	144	144	144	144	144	144	144	144	144	108	72
54	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	54	36
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	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	0	0
60	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	60	40
120	160	160	160	160	160	160	160	160	160	160	160	160	160	160	160	160	160	120	80
60	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	60	40
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	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	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	0	0	0	0	0	0	0	0	0	0	0	0	0
−120	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−160	−120	−80
−240	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−320	−240	−160

Output of convolution of the image with the filter given in the previous slide

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	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	2	2	2	2	2
2	2	2	2	2	2

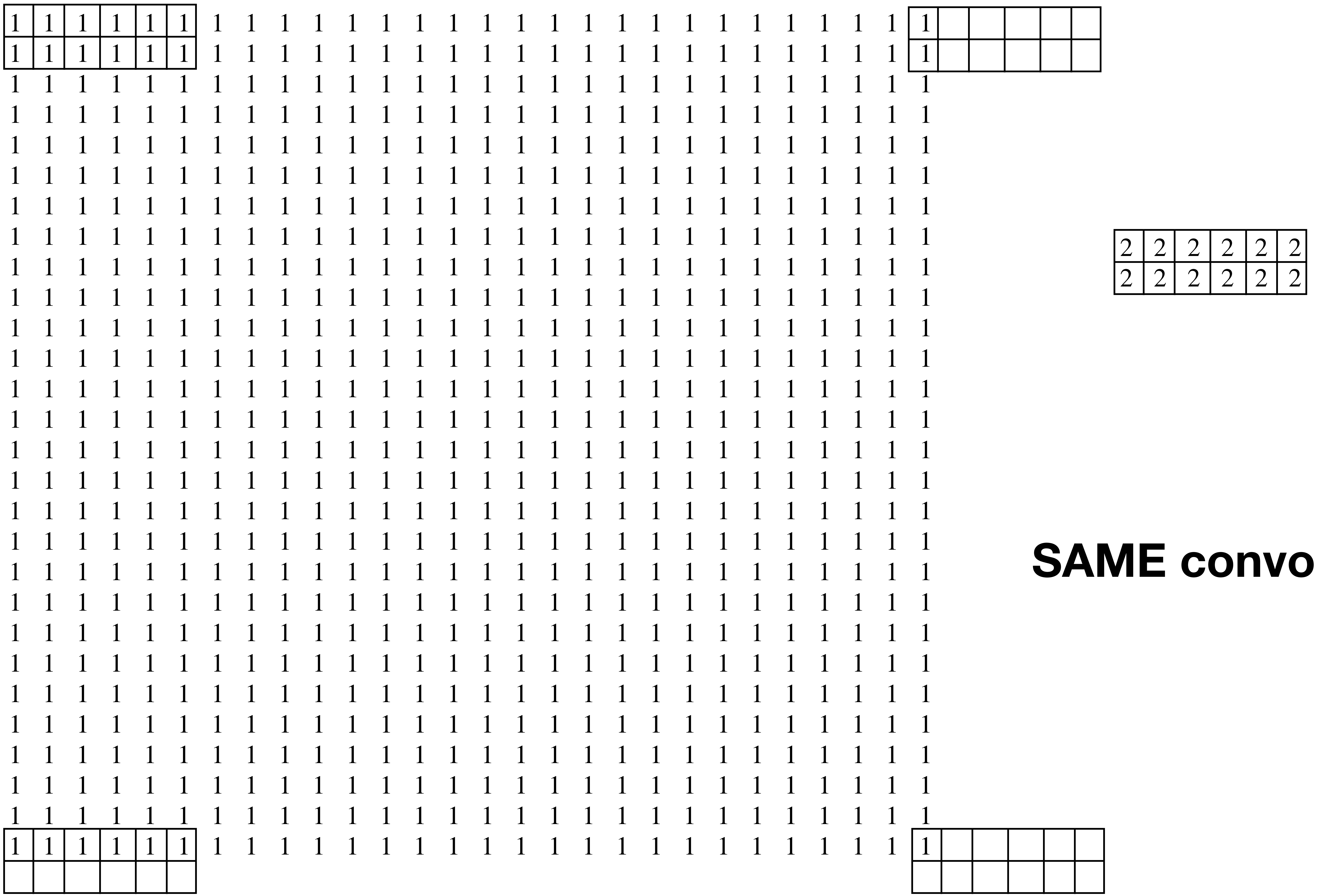
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	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	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	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	2	2	2	2	2
2	2	2	2	2	2

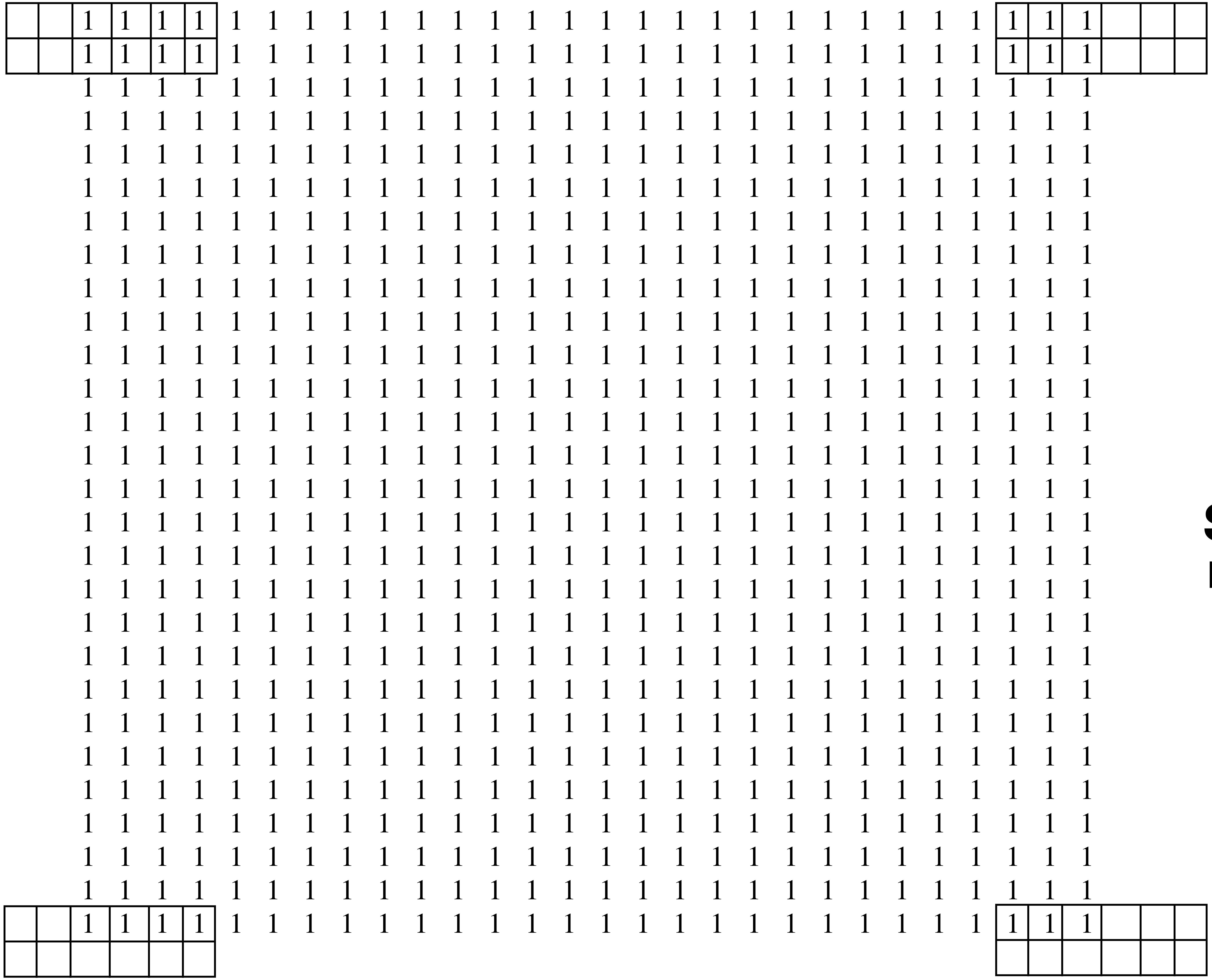
VALID convolution

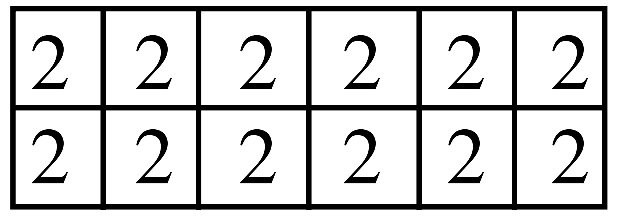
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
2	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
3	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
4	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
5	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
6	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
7	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
8	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
9	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
10	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
11	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
12	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
13	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
14	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
15	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
16	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
17	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
18	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
19	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
21	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
22	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
23	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
25	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
26	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24
27	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24

VALID convolution output shape:
[1 27 23 1]



SAME convolution





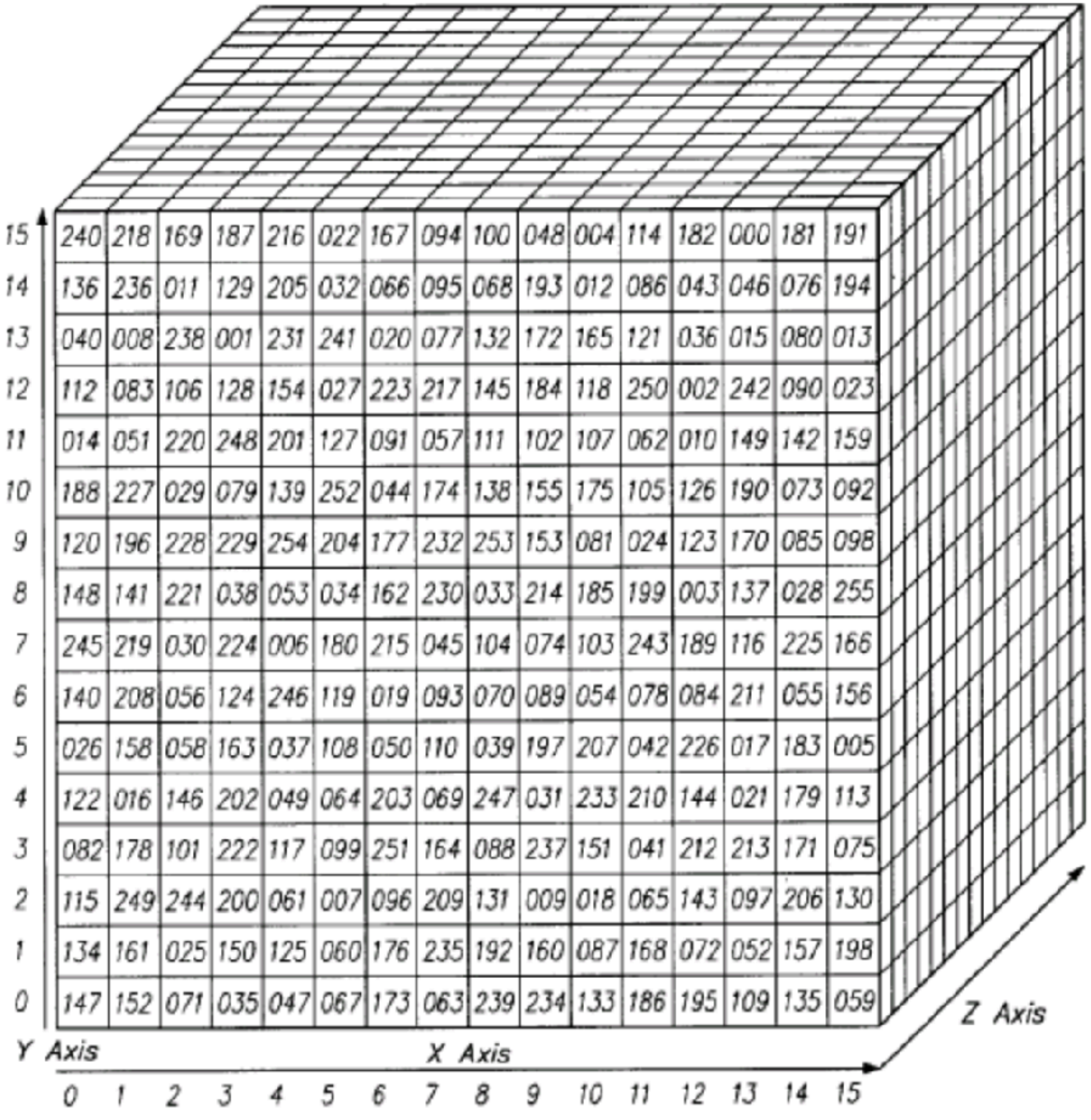
SAME convolution

Padding evenly distributed
at both ends
left & right,
top & bottom.

	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
2	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
3	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
4	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
5	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
6	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
7	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
8	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
9	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
10	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
11	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
12	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
13	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
14	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
15	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
16	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
17	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
18	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
19	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
20	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
21	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
22	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
23	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
24	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
25	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
26	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
27	16	20	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	20	16	12
28	8	10	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	10	8	6

```
SAME convolution
output shape:
[ 1 28 28  1]
```

SAME convolution



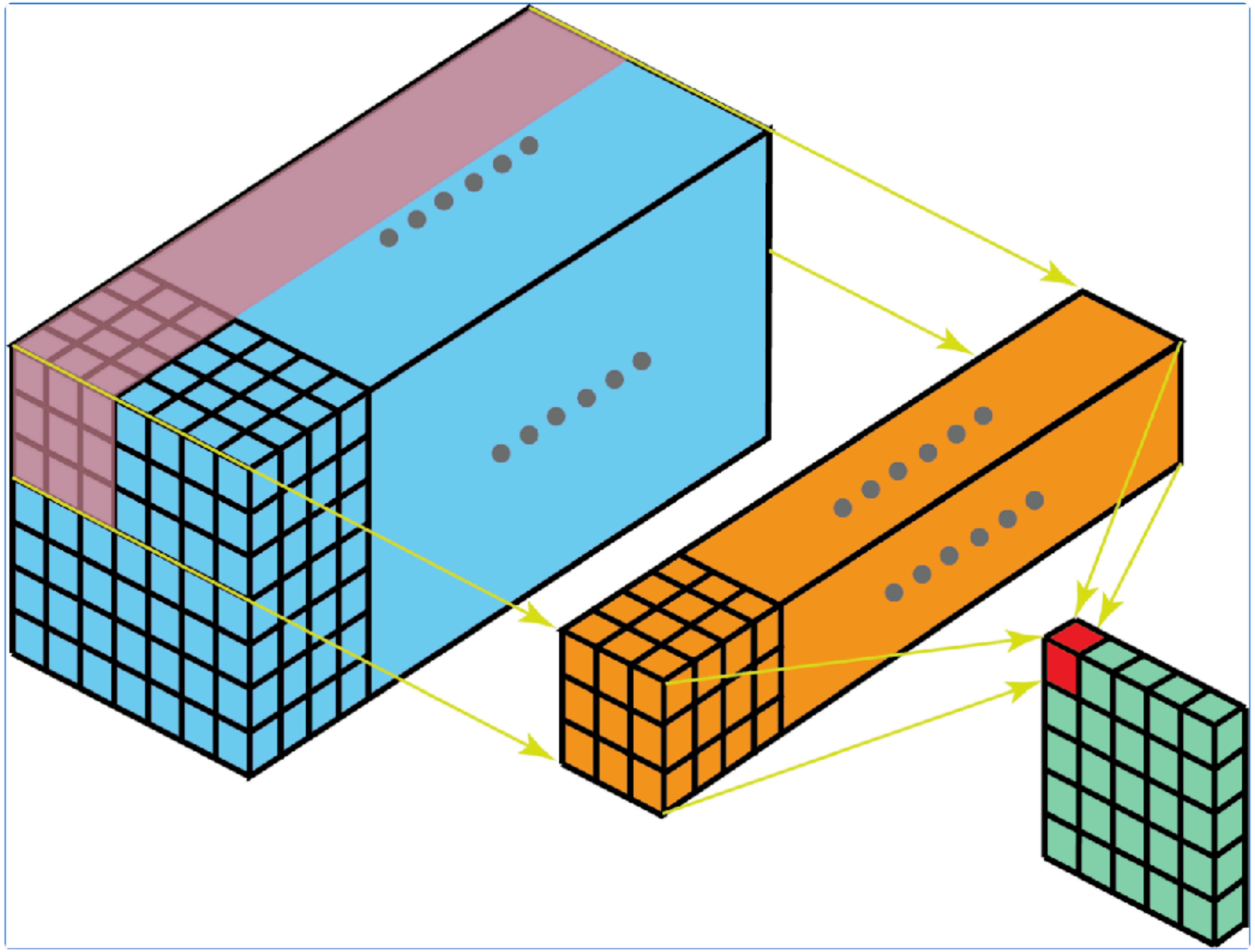
Collection of images in a batch

- **Z axis is the first axis**
  Number of images in the batch

- **Y axis is the second axis**
  Height of an image

- **X axis is the third axis**
  Width of an image

The 4th axis is the #channels



Example of convolution on a single example in the batch

Showing a single image (as a box) having multiple channels

Input to the convolution is a 4D tensor

[batchsize, height, width, #channels]

A batch comprising a single channel 4x4 image can be shown as:

Size: [1, 4, 4, 1]

```
[ [ [ [1], [1], [1], [1] ],
      [ [1], [1], [1], [1] ],
      [ [1], [1], [1], [1] ],
      [ [1], [1], [1], [1] ]   ] ]
```

Convolution filter is a 4D tensor

[height, width, #channels, #filters]. #filters= #OUTPUTchannels

[height, width, #INPUTchannels, #OUTPUTchannels]

A single filter of size 3x2 applied on a single channel image:

Size: [3, 2, 1, 1]

```
[ [ [[0]], [[1] ] ],
  [ [[0]], [[1]] ],
  [ [[0]], [[1]] ]   ]
```

Multi-channel convolution

Output of convolution

Input Image [batchsize, height, width, #channels]

Filter [height, width, #channels, #OUTPUTchannels]

Output Image [batchsize, height, width, #OUTPUTchannels]

#OUTPUTchannels are the number of filters used for convolution

Multi-channel convolution
Box convolution

Advantages of Convolution

- Fewer parameters in the model
  because the same kernel (i.e. the same weights) is used at multiple locations in the image.

- We are putting very strong priors (infinitely strong priors) on the model:
  1) Sharing of weights
  2) Small receptive field (kernel size) ==> ZERO weight connection with all other neurons in the previous layer !

- Repeated features can be easily identified as the kernel moves over different locations. With the same set of kernels one can process larger images also! The network does not need to be retrained to process larger images.

- Max pooling achieves translation invariance.