```
tf.nn.conv2d(input, filter, strides, padding,
             use_cudnn_on_gpu = None,
             data_format = None,
             name = None)
```

input

this is a tensor of shape

```
[ batch        ]  number of images
[ in_height    ]  height of image
[ in_width     ]  width of image
[ in_channels  ]  number of channels
```

filter

this is a kernel tensor of shape

```
[ filter_height  ]  height of filter
[ filter_width   ]  width of filter
[ in_channels    ]  number of channels
[ out_channels   ]  number of filters
```

↑ also the number of channels in result image.

result tensor/image

```
op = tf.nn.conv2d(input, filter, strides = [1,1,1,1], padding = 'SAME')
```

strides: could also be [1,2,2,1] or other...

padding: could also be 'VALID'

4D

for (5)

```
input = tf.Variable(tf.random_normal([1,5,5,5]))
filter = tf.Variable(tf.random_normal([3,3,5,7]))
```

0

input image₀ ← with 1 channel

3	4
5	6

filter₀

2

filter₀'s size:

[1, 1, 1, 1]

width, height, number of channels, number of filters

input image₀'s size:

[1, 2, 2, 1]

number of images, width, height, number of channels

result image₀ = input image₀ .* filter₀

3	4
5	6

.* | 2 | ← filter₀

= | 6 | 8 |
|---|---|
| 10 | 12 |

‖ result image₀'s size: [1, 2, 2, 1]

1

filter₁ =

2

a tube

‖ filter₁'s size: [1, 1, 5, 1]

input image₁ =

3	4	5
6	7	8
9	10	11

‖ input image₁'s size: [1, 3, 3, 5]

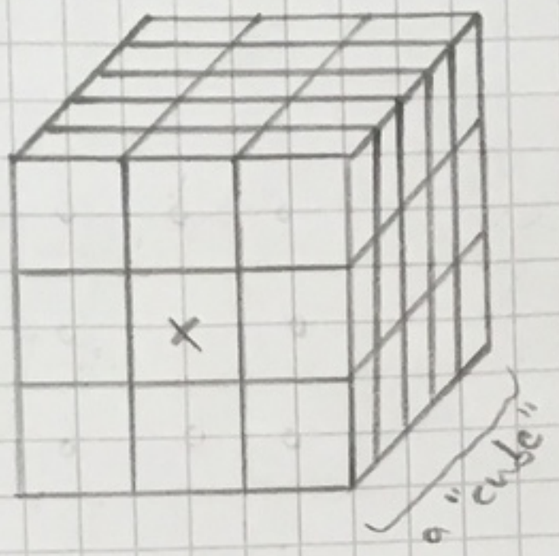
result image₁ =

12	17	9
15	18	6
21	13	7

‖ result image₁'s size: [1, 3, 3, 1]

dot products of all corresponding tubes.

filter$_2$ =



filter$_2$'s size: [3, 3, 5, 1]

inputimage$_2$ = inputimage$_1$

inputimage$_2$'s size:
[1, 3, 3, 5]

resultimage$_2$ = □

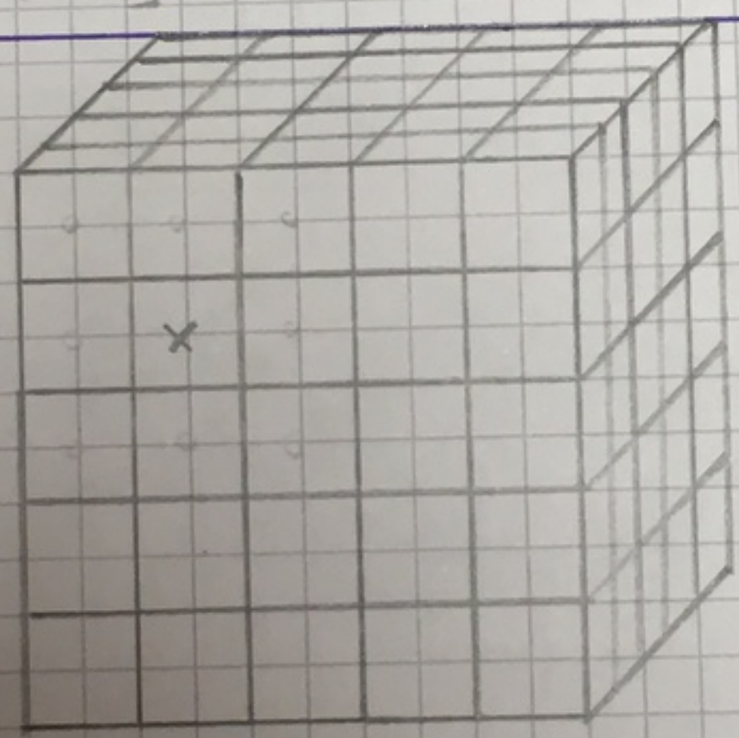
dot-product of everything

resultimage$_2$'s size:
[1, 1, 1, 1]

---

(3)

inputimage$_3$ =

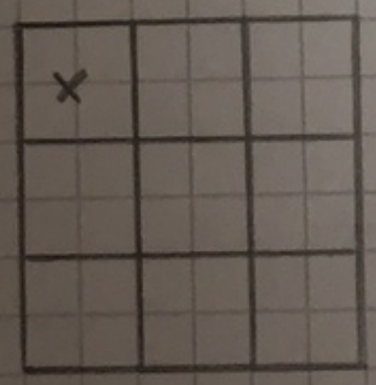


inputimage$_3$'s size:
[1, 5, 5, 5]

filter$_3$'s size:
[3, 3, 5, 1]

filter$_3$ = filter$_2$

resultimage$_3$ =



dot products of all corresponding cubes.

resultimage$_3$'s size:
[1, 3, 3, 1]

padding = 'VALID'

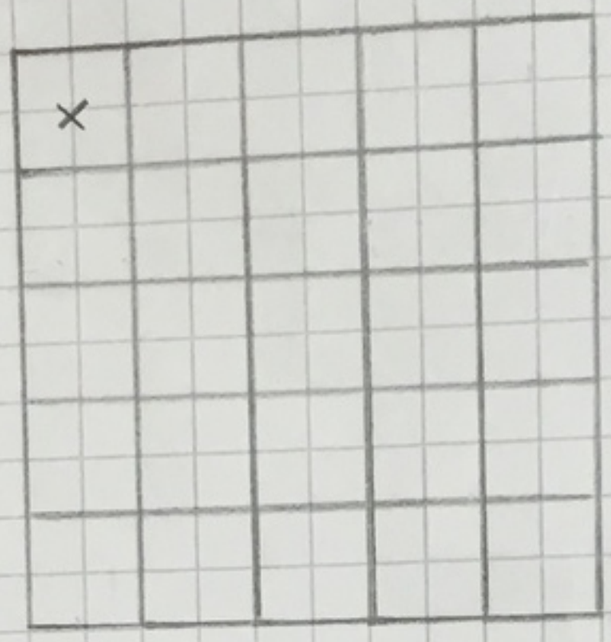
(4)

$input image_4 = input image_3$

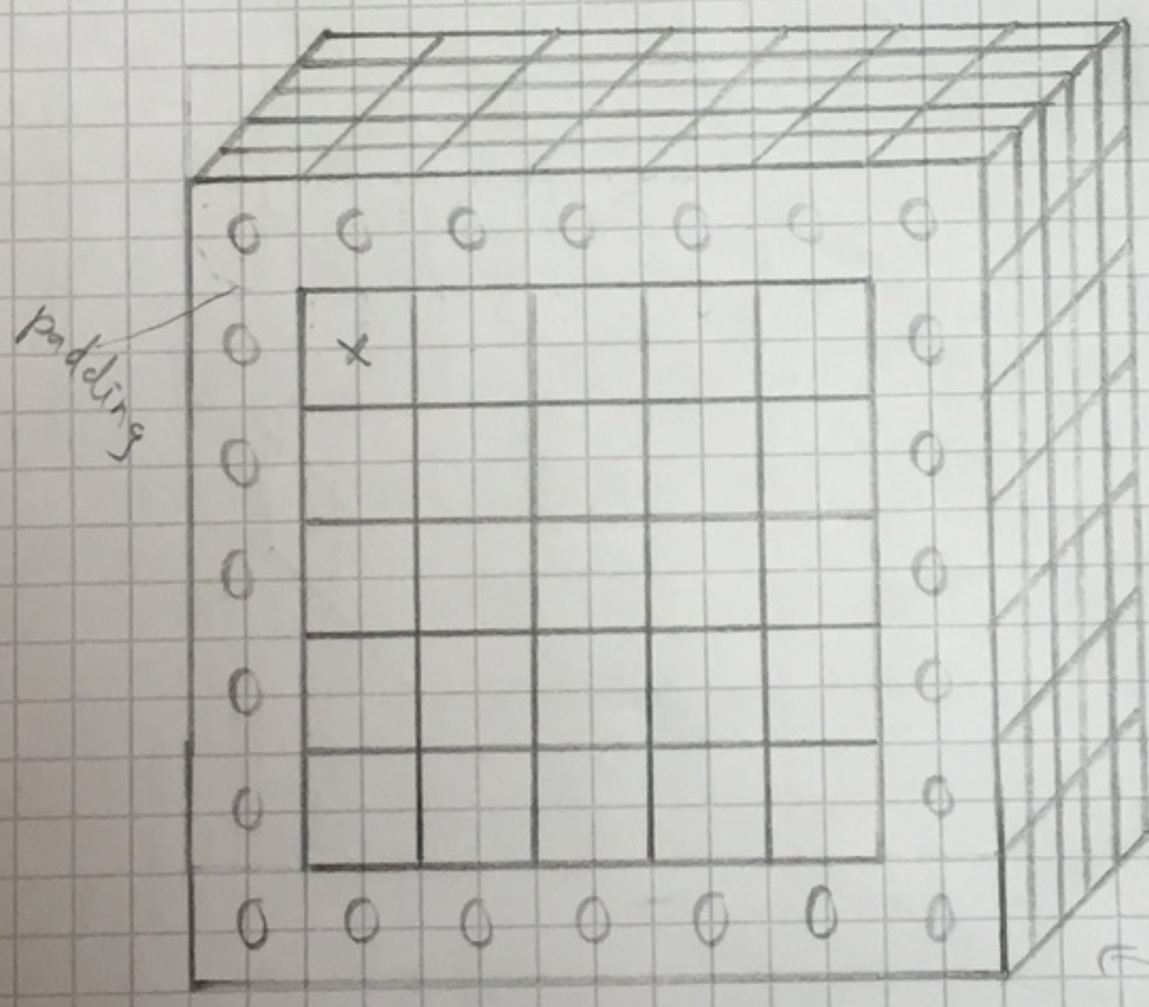
$filter_4 = filter_2$

padding = 'SAME'

$result image_4$ =



$input image_4$ (including padding) =



$input image_4$'s size:
[1, 5, 5, 5]

$filter_4$'s size:
[3, 3, 5, 1]

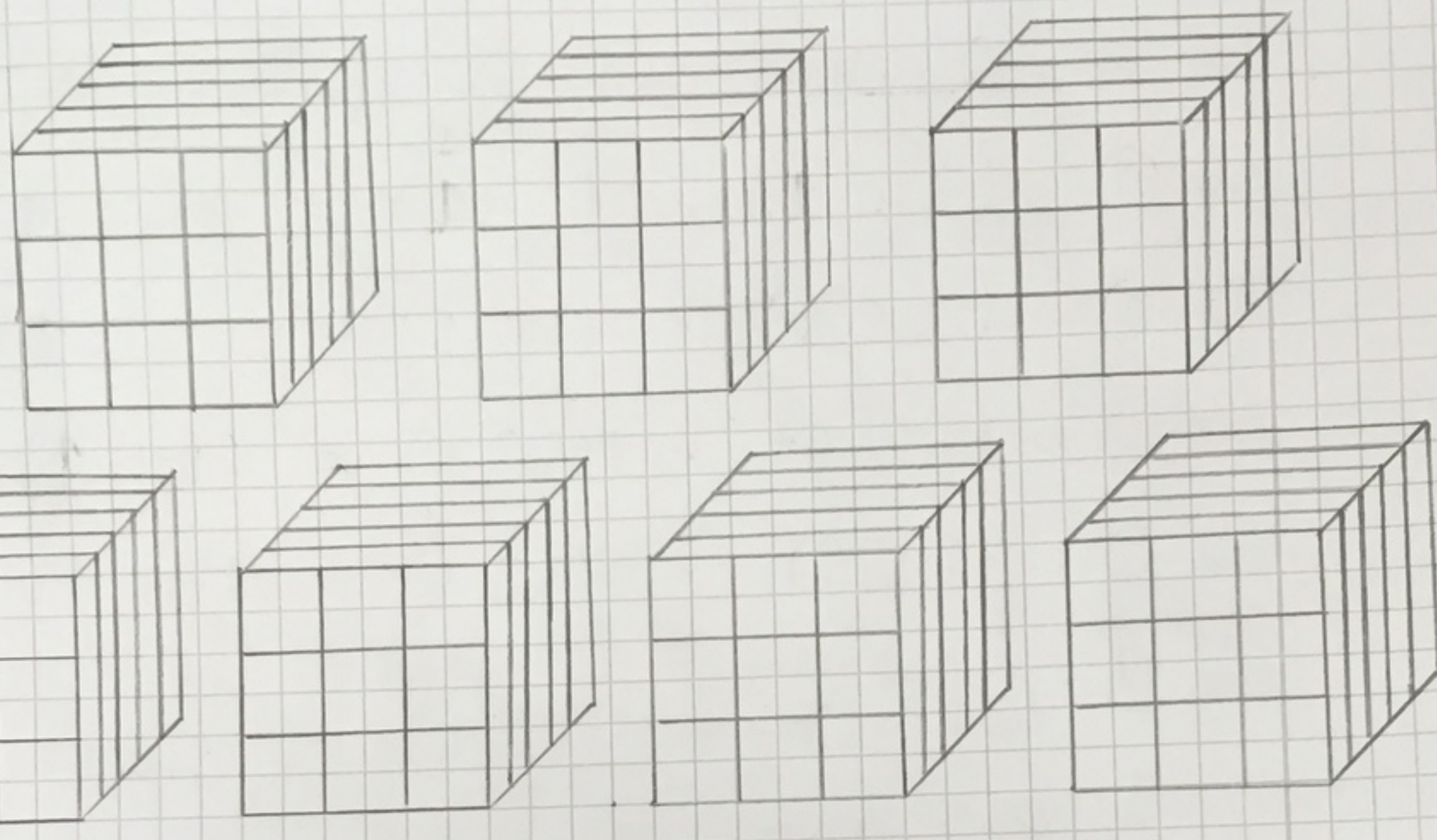
$result image_4$'s size:
[1, 5, 5, 1]

size of $input image_4$:
[1, 5, 5, 5]

size of $input image_4$ with padding:
[1, 7, 7, 5]

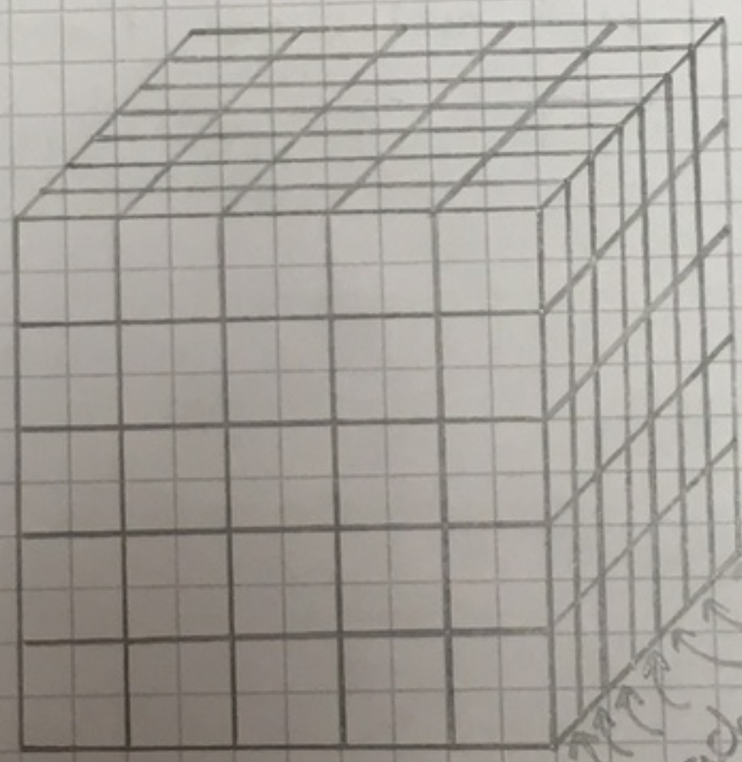
← dot products of all corresponding cubes

$filter_5 = filter_2$ but we use 7 separate ones ‖ $filter_5$'s size: [3,3,5,7]



$inputimage_5 = inputimage_4$ (with padding) ‖ size: [1,5,5,5]
↑ without padding

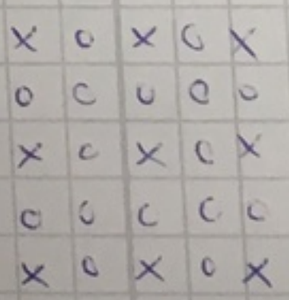
$resultimage_5 =$



‖ size: [1,5,5,7]

(7 times) each one is created with a different filter

(6) vary strides, e.g. strides [1,2,2,1]



(7) more than one image: e.g. $inputimage_7$'s size: [10,5,5,5] ⟶ stack of 10 images
$resultimage_7$'s size: [10,5,5,7]

Mathematical Notation

We use 3D matrices (width, height, depth) of the form

image × filter = result

case ⓪ : (2,2,1) × (1,1,1) = (2,2,1)

case ① : (3,3,5) × (1,1,5) = (3,3,1)

case ② : (3,3,5) × (3,3,5) = (1,1,1)

case ③ : (5,5,5) × (3,3,5) = (3,3,1)

case ④ : without padding
↳ (5,5,5) × (3,3,5) = (5,5,1)

with padding
↳ (7,7,5) × (3,3,5) = (5,5,1)

case ⑤ : with padding and 7 repetitions with 7 filters
↳ (7,7,5) × 7×(3,3,5) = (5,5,7)

① Flatten Filter

filter

$$\begin{bmatrix} \text{filter\_height} \\ \text{filter\_width} \\ \text{in\_channels} \\ \text{out\_channels} \end{bmatrix} \xrightarrow{\text{flatten}} \begin{bmatrix} \text{filter\_height} * \text{filter\_width} * \text{in\_channels} \\ \text{out\_channels} \end{bmatrix}$$

② Form "virtual" tensor

input

$$\begin{bmatrix} \text{batch} \\ \text{in\_height} \\ \text{in\_width} \\ \text{in\_channels} \end{bmatrix} \xrightarrow[\text{patches}]{\text{extract image}} \begin{bmatrix} \text{batch} \\ \text{out\_height} \\ \text{out\_width} \\ \text{filter\_height} * \text{filter\_width} * \text{in\_channels} \end{bmatrix}$$

Filter

4D column vector

$$\begin{pmatrix} fh \\ fw \\ ic \\ oc \end{pmatrix} \xrightarrow[\text{filter}]{\text{flatten}} \begin{pmatrix} \overbrace{fh \cdot fw \cdot ic}^{z} \\ oc \end{pmatrix}$$

F_0 → F_1

2D column vector

Input

4D column vector

$$\begin{pmatrix} b \\ ih \\ iw \\ ic \end{pmatrix} \xrightarrow[\text{image patch}]{\text{extract}} [\, b \mid oh \mid ow \mid fh \cdot fw \cdot ic \,]$$

I_0 → I_1

4D row vector

(3)

Output

$$R_0 = F_1 \cdot I_1$$

$$= \begin{pmatrix} z \\ oc \end{pmatrix}_{2D} * \underbrace{\begin{pmatrix} b & oh & ow & z \end{pmatrix}}_{4D}$$

$$= \underbrace{\begin{pmatrix} z \cdot b + z \cdot oh + z \cdot ow + z \cdot z \\ oc \cdot b + oc \cdot oh + oc \cdot ow + oc \cdot z \end{pmatrix}}_{2D}$$