Lecture 3: Convolutional Neural Networks

Today:

Today:

- Expectations of visual recognition systems (10%)

Today:

- Expectations of visual recognition systems (10%)
- A Conv layer, stride, padding etc. (25%)

Today:

- Expectations of visual recognition systems (10%)
- A Conv layer, stride, padding etc. (25%)
- Conv2D (20%)

Today:

- Expectations of visual recognition systems (10%)
- A Conv layer, stride, padding etc. (25%)
- Conv2D (20%)
- Transposed Conv (5%)

Today:

• Expectations of visual recognition systems	(10%)
• A Conv layer, stride, padding etc.	(25%)
• Conv2D	(20%)
• Transposed Conv	(5%)
• Implementation and backprop	(20%)

Today:

- Expectations of visual recognition systems (10%)
- A Conv layer, stride, padding etc. (25%)
- Conv2D (20%)
- Transposed Conv (5%)
- Implementation and backprop (20%)
- What is encoded in feature maps? (10%)

Today:

- Expectations of visual recognition systems (10%)
- A Conv layer, stride, padding etc. (25%)
- Conv2D (20%)
- Transposed Conv (5%)
- Implementation and backprop (20%)
- What is encoded in feature maps? (10%)
- Max-pooling, convnet and conv variants (10%)


vertical
line


vertical
line


vertical
line
horizontal
line


key


key
key
key
key
key
key
key
key
key
key
key
key


keyboard
key
key
key
key
key
key
key
key
key
key
key
key

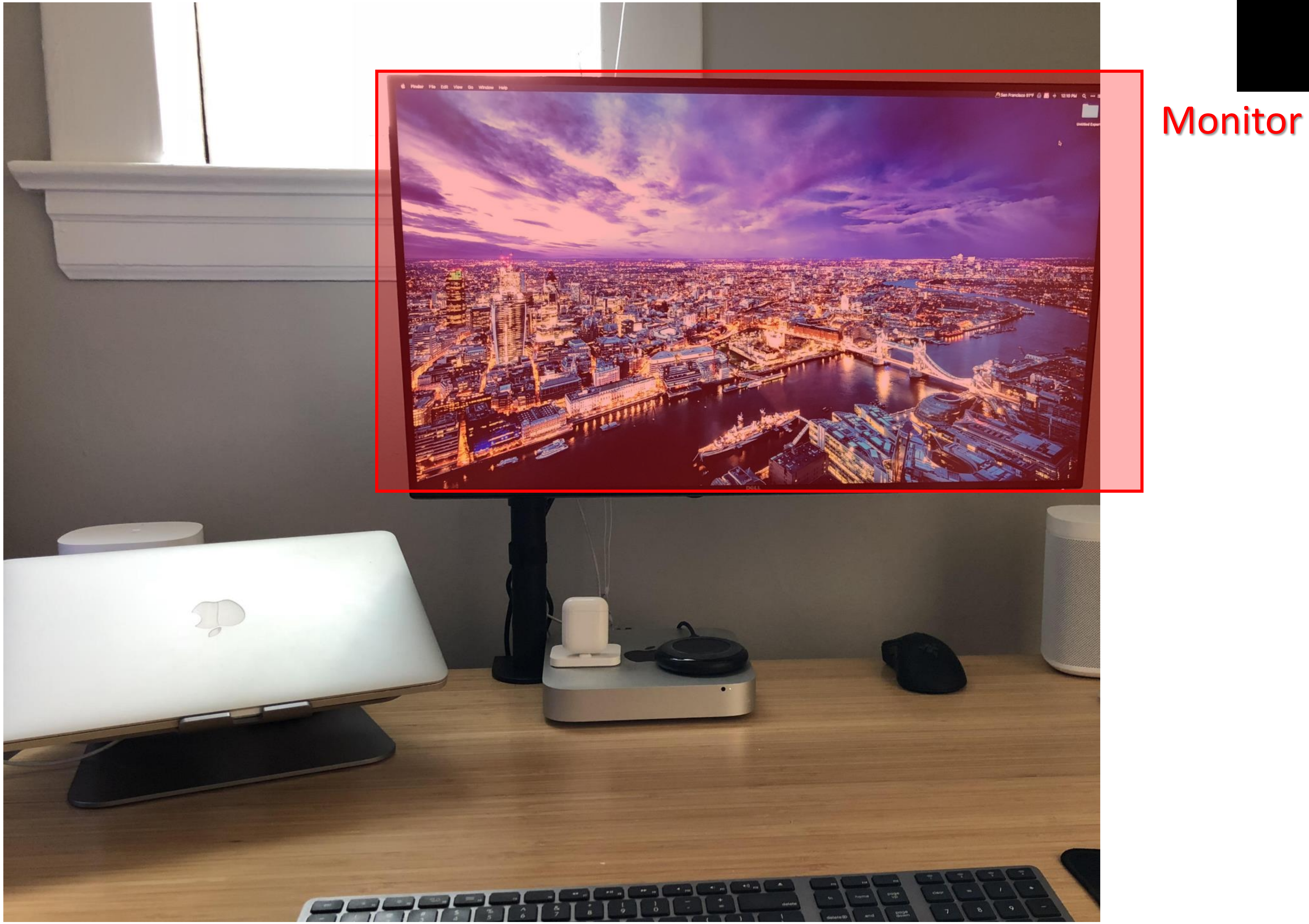

Monitor
keyboard
key
key
key
key
key
key
key
key
key
key
key
key


Monitor
key
key
key
key
key
key
key
key
key
key
key
key
keyboard
Mouse


Monitor
Desk
keyboard
key
key
key
key
key
key
key
key
key
key
key
key
Mouse


Monitor


Monitor


Monitor

Expectations of visual recognition network

Expectations of visual recognition network

1. Maintain 2D structure logic

Expectations of visual recognition network

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)

Expectations of visual recognition network

1. Maintain 2D structure logic

2. Shift invariant (actually, equivariant)

3. Consider only local correlations

Expectations of visual recognition network

1. Maintain 2D structure logic

2. Shift invariant (actually, equivariant)

3. Consider only local correlations

4. Hierarchically growing field of view

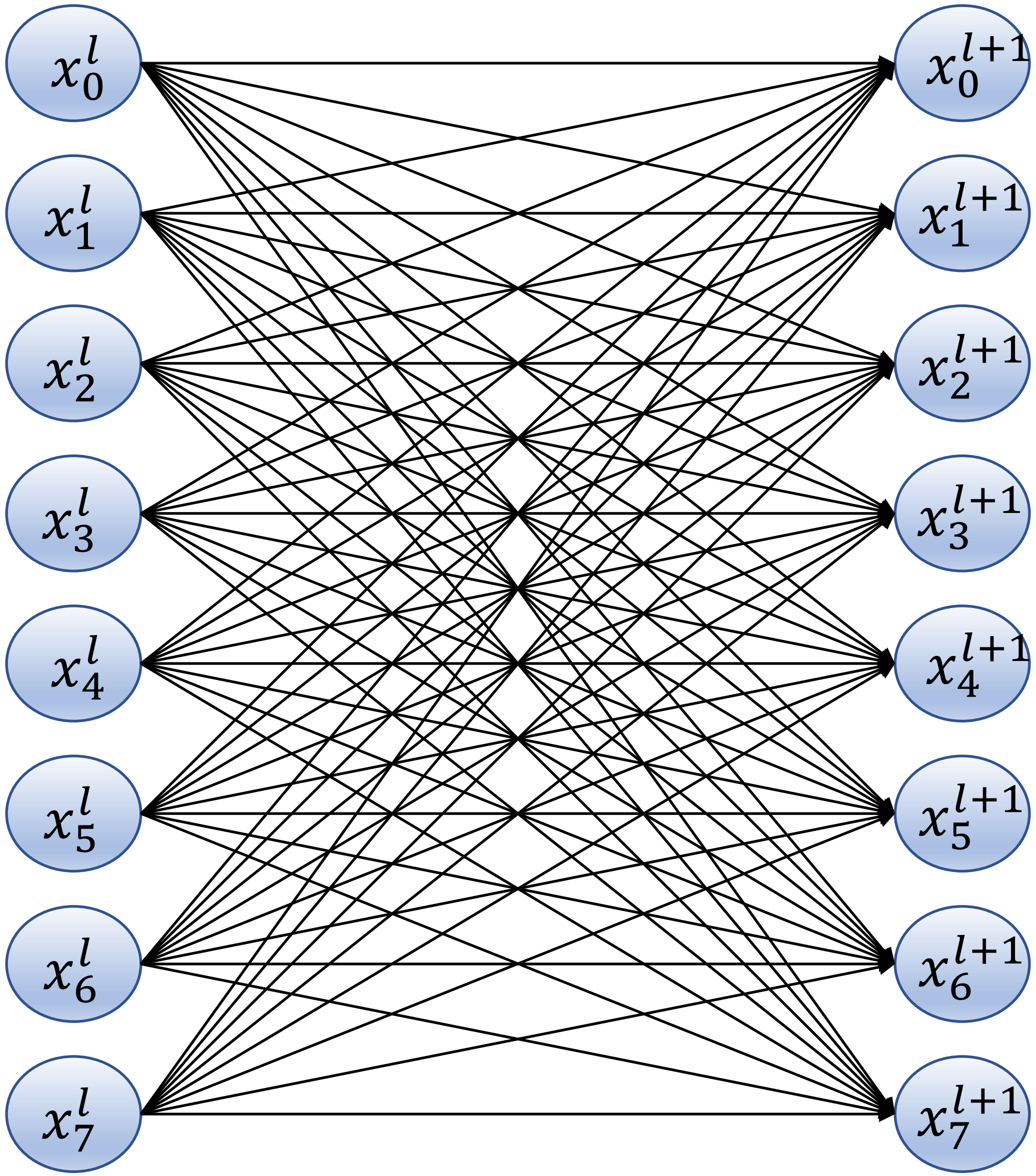
Expectations of visual recognition network

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity

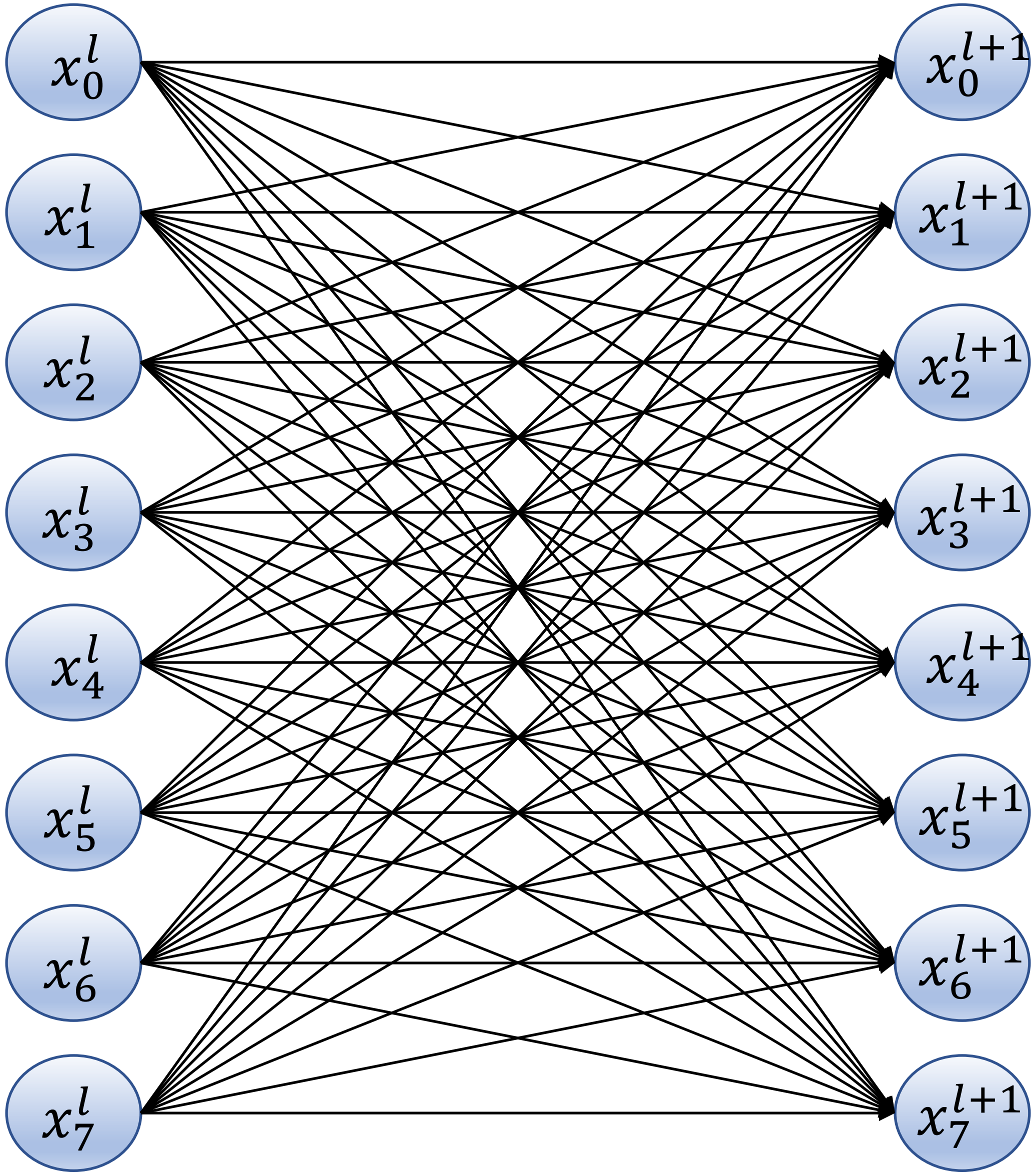
Expectations of visual recognition network

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

Fully connected layer

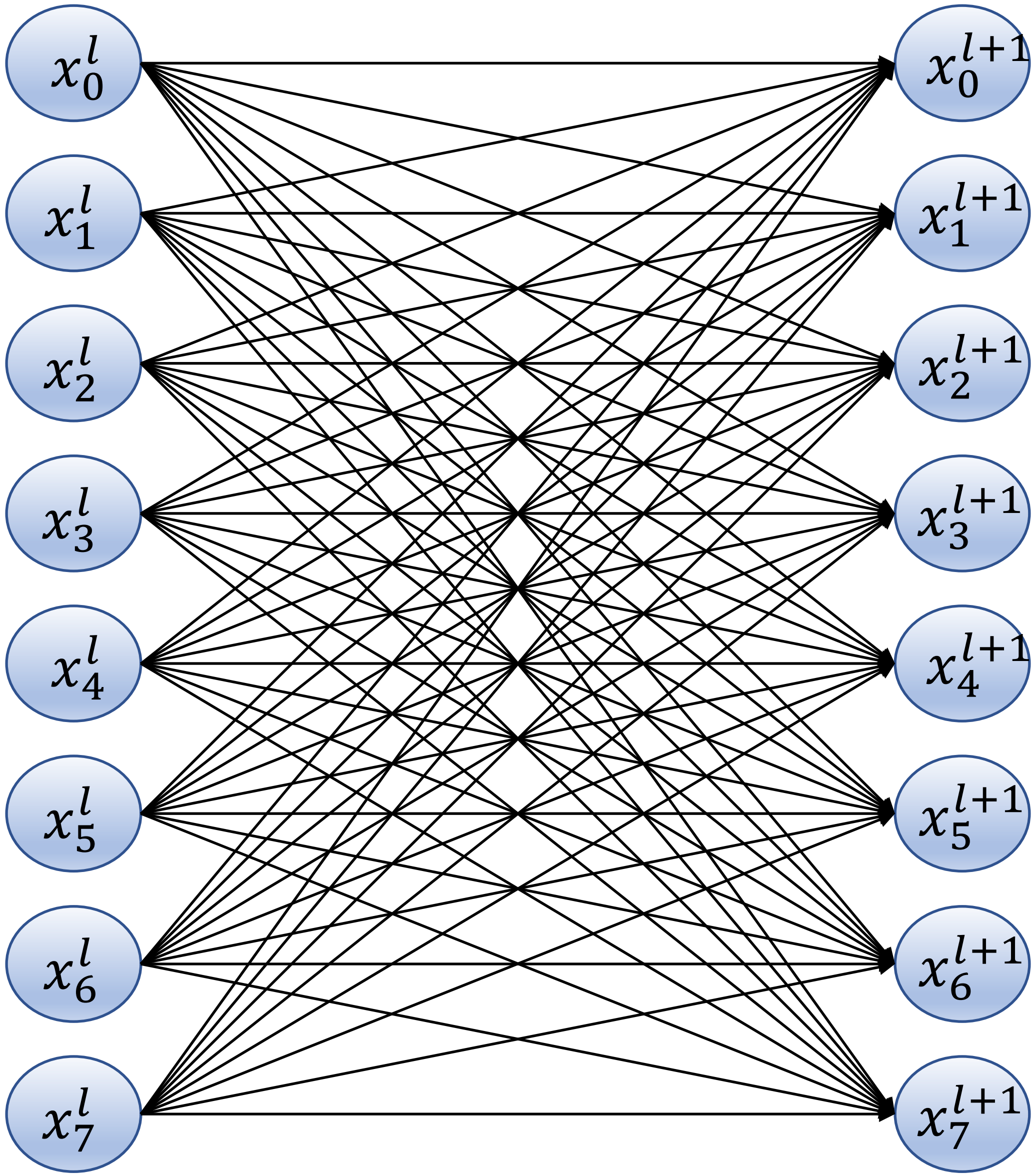


Fully connected layer



$$\begin{bmatrix} w_{00}^l & w_{01}^l & & \\ w_{10}^l & w_{11}^l & & \cdots \\ & & & \\ & \vdots & & \ddots \\ & & & \end{bmatrix}$$

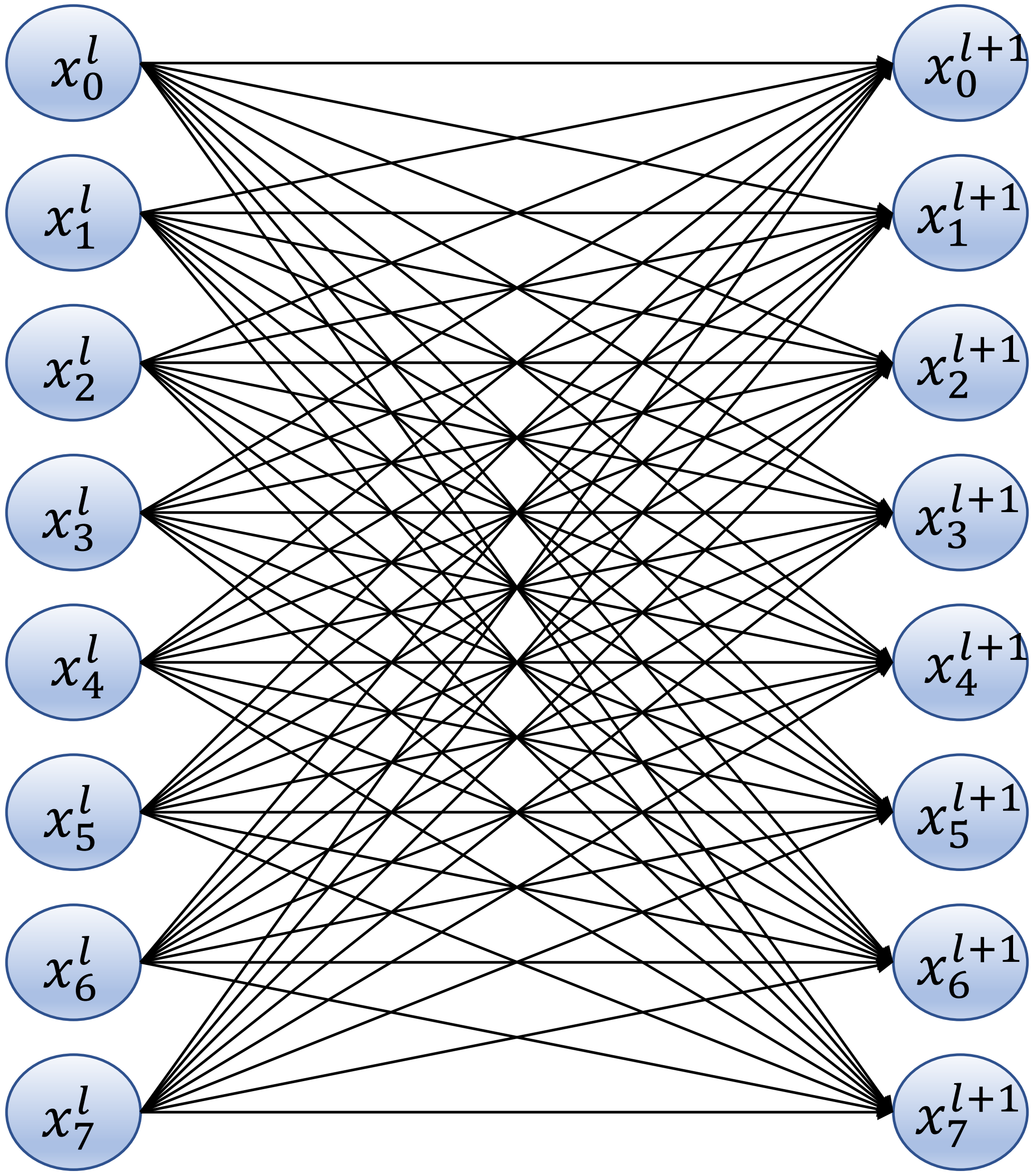
Fully connected layer



$$\begin{bmatrix} w_{00}^l & w_{01}^l & \cdots \\ w_{10}^l & w_{11}^l & \\ & \vdots & \ddots \end{bmatrix}$$

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

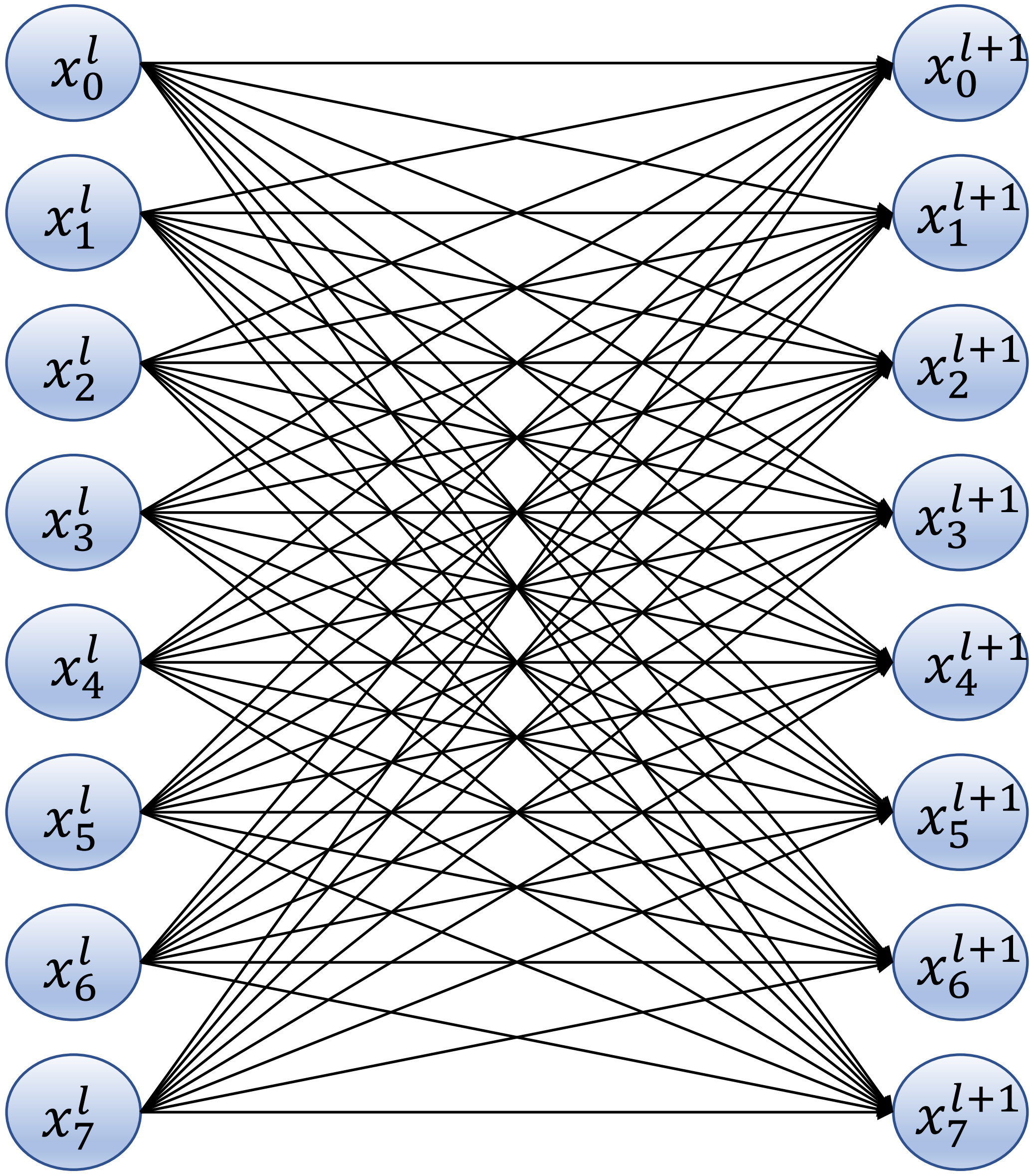
Fully connected layer



$$\begin{bmatrix} w_{00}^l & w_{01}^l & \\ w_{10}^l & w_{11}^l & \cdots \\ & \vdots & \ddots \end{bmatrix}$$

1. ✗ Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

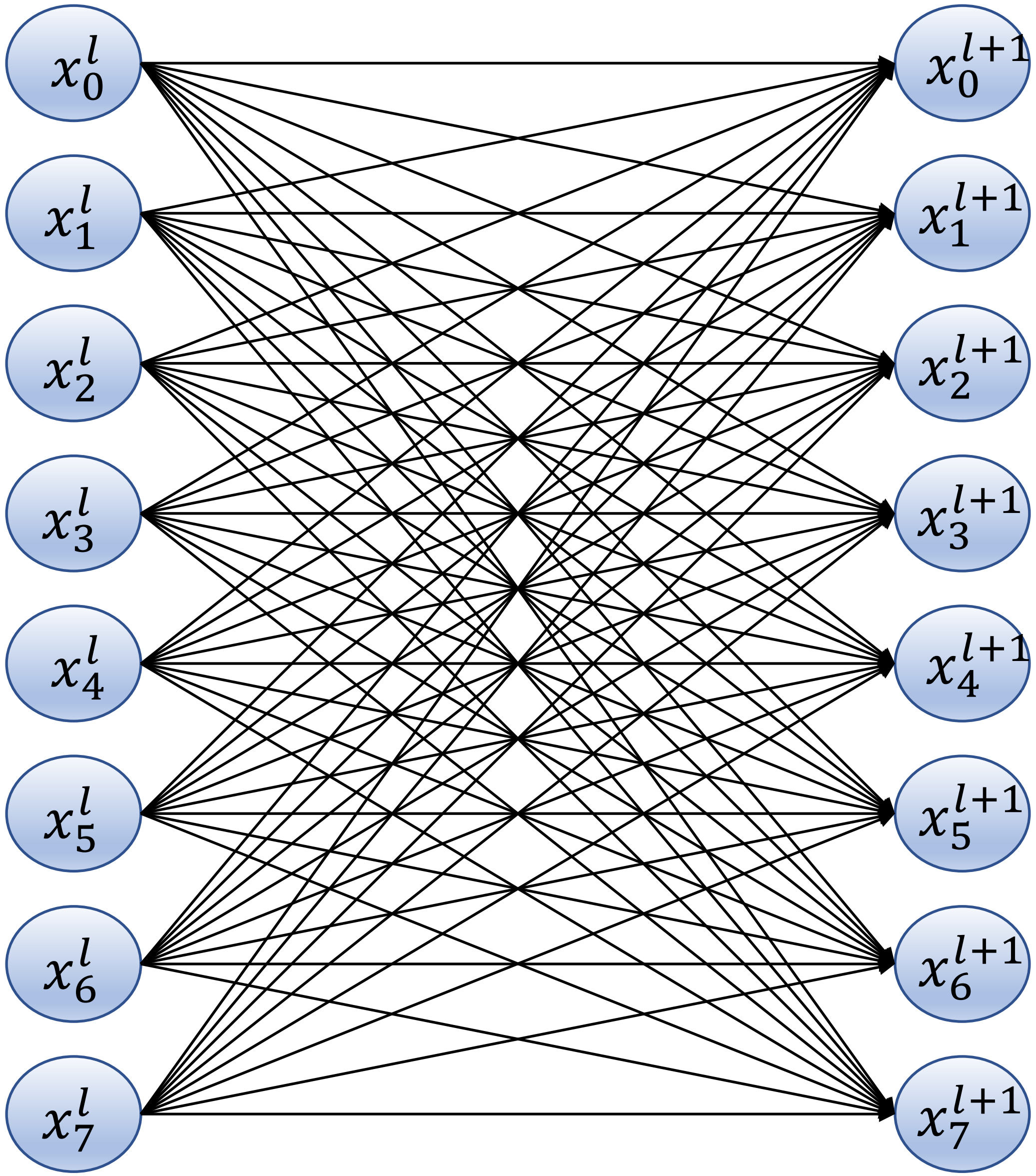
Fully connected layer



$$\begin{bmatrix} w_{00}^l & w_{01}^l & & \\ w_{10}^l & w_{11}^l & & \cdots \\ & & & \\ & \vdots & & \ddots \\ & & & \end{bmatrix}$$

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

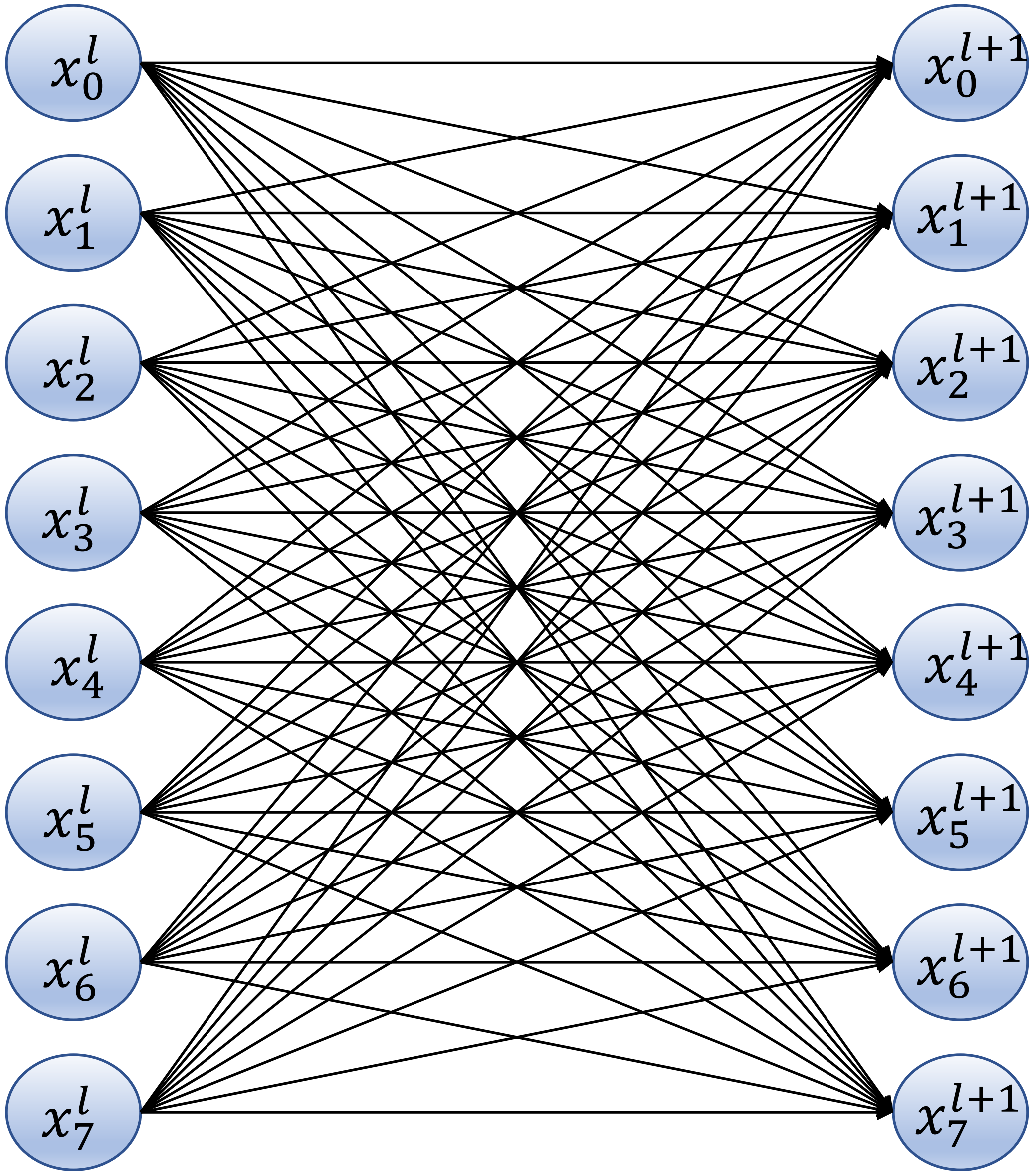
Fully connected layer



$$\begin{bmatrix} w_{00}^l & w_{01}^l & \\ w_{10}^l & w_{11}^l & \cdots \\ & \vdots & \ddots \end{bmatrix}$$

1. ✗ Maintain 2D structure logic
2. ✗ Shift invariant (actually, equivariant)
3. ✗ Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

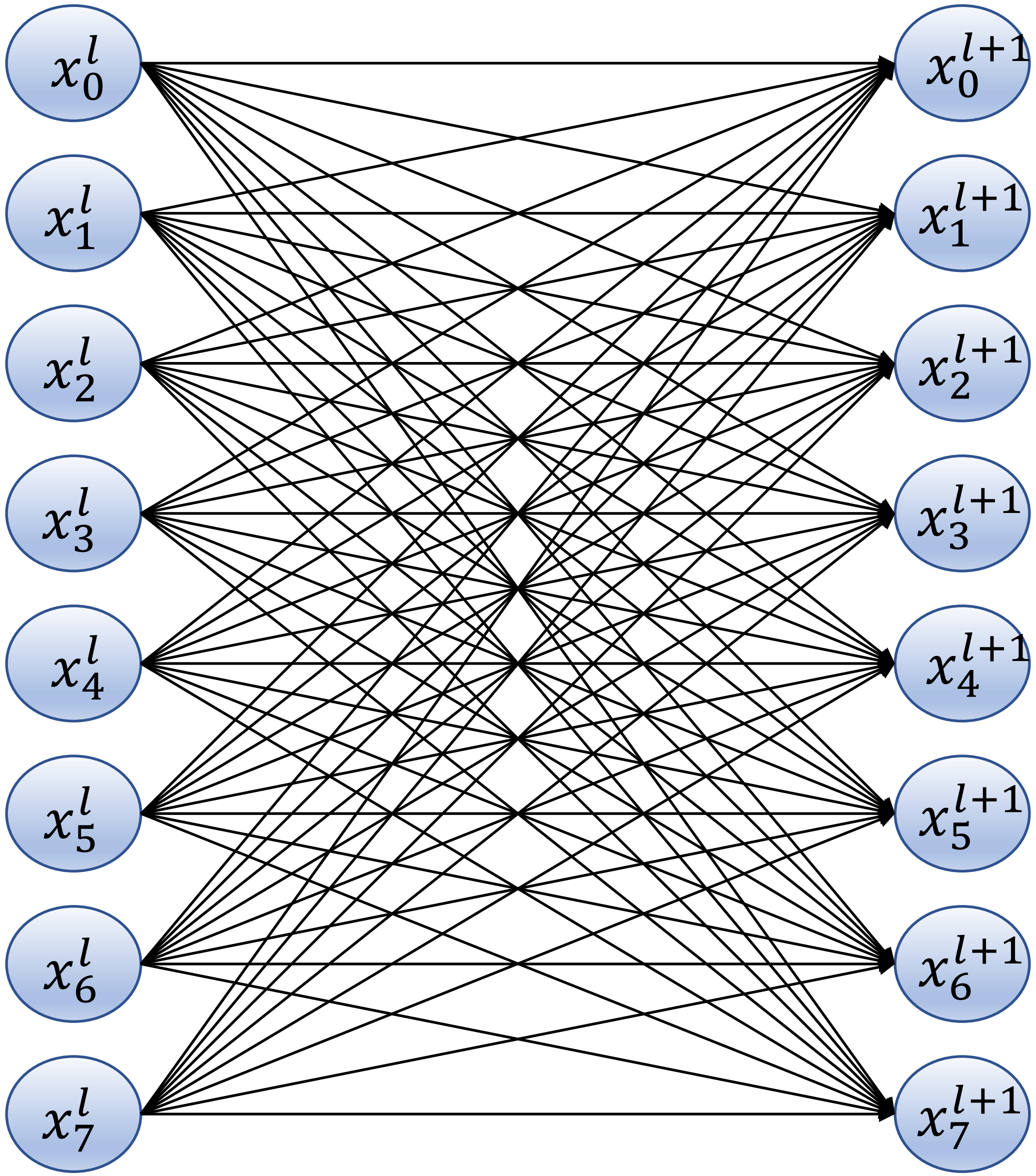
Fully connected layer



$$\begin{bmatrix} w_{00}^l & w_{01}^l & & \\ w_{10}^l & w_{11}^l & & \cdots \\ & \vdots & & \ddots \end{bmatrix}$$

1. ❌ Maintain 2D structure logic
2. ❌ Shift invariant (actually, equivariant)
3. ❌ Consider only local correlations
4. ❌ Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

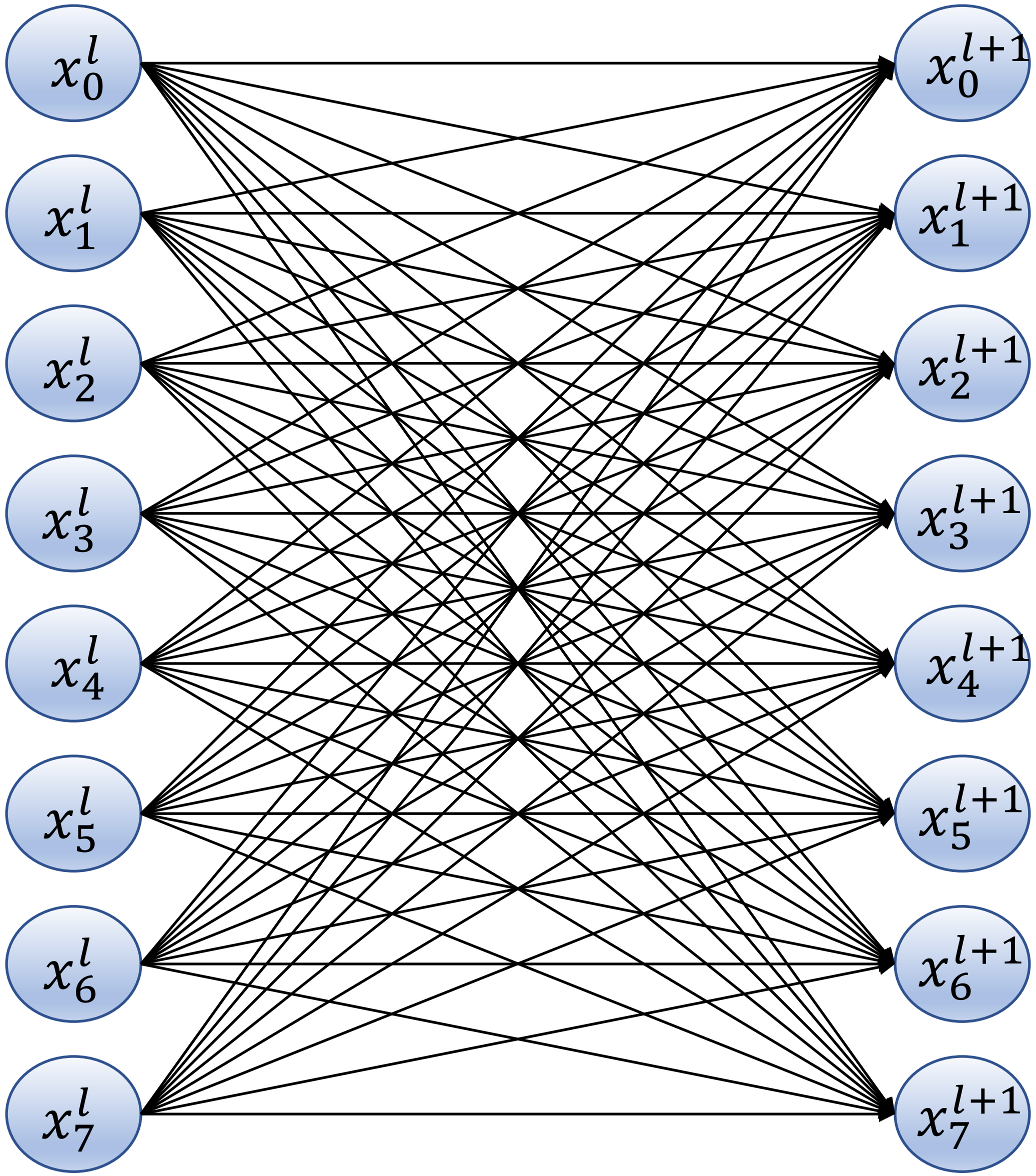
Fully connected layer



$$\begin{bmatrix} w_{00}^l & w_{01}^l & & \\ w_{10}^l & w_{11}^l & & \cdots \\ & & & \\ & \vdots & & \ddots \\ & & & \end{bmatrix}$$

❌ 1. Maintain 2D structure logic
❌ 2. Shift invariant (actually, equivariant)
❌ 3. Consider only local correlations
❌ 4. Hierarchically growing field of view
✓ 5. Hierarchically progressing complexity
6. Reasonable amount of params

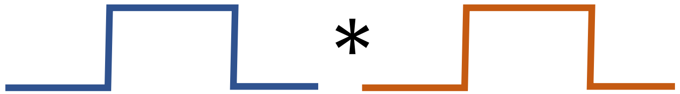
Fully connected layer

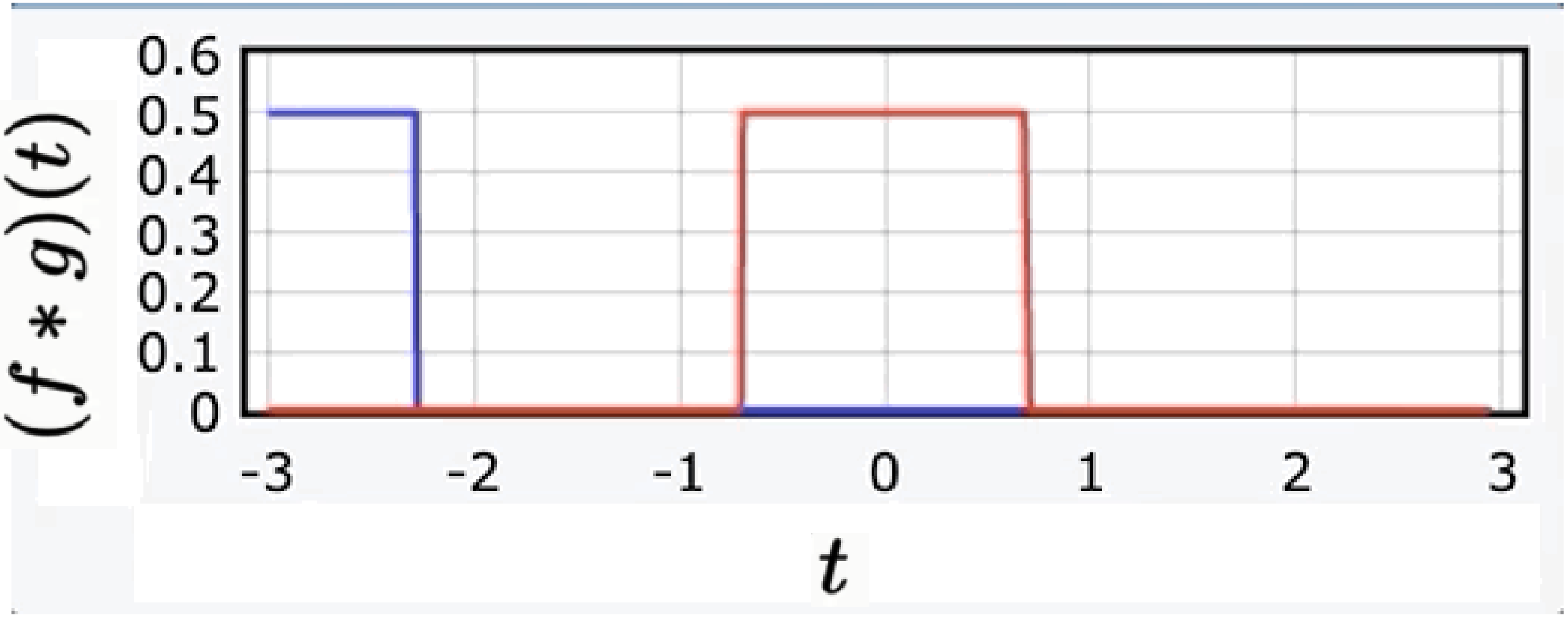


$$\begin{bmatrix} w_{00}^l & w_{01}^l & & \\ w_{10}^l & w_{11}^l & & \cdots \\ & & & \\ & \vdots & & \ddots \\ & & & \end{bmatrix}$$

✗ 1. Maintain 2D structure logic
✗ 2. Shift invariant (actually, equivariant)
✗ 3. Consider only local correlations
✗ 4. Hierarchically growing field of view
✓ 5. Hierarchically progressing complexity
✗ 6. Reasonable amount of params

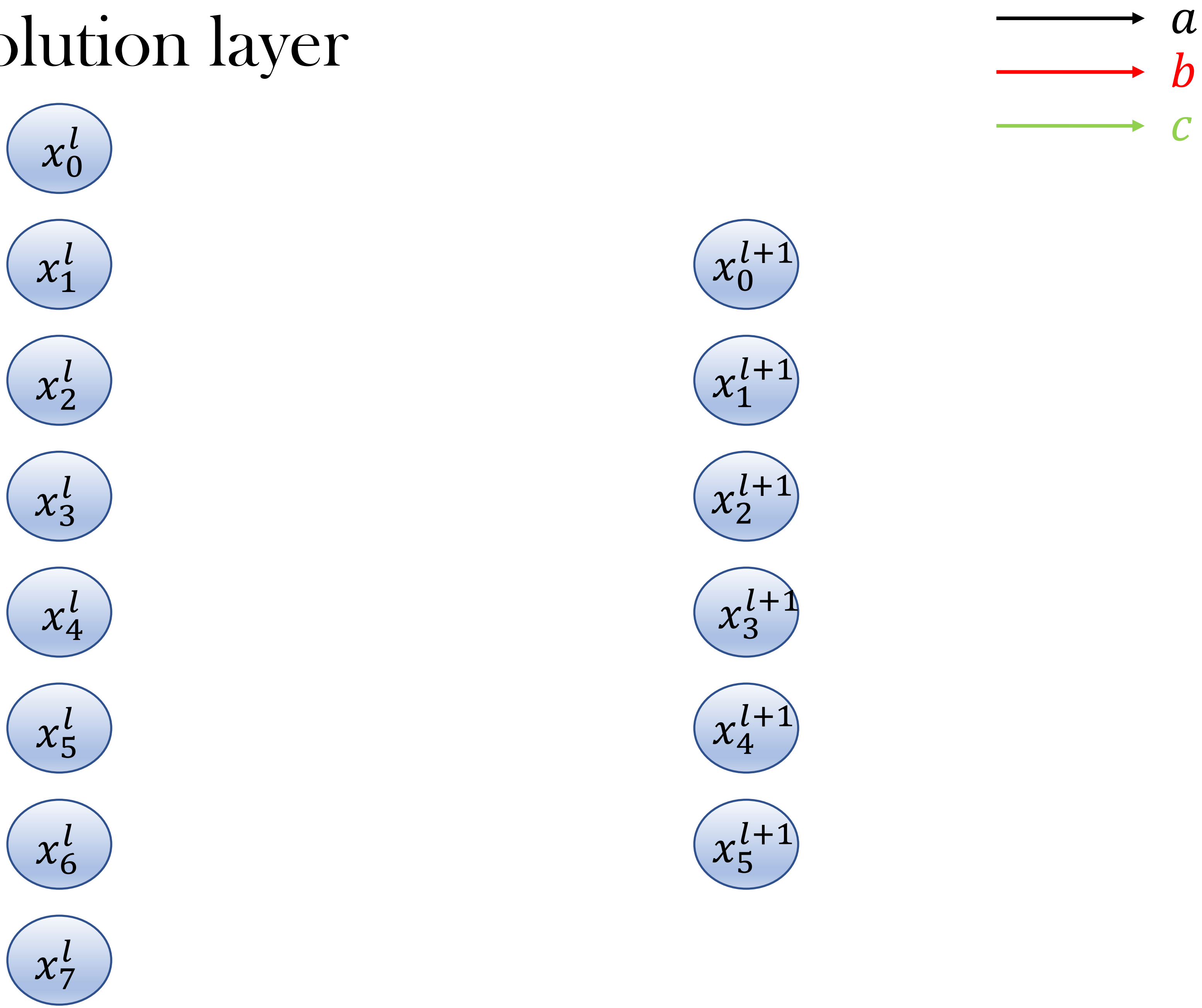
Convolution



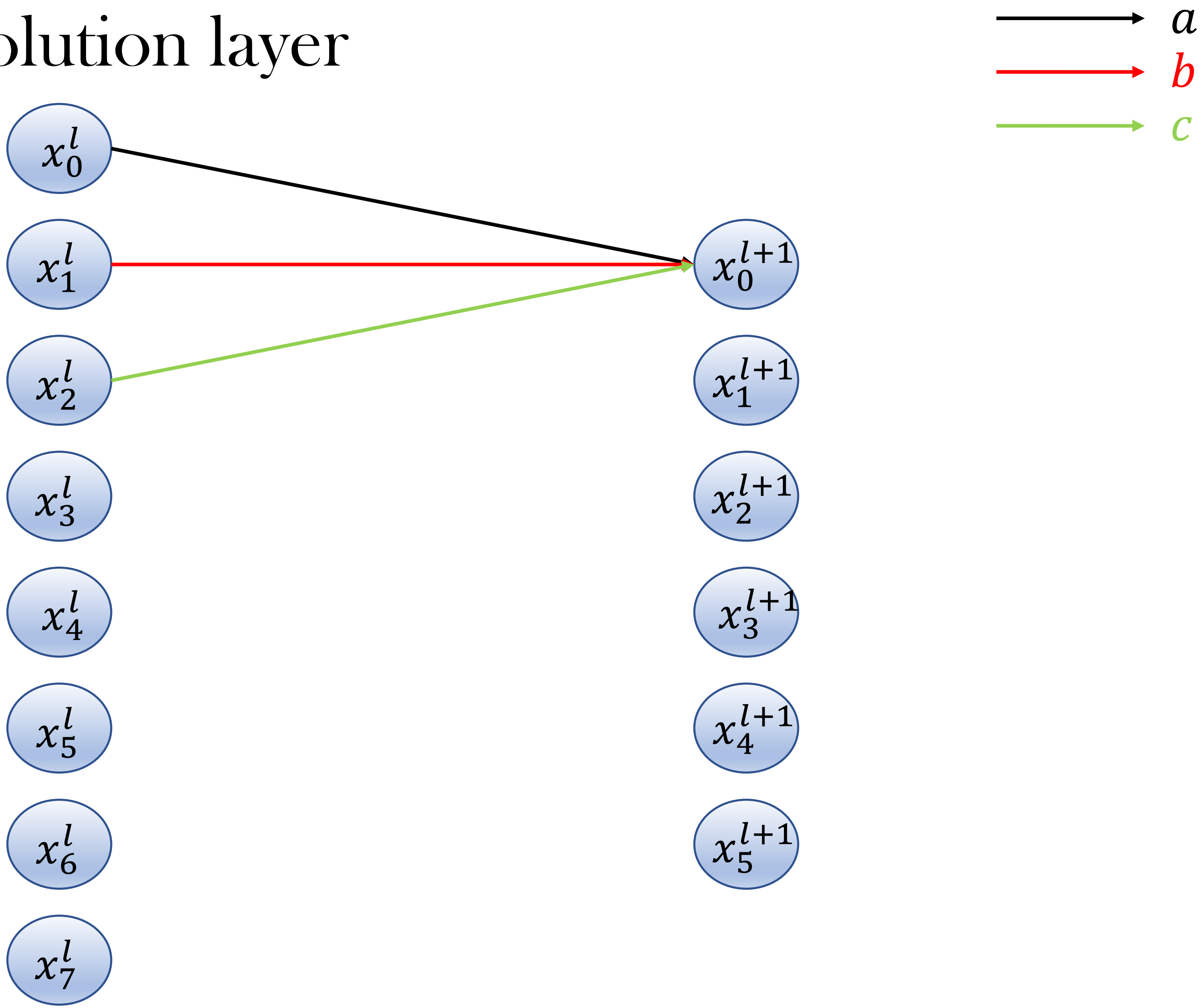


$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

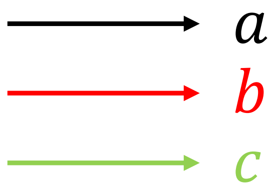
Convolution layer

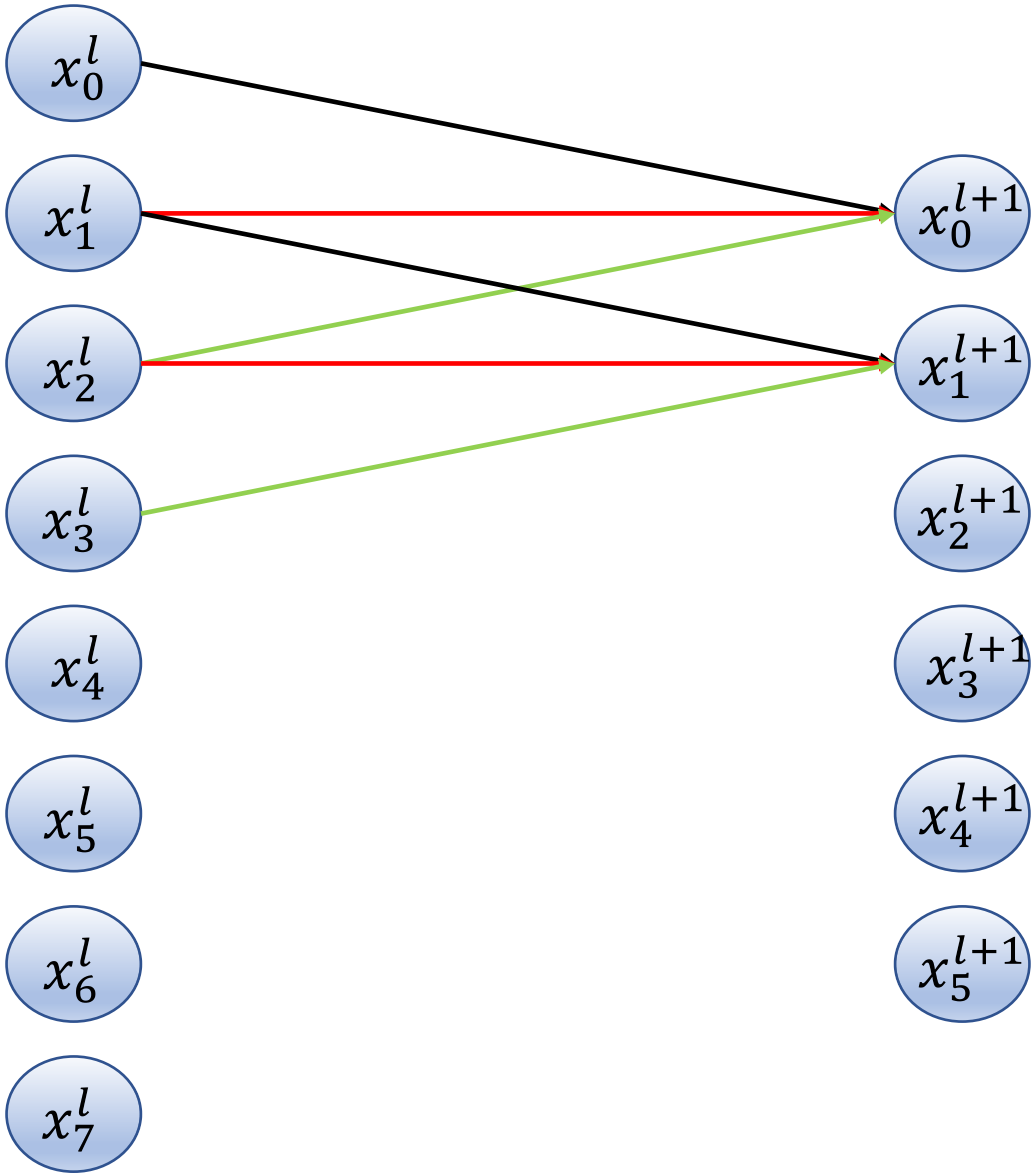


Convolution layer

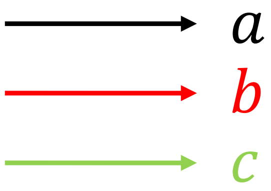


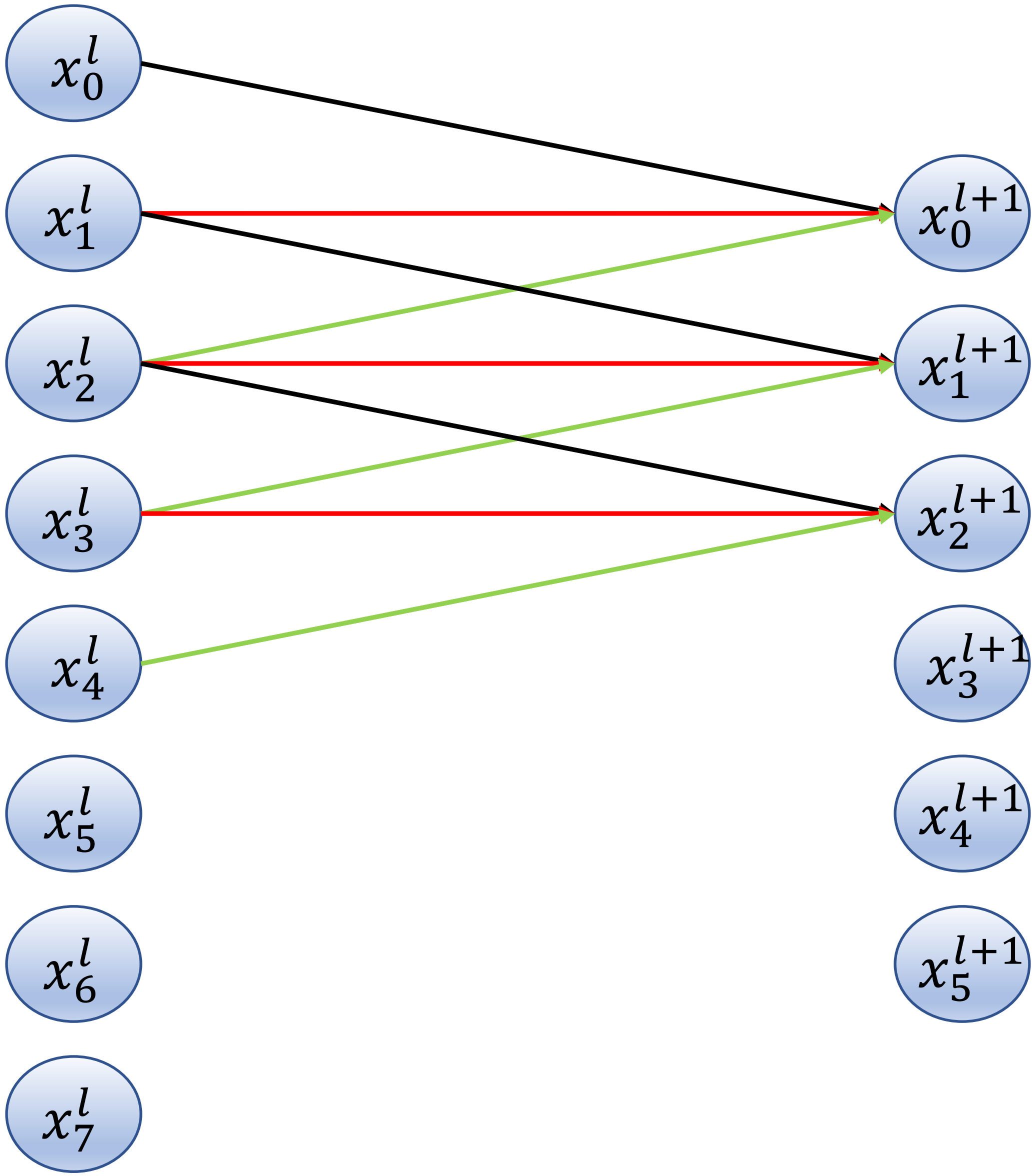
Convolution layer





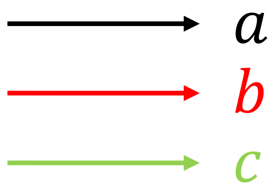
Convolution layer

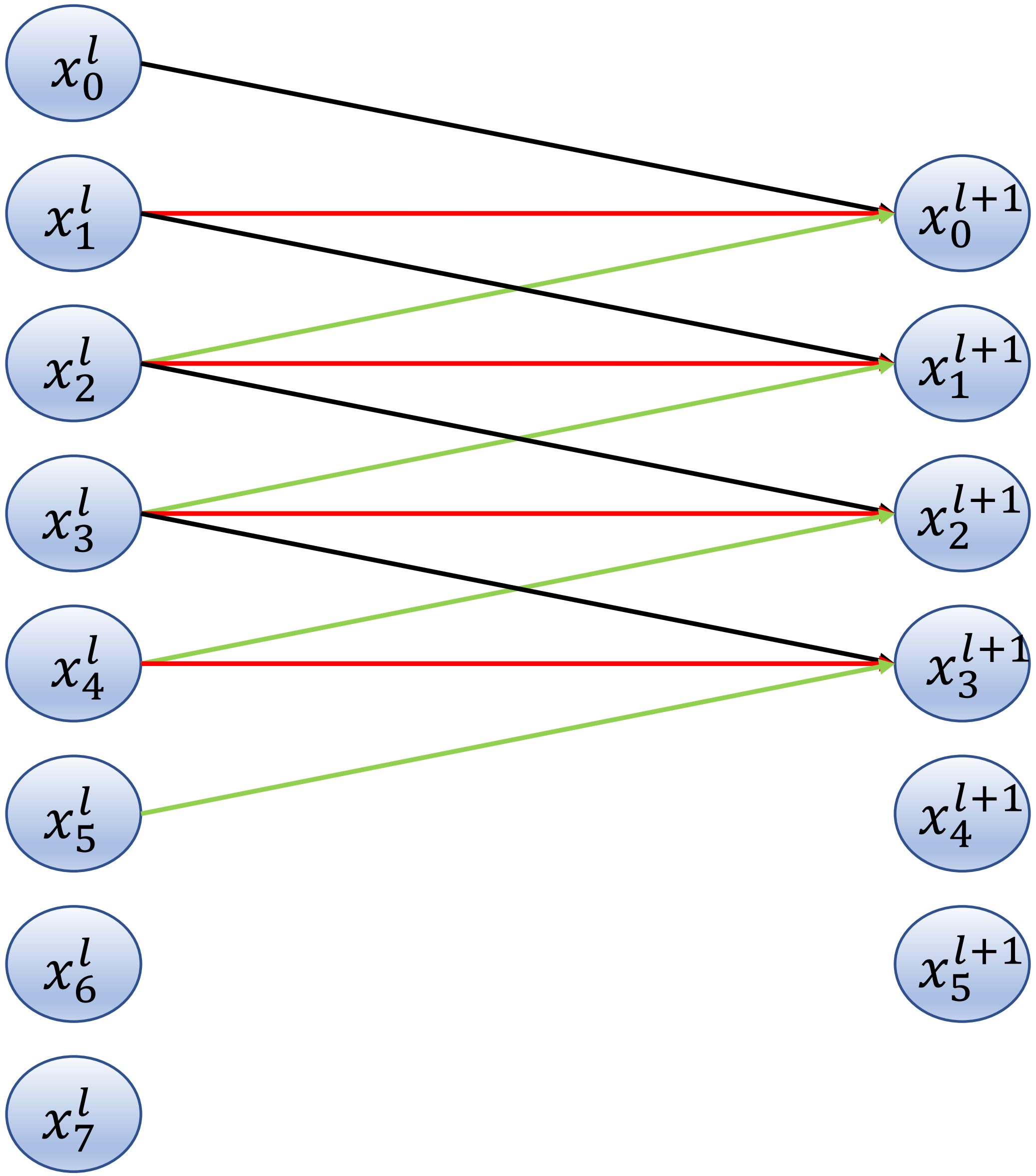




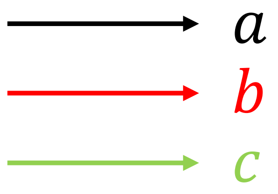
Convolution layer

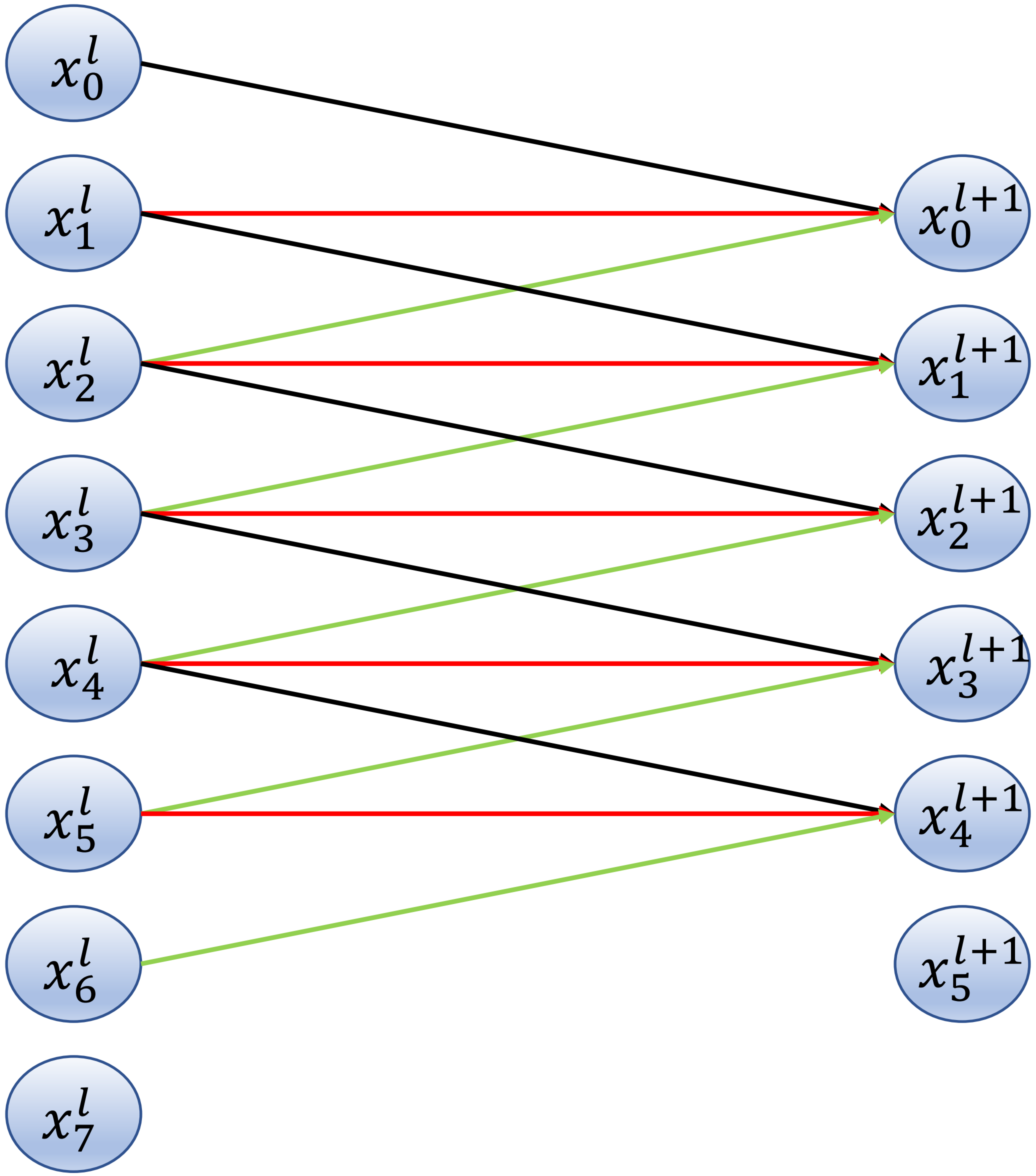
[redacted]



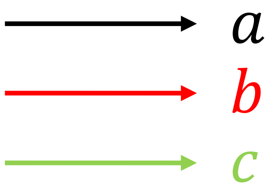


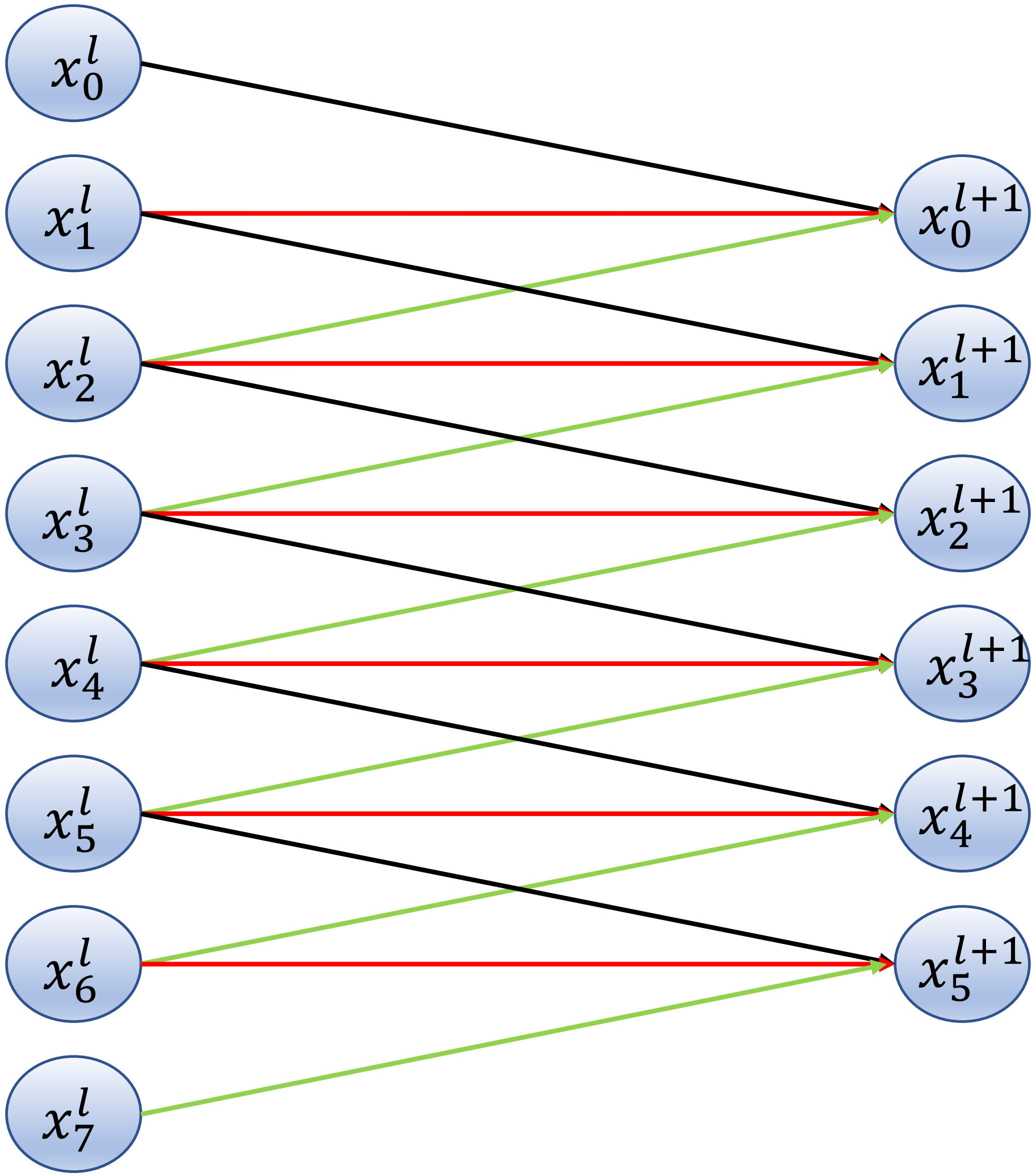
Convolution layer



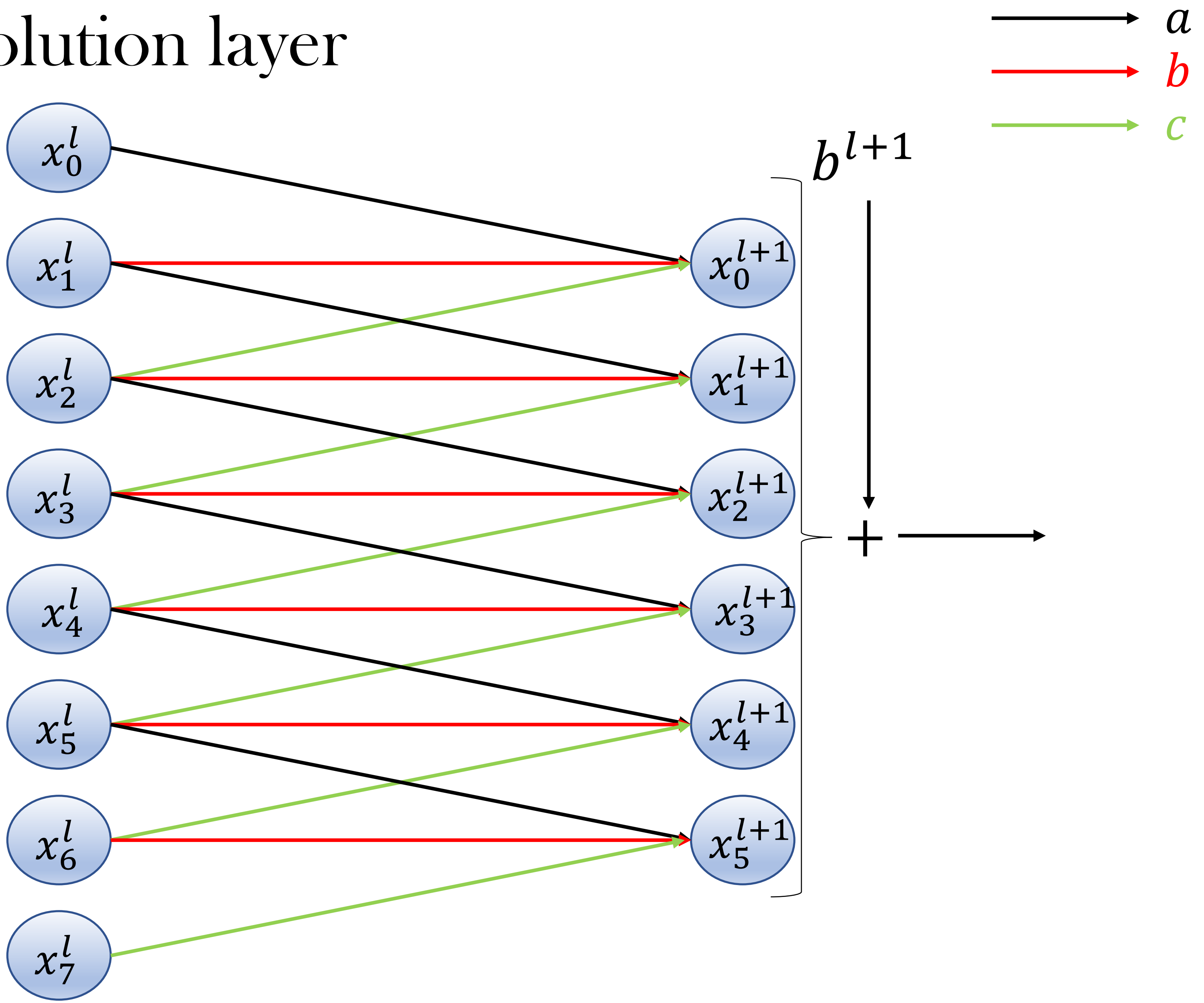


Convolution layer

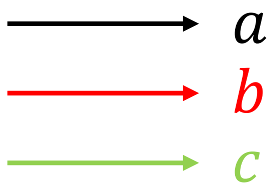


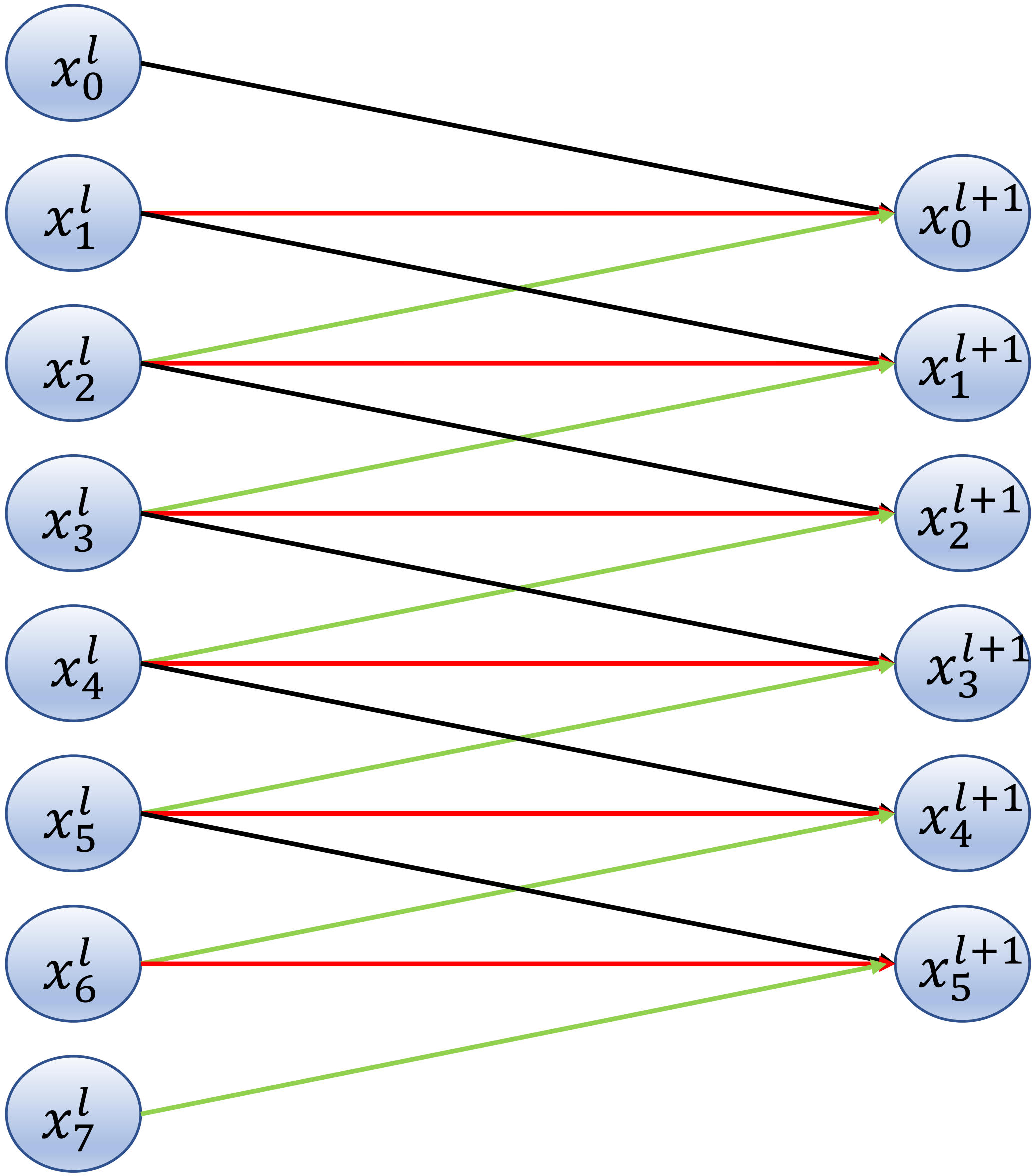


Convolution layer

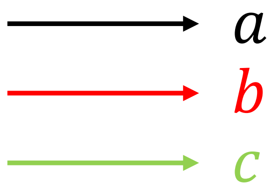


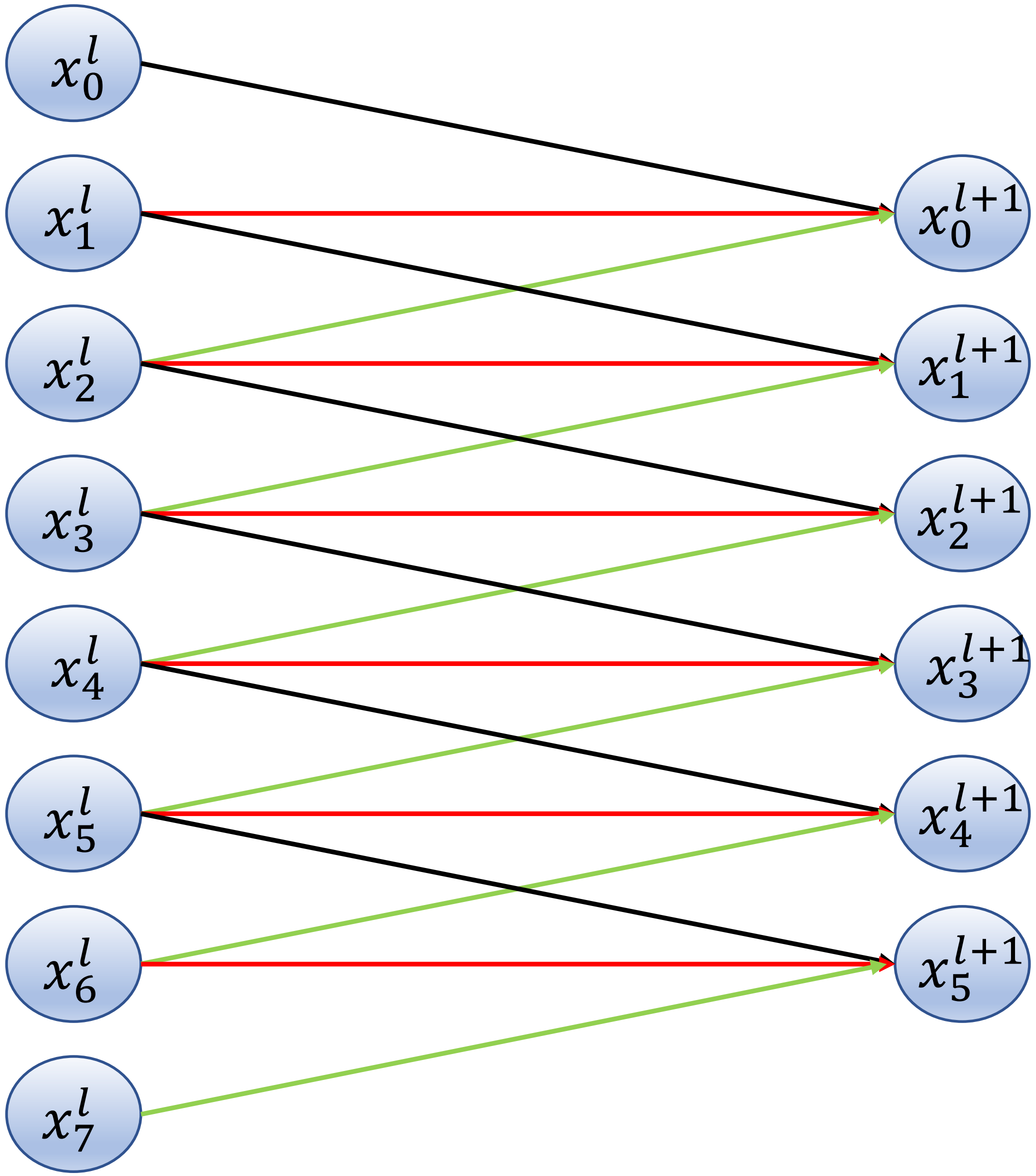
Convolution layer



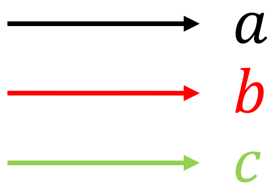


Convolution layer

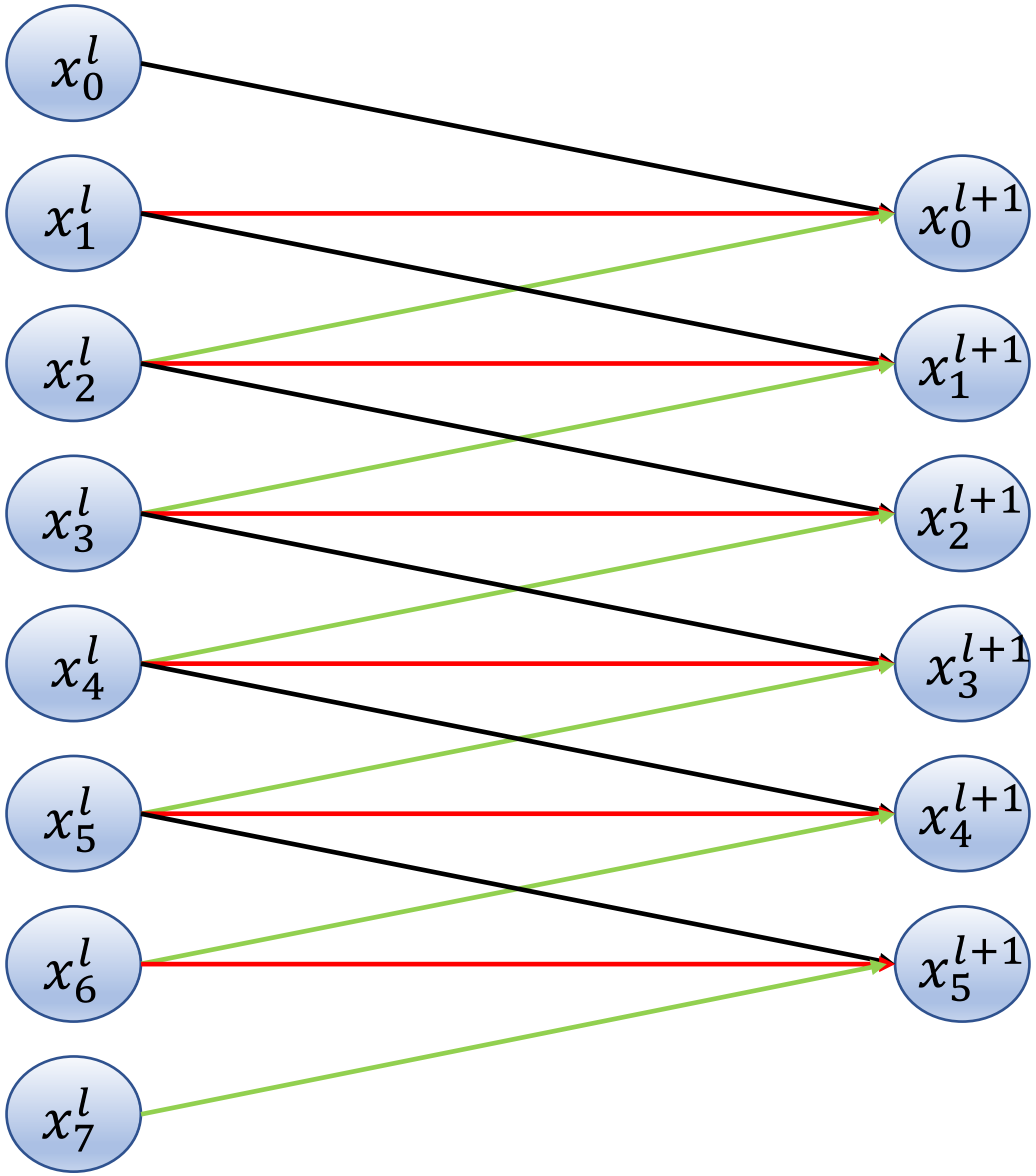




Convolution layer

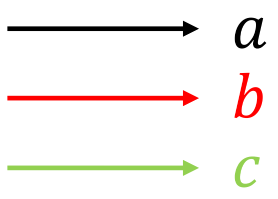


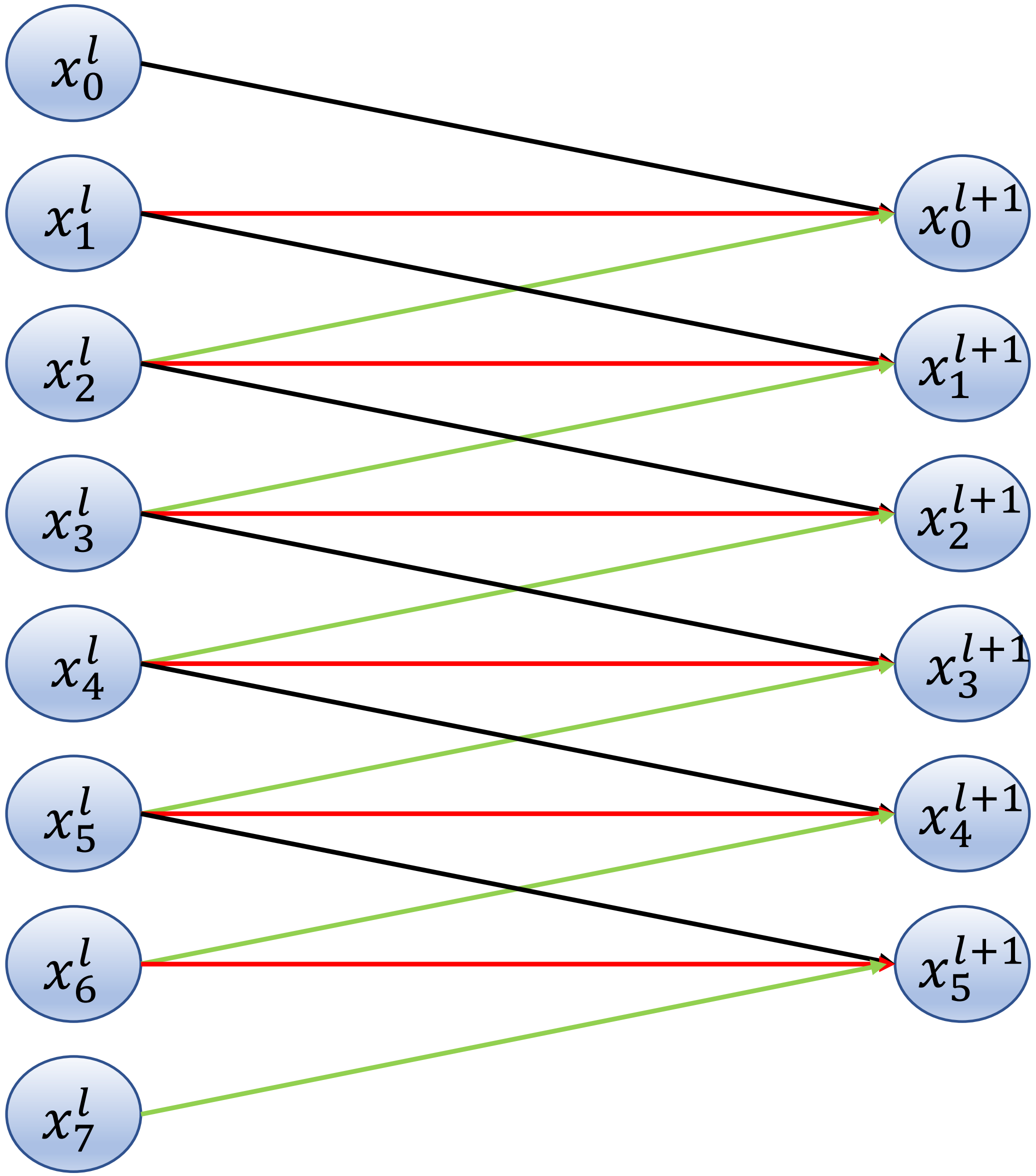
[redacted]



$$\begin{bmatrix} \boldsymbol{a} & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

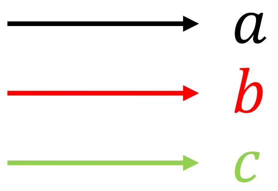
Convolution layer

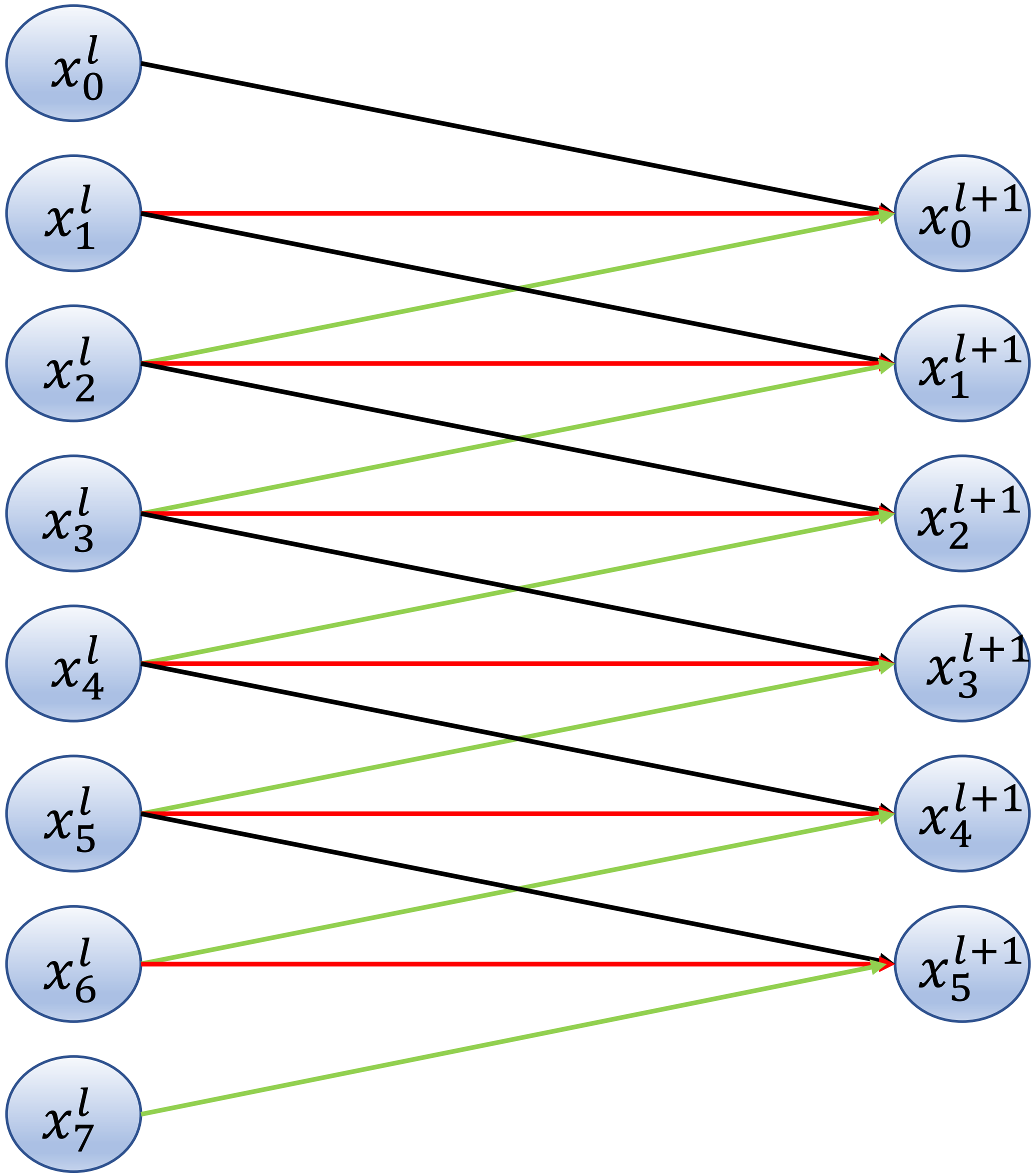




$$\begin{bmatrix} \boldsymbol{a} & \boldsymbol{b} & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \end{bmatrix}$$

Convolution layer





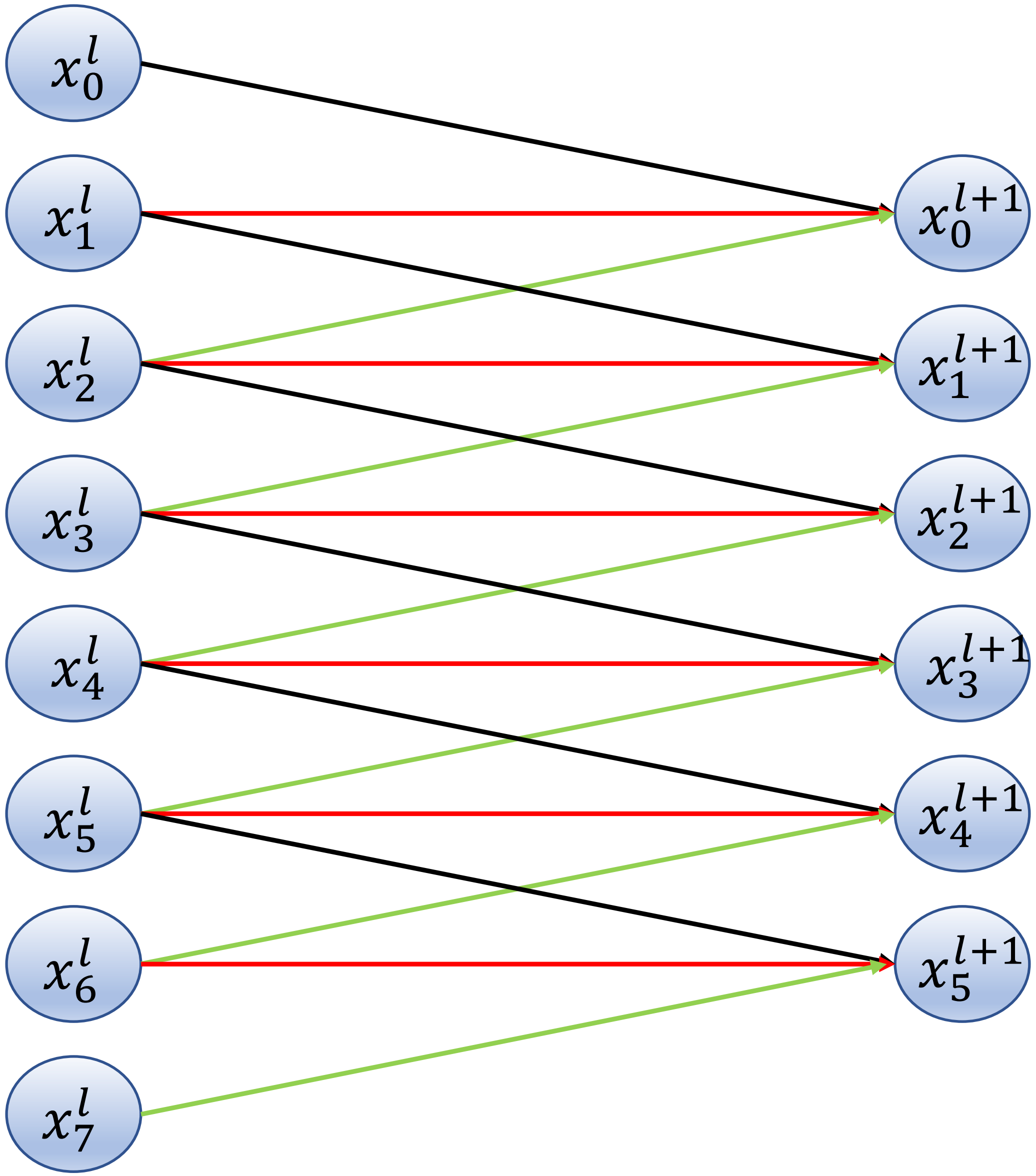
$$\begin{bmatrix} \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

Convolution layer

$\longrightarrow a$

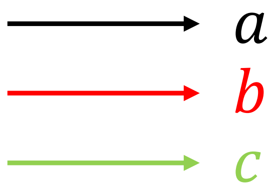
$\longrightarrow b$

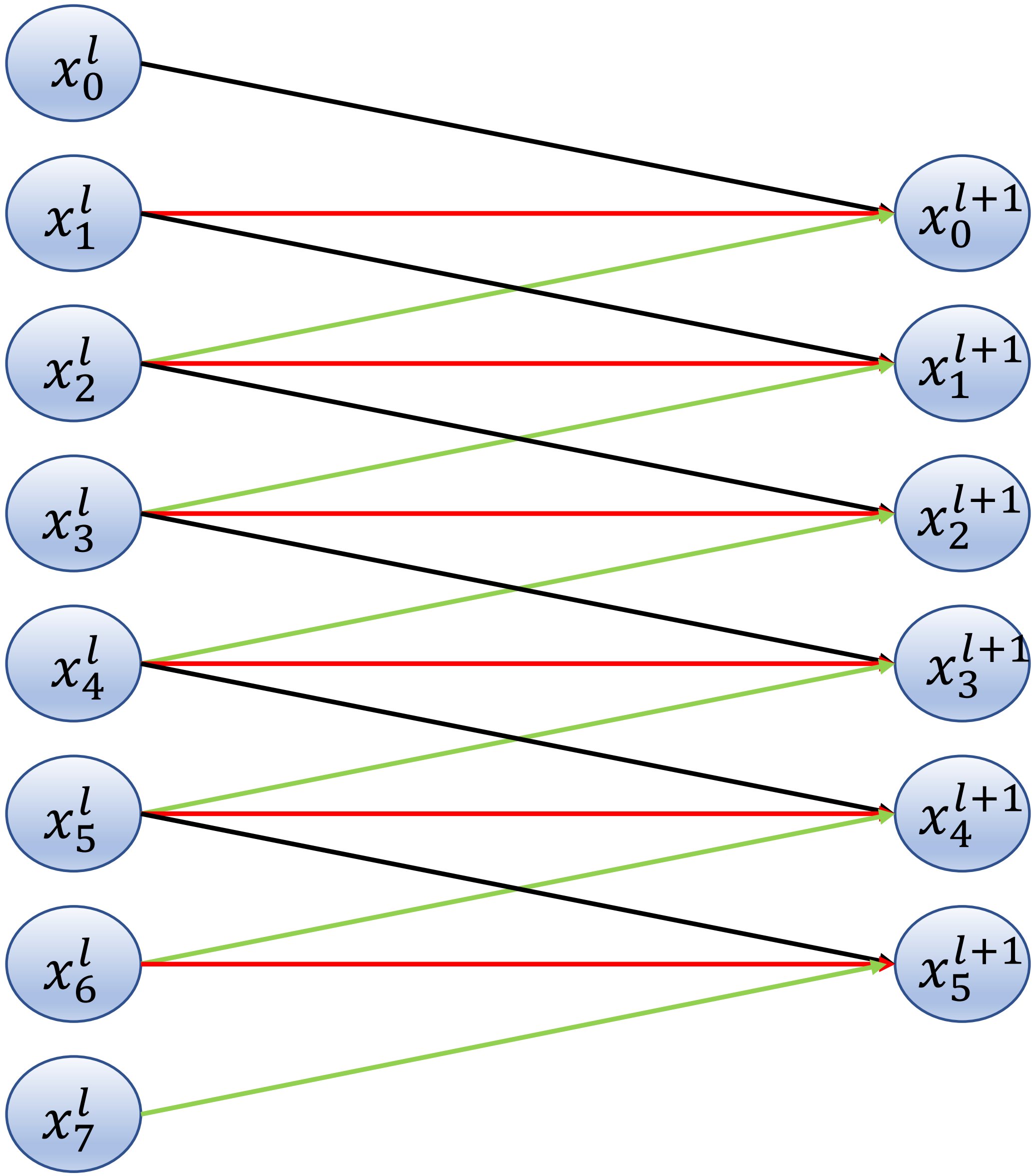
$\longrightarrow c$



$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 & 0 \\ 0 & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

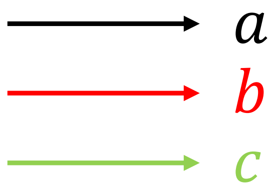
Convolution layer

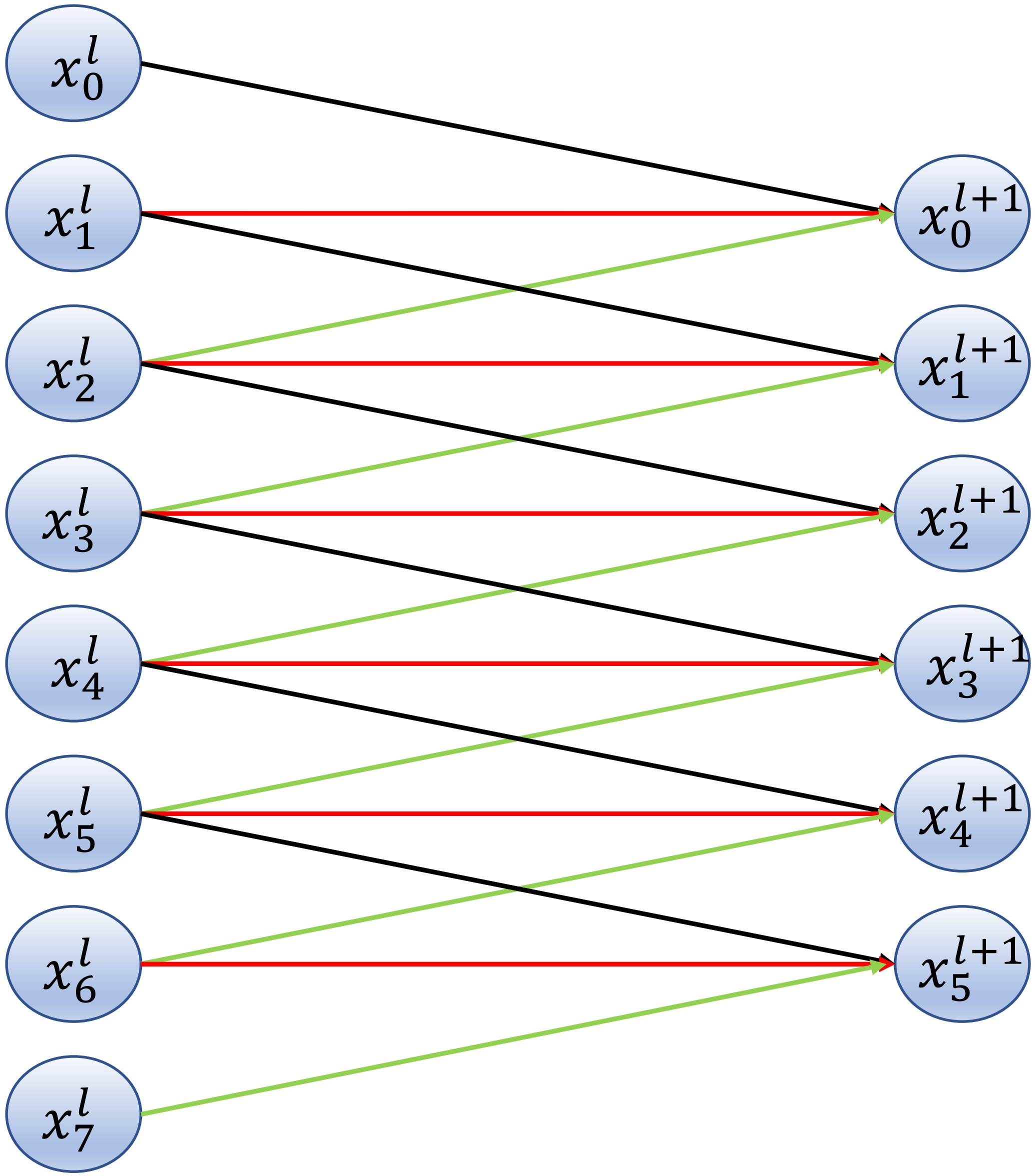




$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 & 0 \\ 0 & a & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

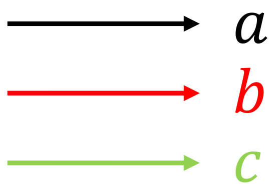
Convolution layer

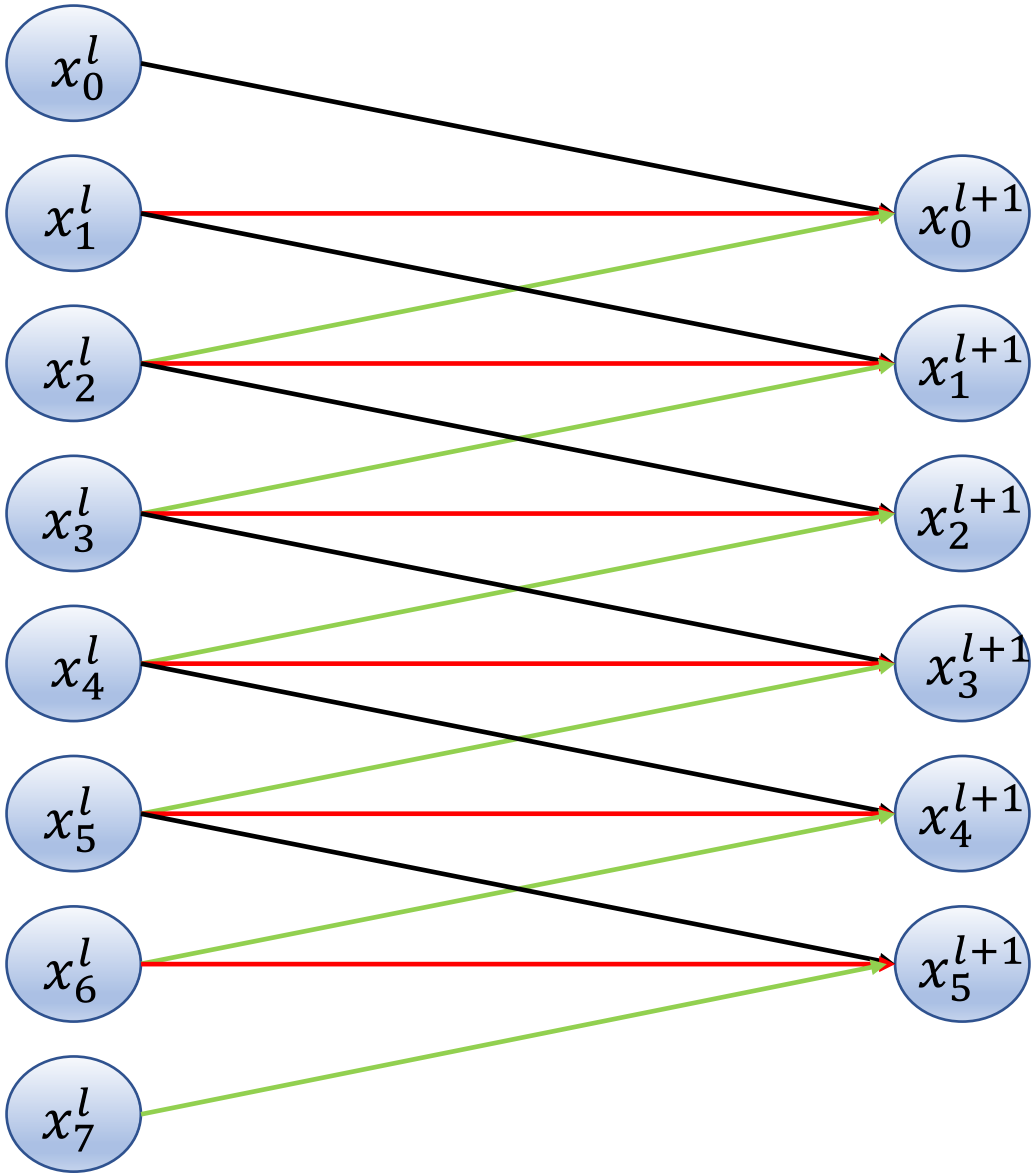




$$\begin{bmatrix} \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & 0 & 0 & 0 & 0 & 0 \\ 0 & \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & 0 & 0 & 0 & 0 \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

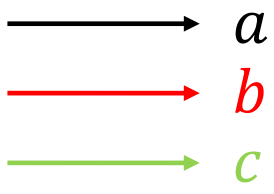
Convolution layer

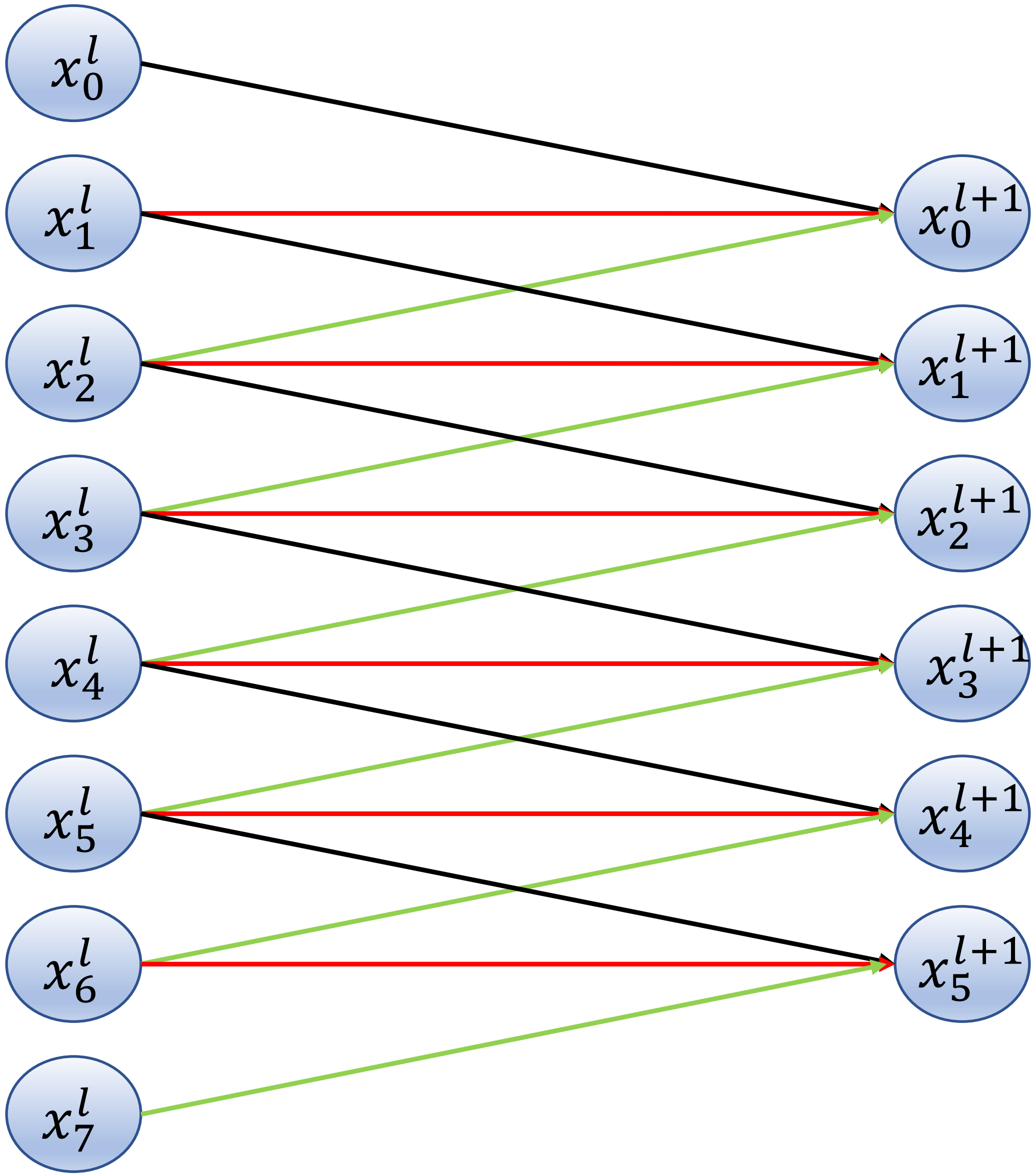




$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 & 0 \\ 0 & a & b & c & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 & 0 \\ 0 & 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c & 0 \\ 0 & 0 & 0 & 0 & 0 & a & b & c \end{bmatrix}$$

Convolution layer

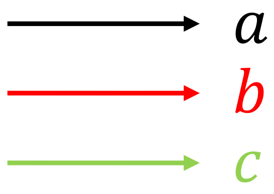


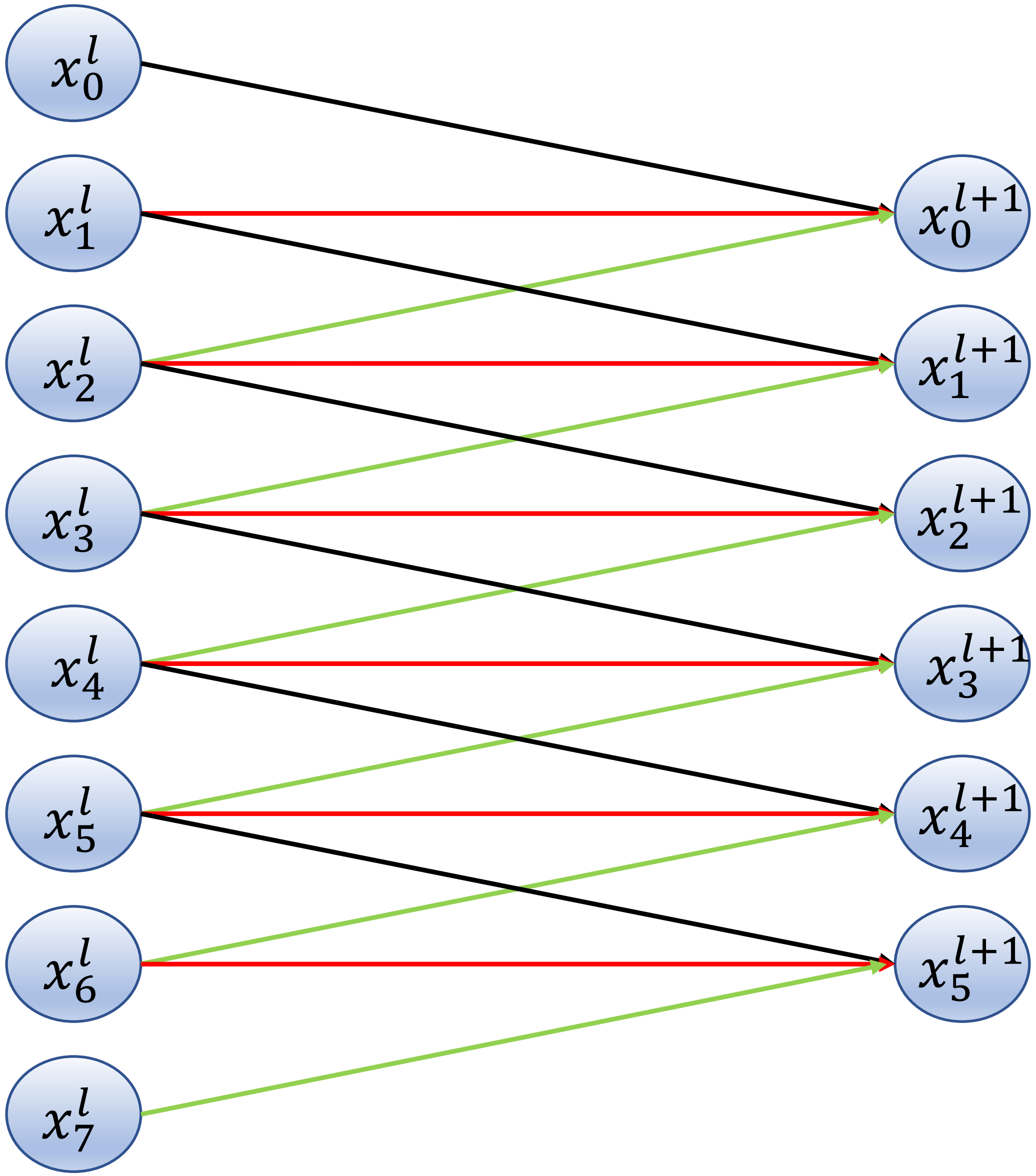


$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 & 0 \\ 0 & a & b & c & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 & 0 \\ 0 & 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c & 0 \\ 0 & 0 & 0 & 0 & 0 & a & b & c \end{bmatrix}$$

Toeplitz matrix

Convolution layer



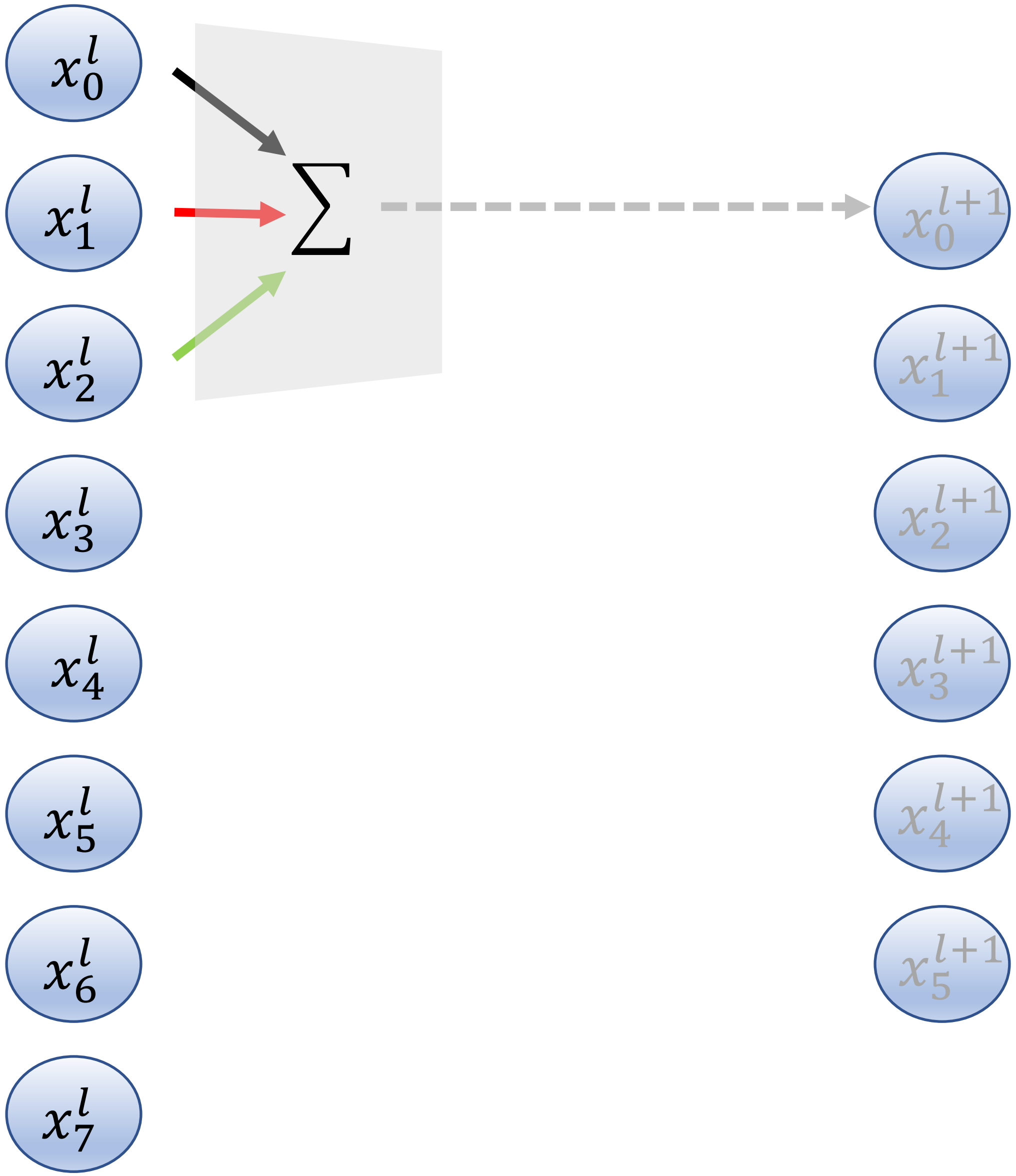


$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 & 0 \\ 0 & a & b & c & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 & 0 \\ 0 & 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c & 0 \\ 0 & 0 & 0 & 0 & 0 & a & b & c \end{bmatrix}$$

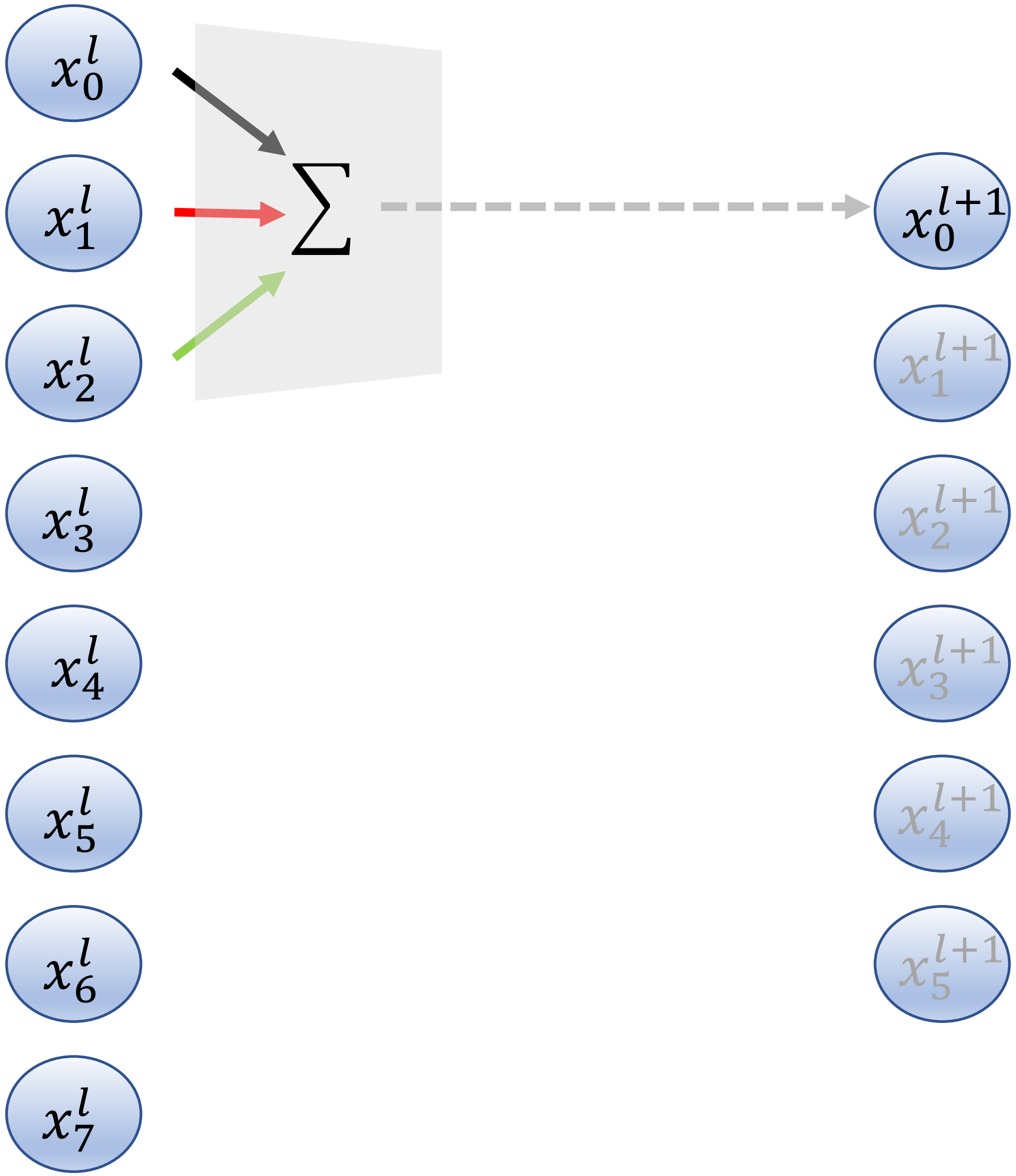
Special case of Linear Layer

Toeplitz matrix

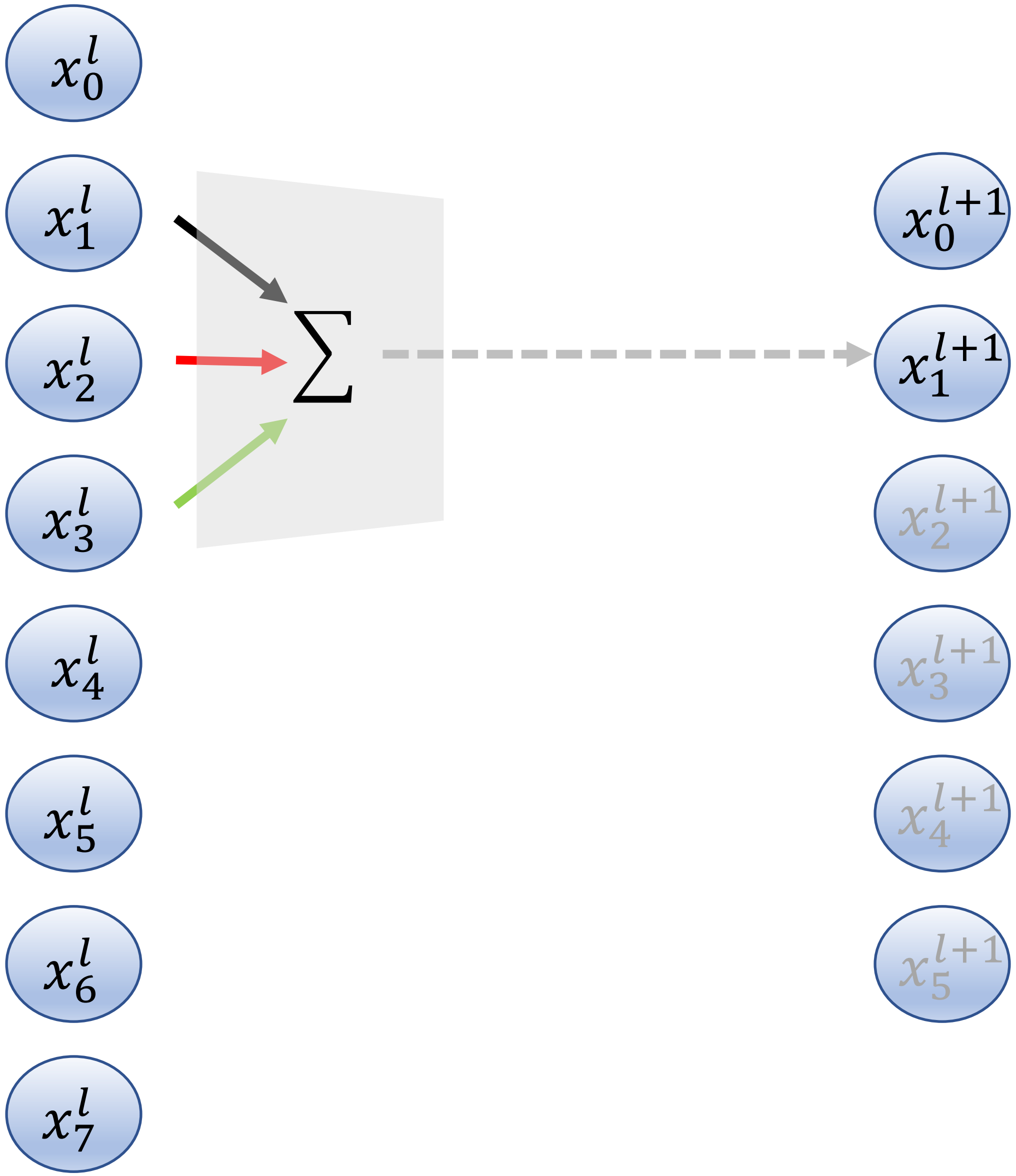
Convolution Filter



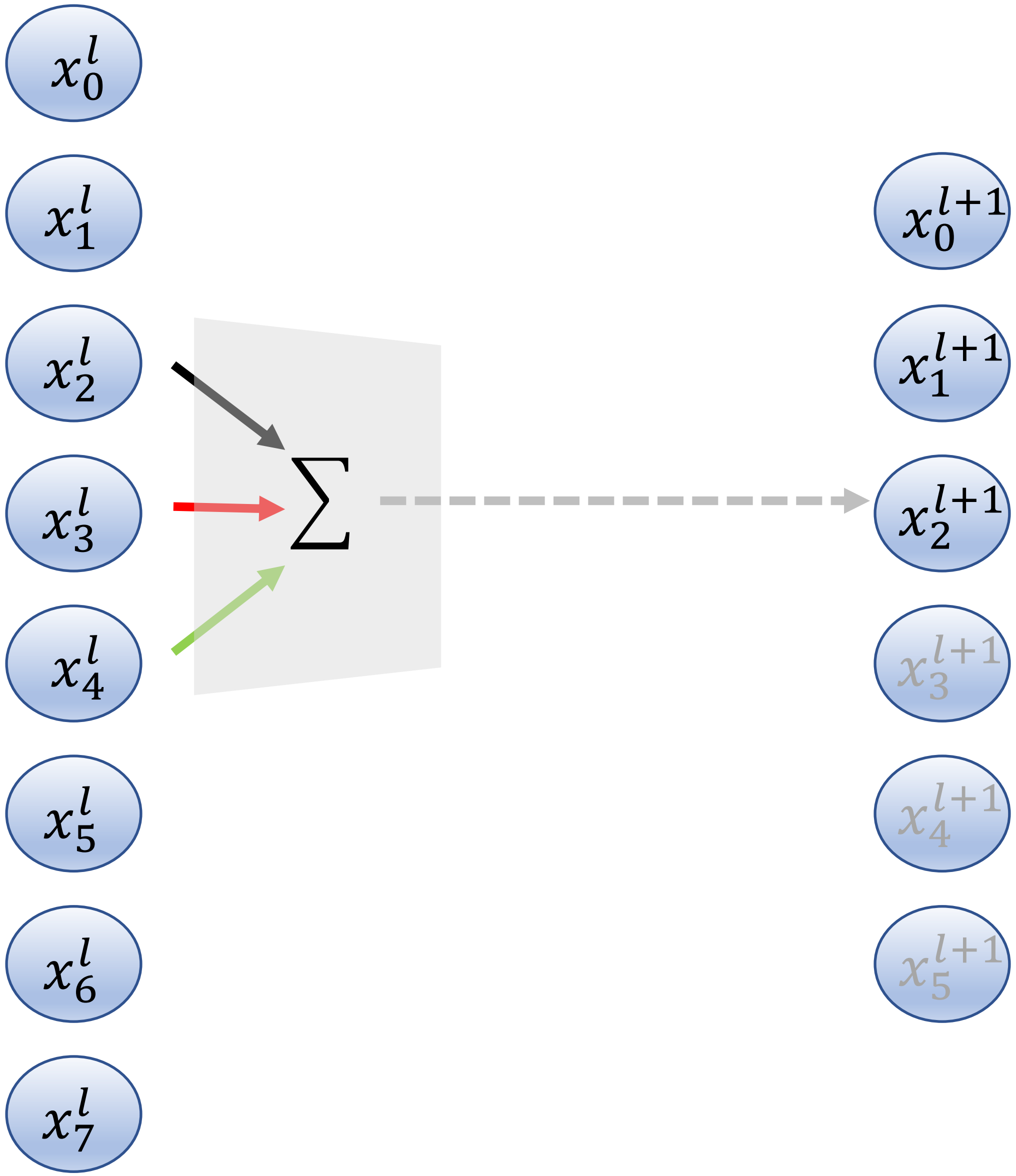
Convolution Filter



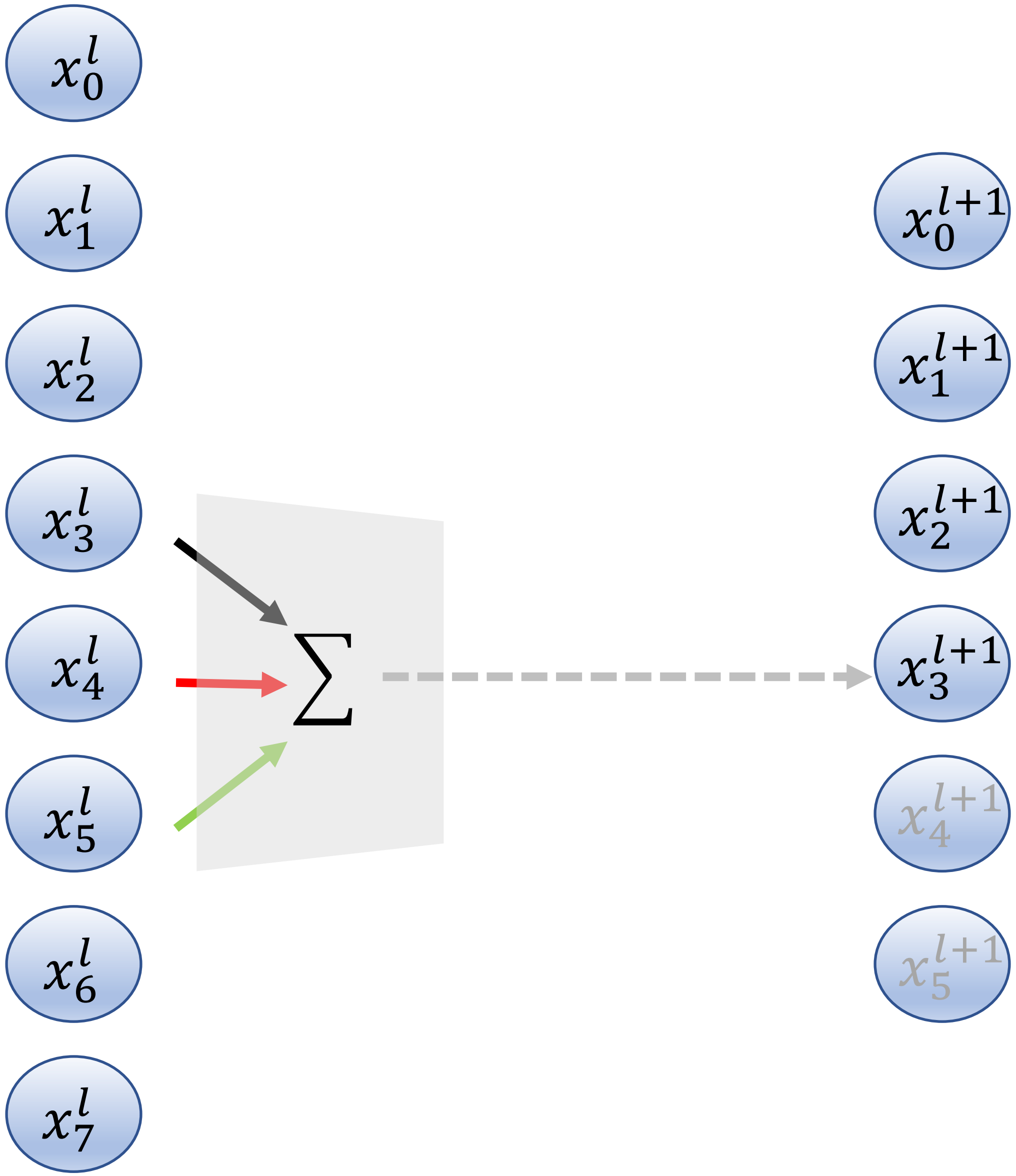
Convolution Filter



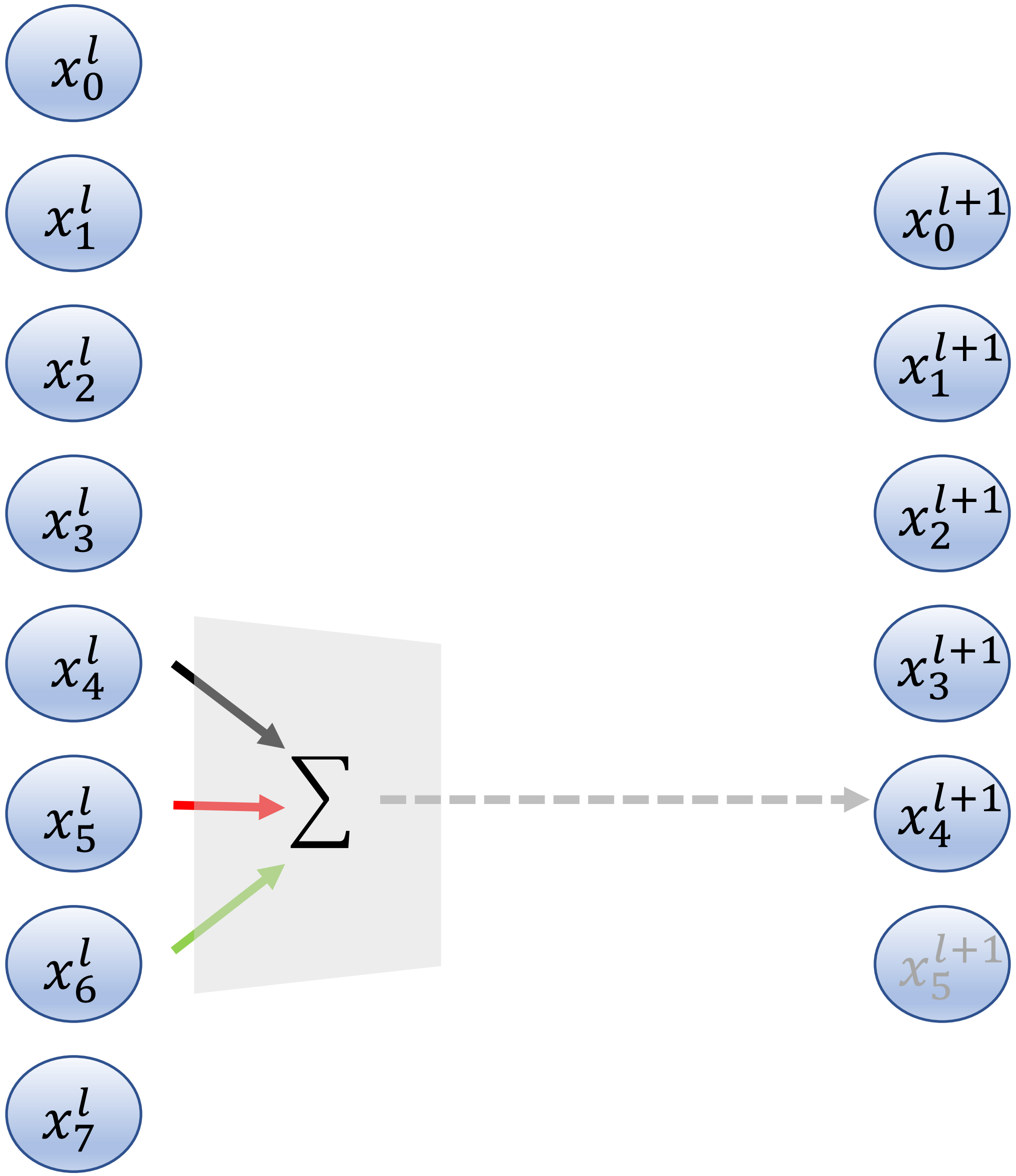
Convolution Filter



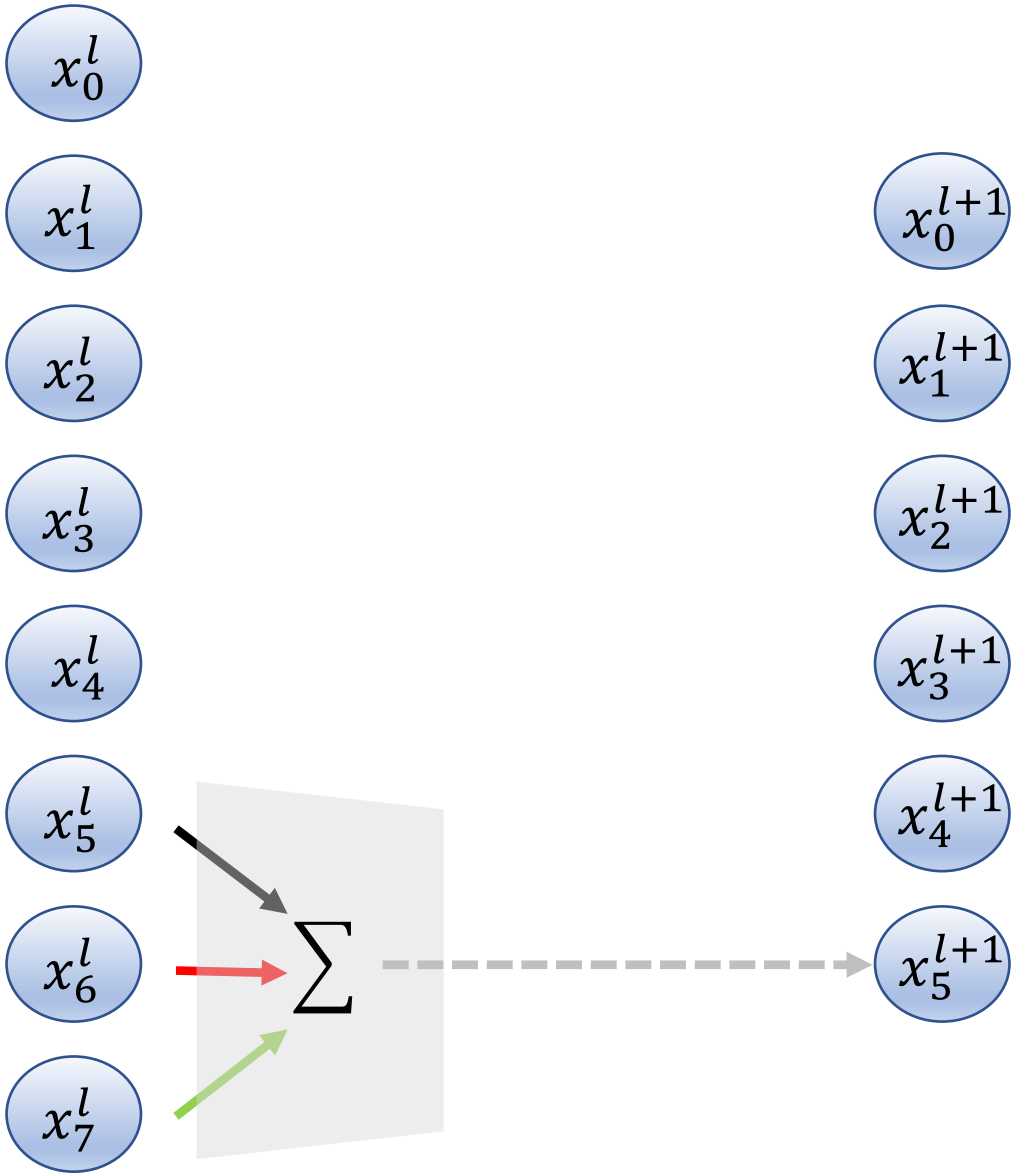
Convolution Filter



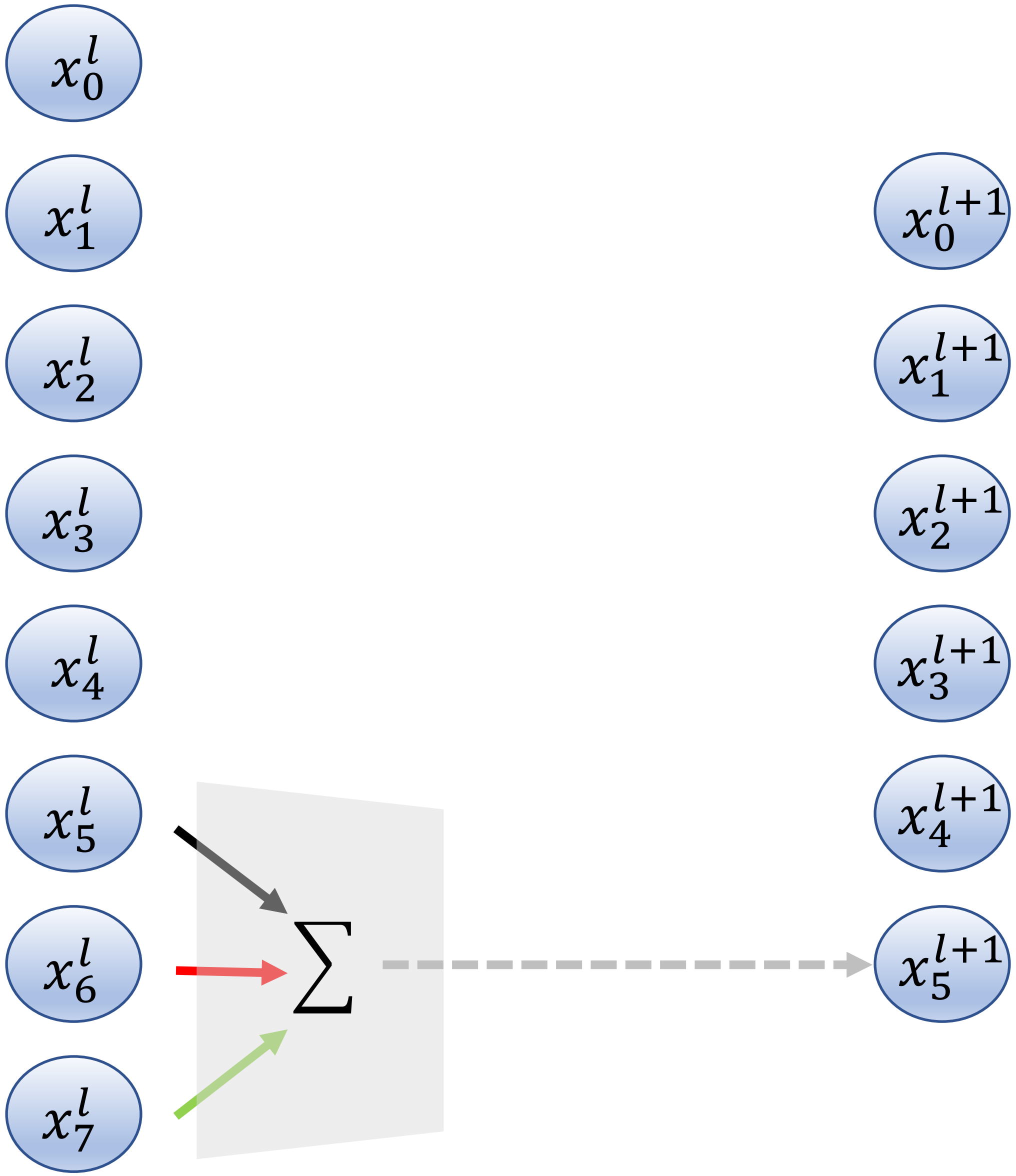
Convolution Filter



Convolution Filter

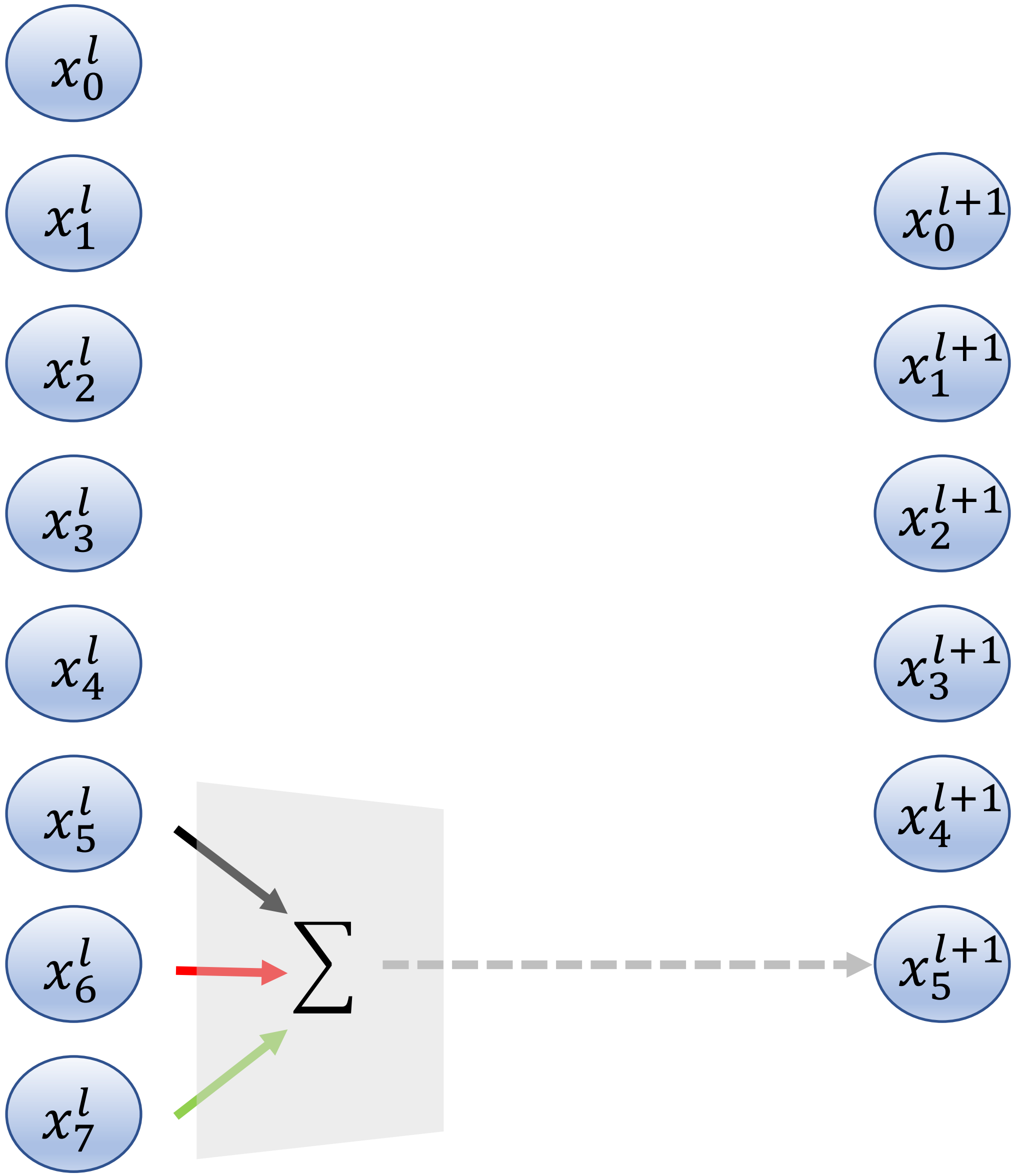


Convolution Filter



Q: Given input size N and filter size K, find output size.

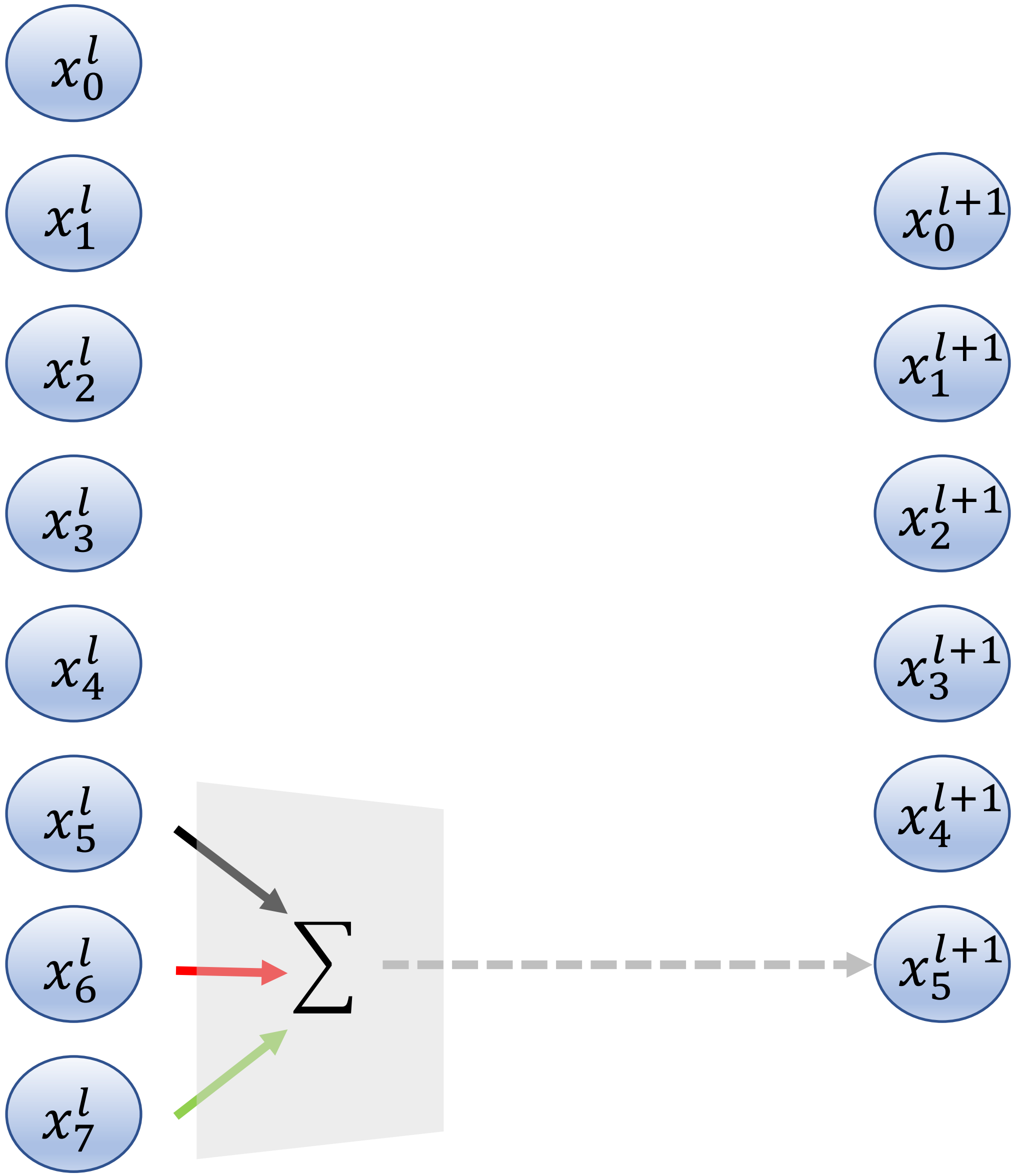
Convolution Filter



Q: Given input size N and filter size K, find output size.

A: $N - K + 1$

Convolution Filter

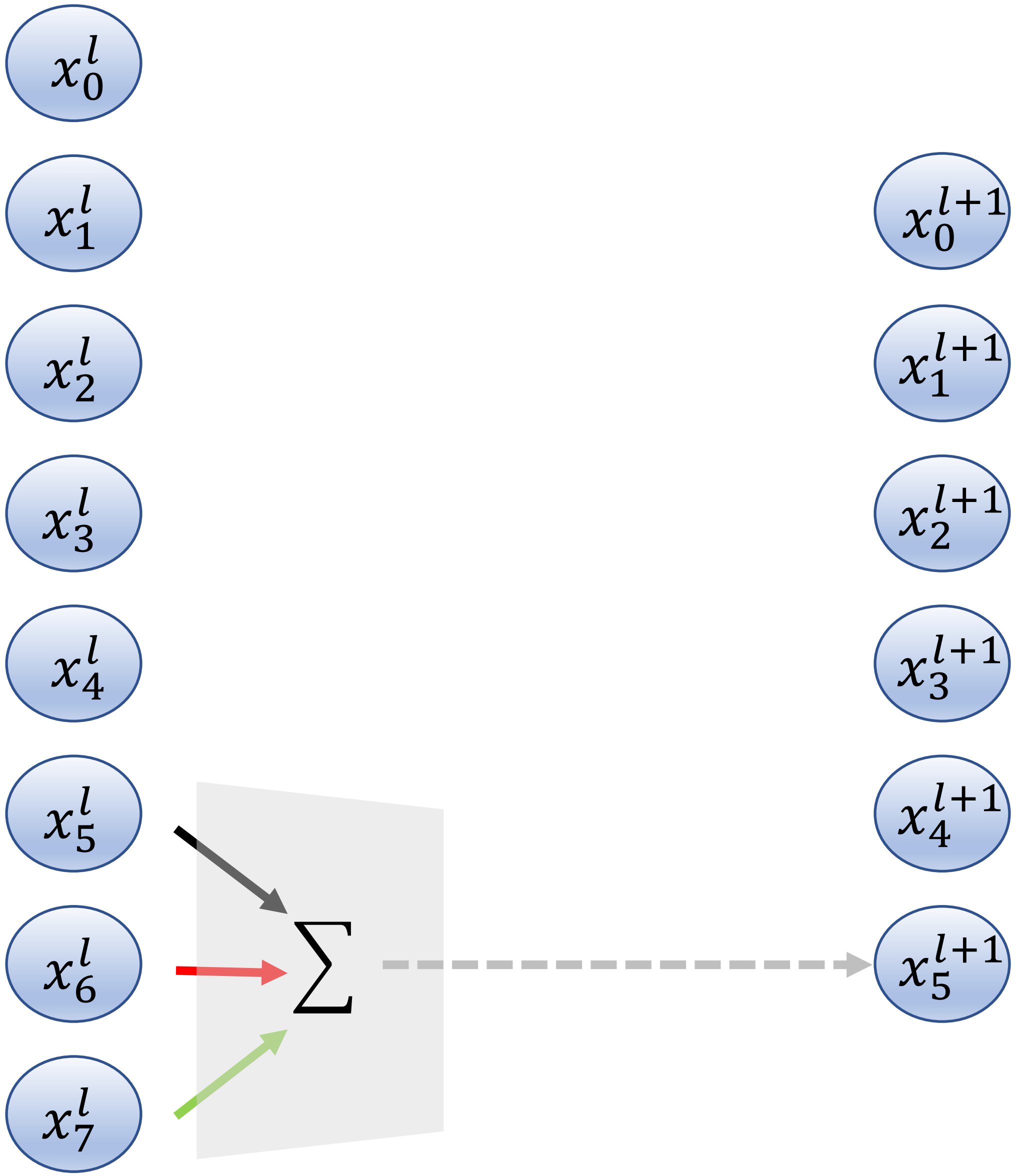


Q: Given input size N and filter size K, find output size.

A: $N - K + 1$

Q: is this a convolution?

Convolution Filter

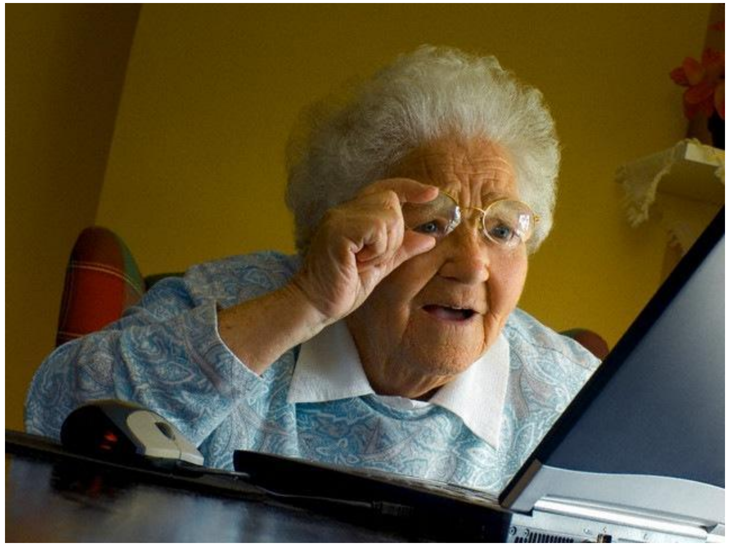


Q: Given input size N and filter size K, find output size.

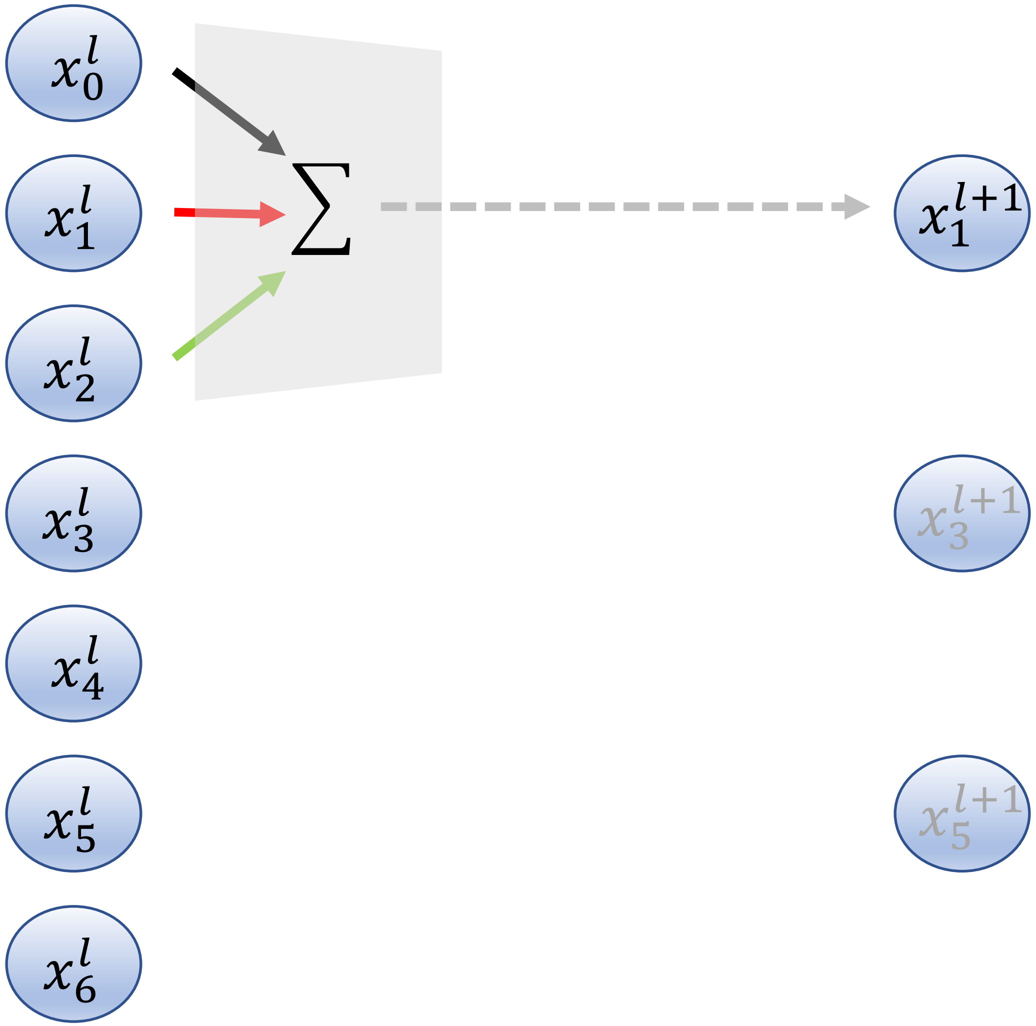
A: $N - K + 1$

Q: is this a convolution?

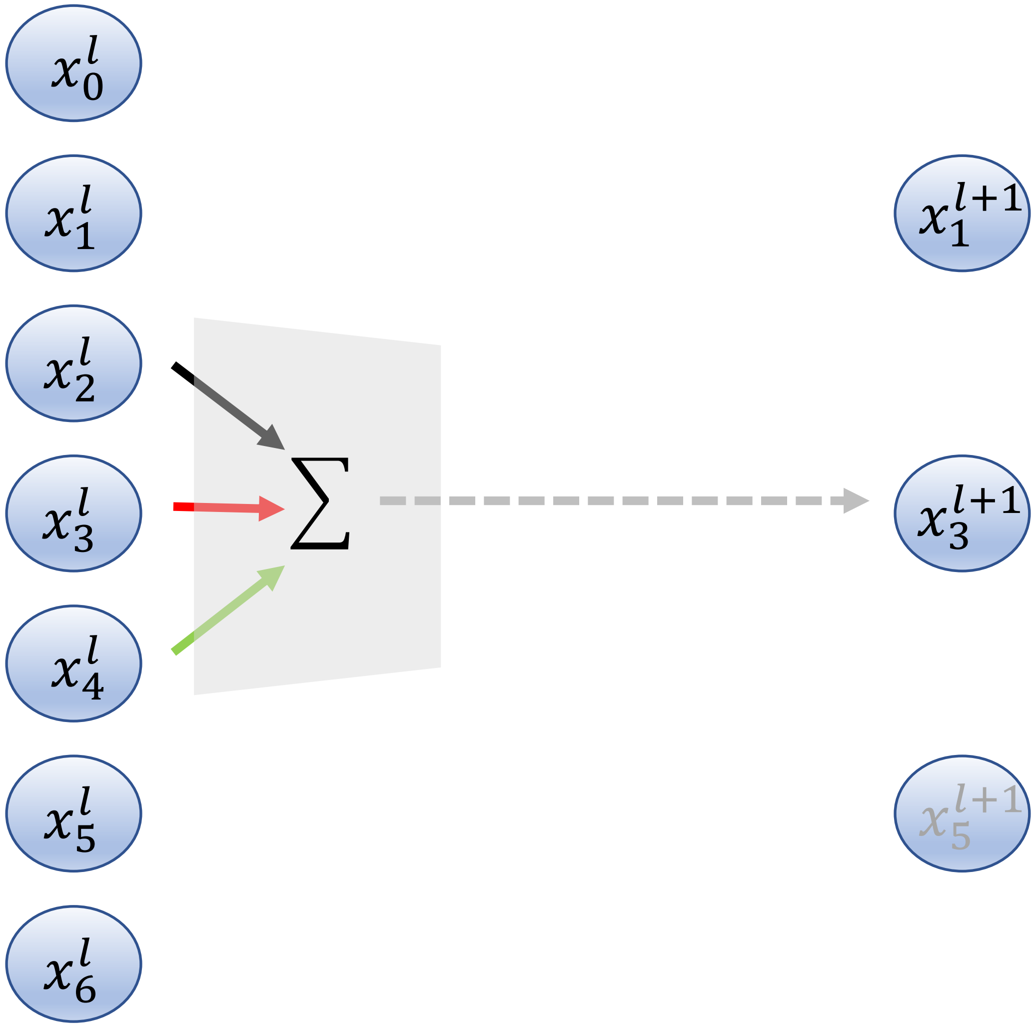
A: Yes, but with the flipped filter.
This is cross-correlation.



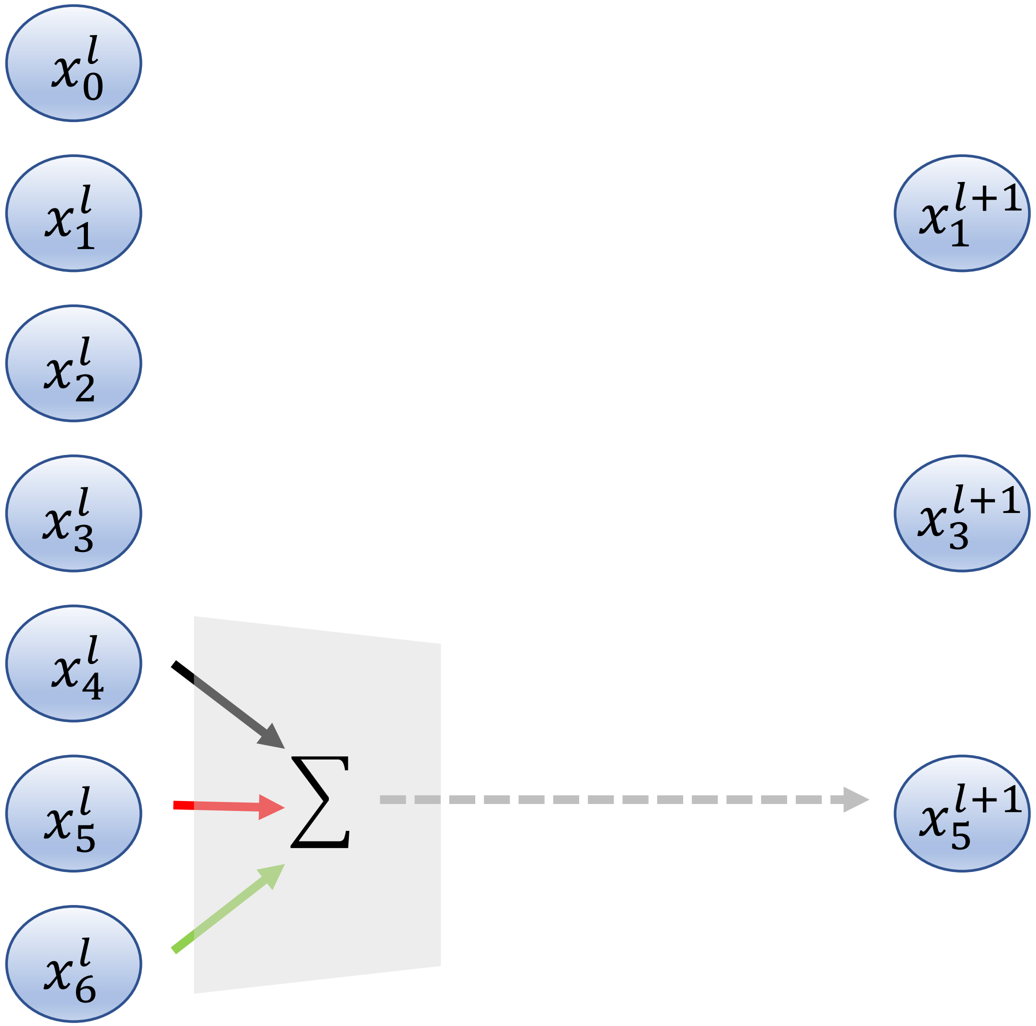
Stride



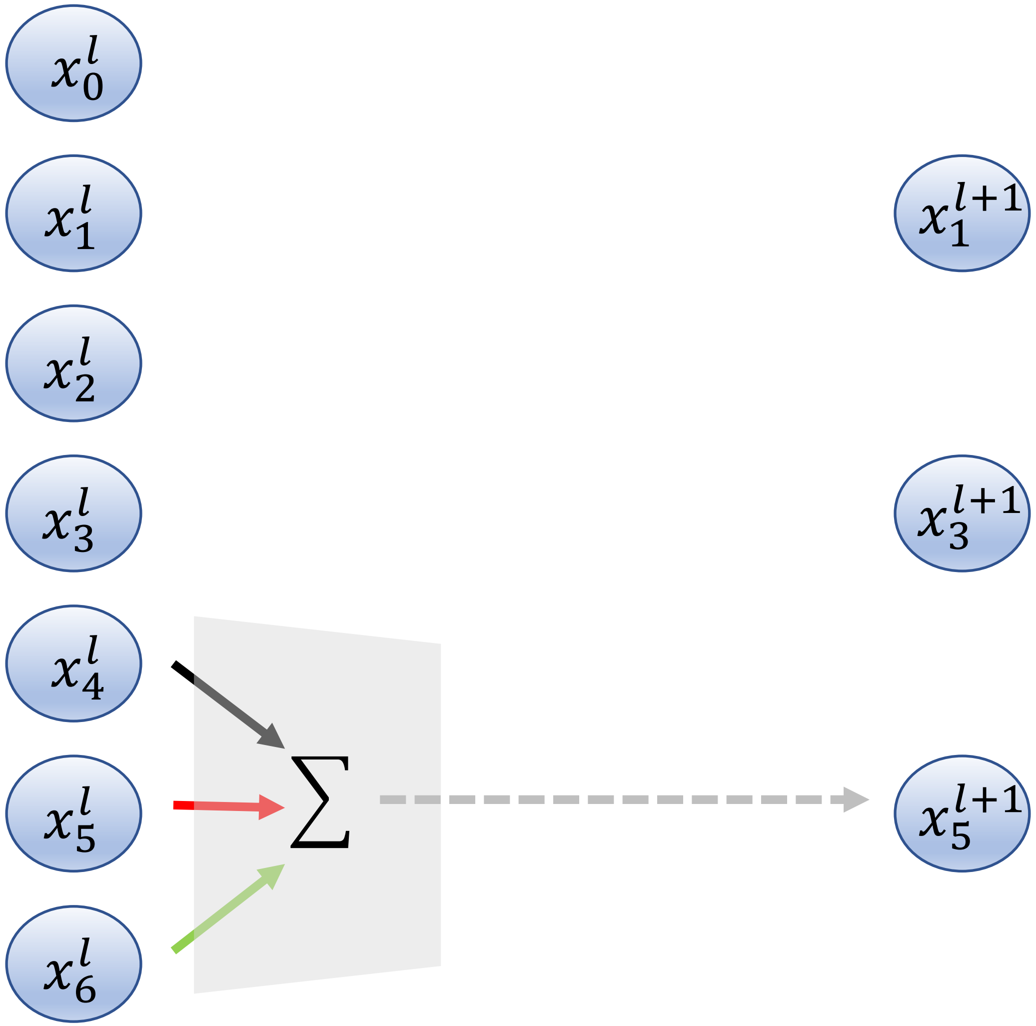
Stride



Stride

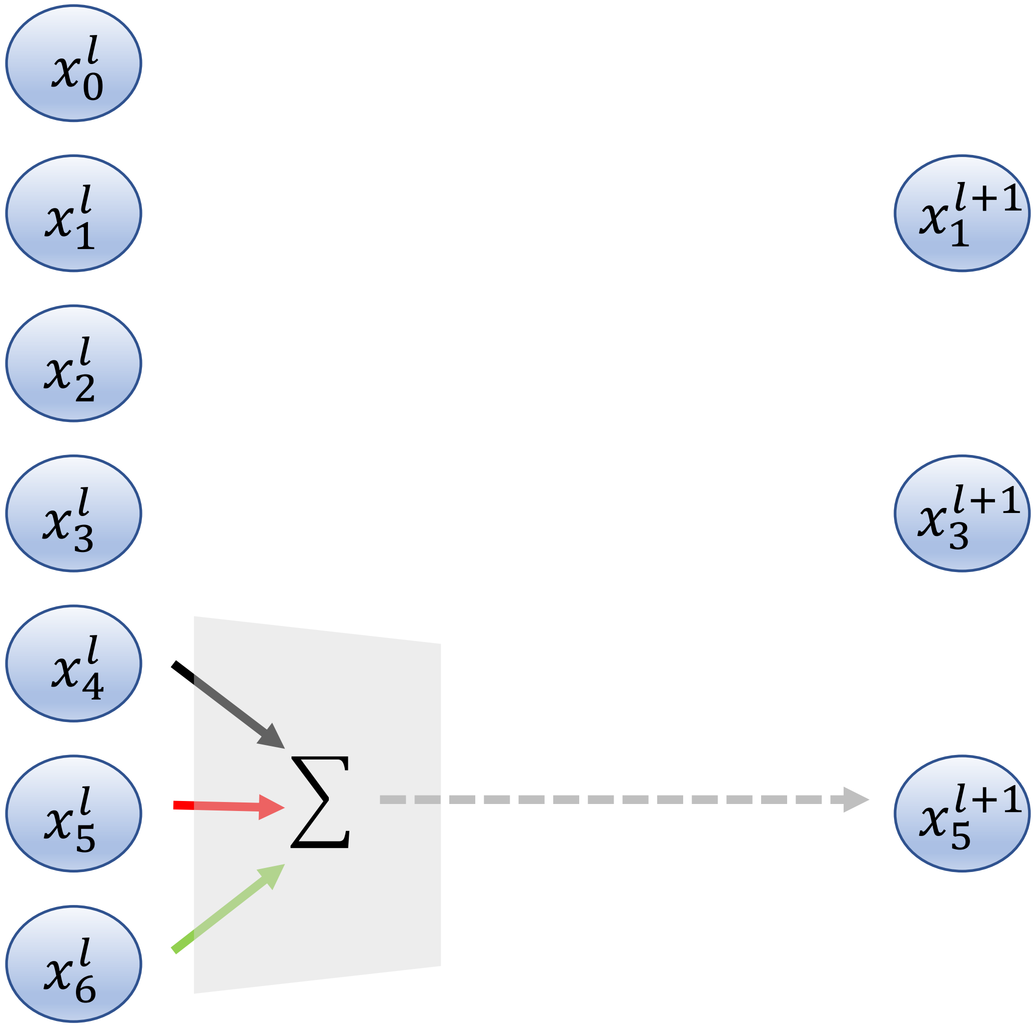


Stride



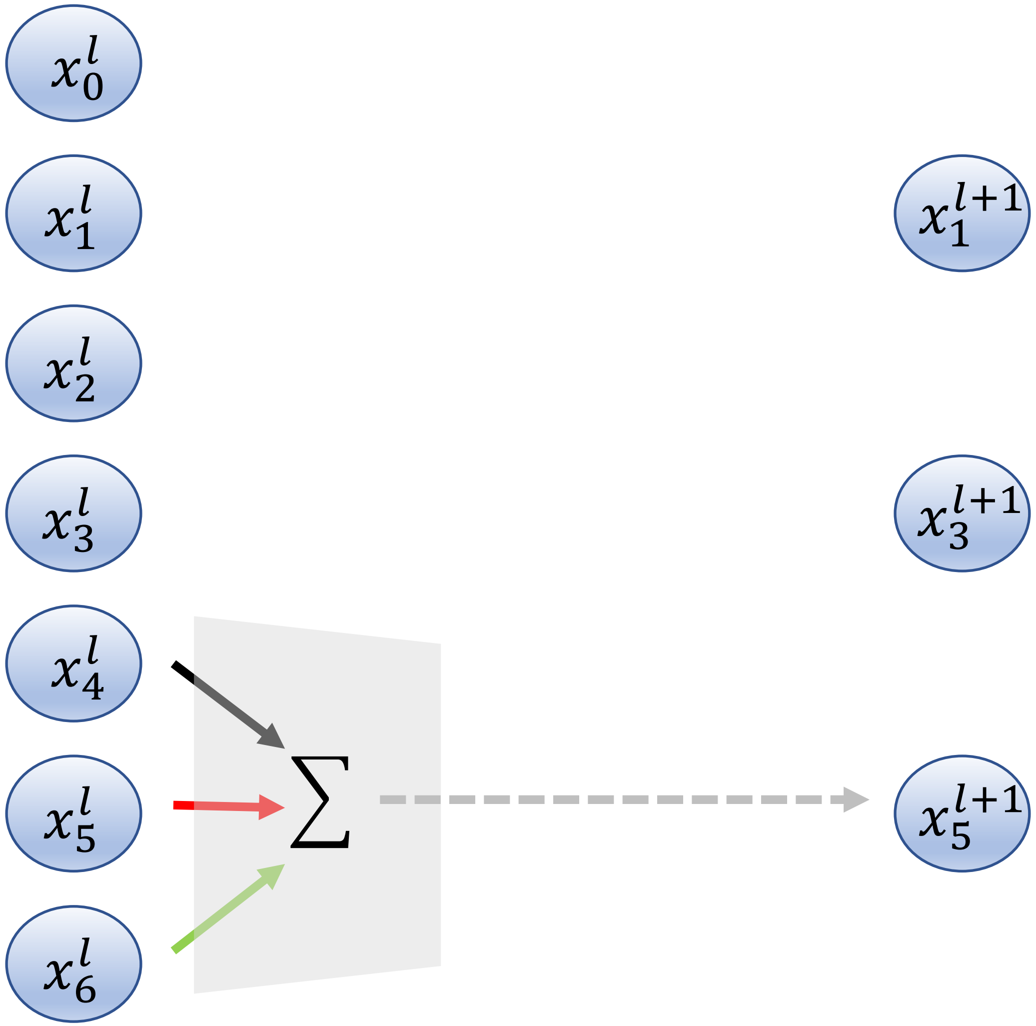
$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 \\ 0 & a & b & c & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & a & b & c & 0 \\ 0 & 0 & 0 & 0 & a & b & c \\ 0 & 0 & 0 & 0 & 0 & a & b \end{bmatrix}$$

Stride



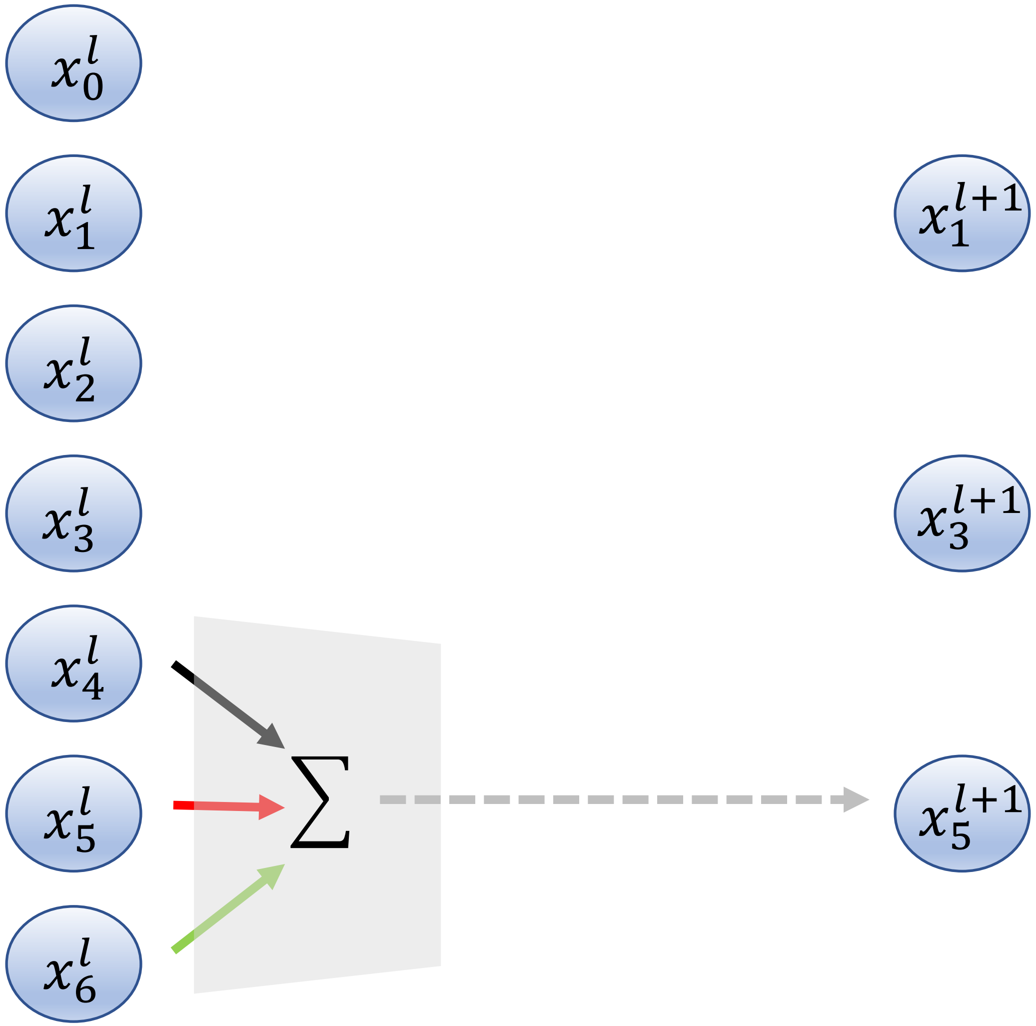
$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 \\ 0 & a & b & c & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & a & b & c & 0 \\ 0 & 0 & 0 & 0 & a & b & c \\ 0 & 0 & 0 & 0 & 0 & a & b \end{bmatrix}$$

Stride



$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c \end{bmatrix}$$

Stride

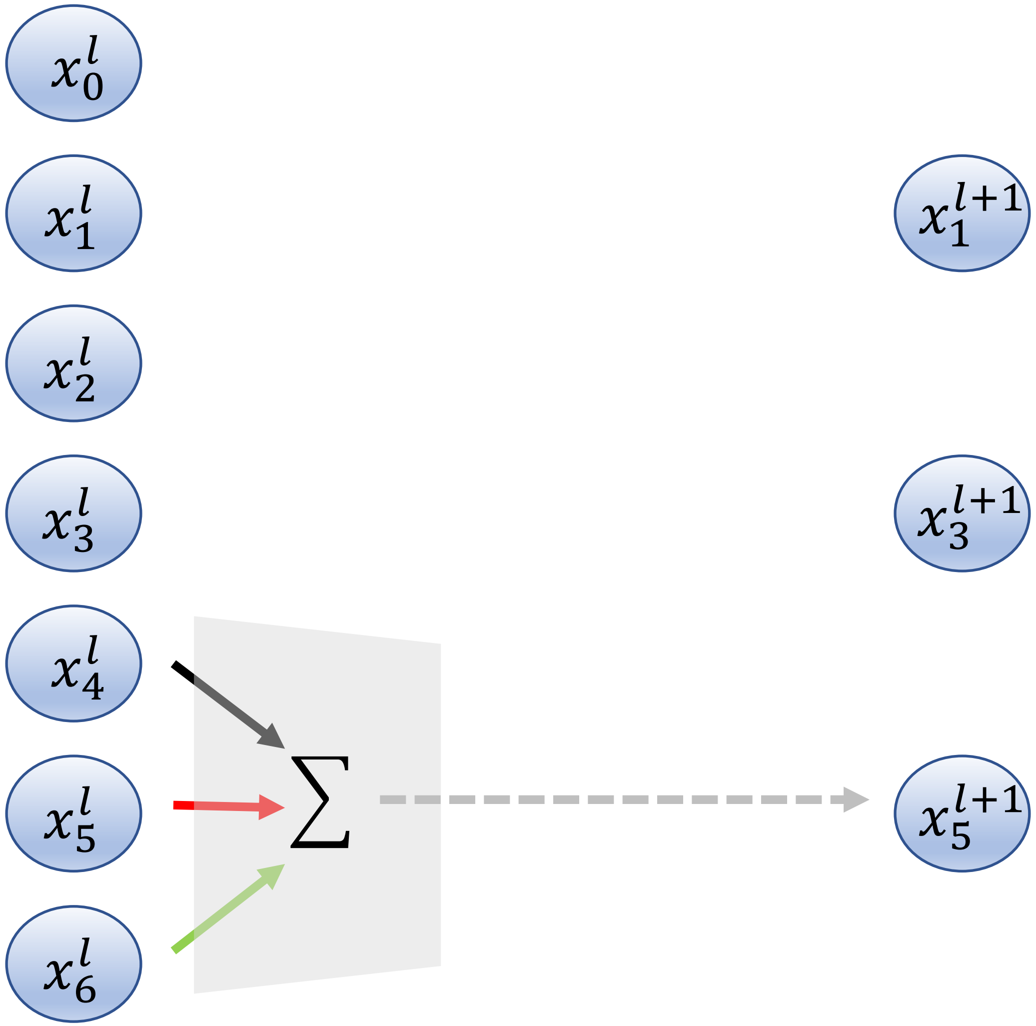


$$\begin{bmatrix} \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} \end{bmatrix}$$

Q: Given input size N, filter size K and stride S , find output size.

Stride

[redacted]

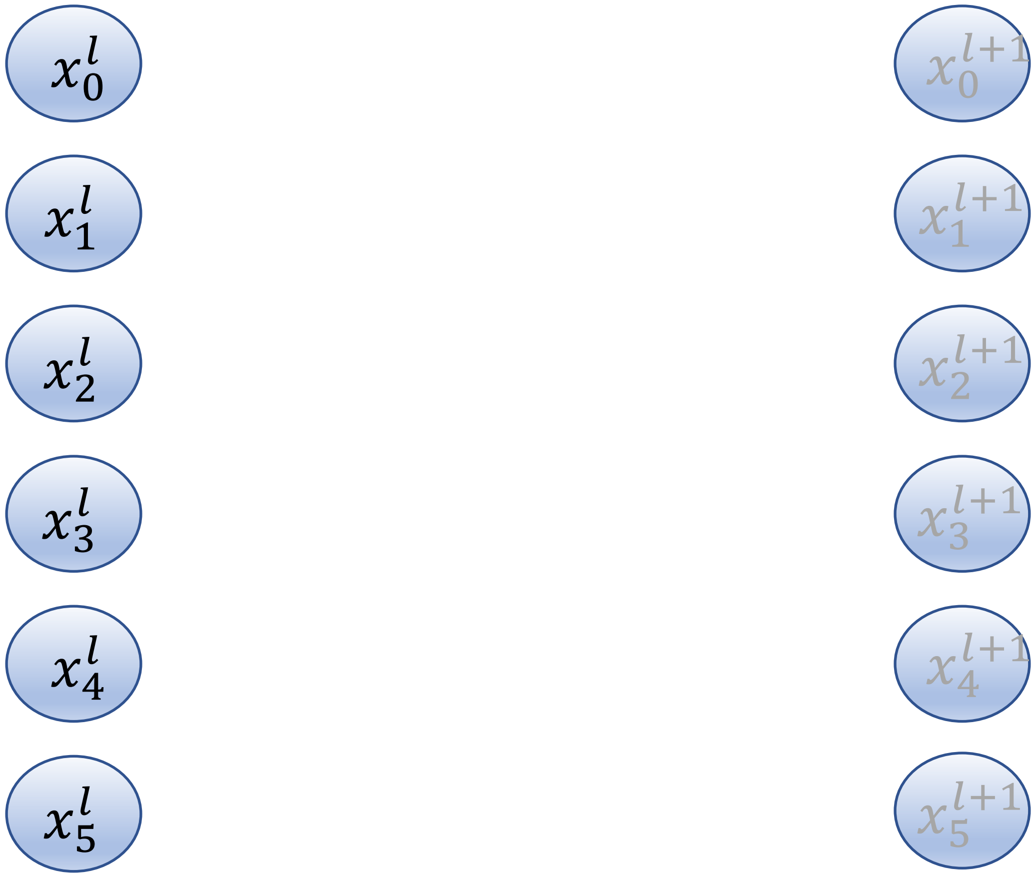


$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c \end{bmatrix}$$

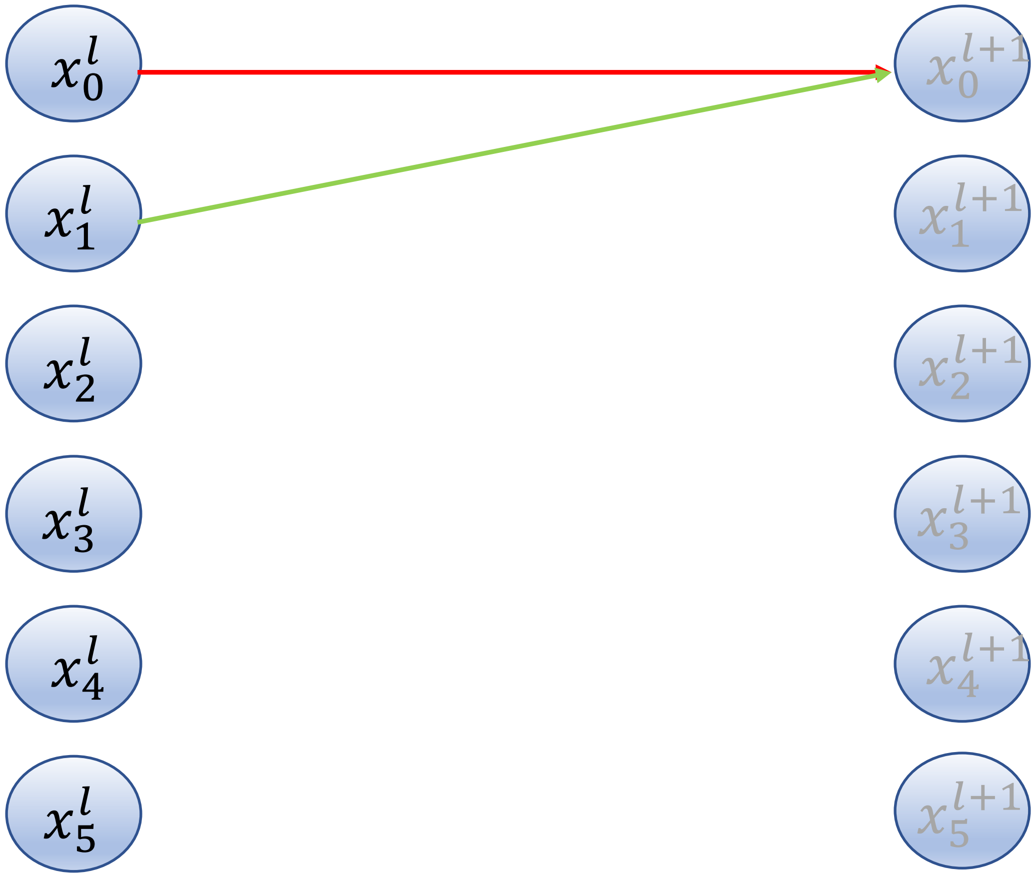
Q: Given input size N, filter size K and stride S , find output size.

A: $\left\lfloor \frac{N-K}{S} + 1 \right\rfloor$

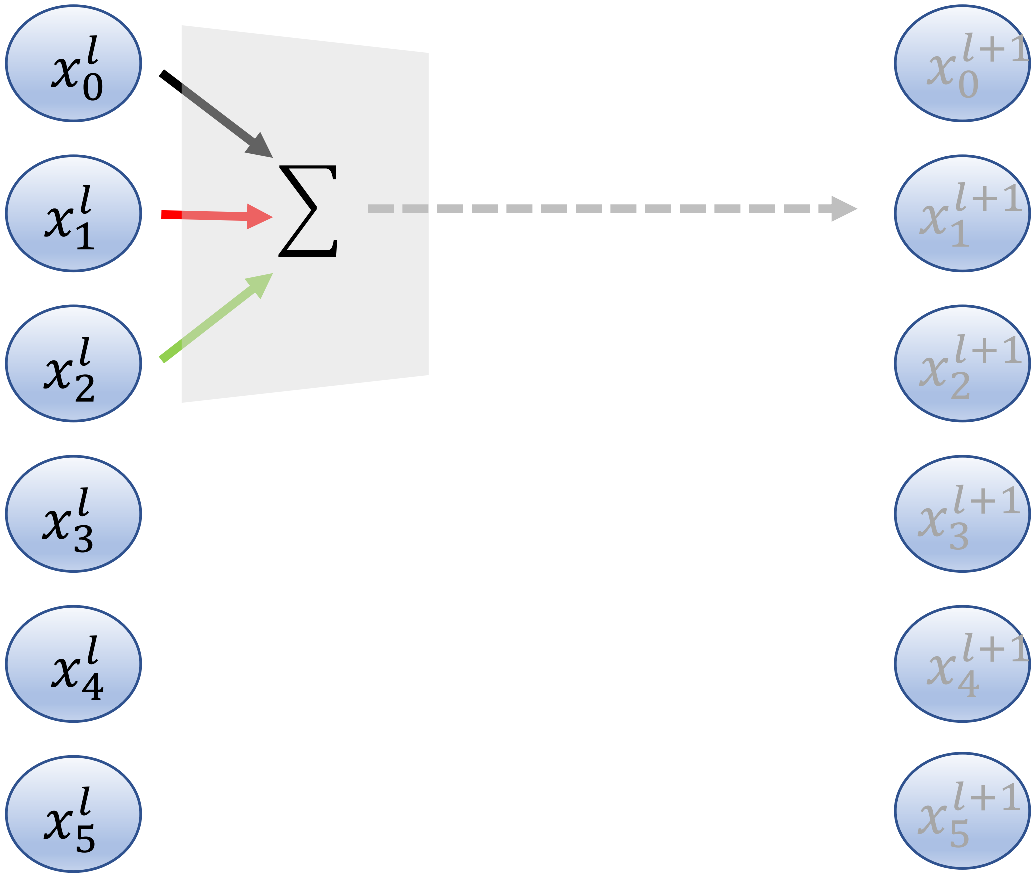
Padding



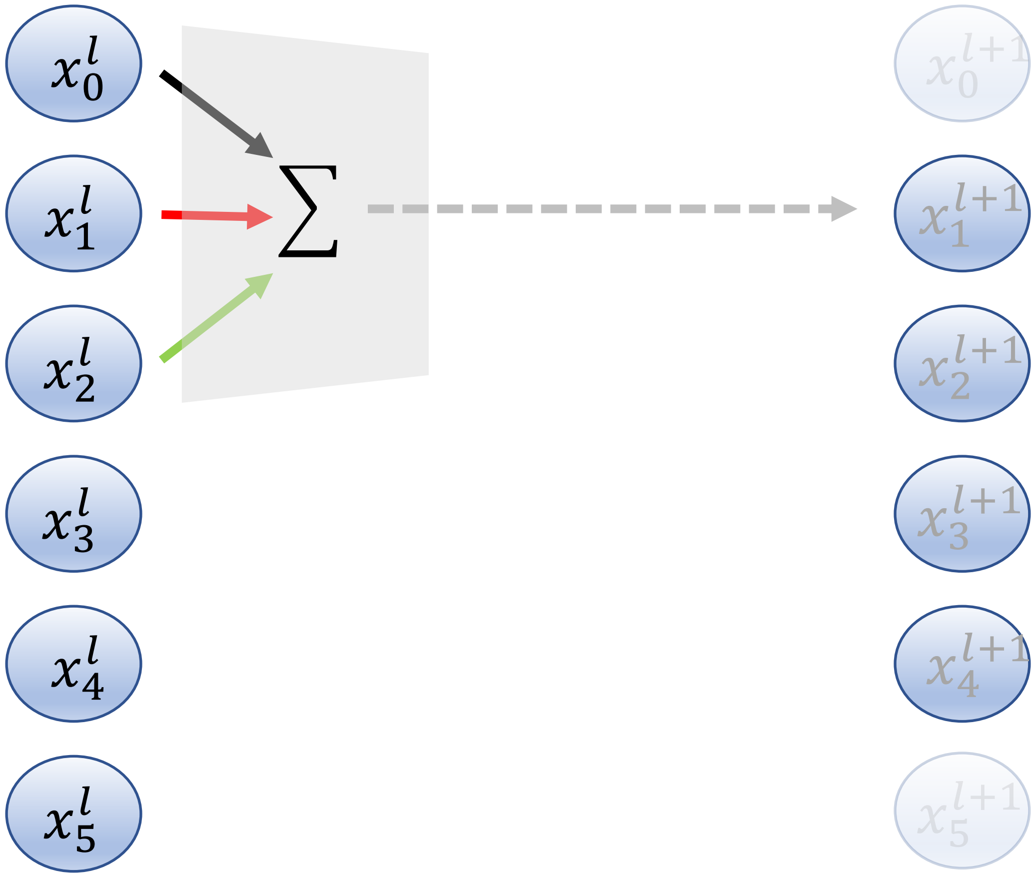
Padding



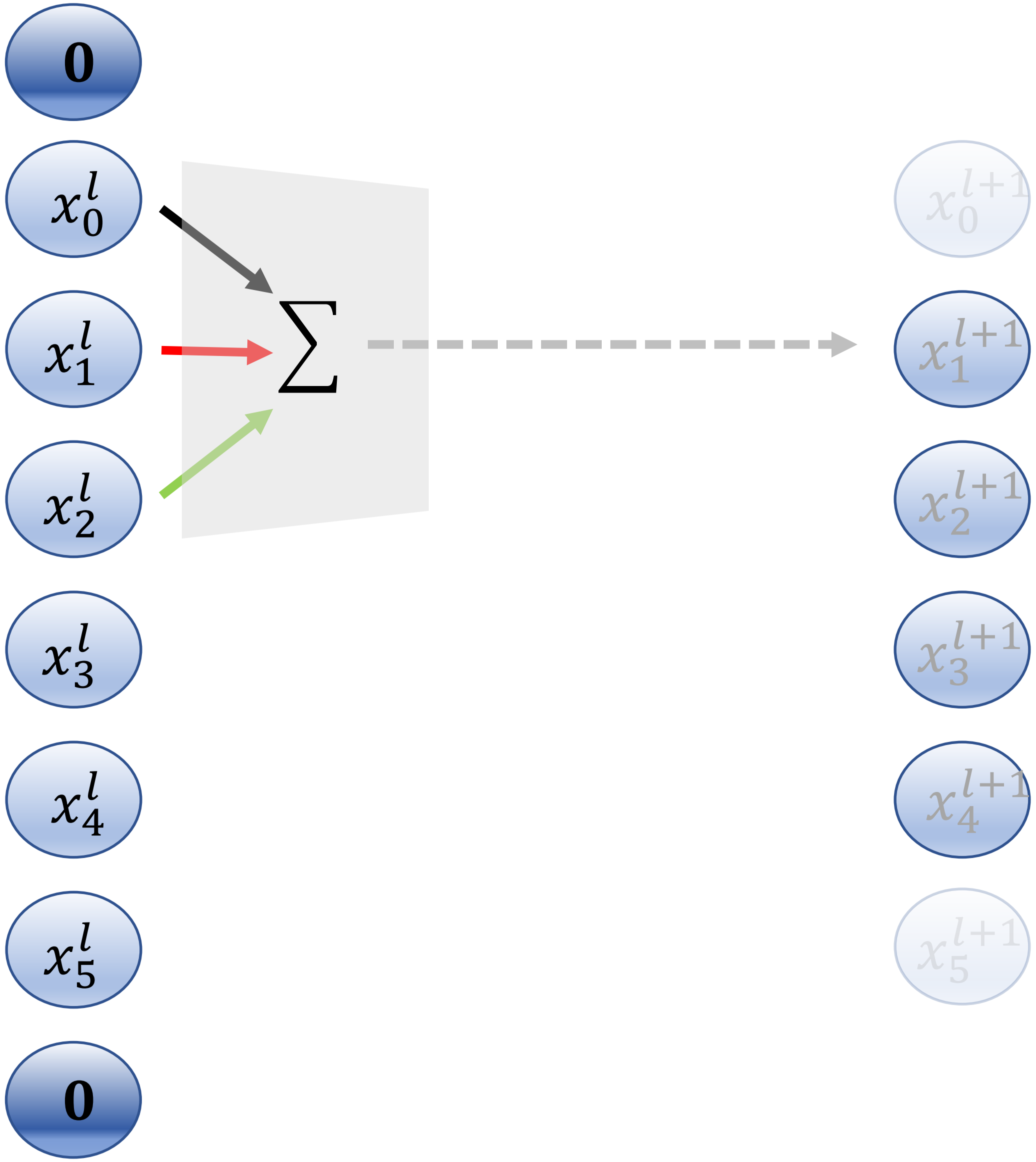
Padding



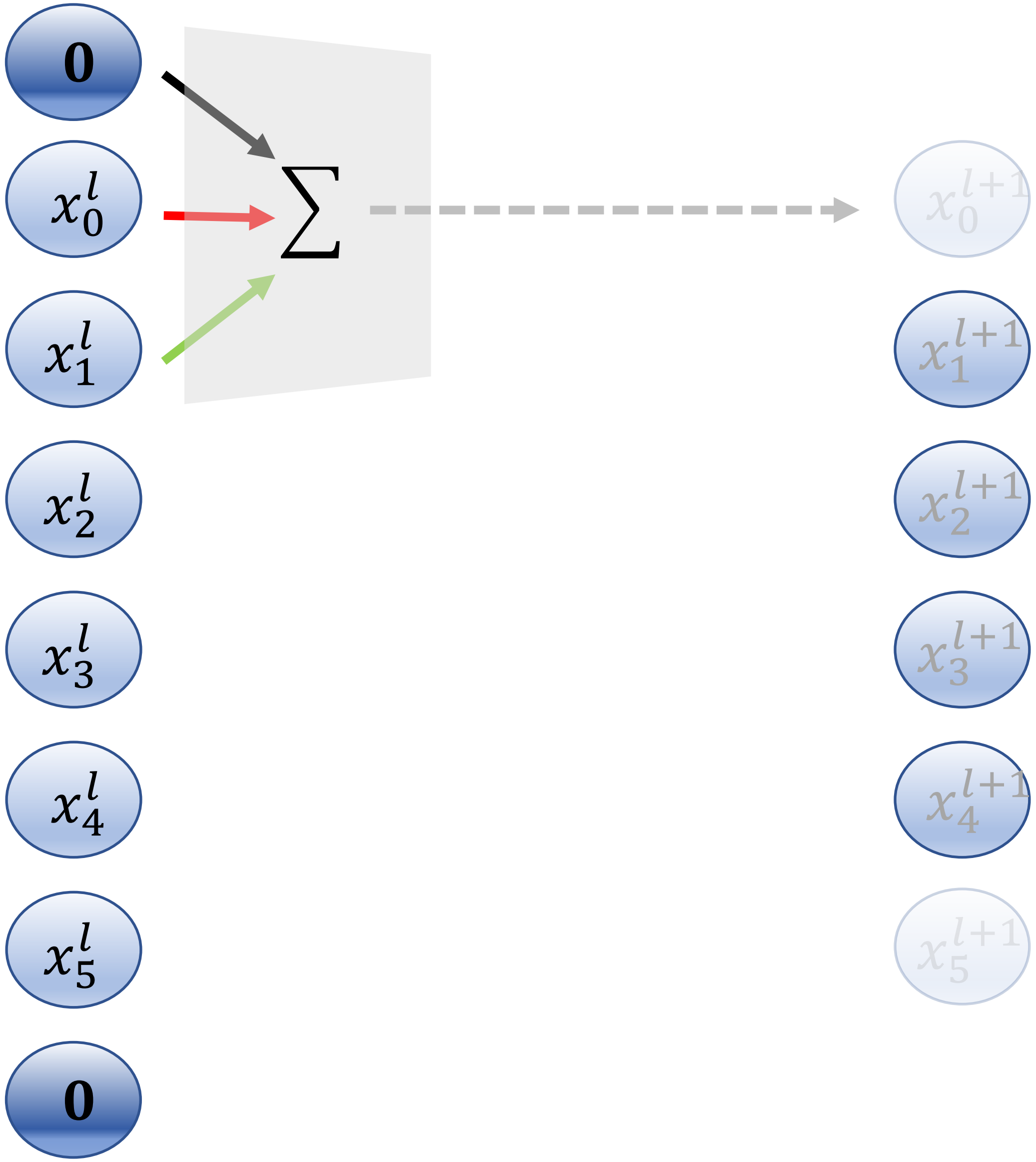
Padding



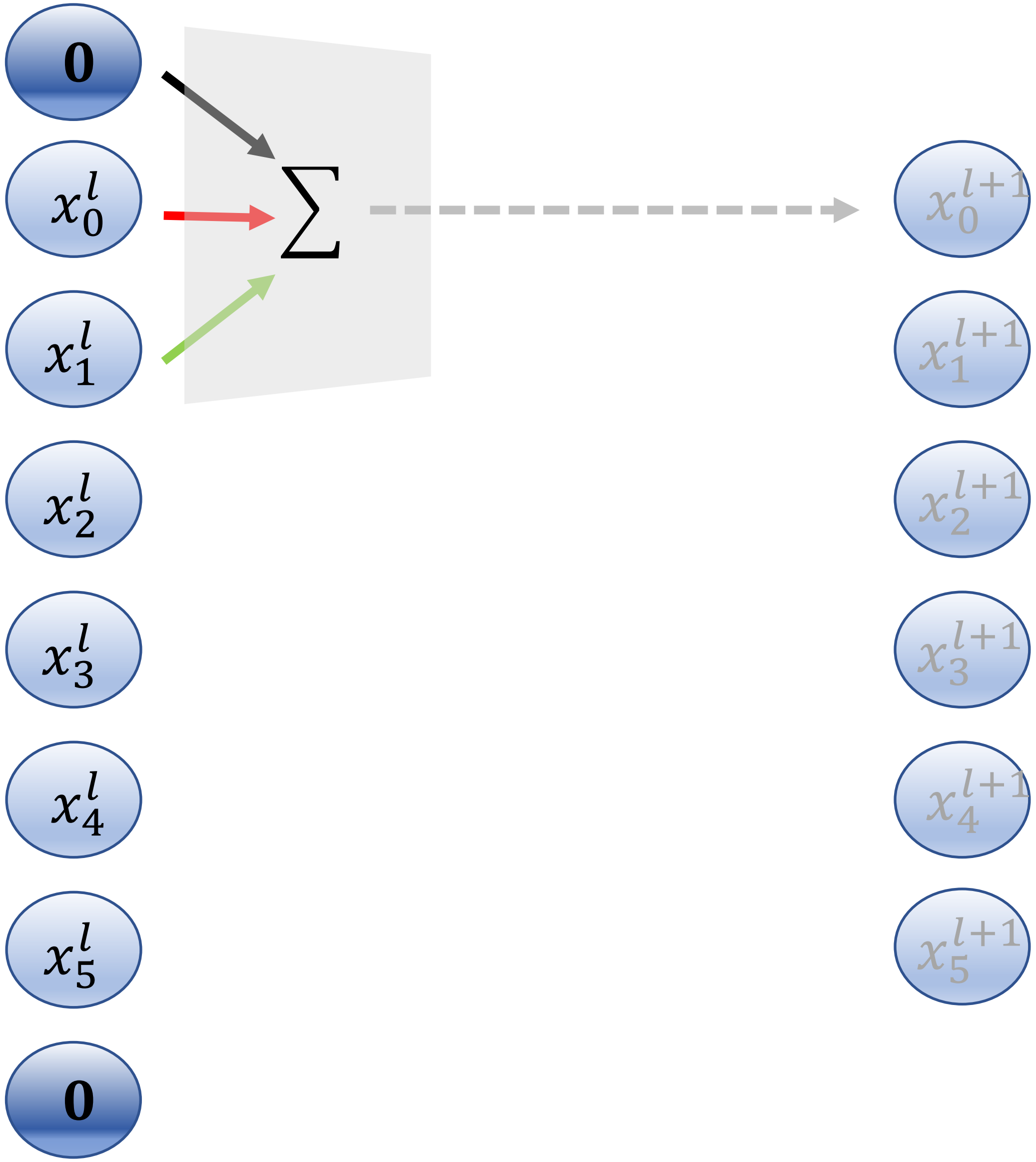
Padding



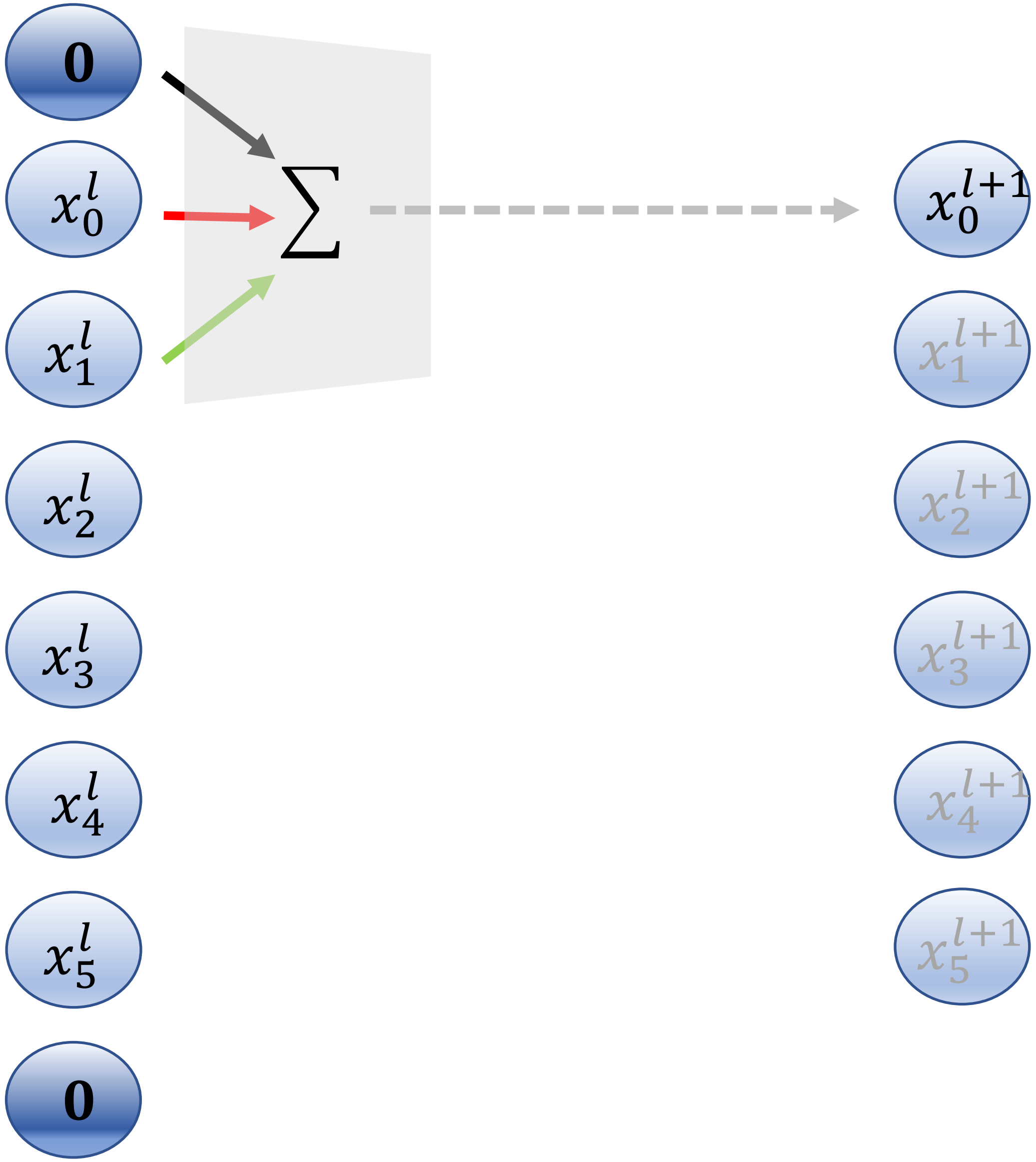
Padding



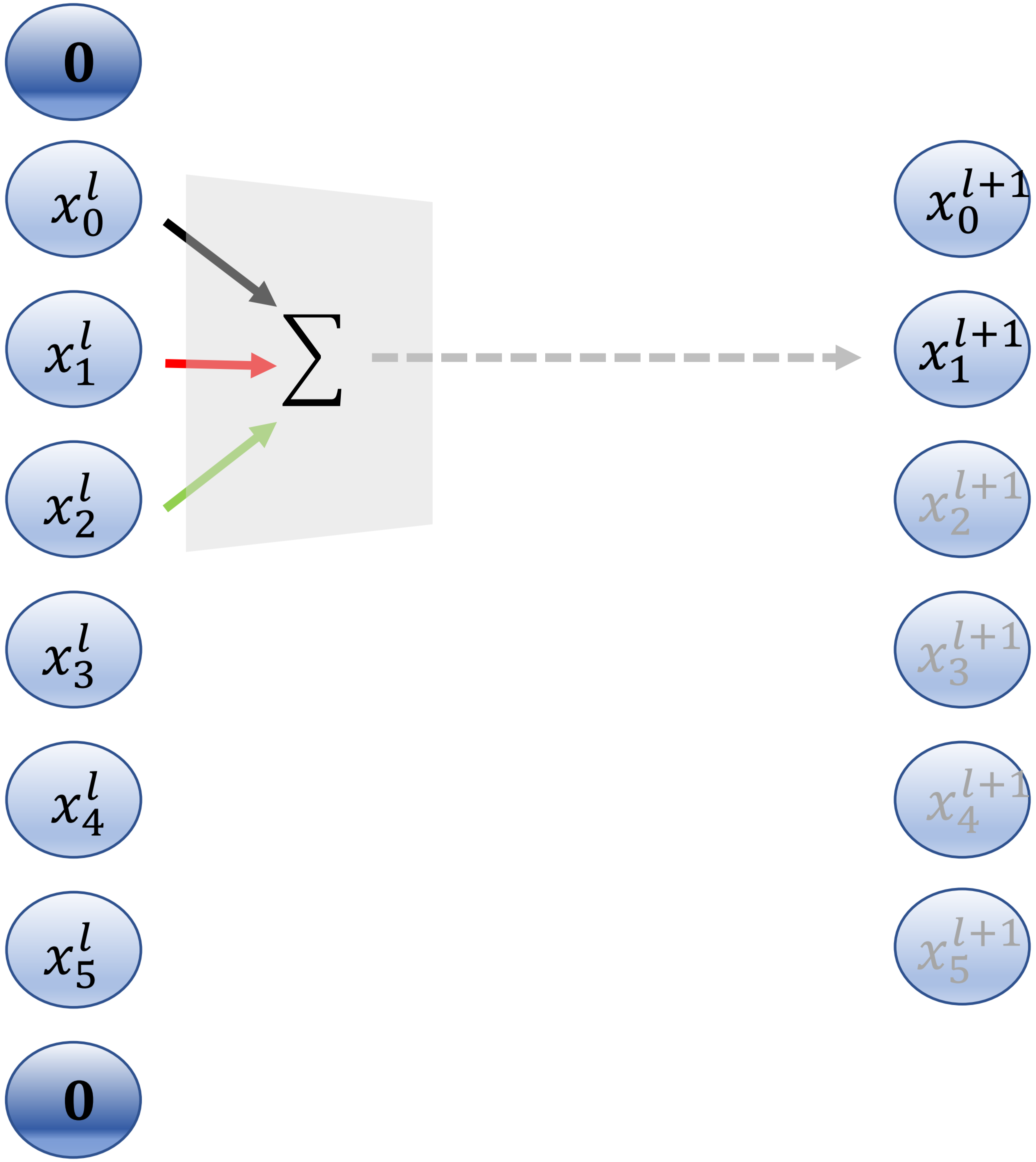
Padding



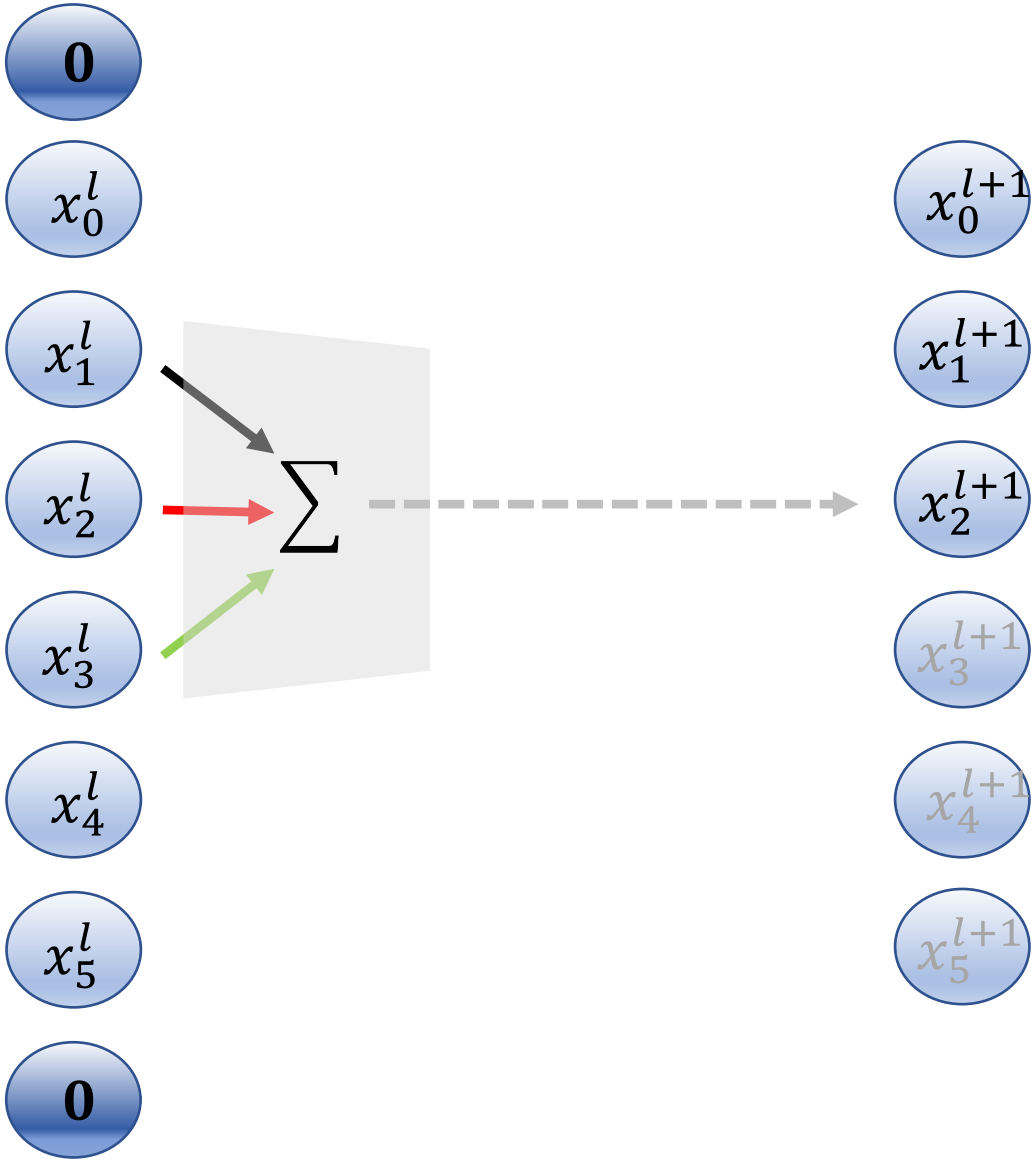
Padding



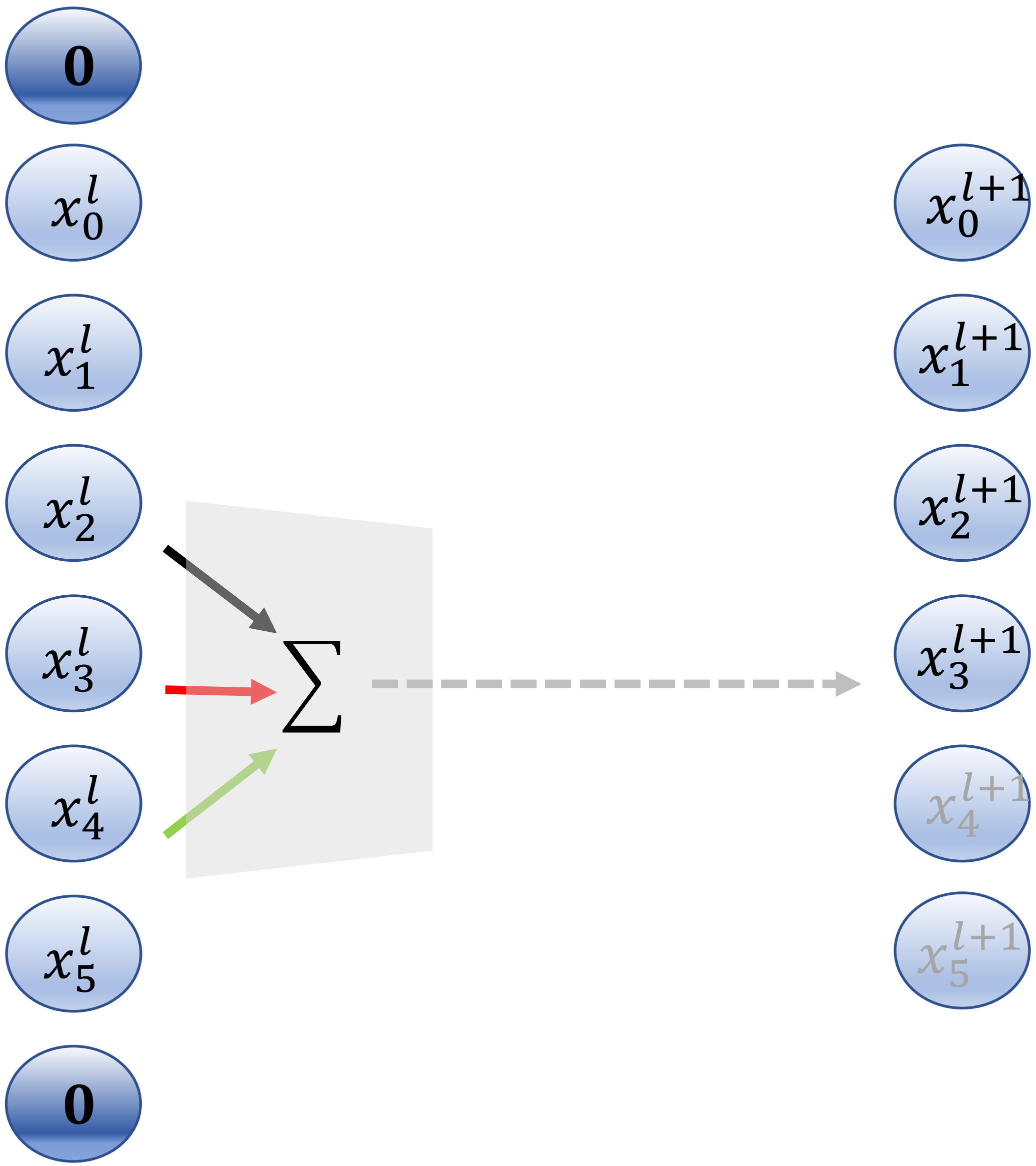
Padding



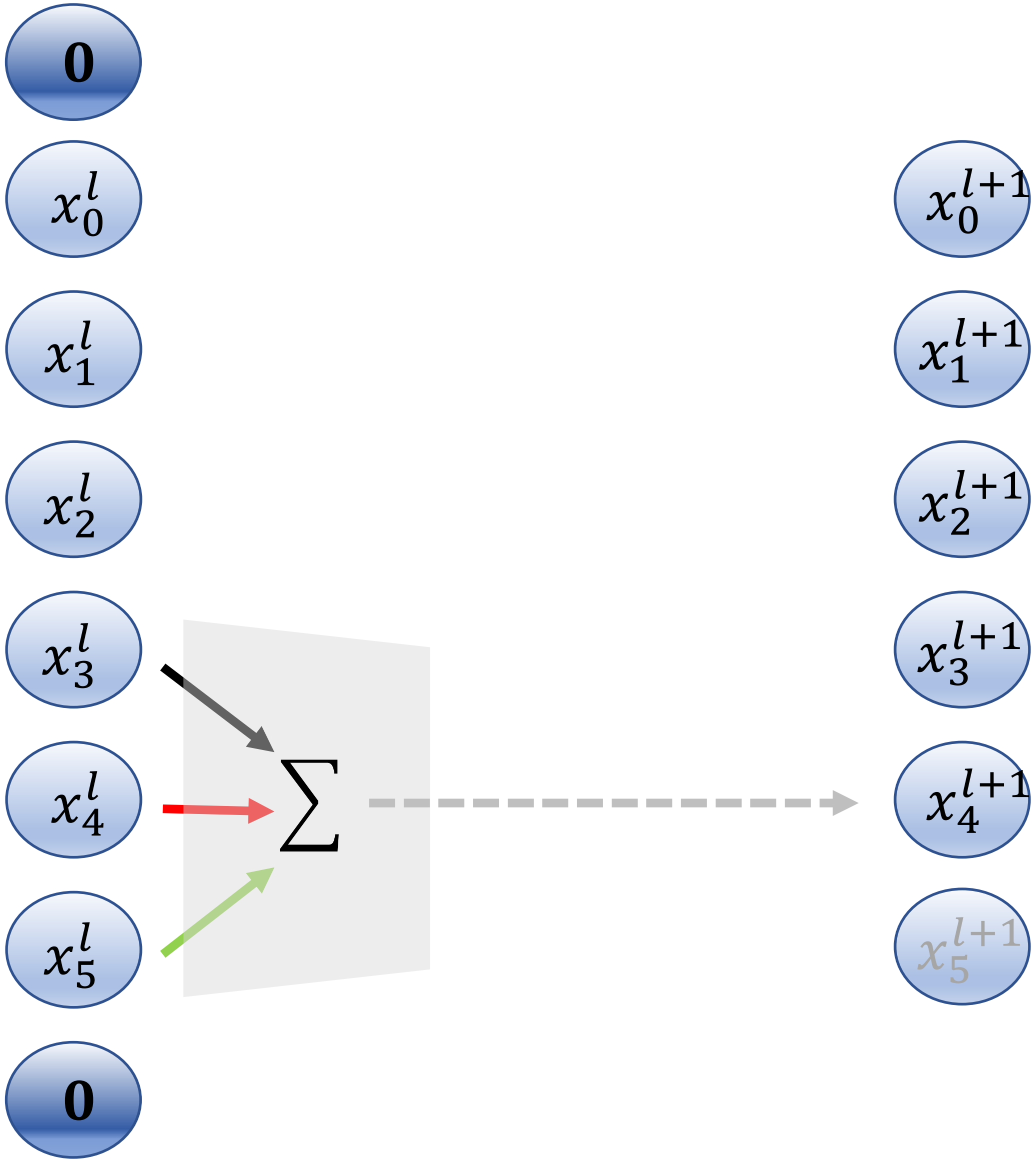
Padding



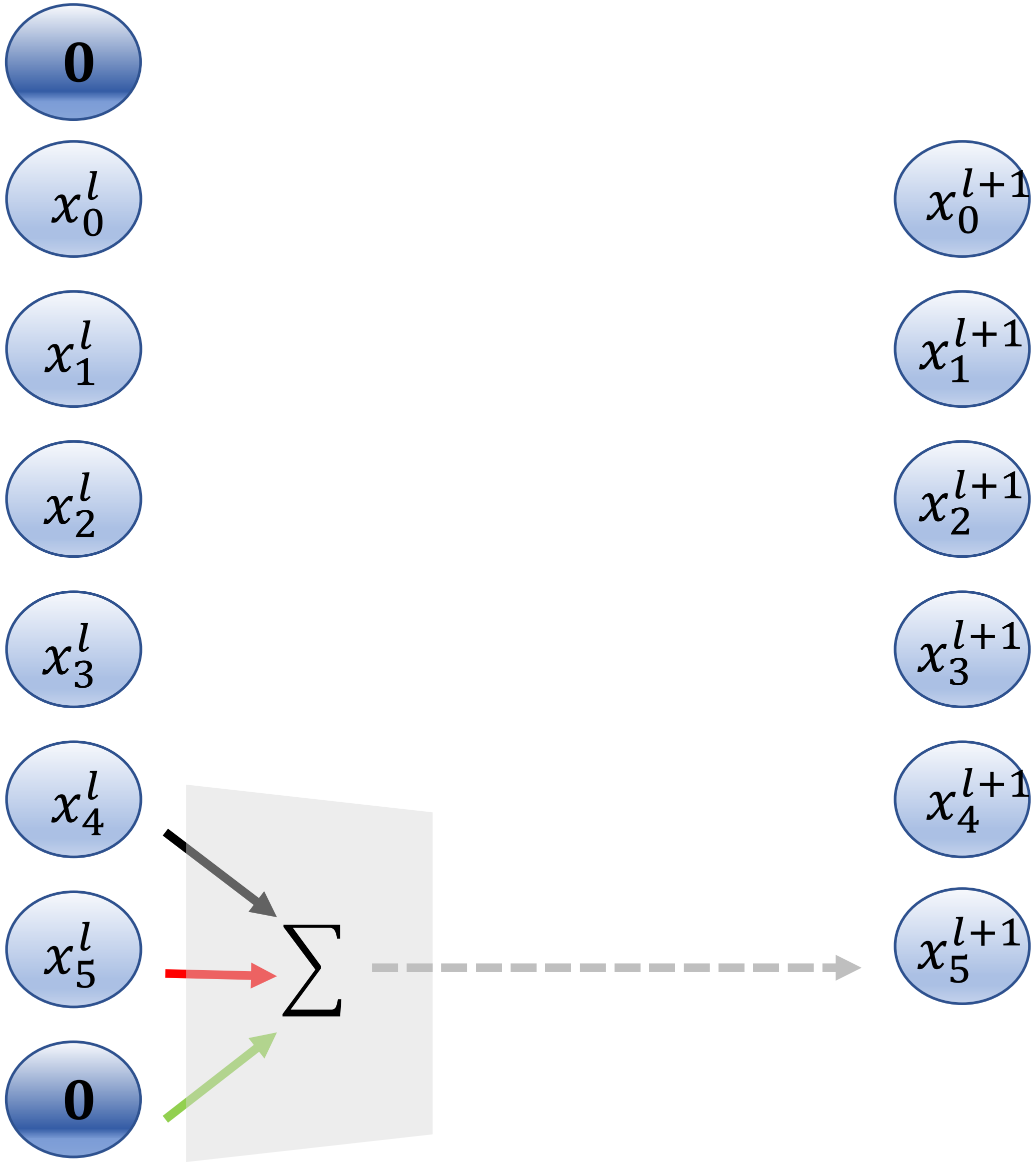
Padding



Padding

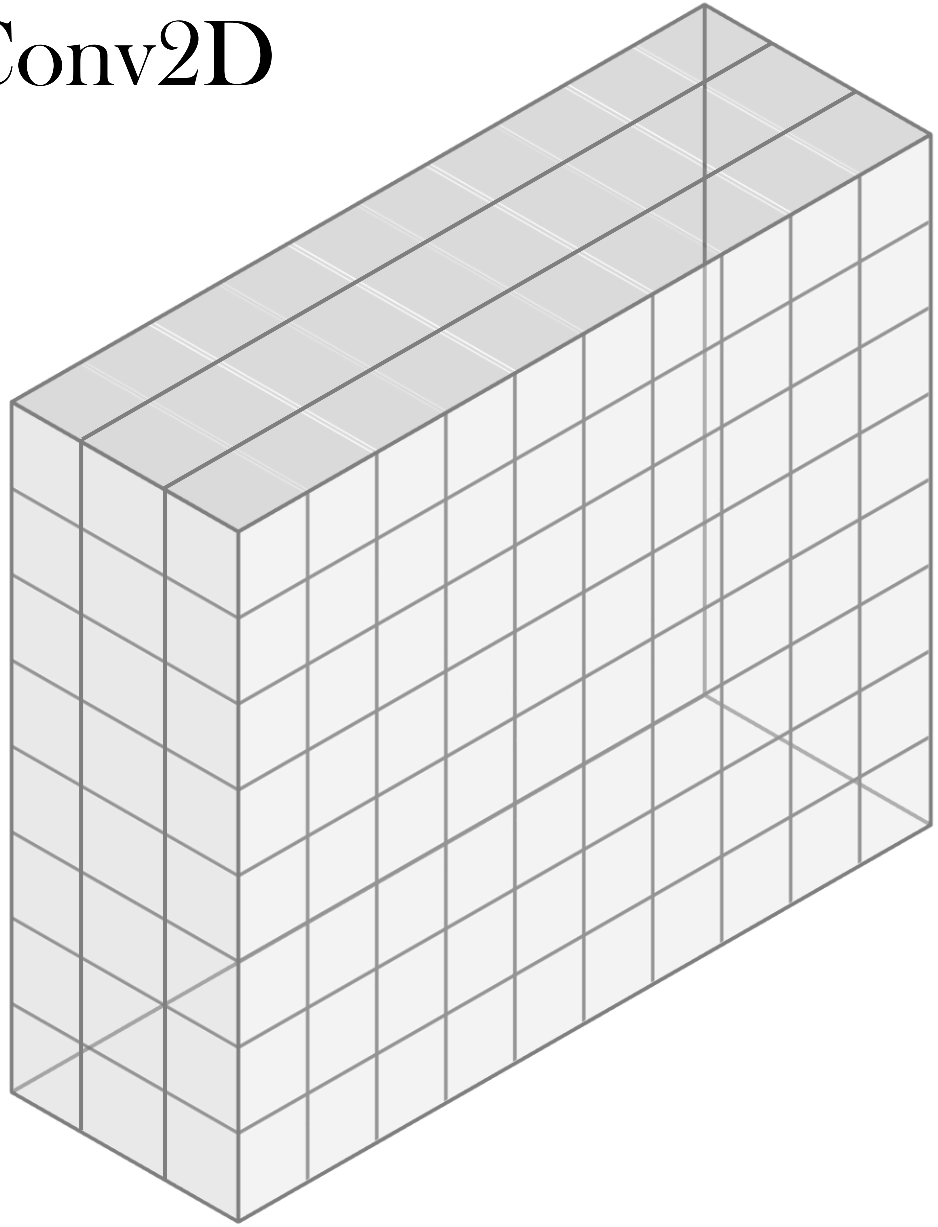


Padding

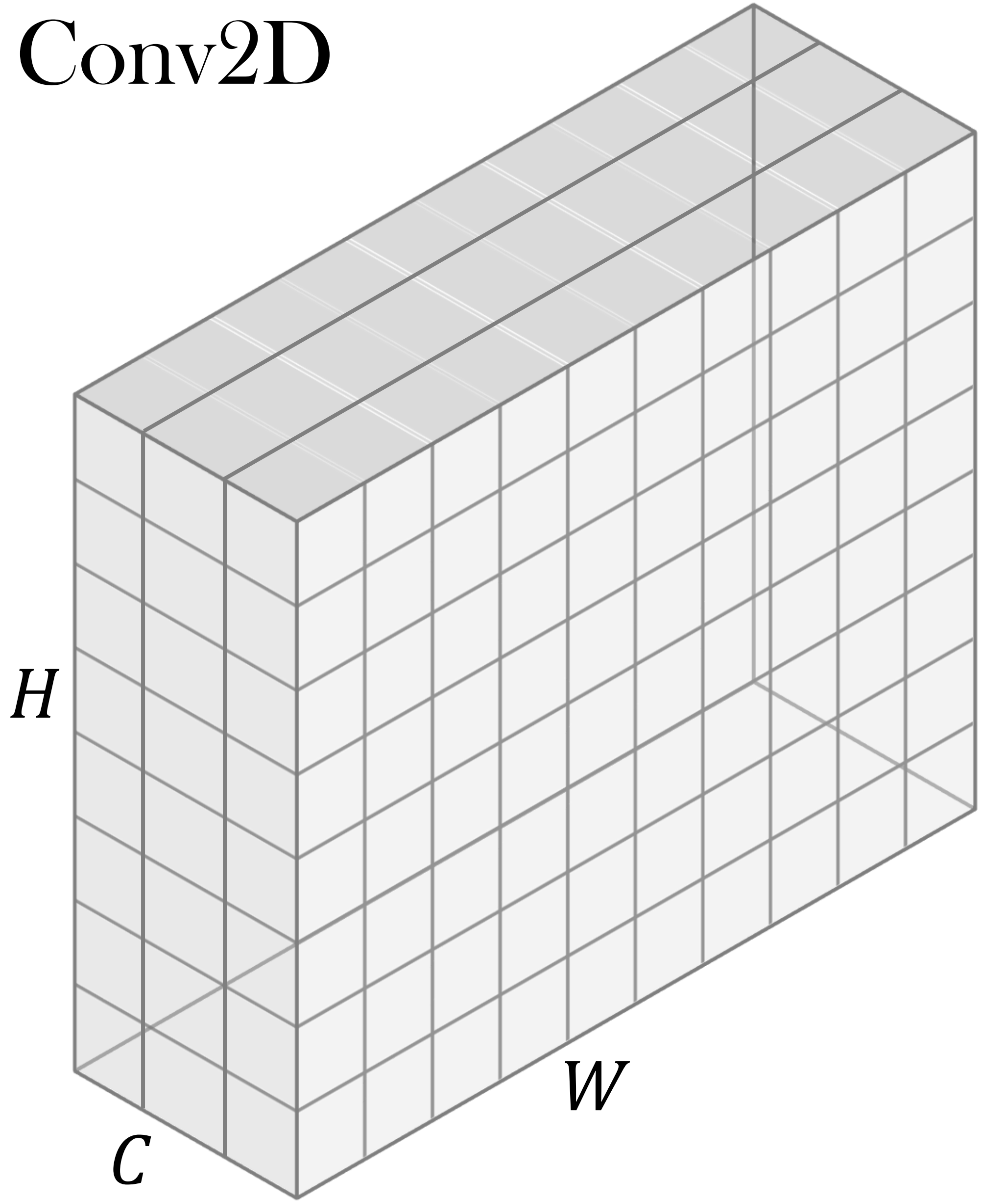


Conv2D

Conv2D

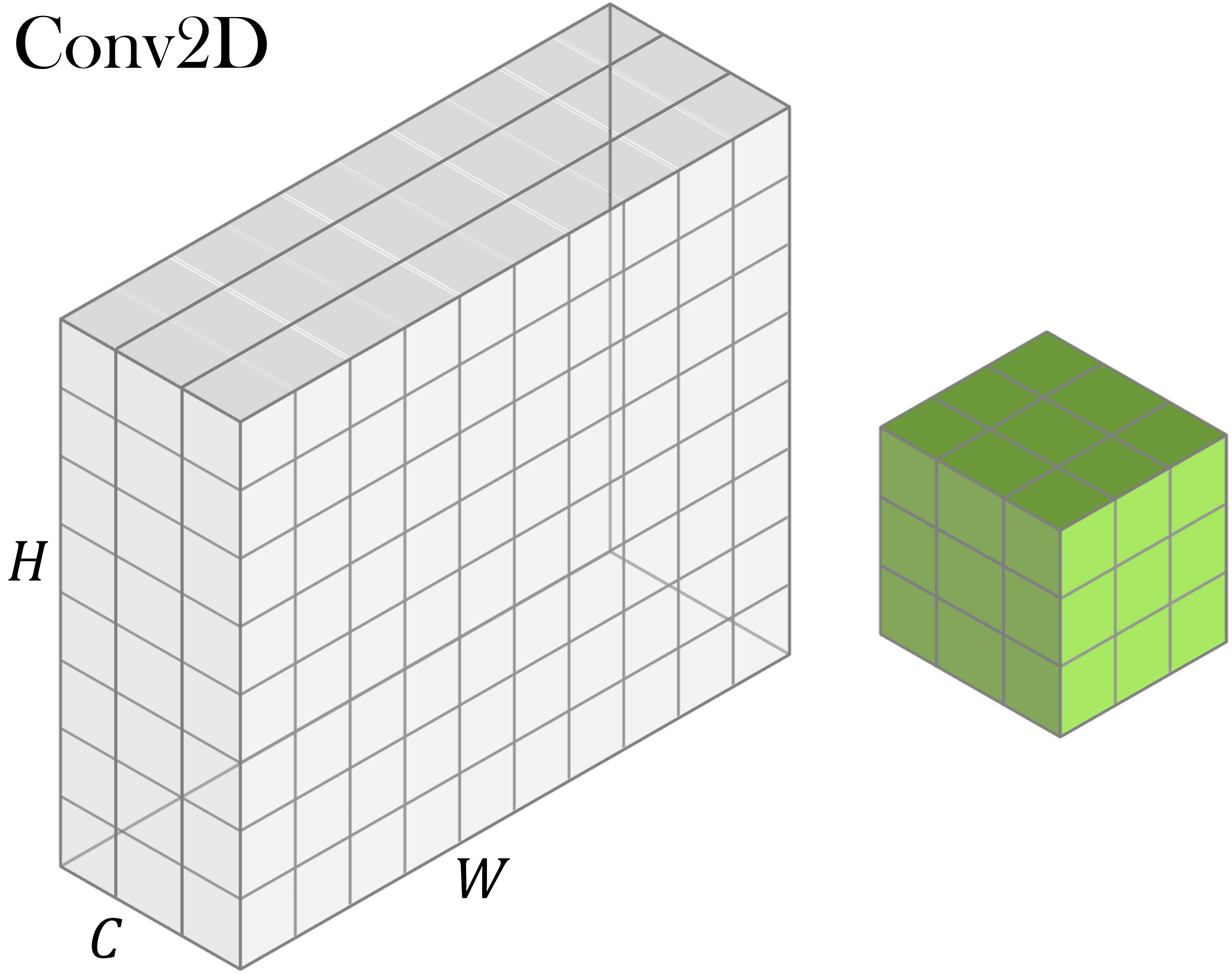


Conv2D



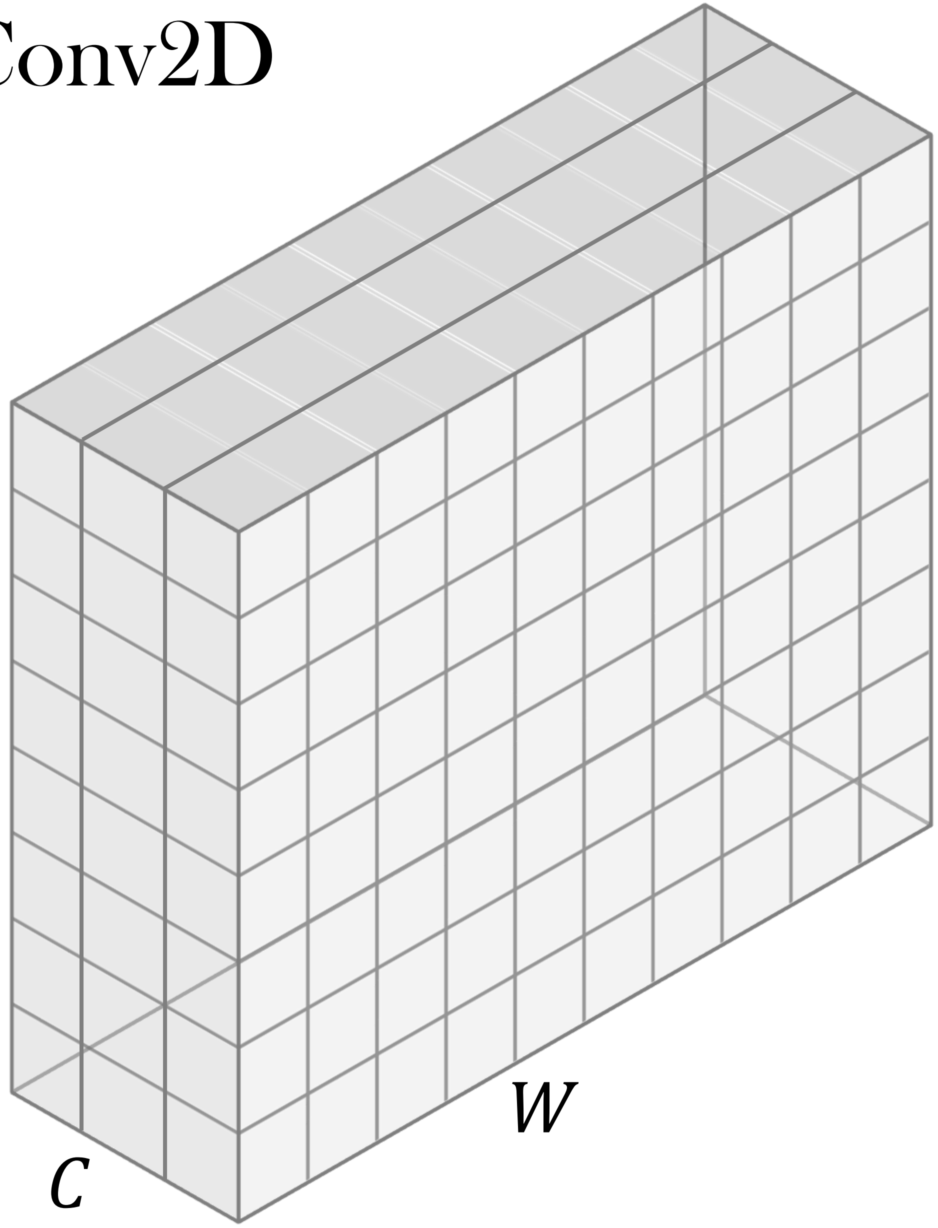
N, C, H, W

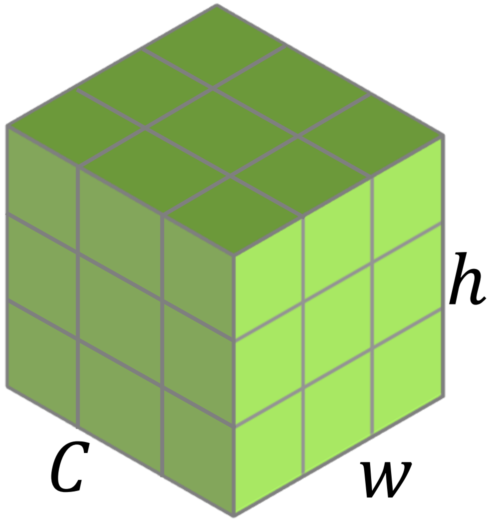
Conv2D



N, C, H, W

Conv2D

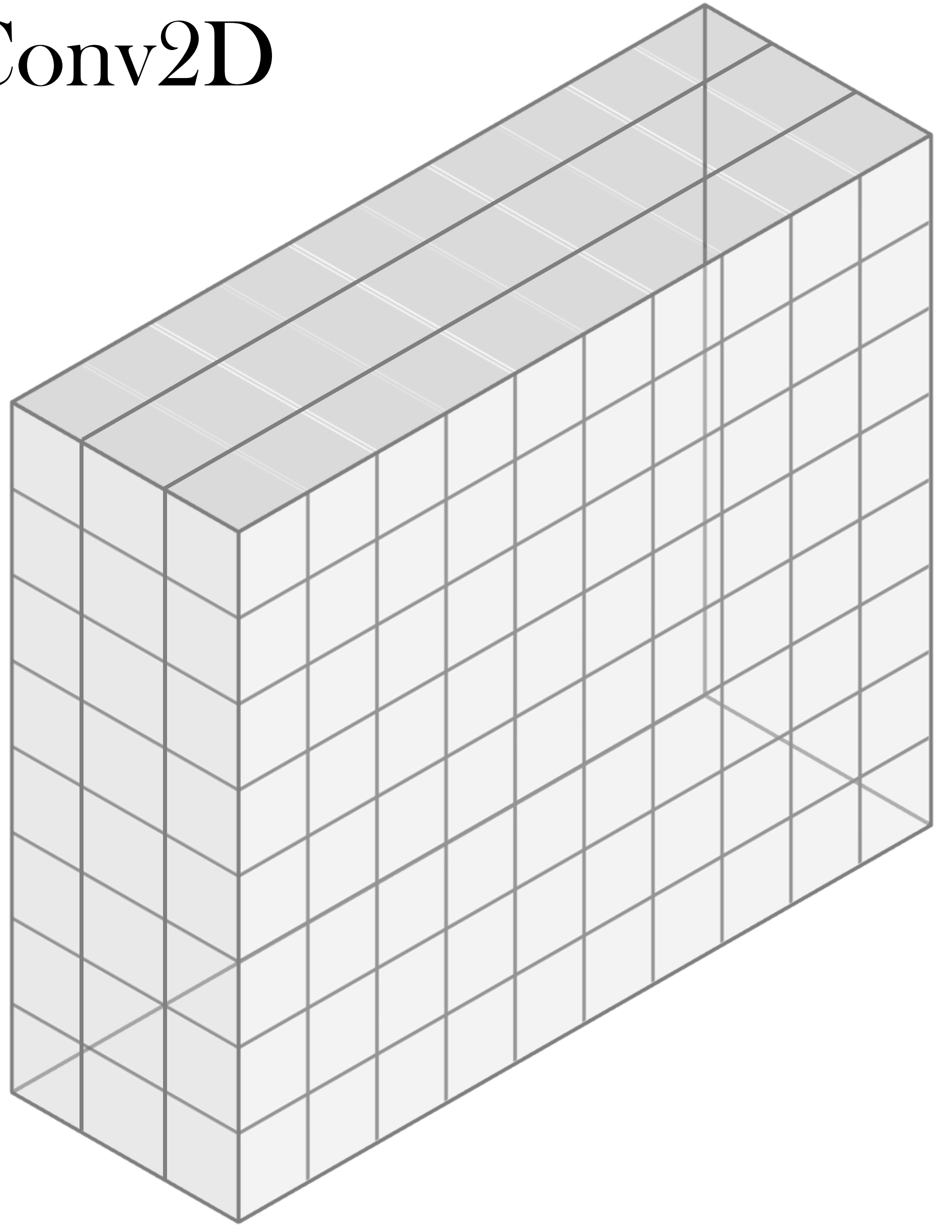




N, C, H, W

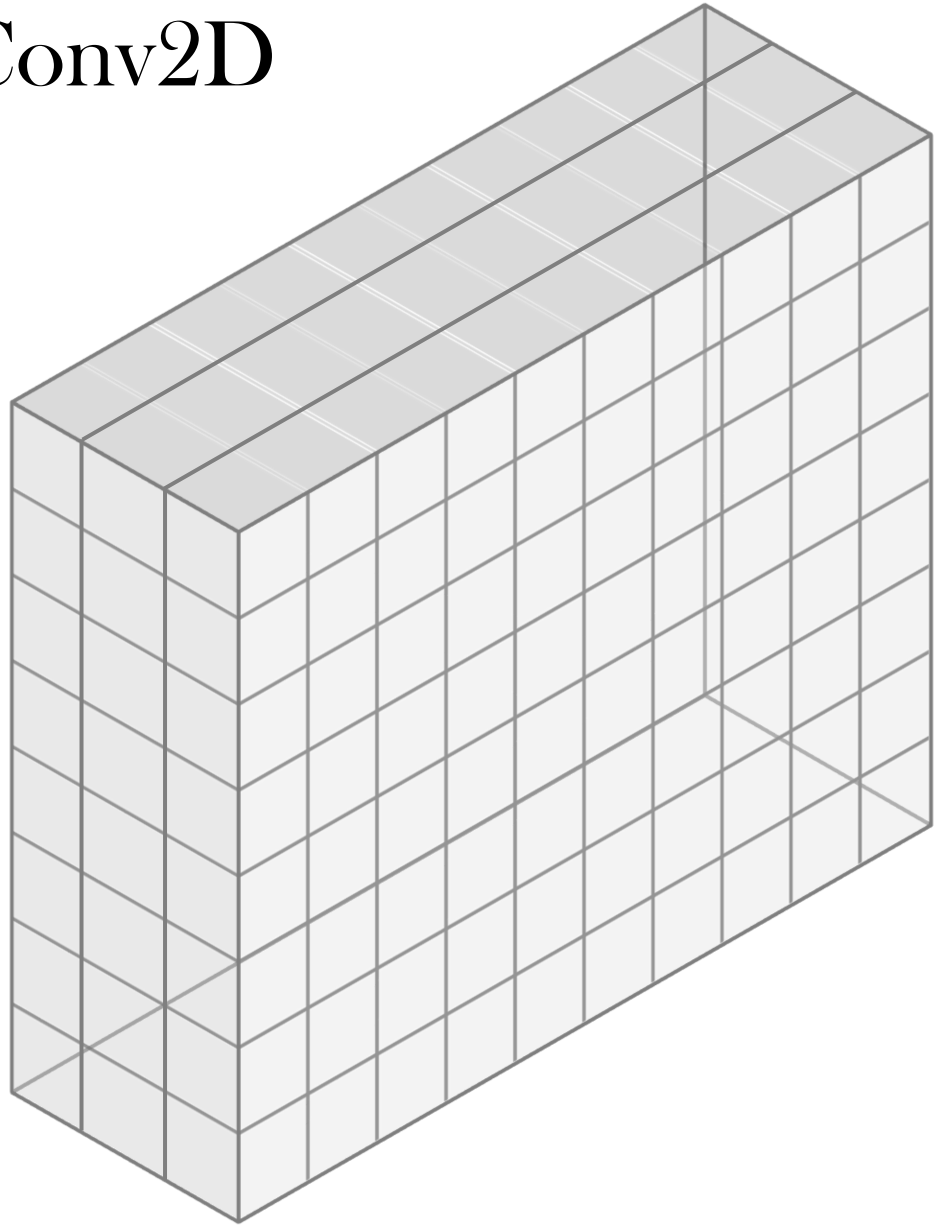
C, h, w

Conv2D



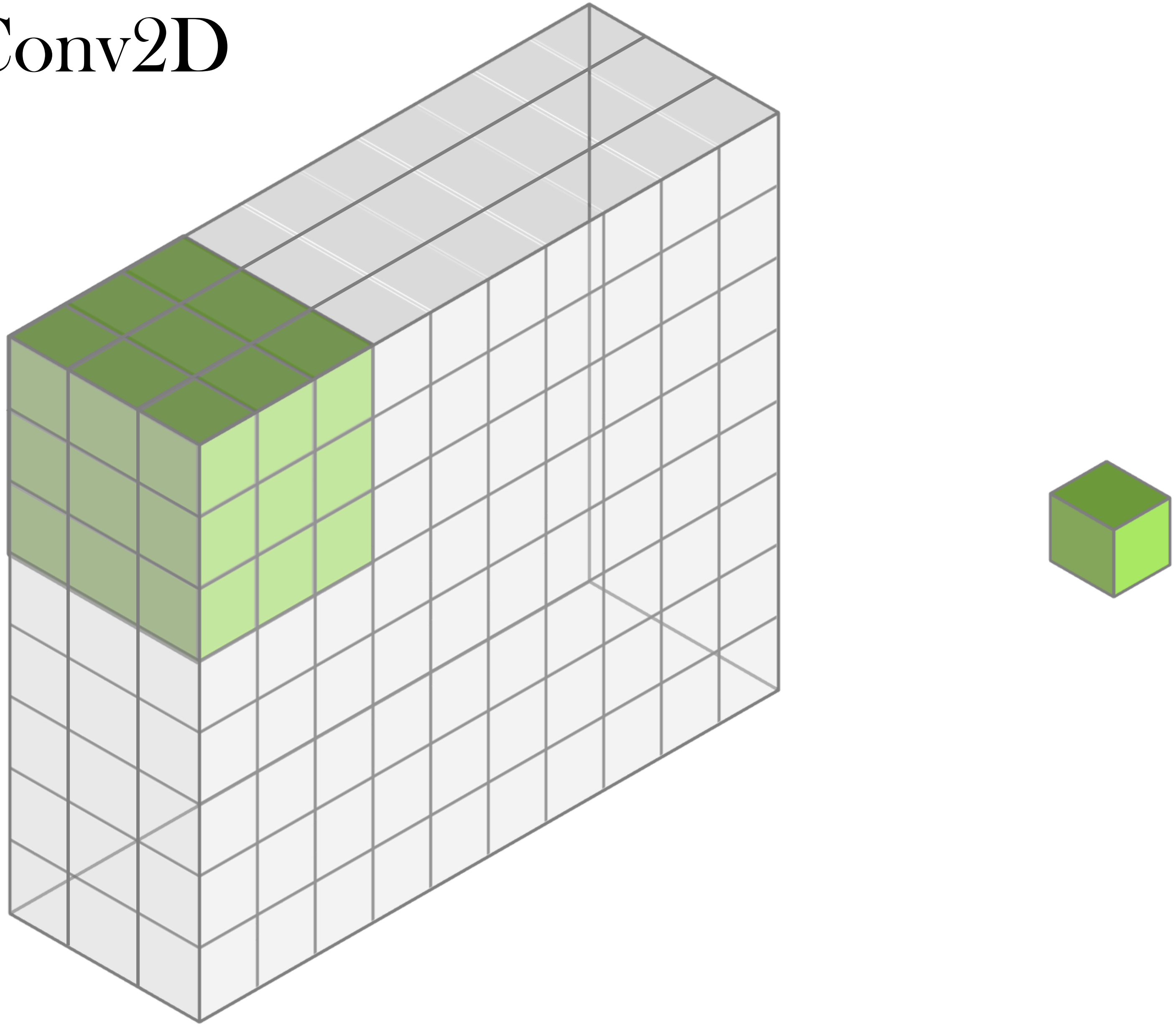
N, C, H, W

Conv2D



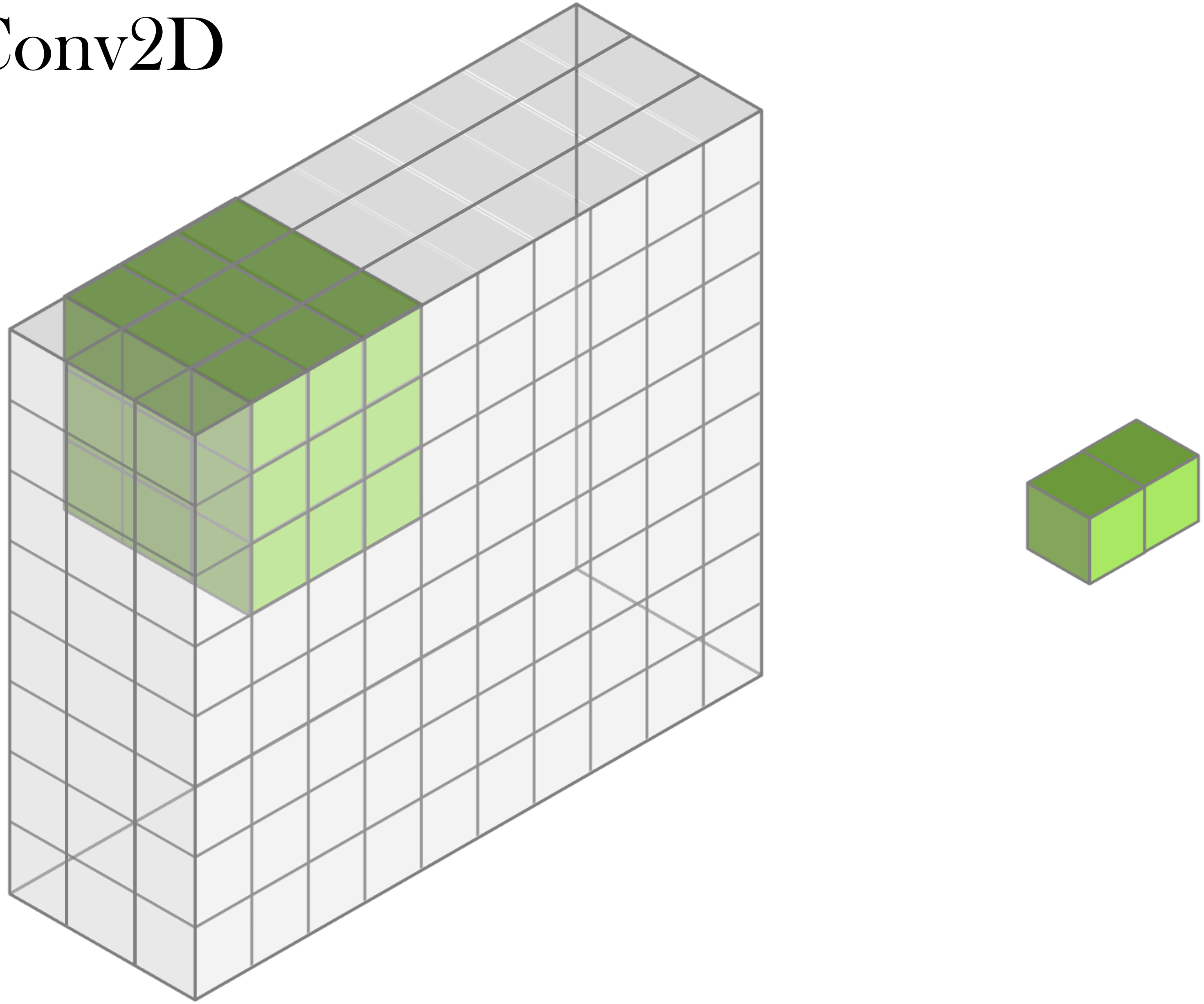
N, C, H, W

Conv2D



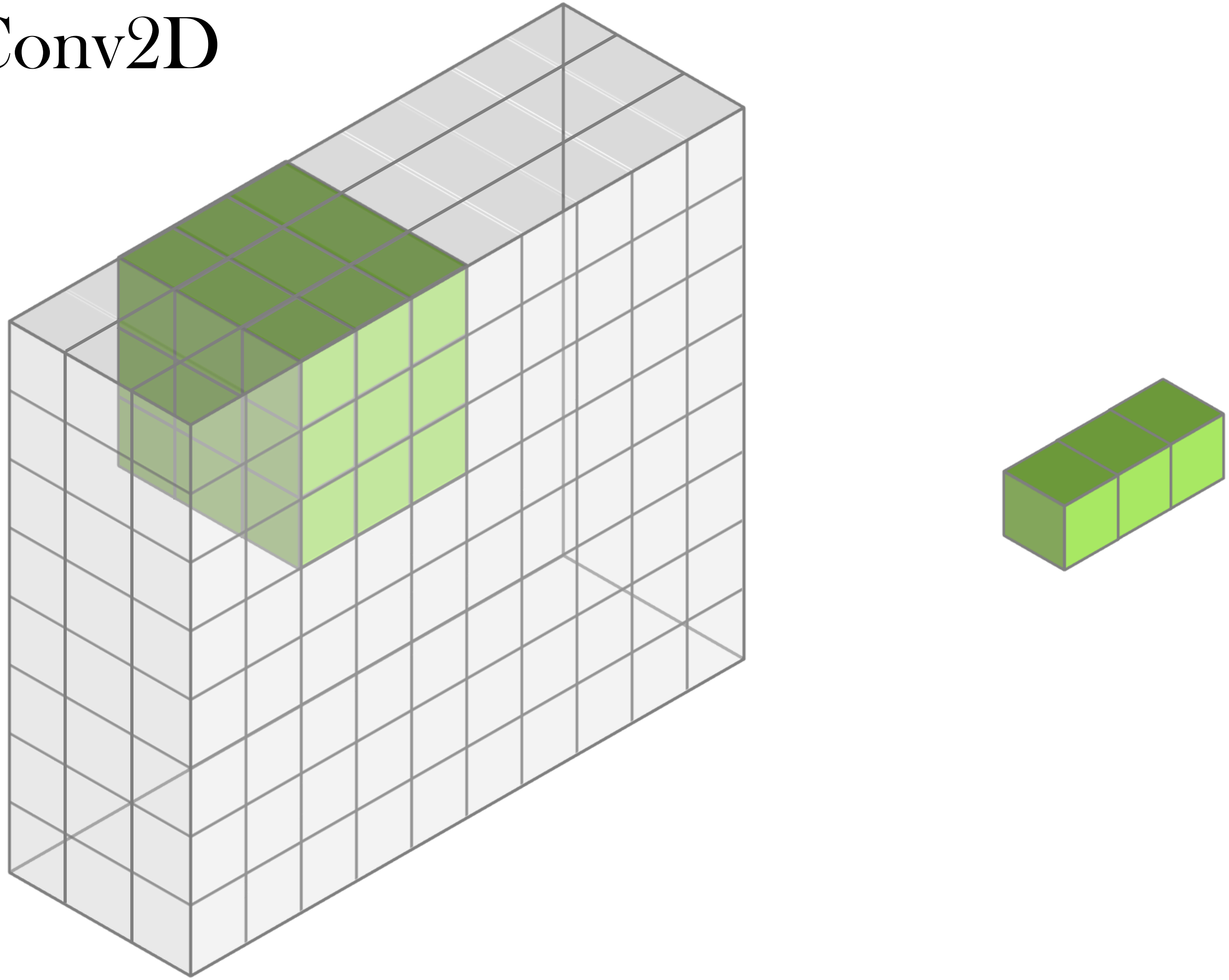
N, C, H, W

Conv2D



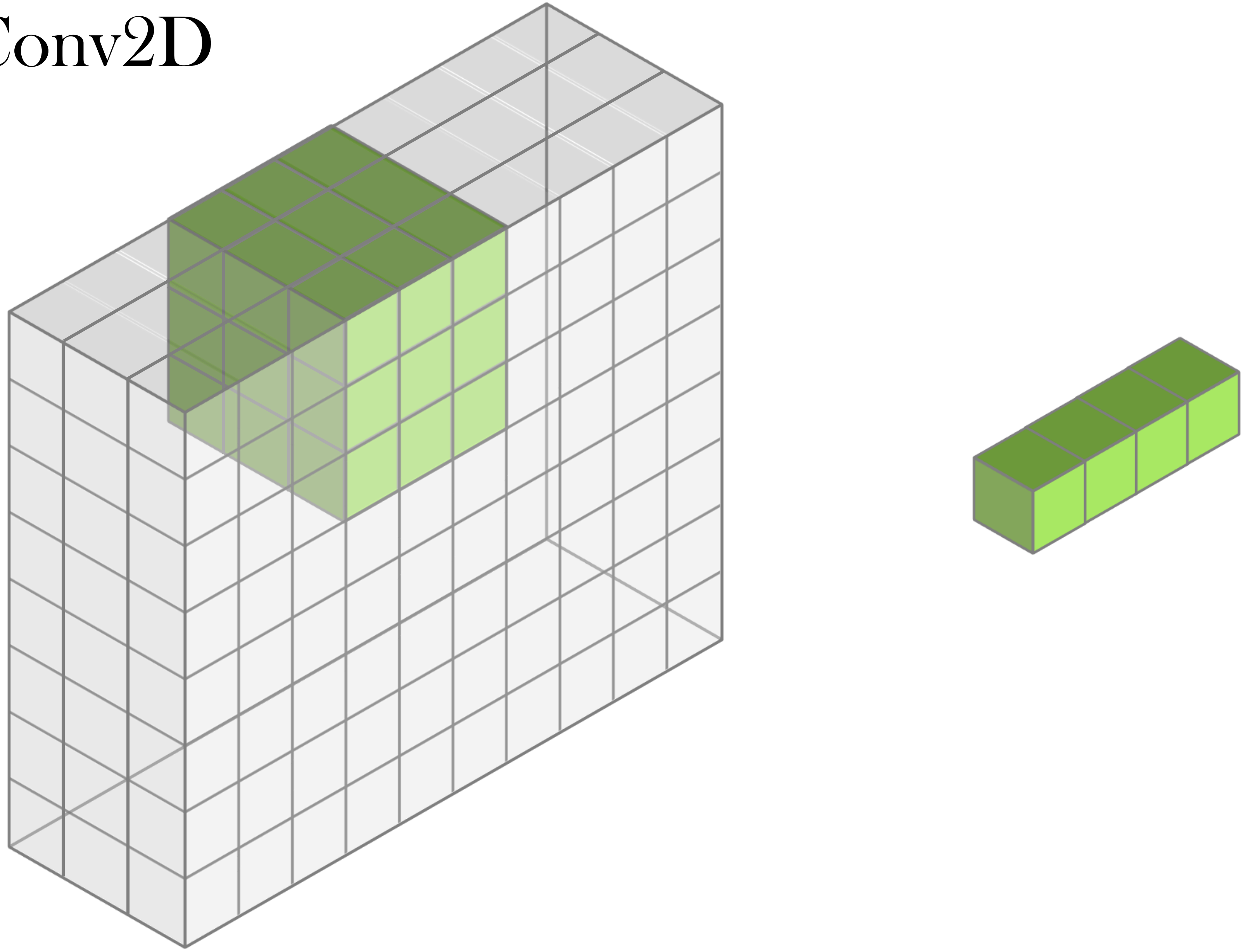
$$N, C, H, W$$

Conv2D



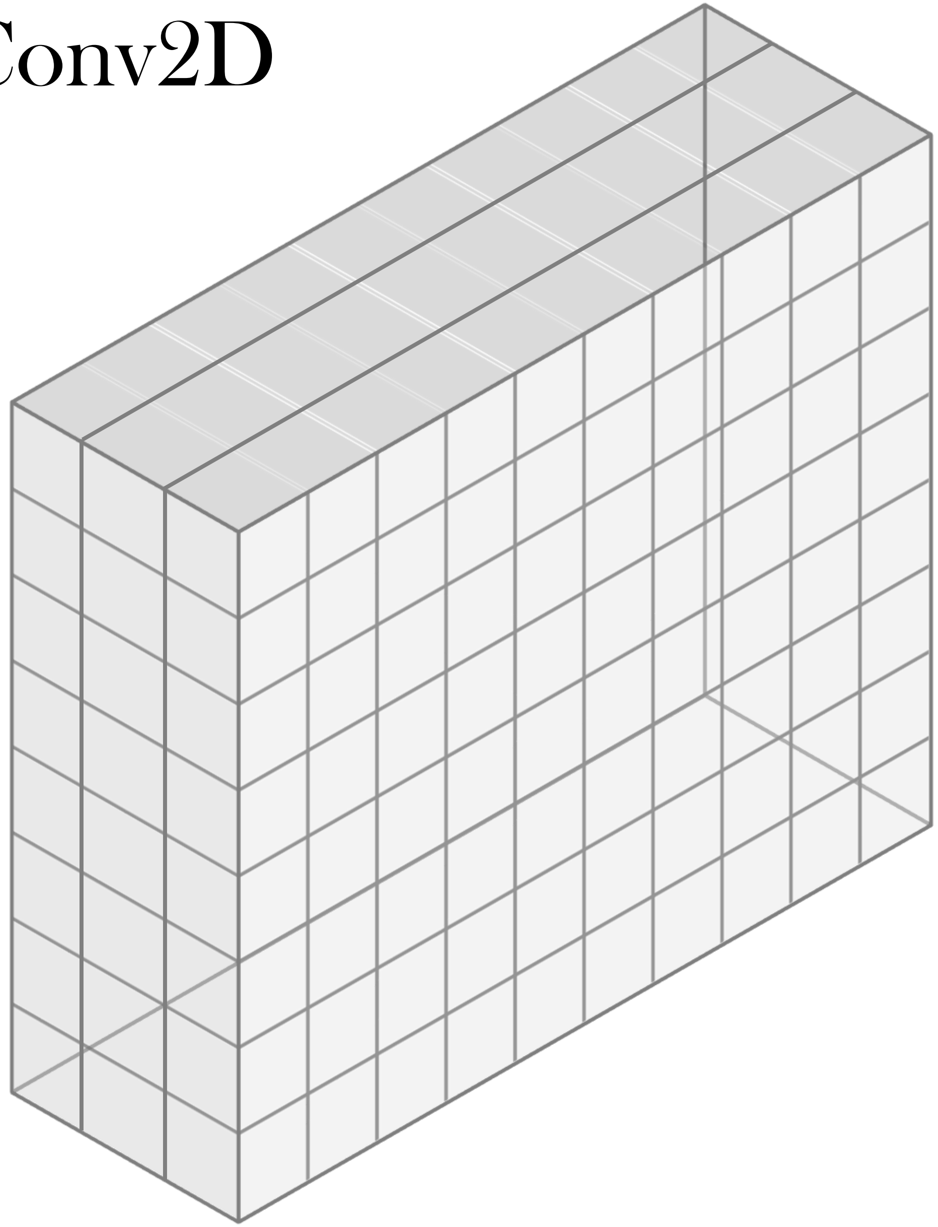
$$N, C, H, W$$

Conv2D

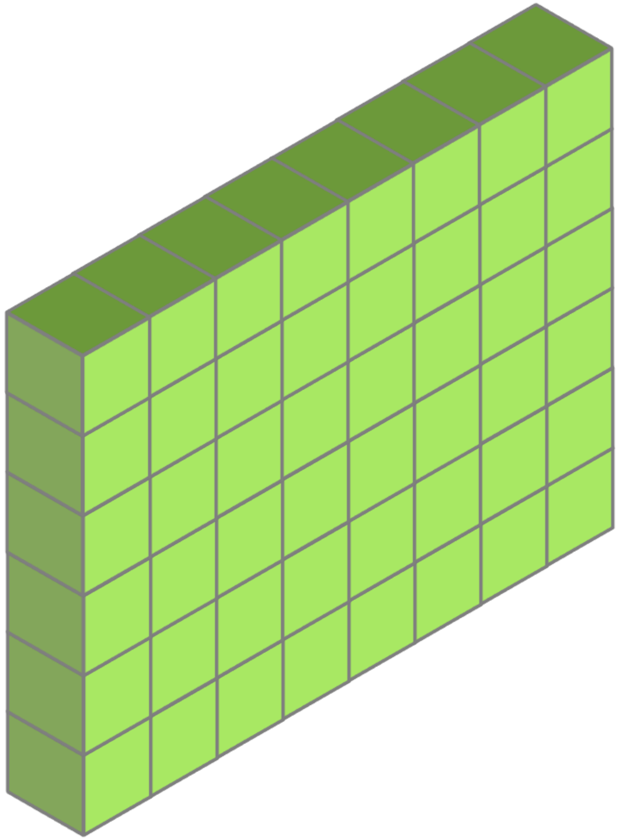


N, C, H, W

Conv2D

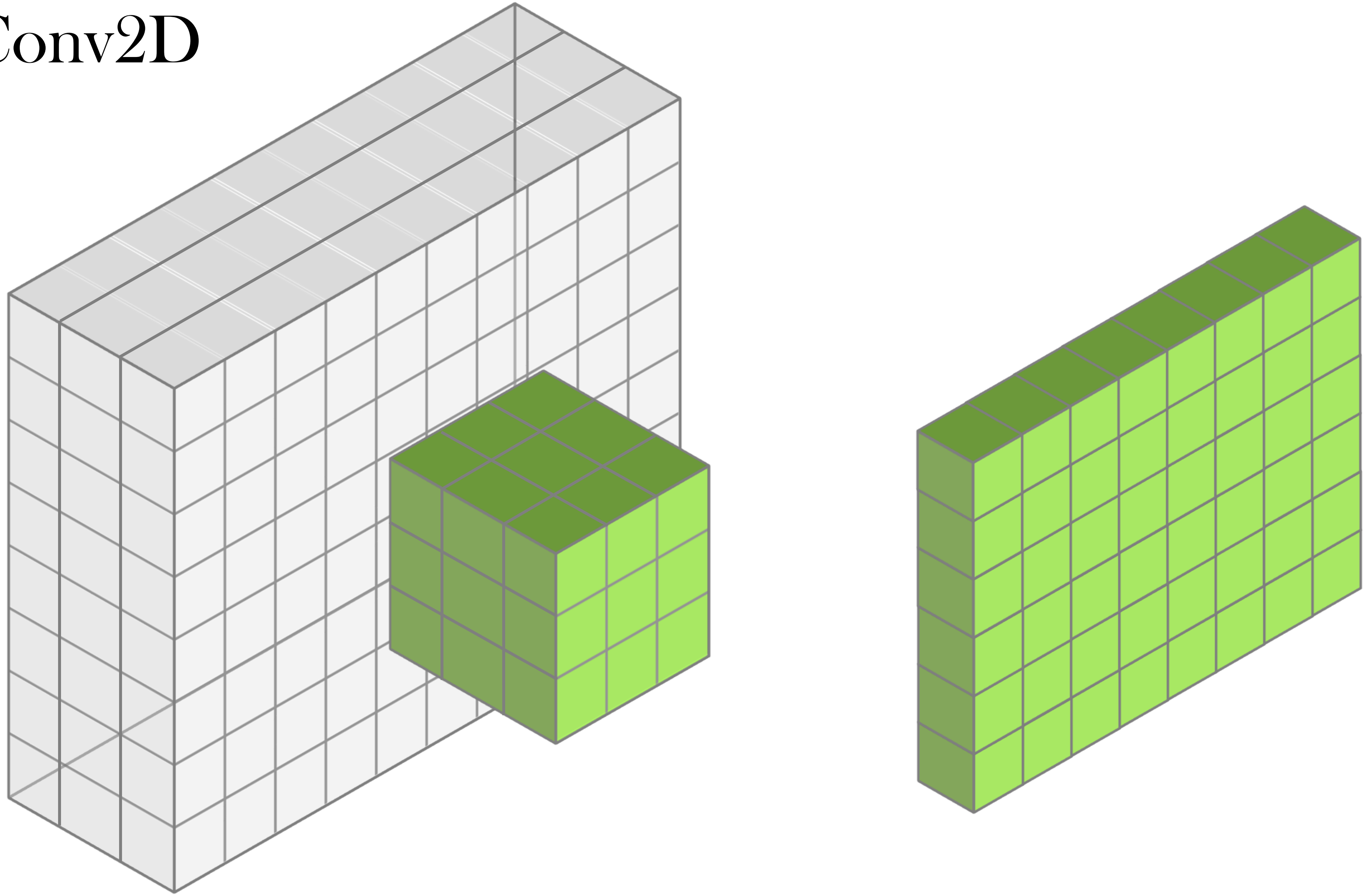


N, C, H, W

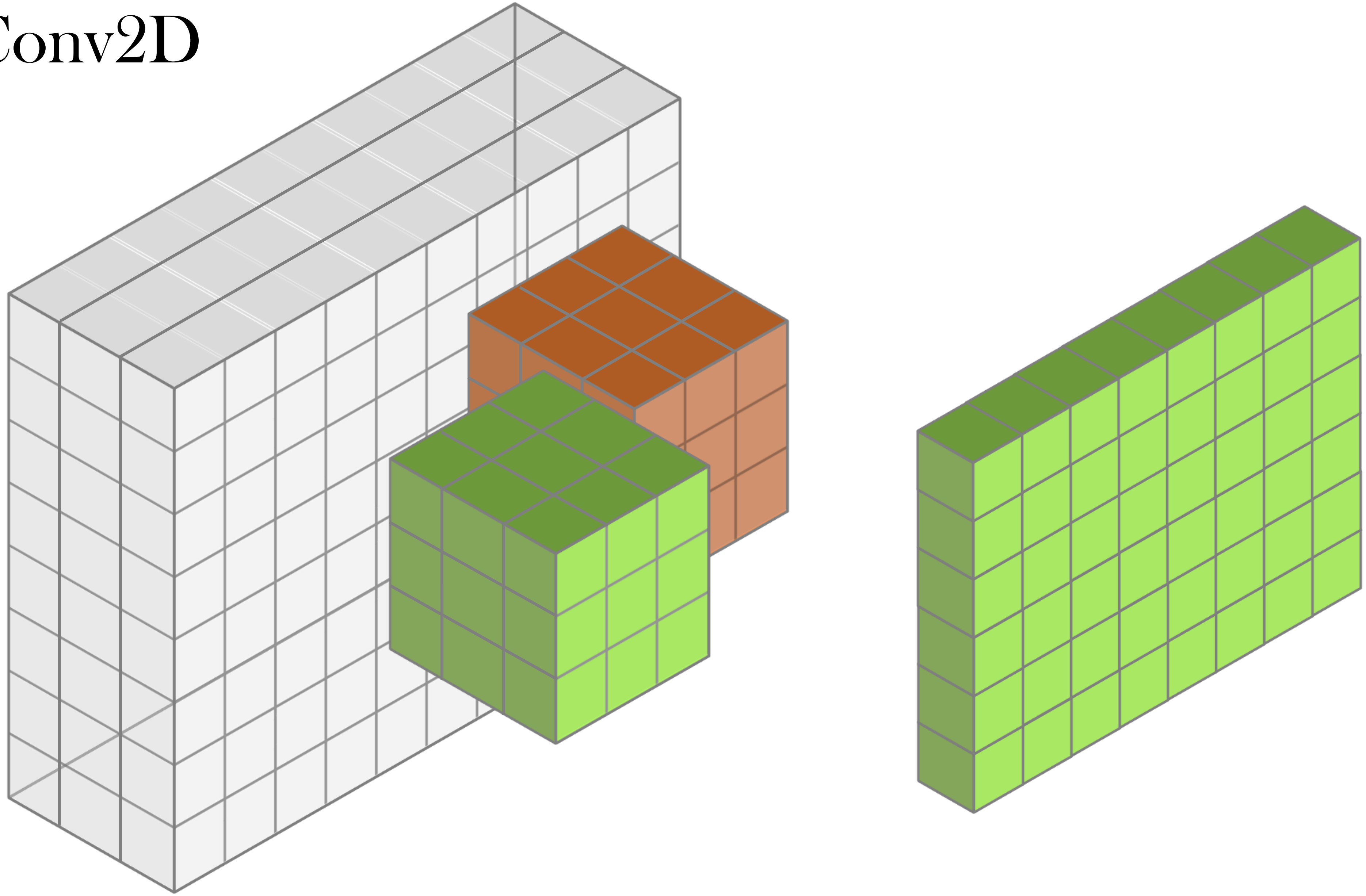


$[N, 1, \widetilde{H}, \widetilde{W}]$

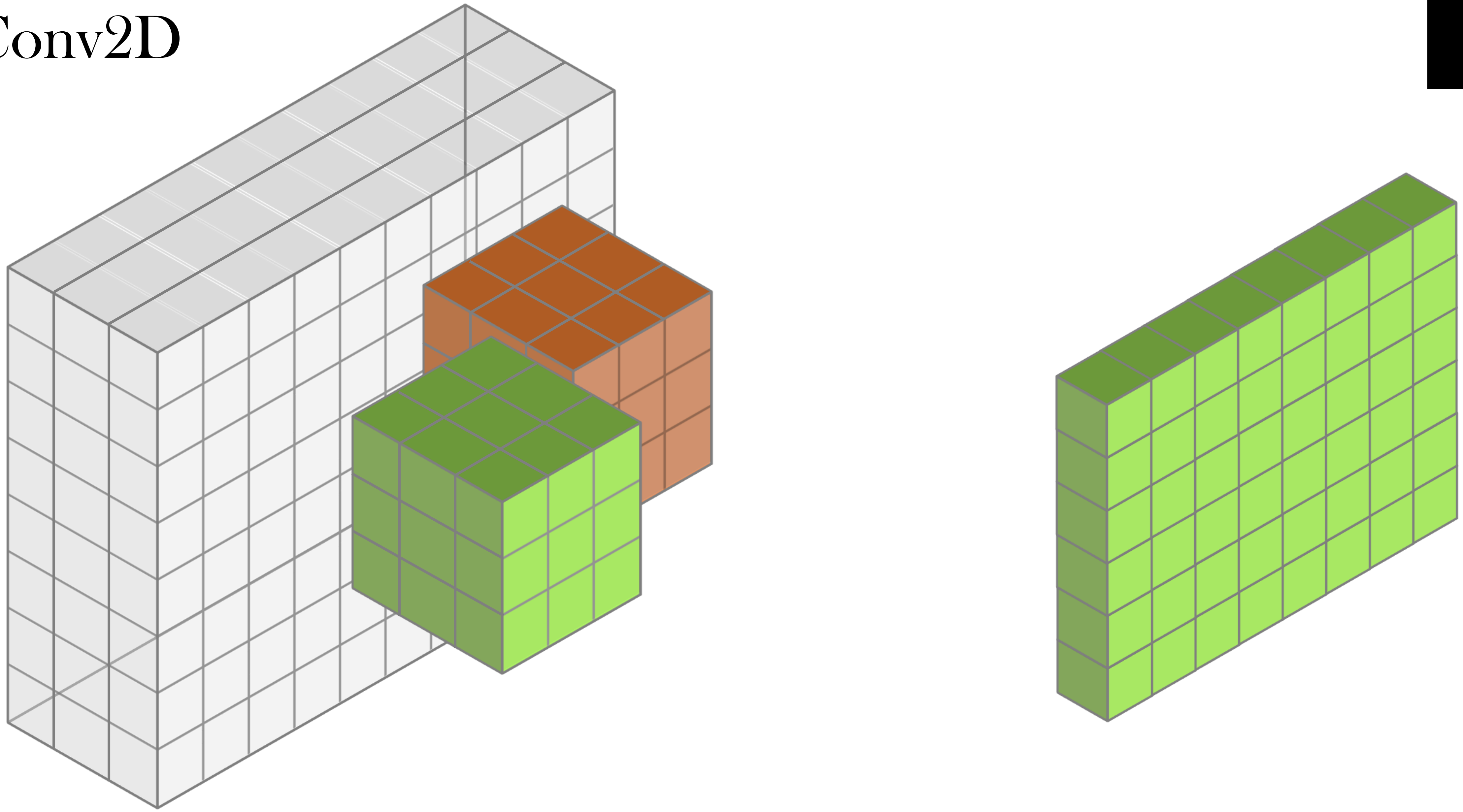
Conv2D



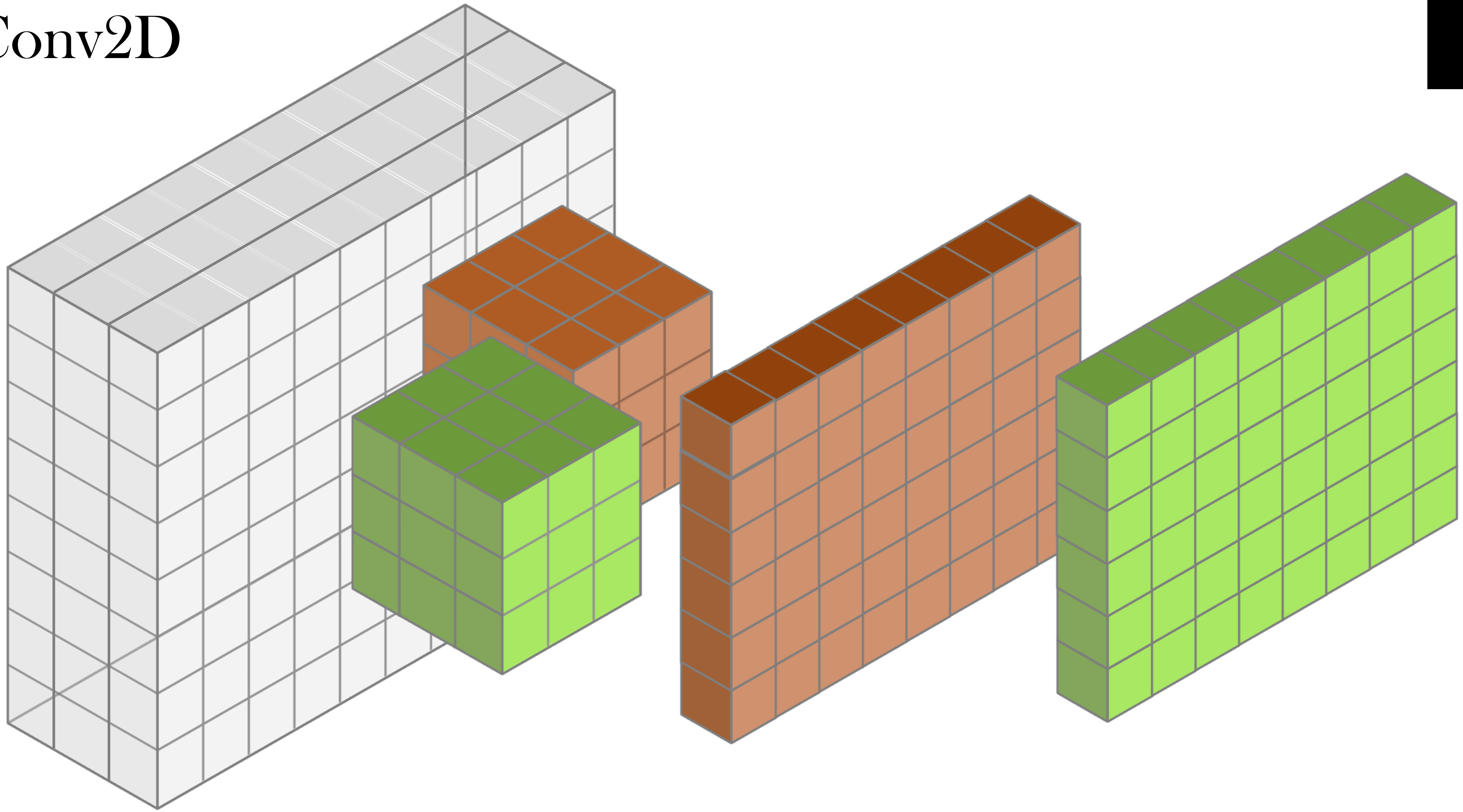
Conv2D



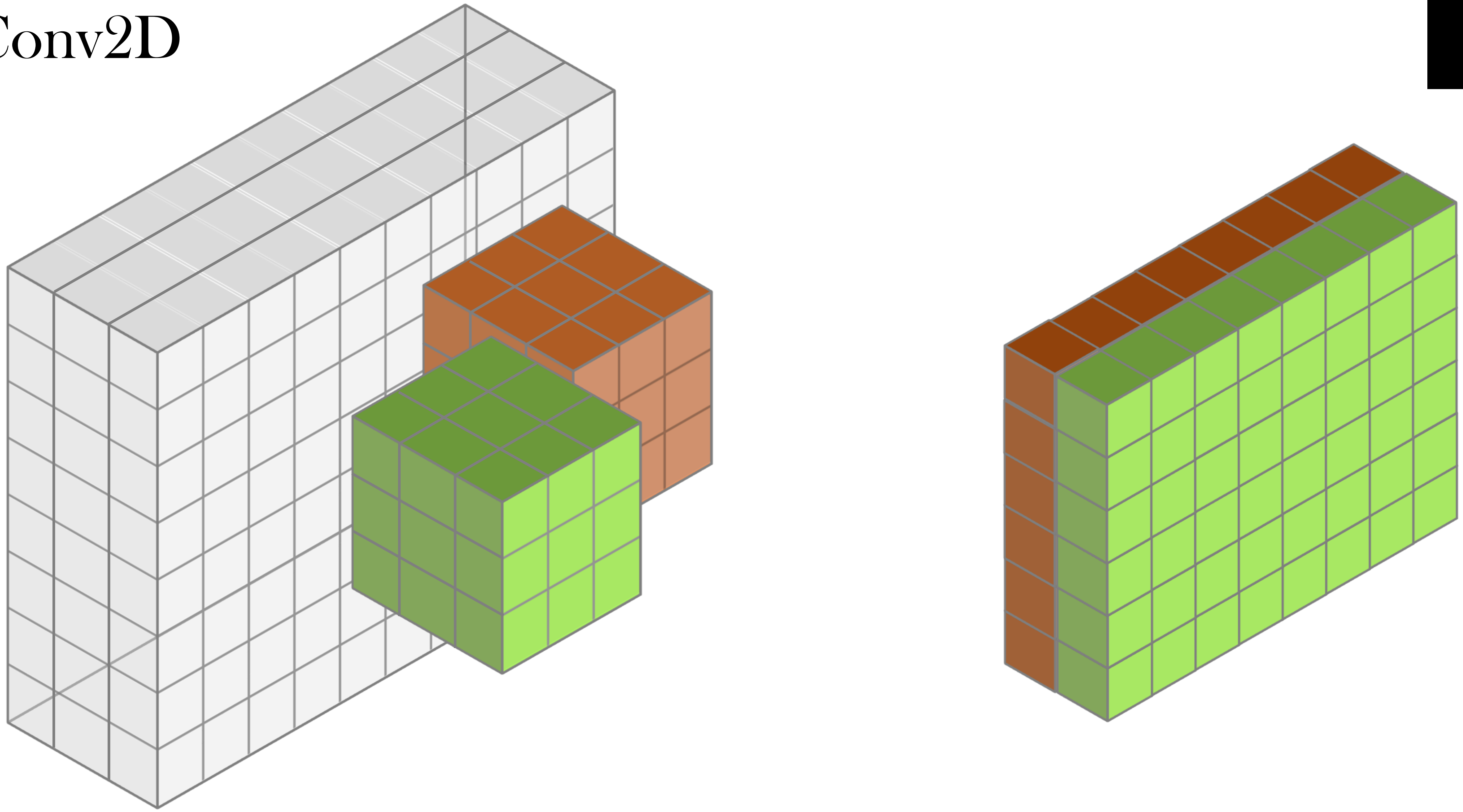
Conv2D



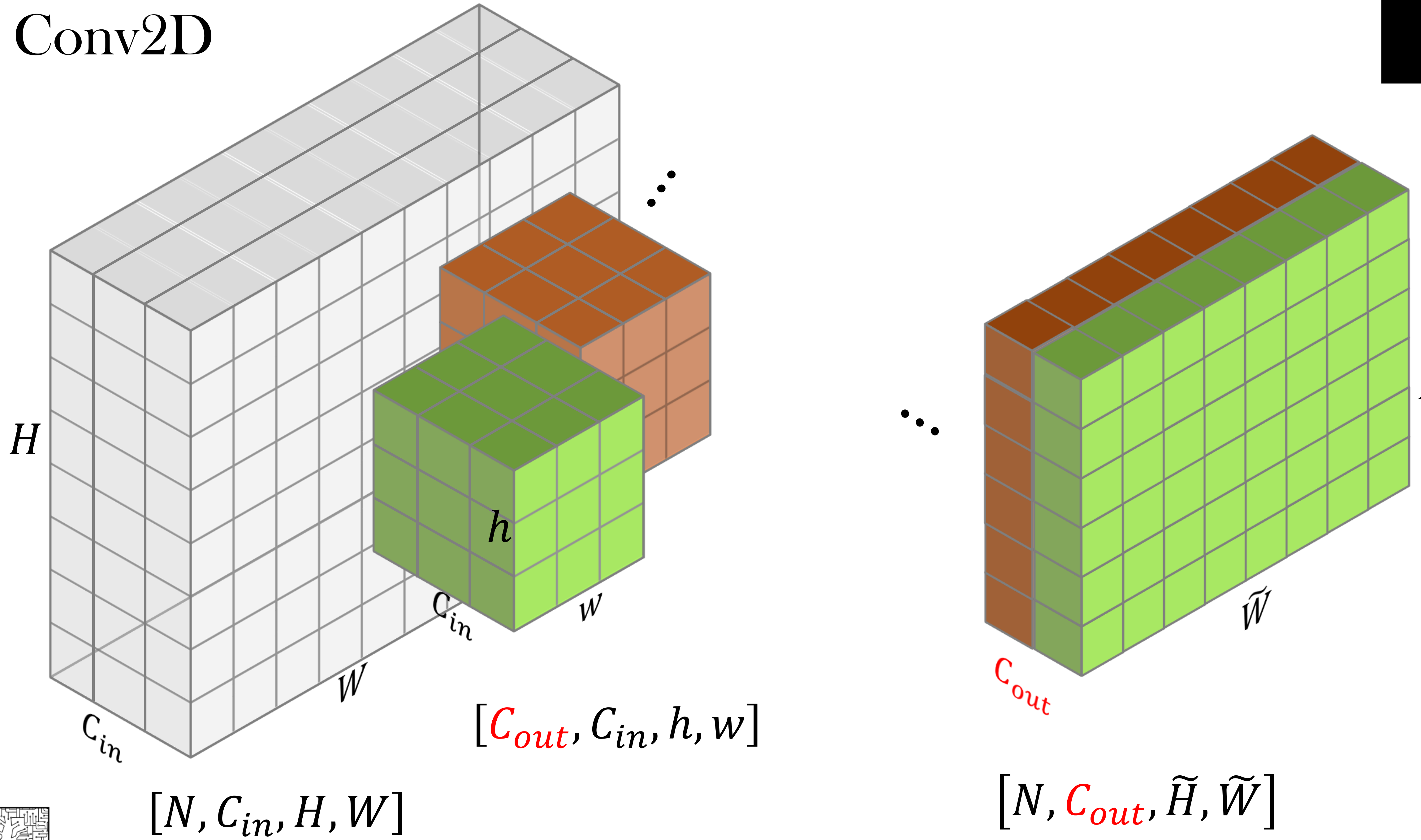
Conv2D



Conv2D



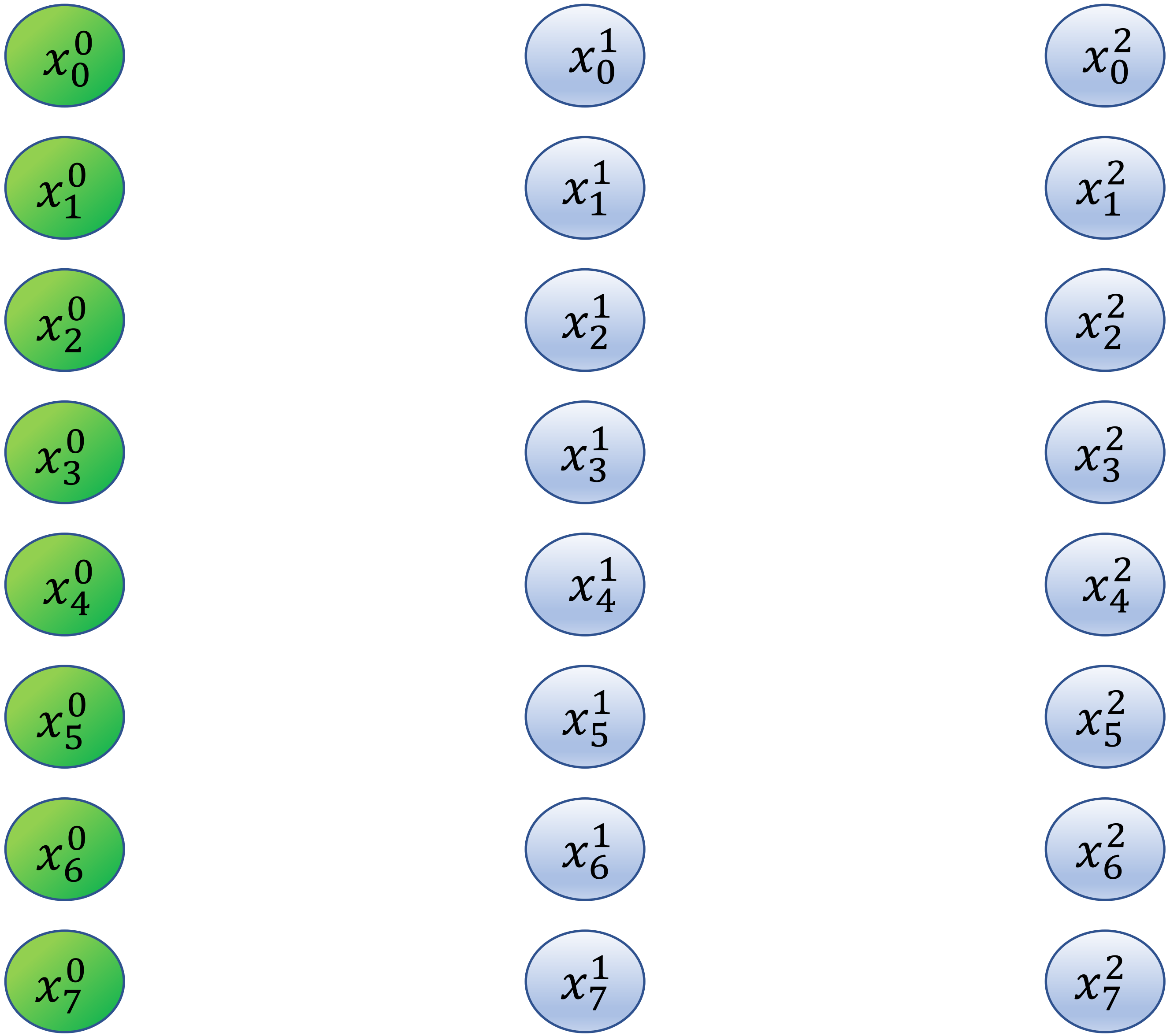
Conv2D



H
C_{in}
W
h
C_{in}
w
$\tilde{H}$
$\tilde{W}$
C_{out}
$[C_{out}, C_{in}, h, w]$
$[N, C_{in}, H, W]$
$[N, C_{out}, \tilde{H}, \tilde{W}]$

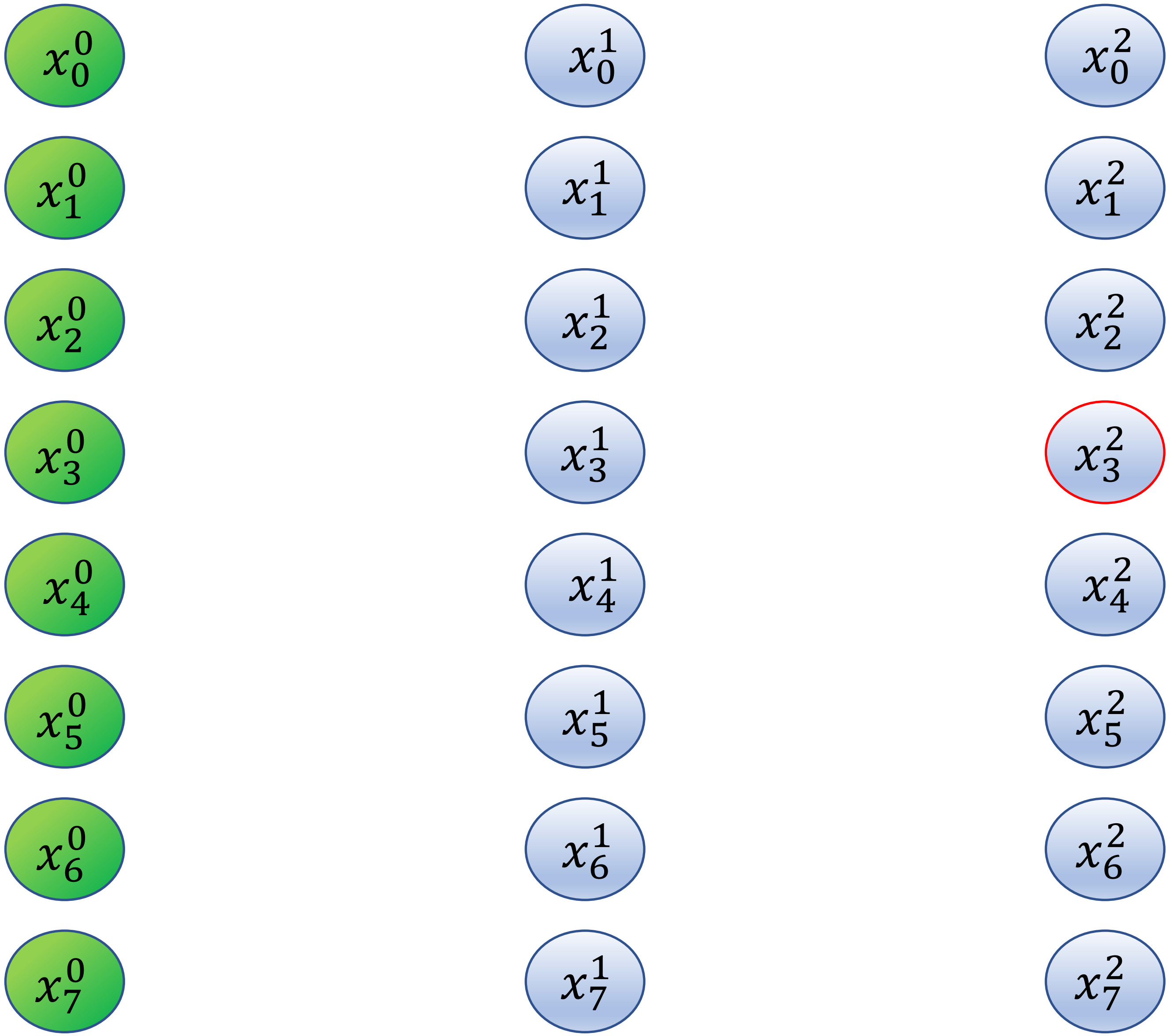
Receptive Field

$[Filter\ size: 3]$



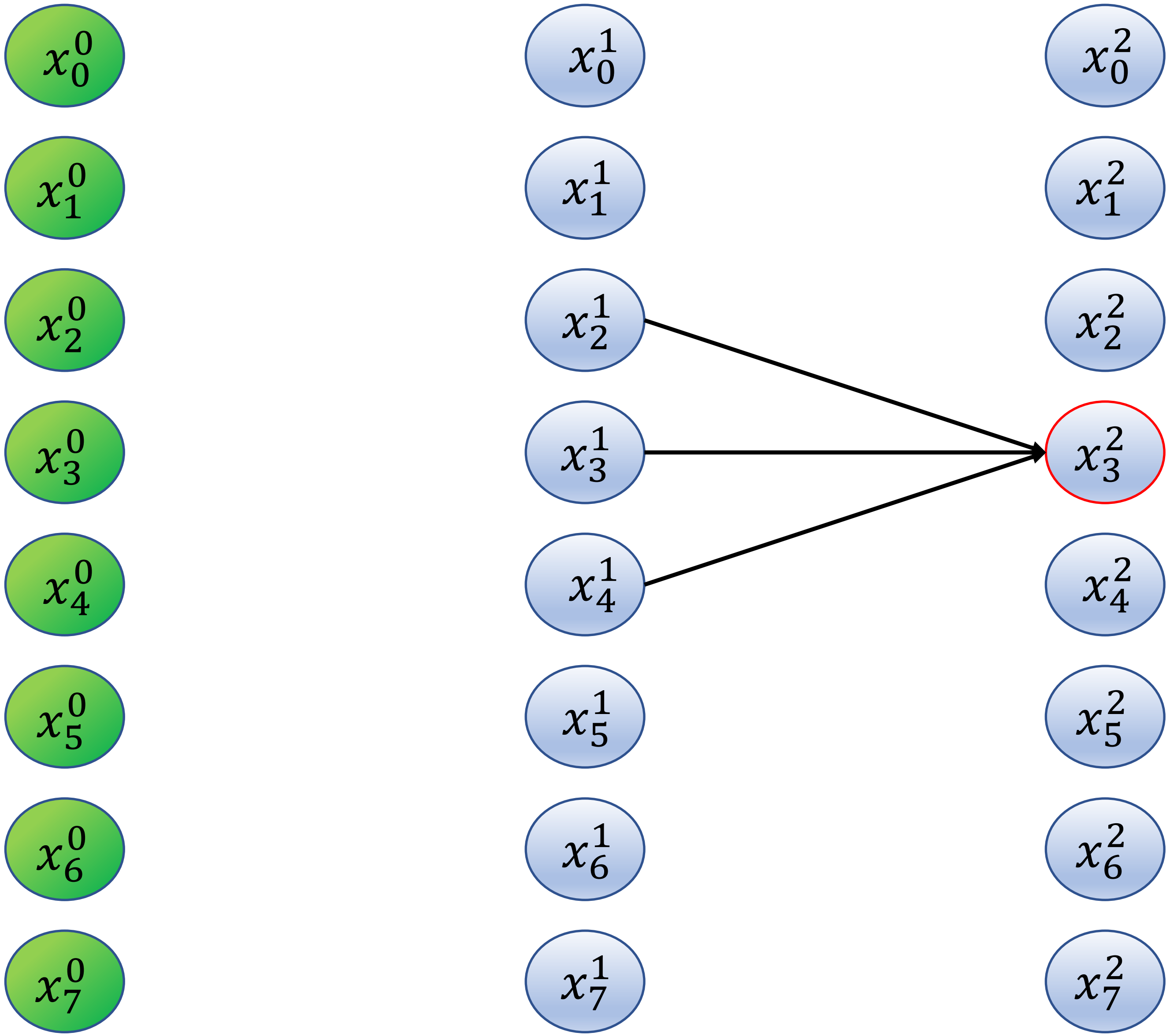
Receptive Field

$[Filter\ size{:}\ 3]$



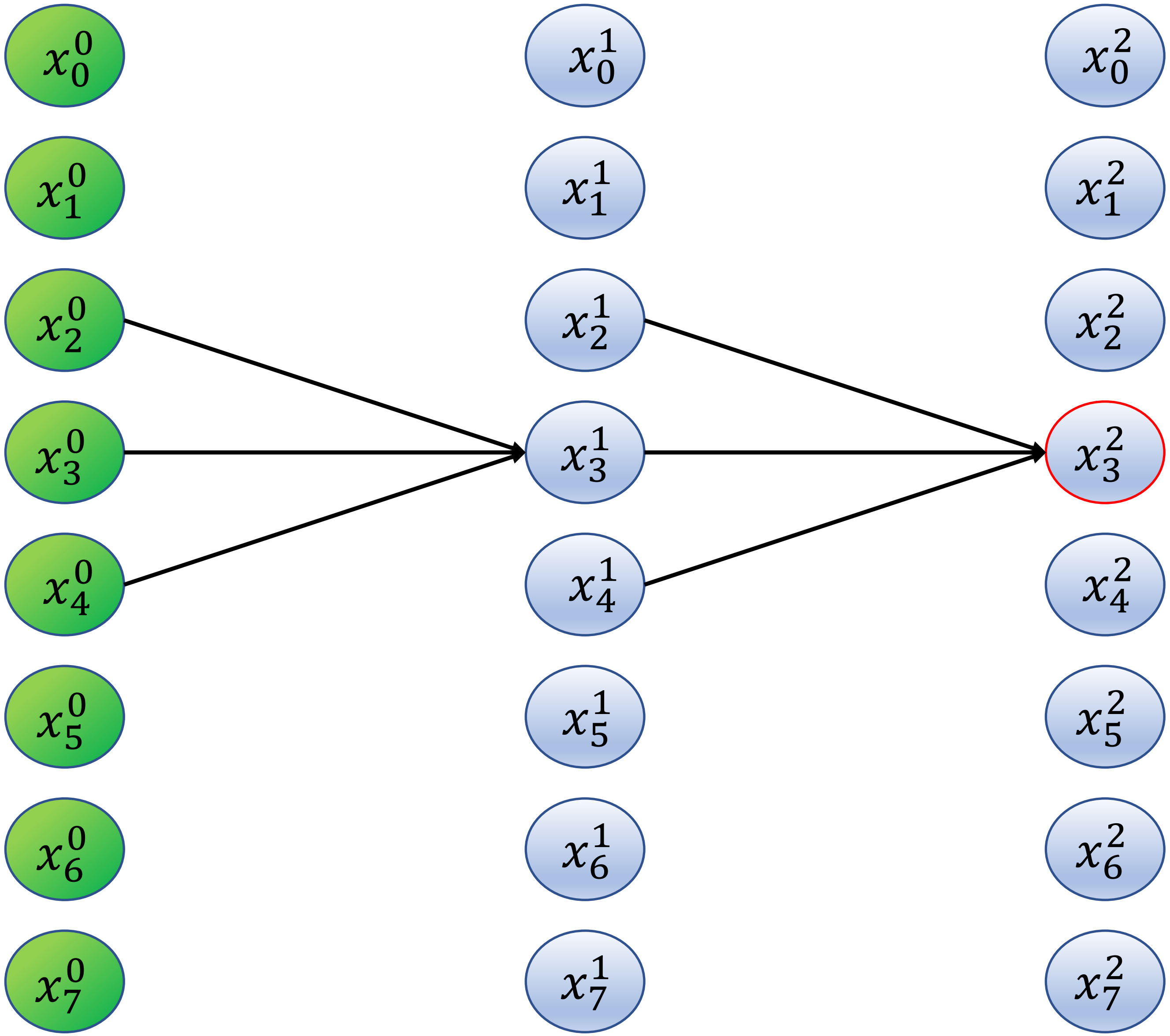
Receptive Field

$[Filter\ size\!: 3]$



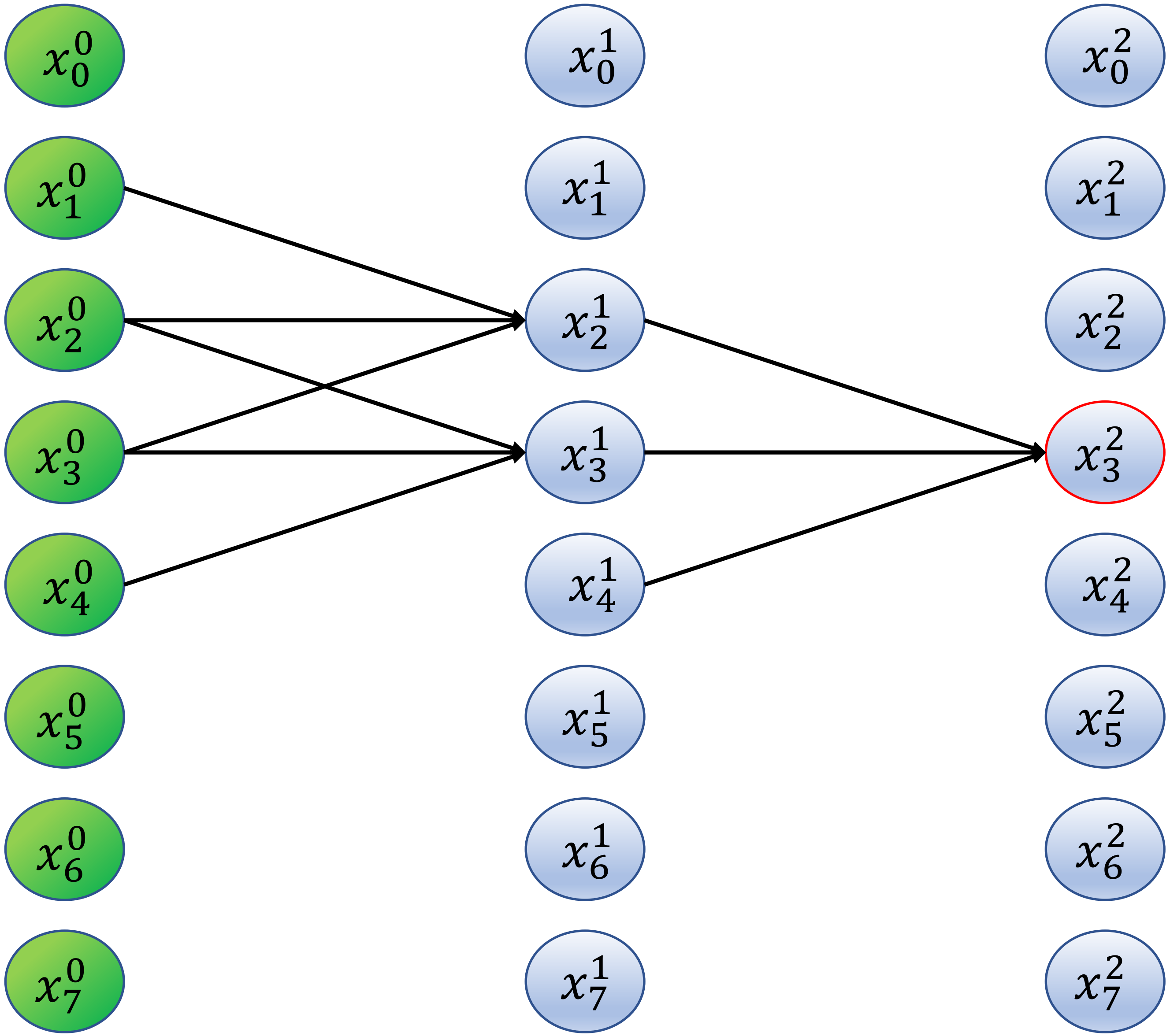
Receptive Field

$[Filter\ size: 3]$



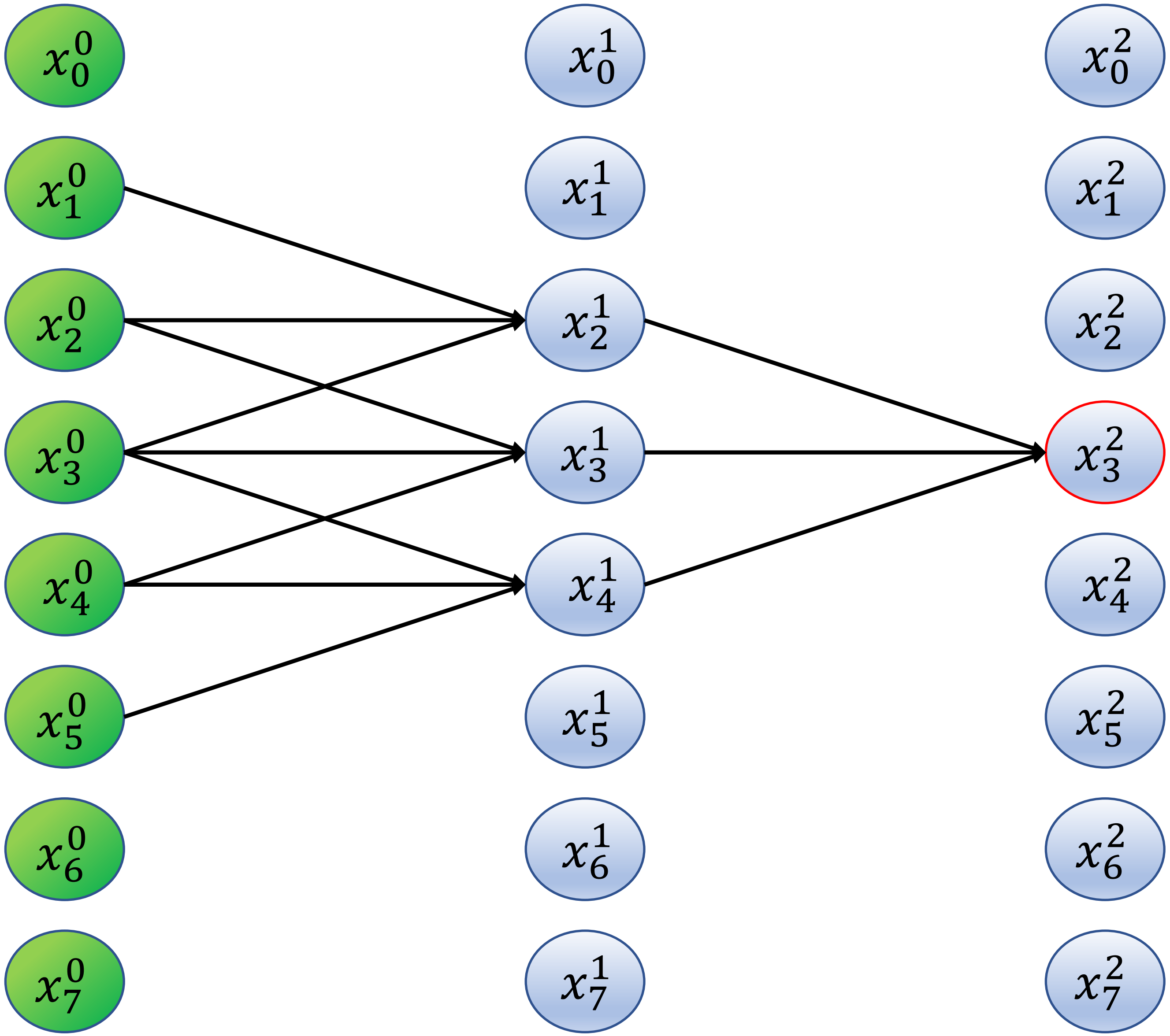
Receptive Field

$[Filter\ size: 3]$



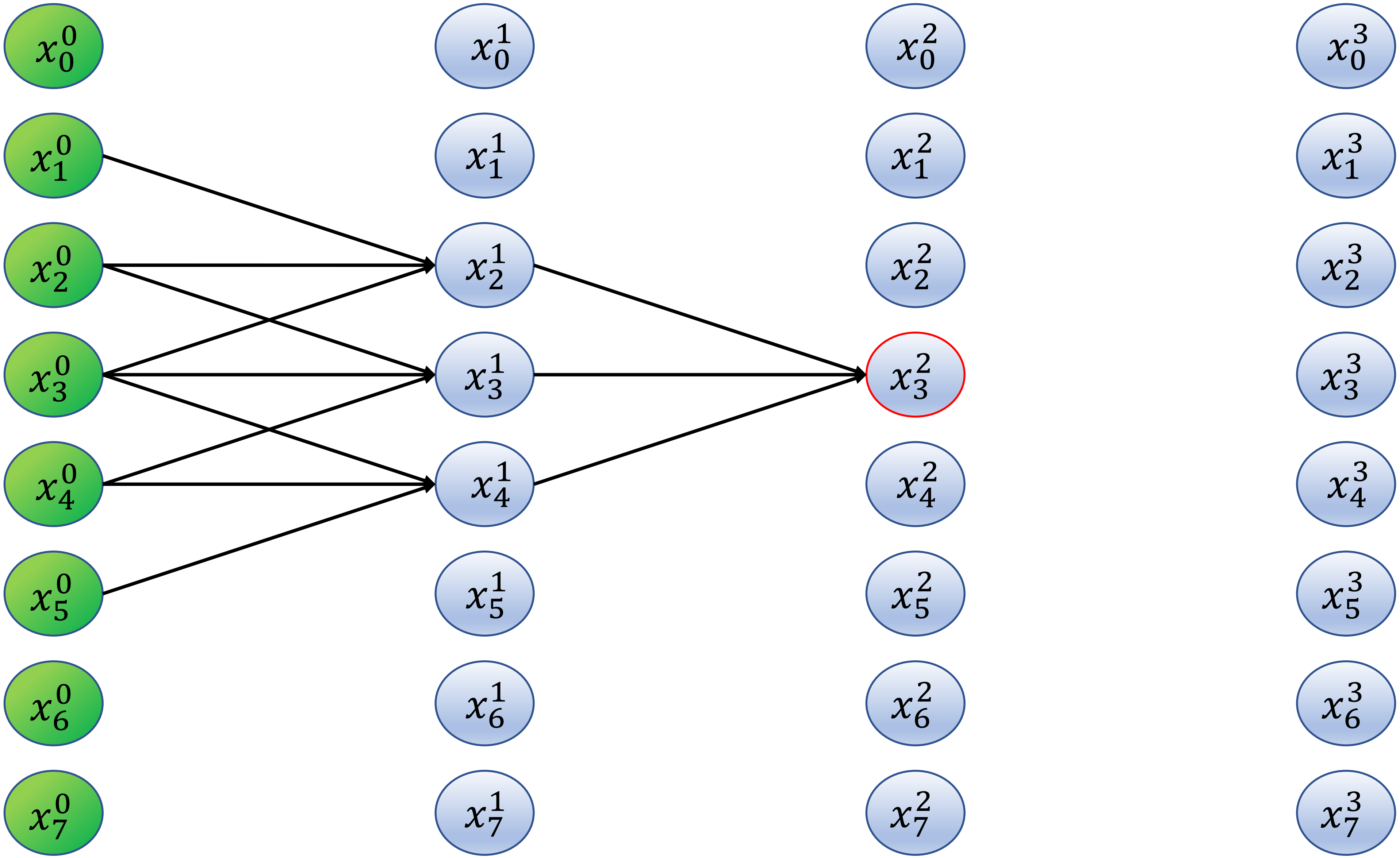
Receptive Field

$[Filter\ size: 3]$



Receptive Field

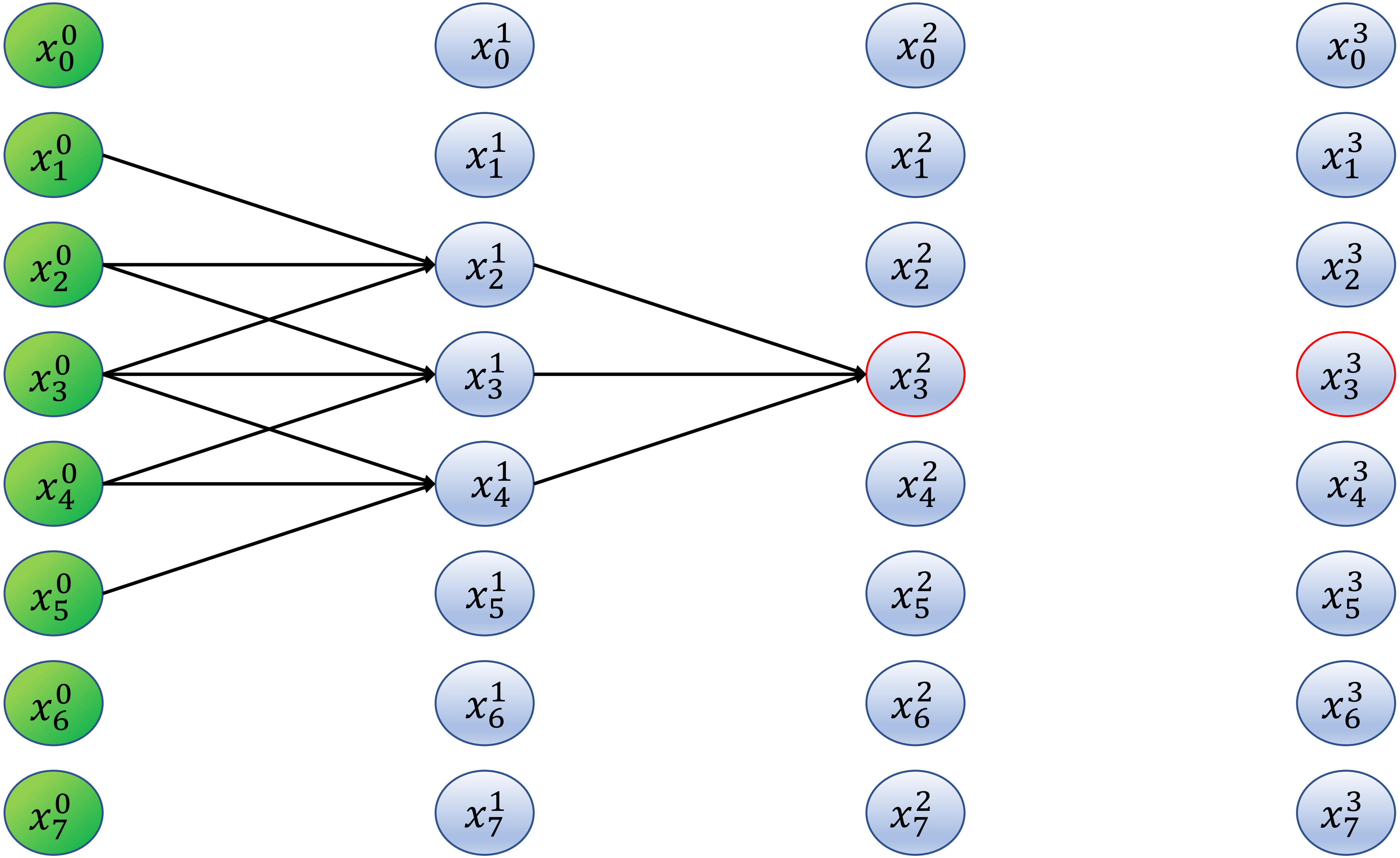
$[\textit{Filter size}: 3]$



Receptive Field

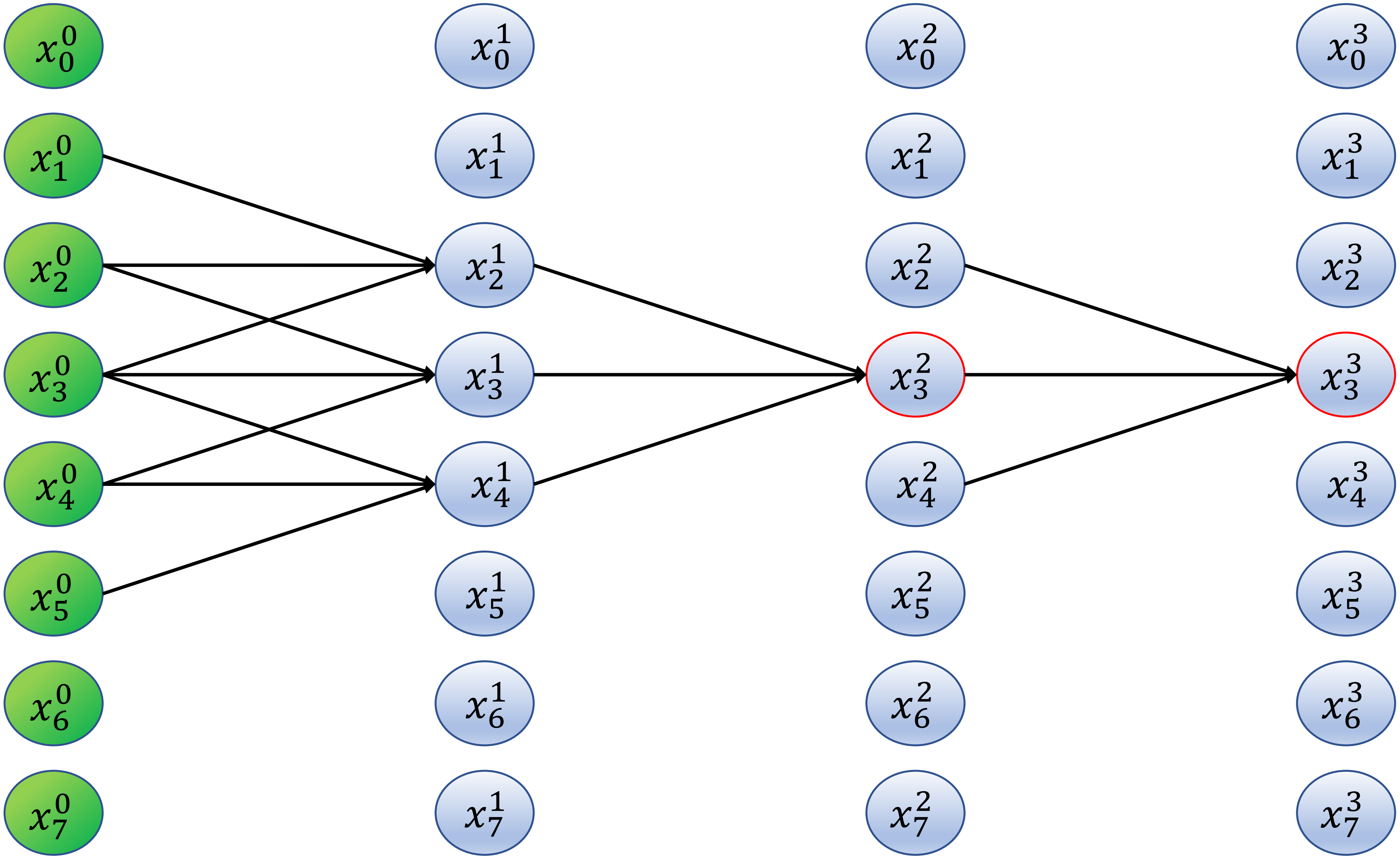
$[Filter\ size: 3]$

[redacted]



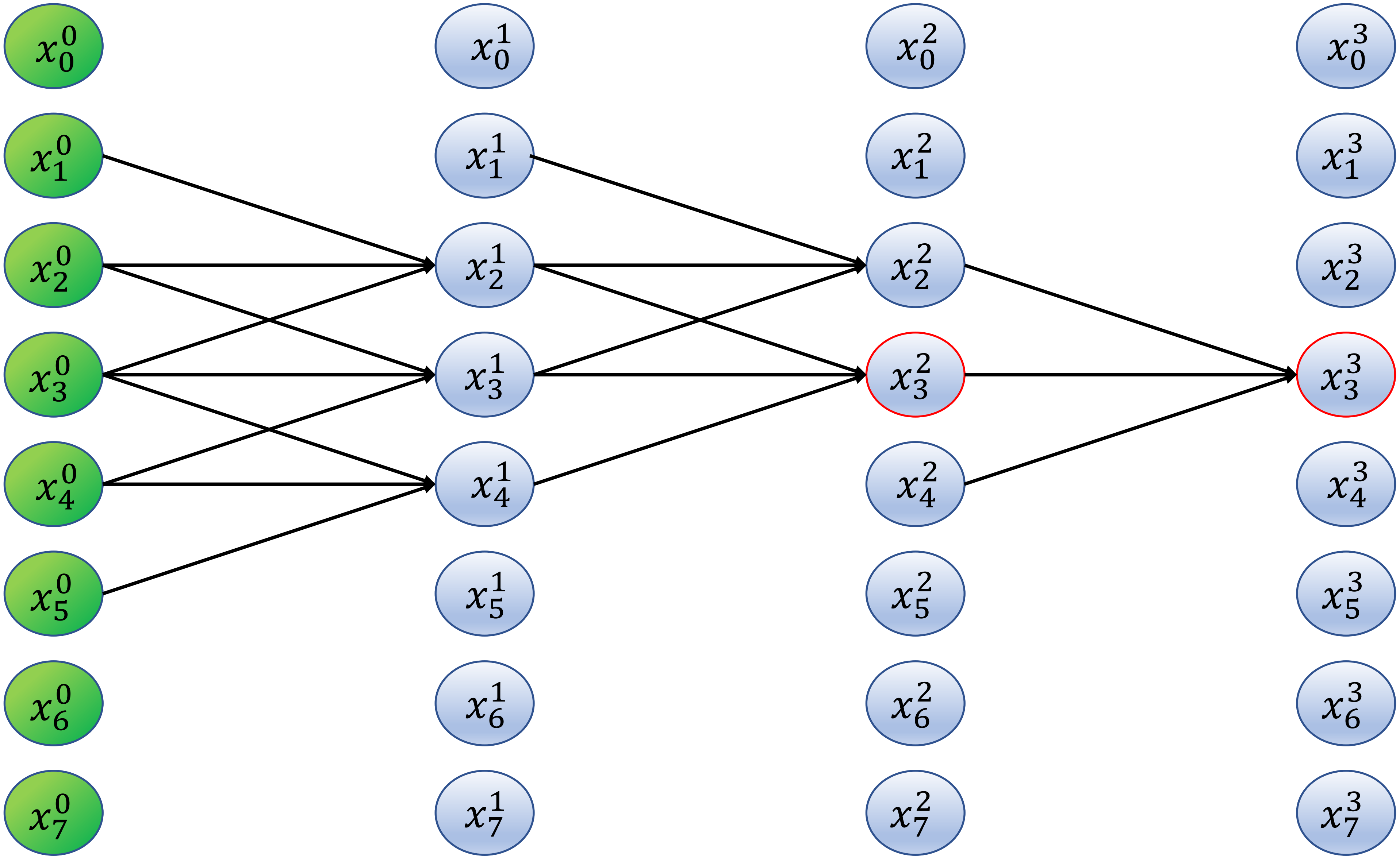
Receptive Field

$[Filter\ size: 3]$



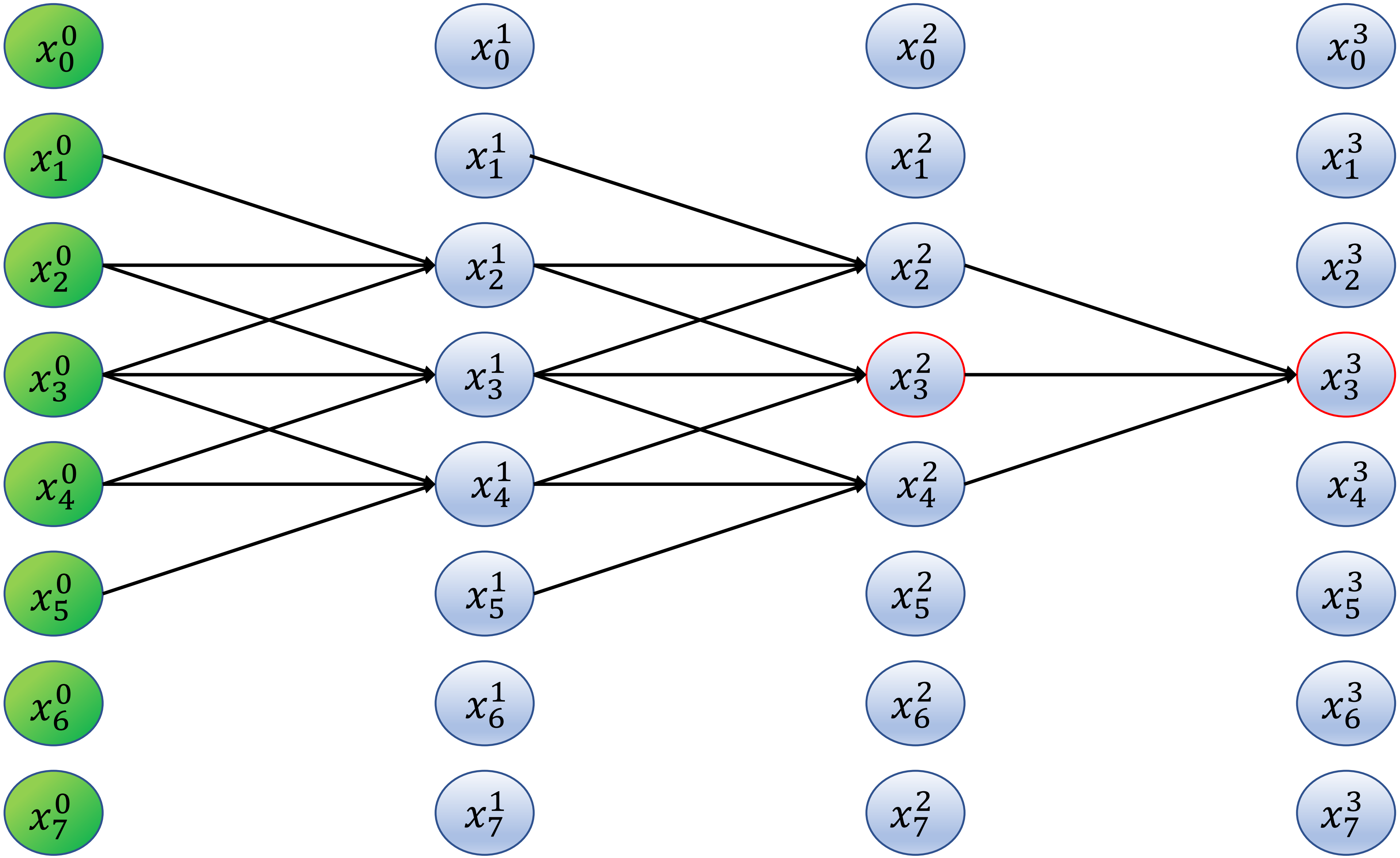
Receptive Field

$[Filter\ size: 3]$



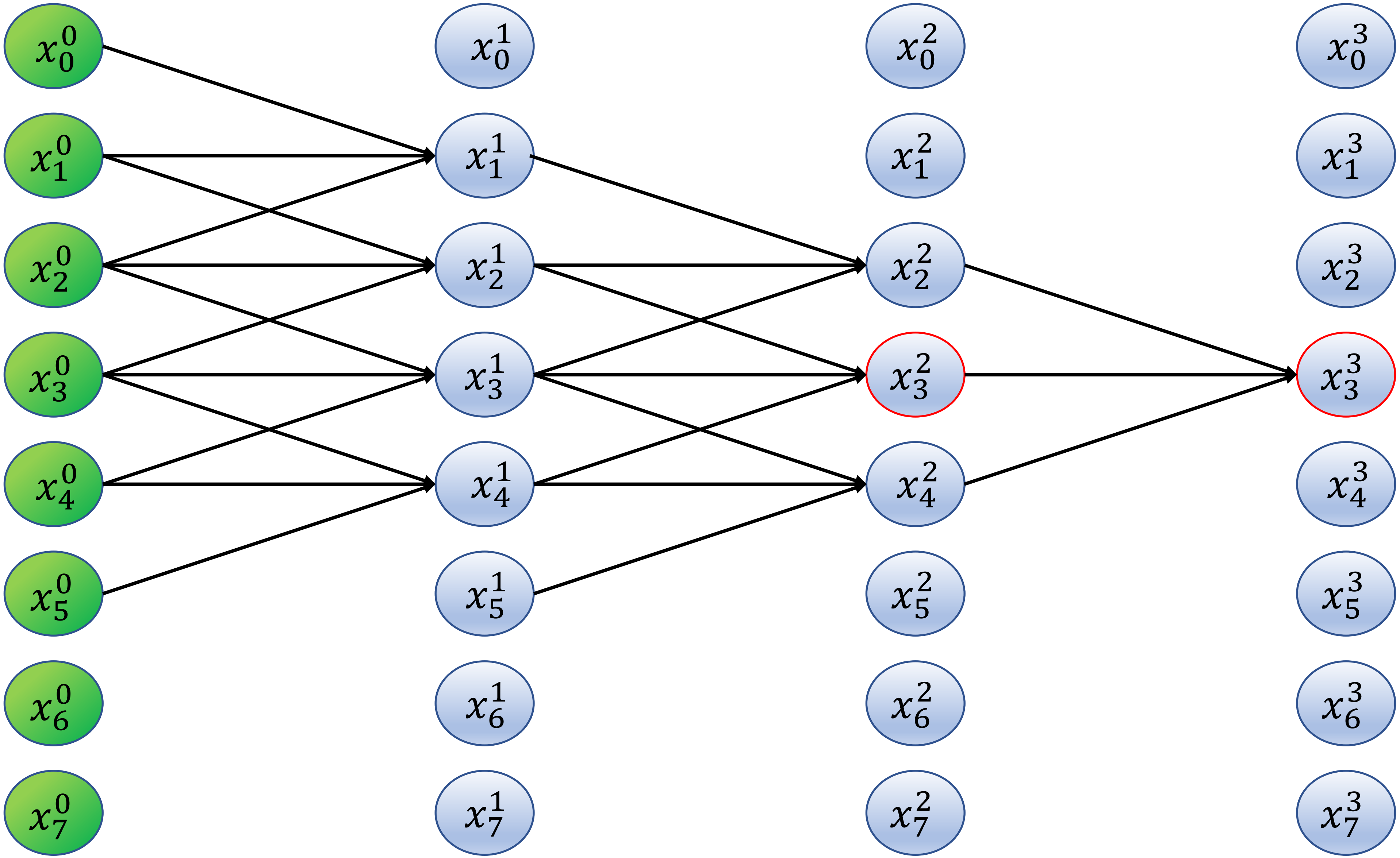
Receptive Field

$[Filter\ size: 3]$



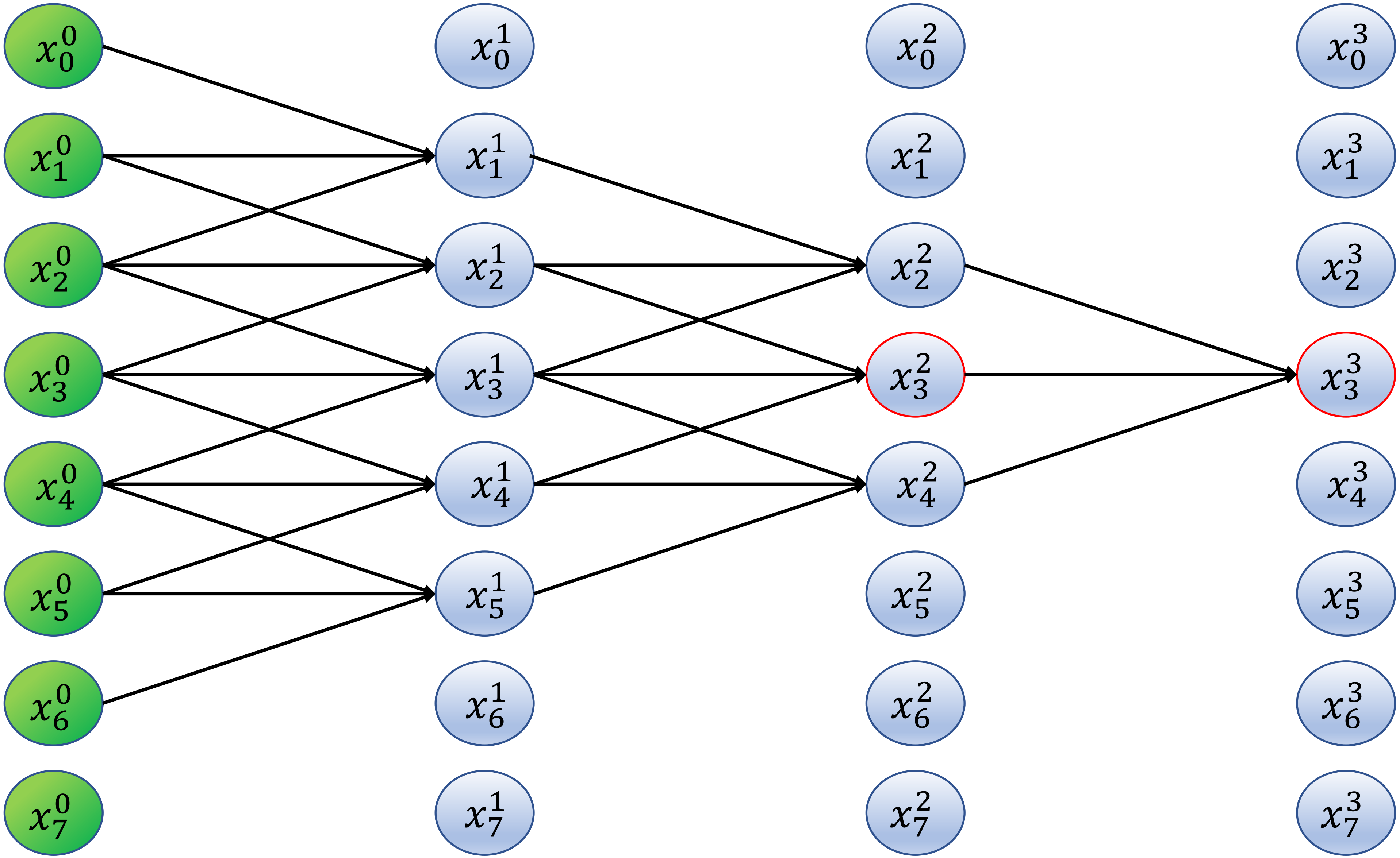
Receptive Field

$[Filter\ size\!: 3]$



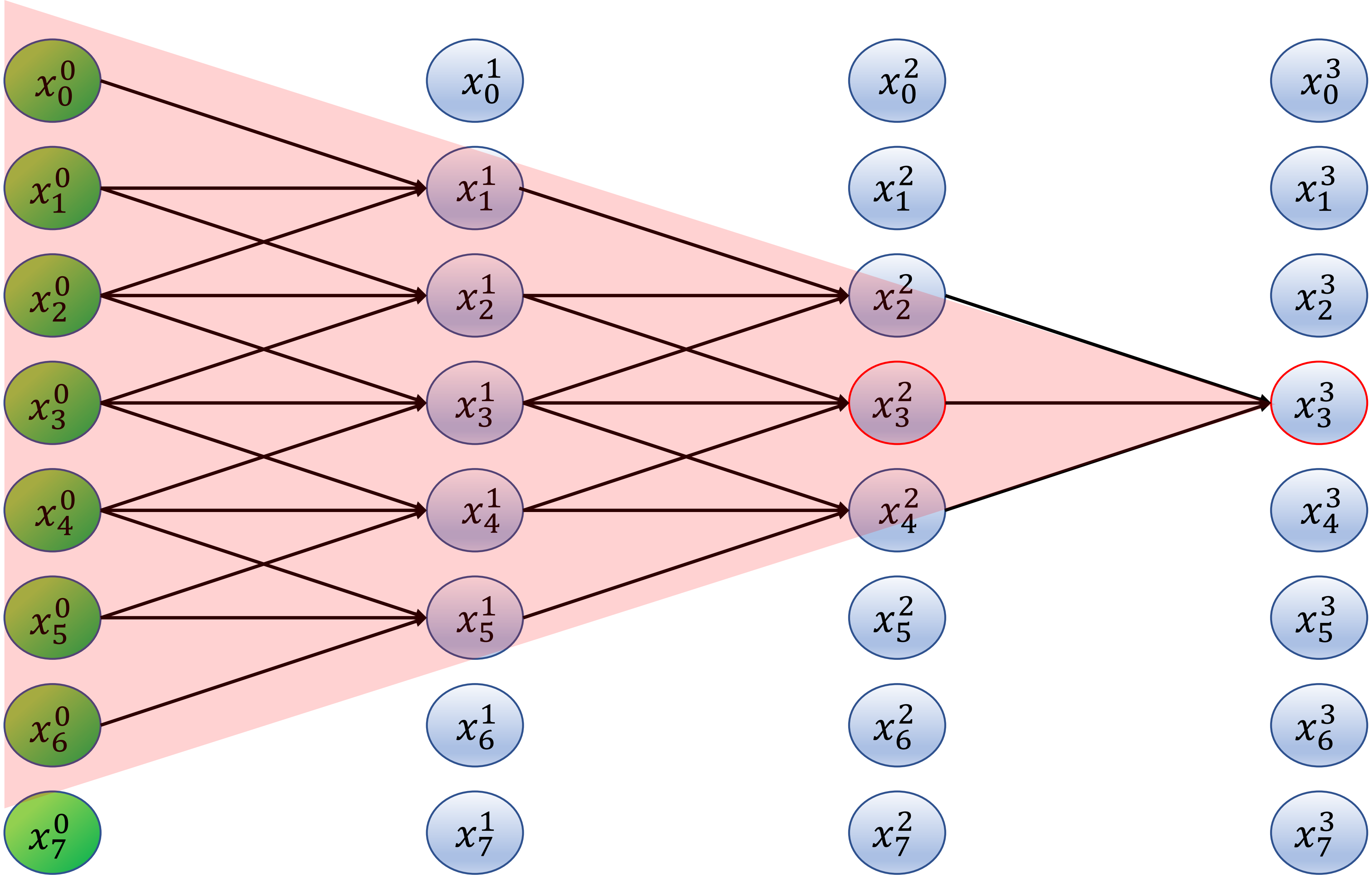
Receptive Field

$[Filter\ size: 3]$

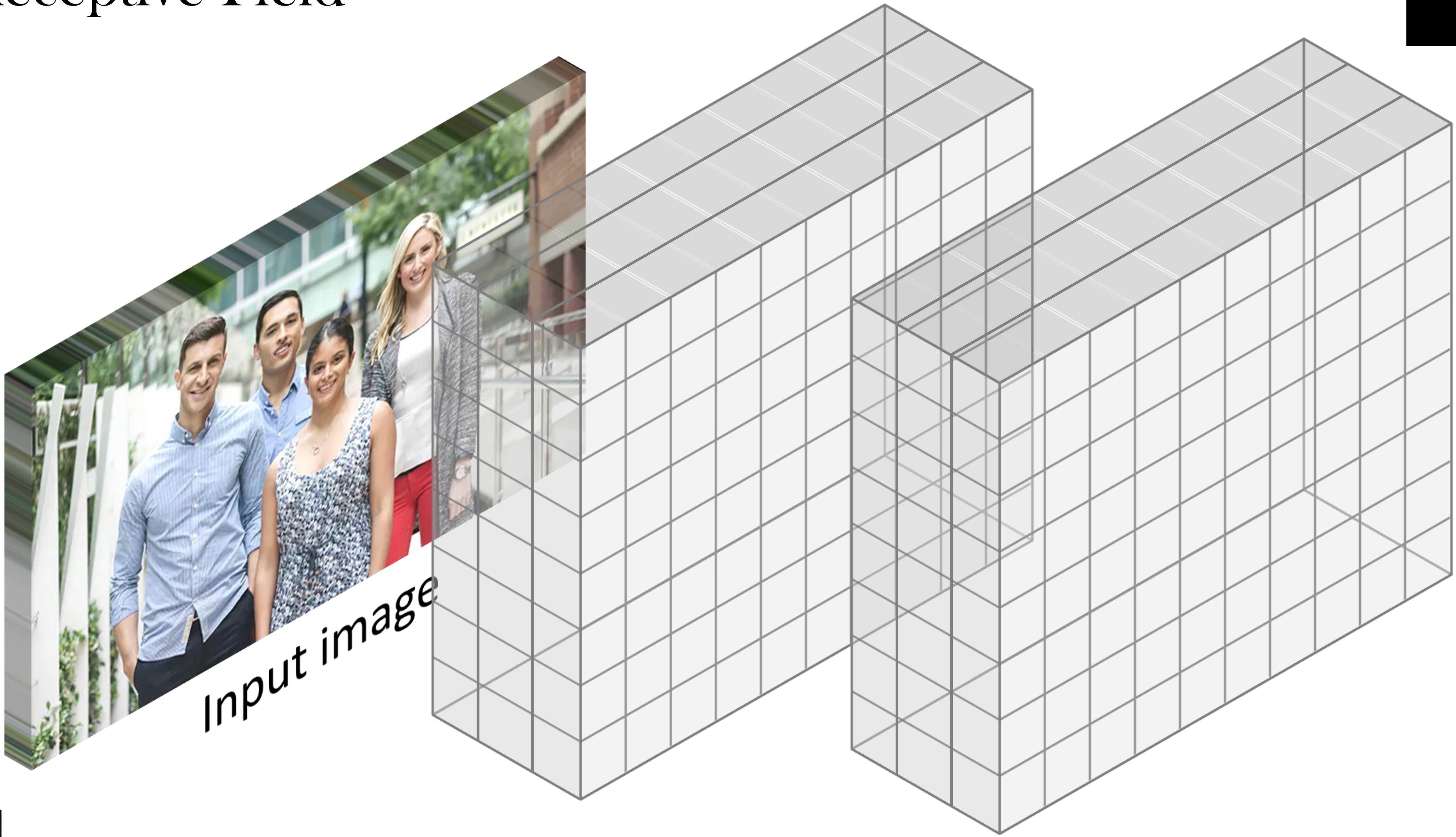


Receptive Field

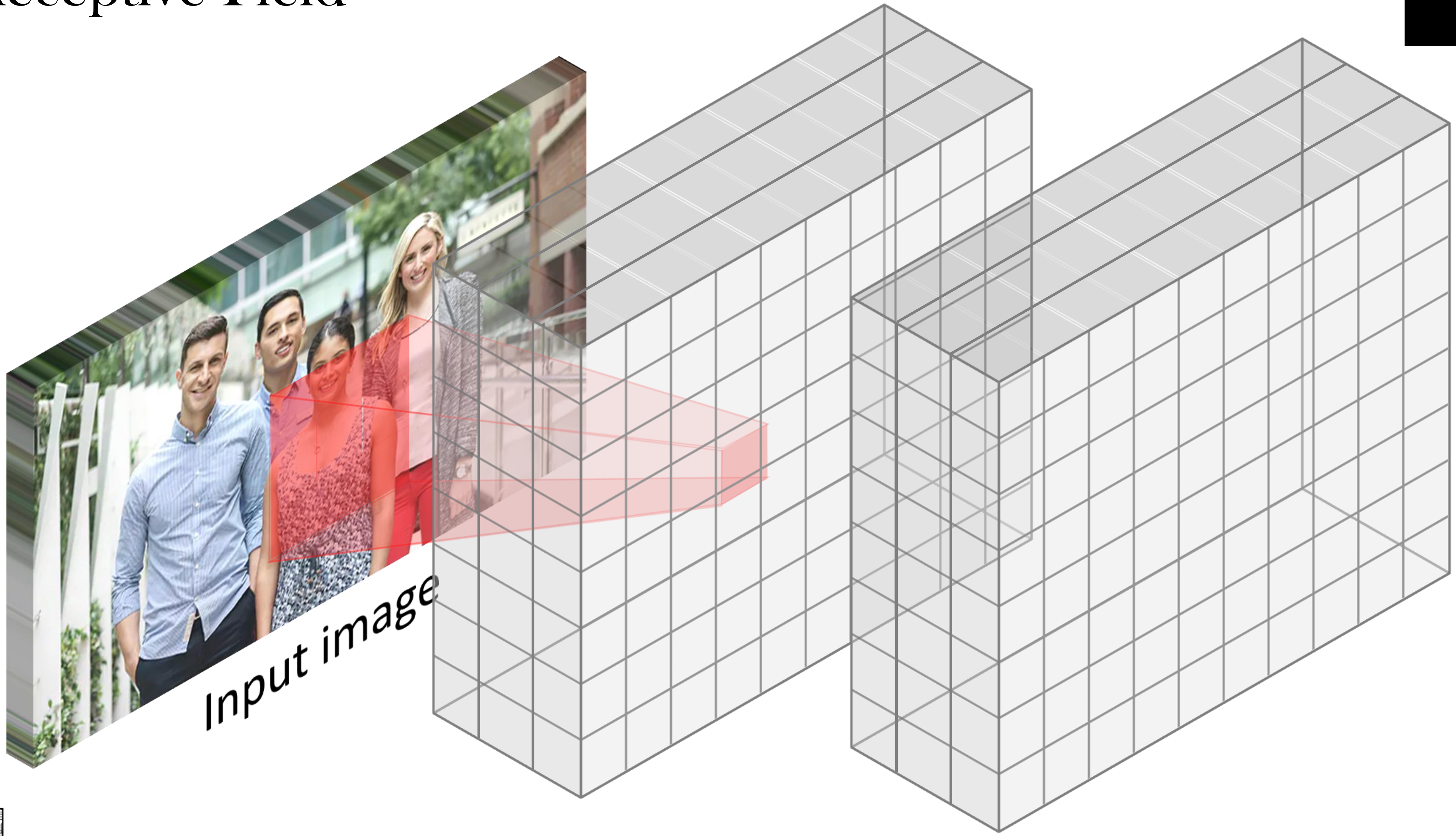
$[Filter\ size: 3]$



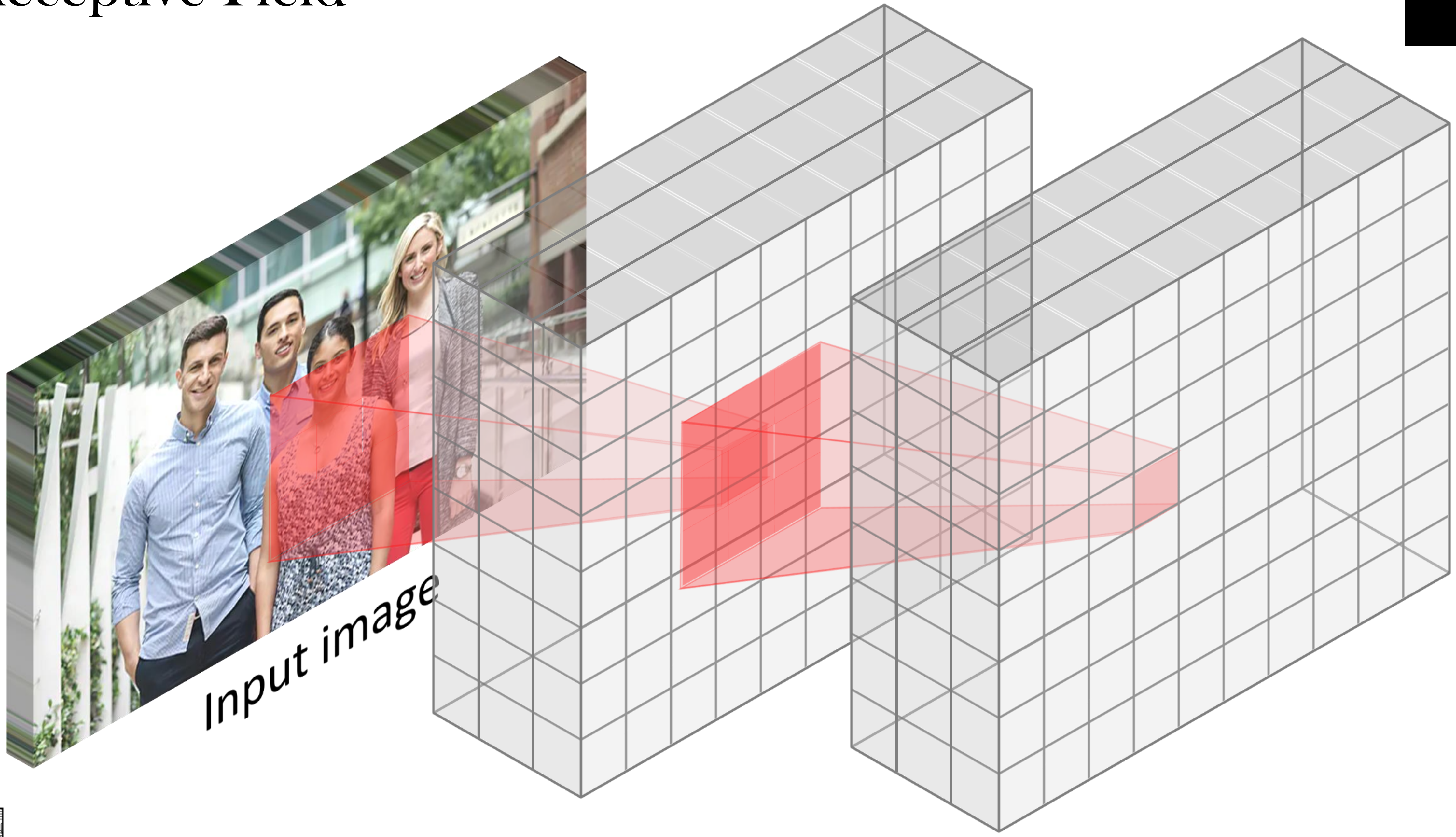
Receptive Field



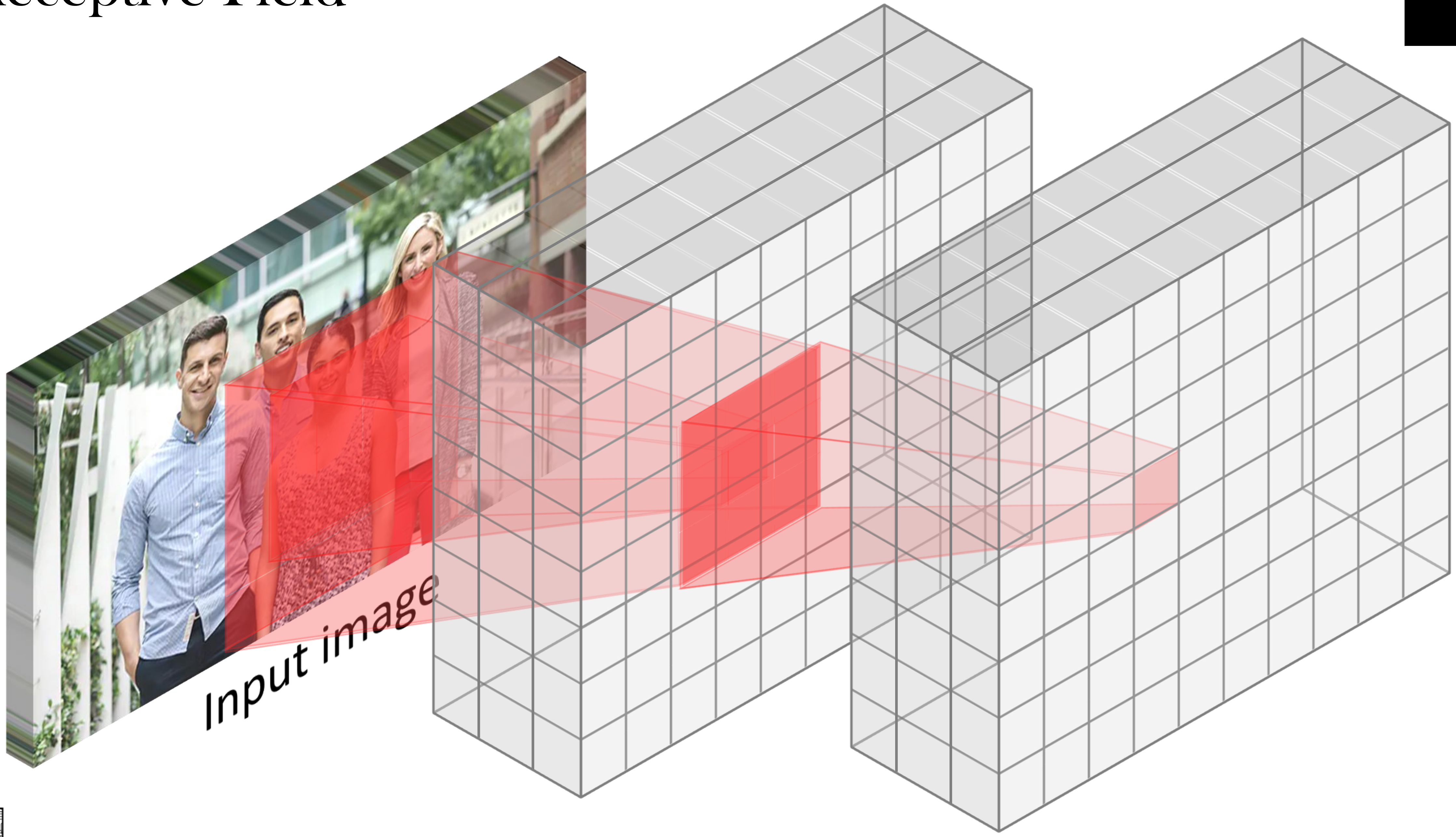
Receptive Field



Receptive Field



Receptive Field



Convs rock!

Convs rock!

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

Convs rock!

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

Convs rock!

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

Convs rock!

✓ 1. Maintain 2D structure logic

✓ 2. Shift invariant (actually, equivariant)

✓ 3. Consider only local correlations

4. Hierarchically growing field of view

5. Hierarchically progressing complexity

6. Reasonable amount of params

Convs rock!

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

Convs rock!

1. Maintain 2D structure logic
2. Shift invariant (actually, equivariant)
3. Consider only local correlations
4. Hierarchically growing field of view
5. Hierarchically progressing complexity
6. Reasonable amount of params

Convs rock!

✓ 1. Maintain 2D structure logic

✓ 2. Shift invariant (actually, equivariant)

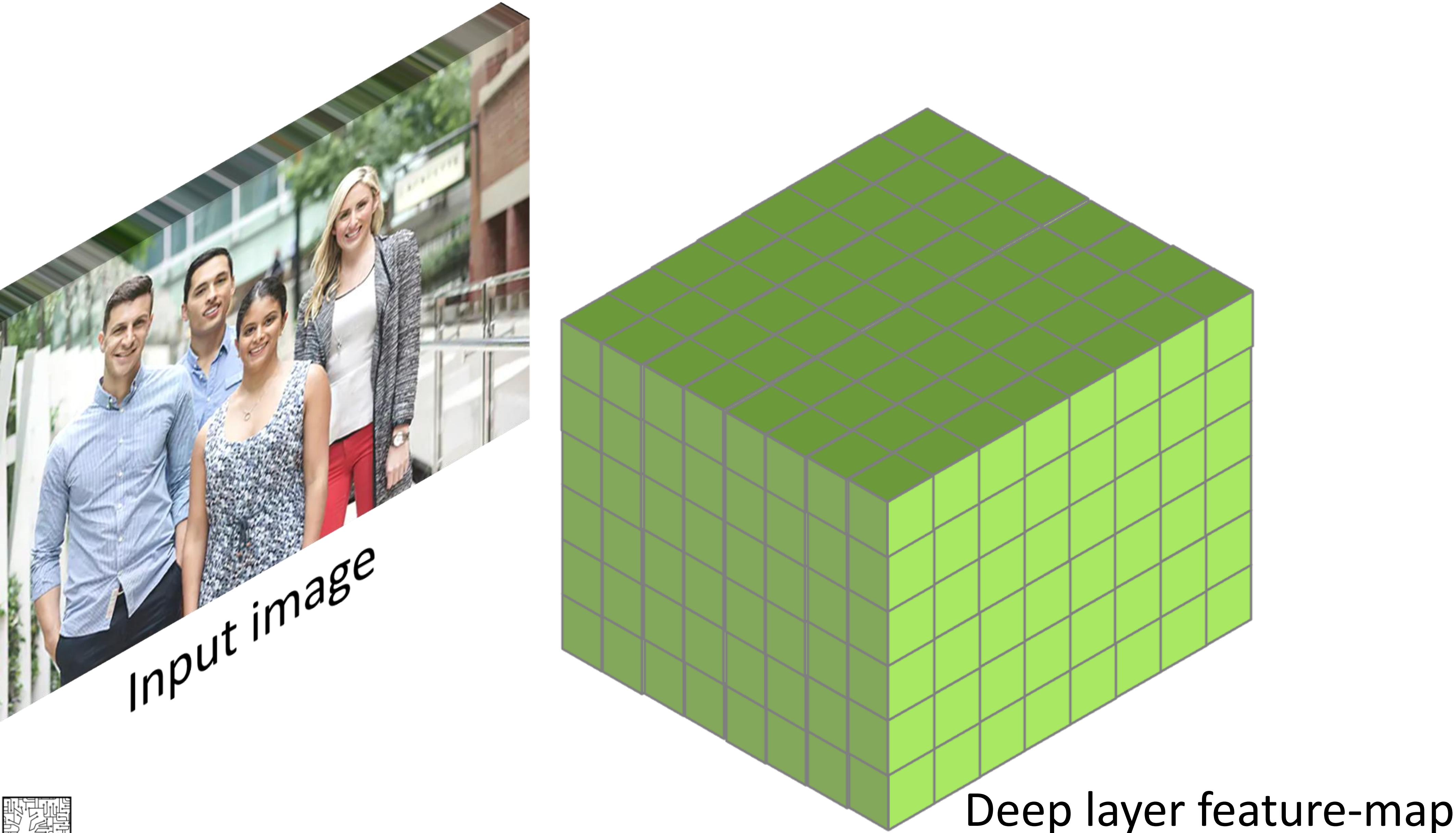
✓ 3. Consider only local correlations

✓ 4. Hierarchically growing field of view

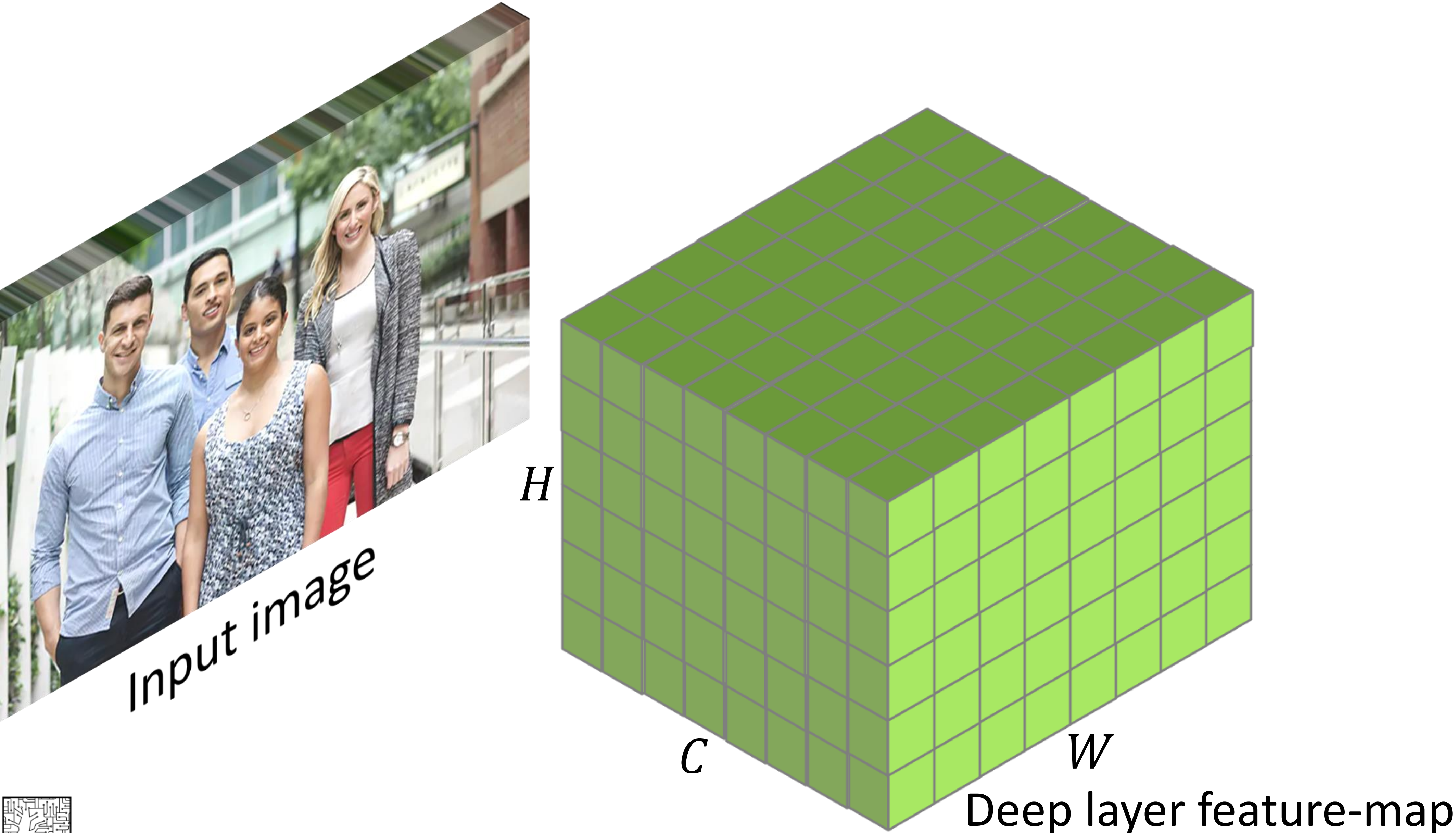
✓ 5. Hierarchically progressing complexity

✓ 6. Reasonable amount of params

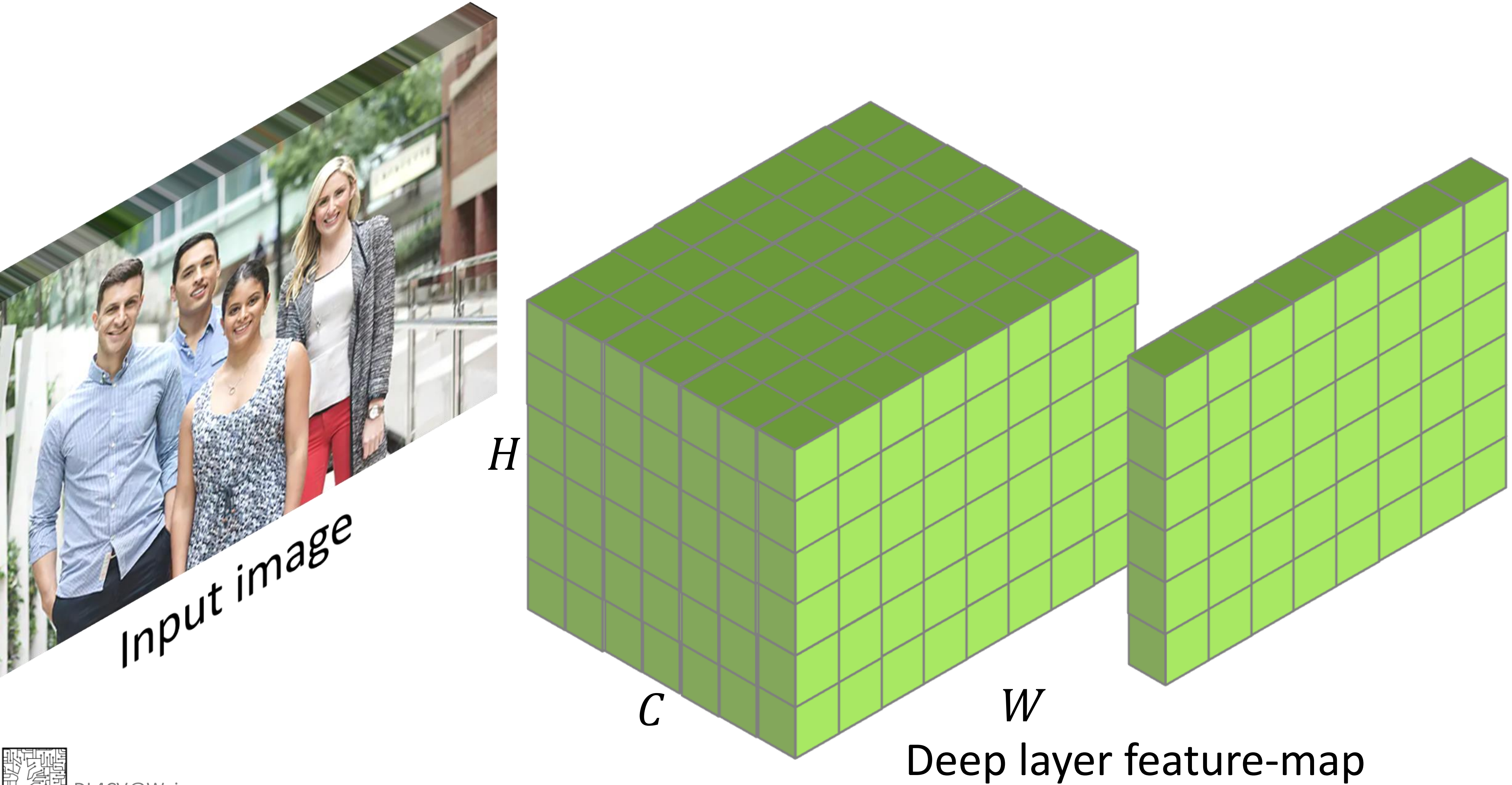
Two important intuitions about feature maps



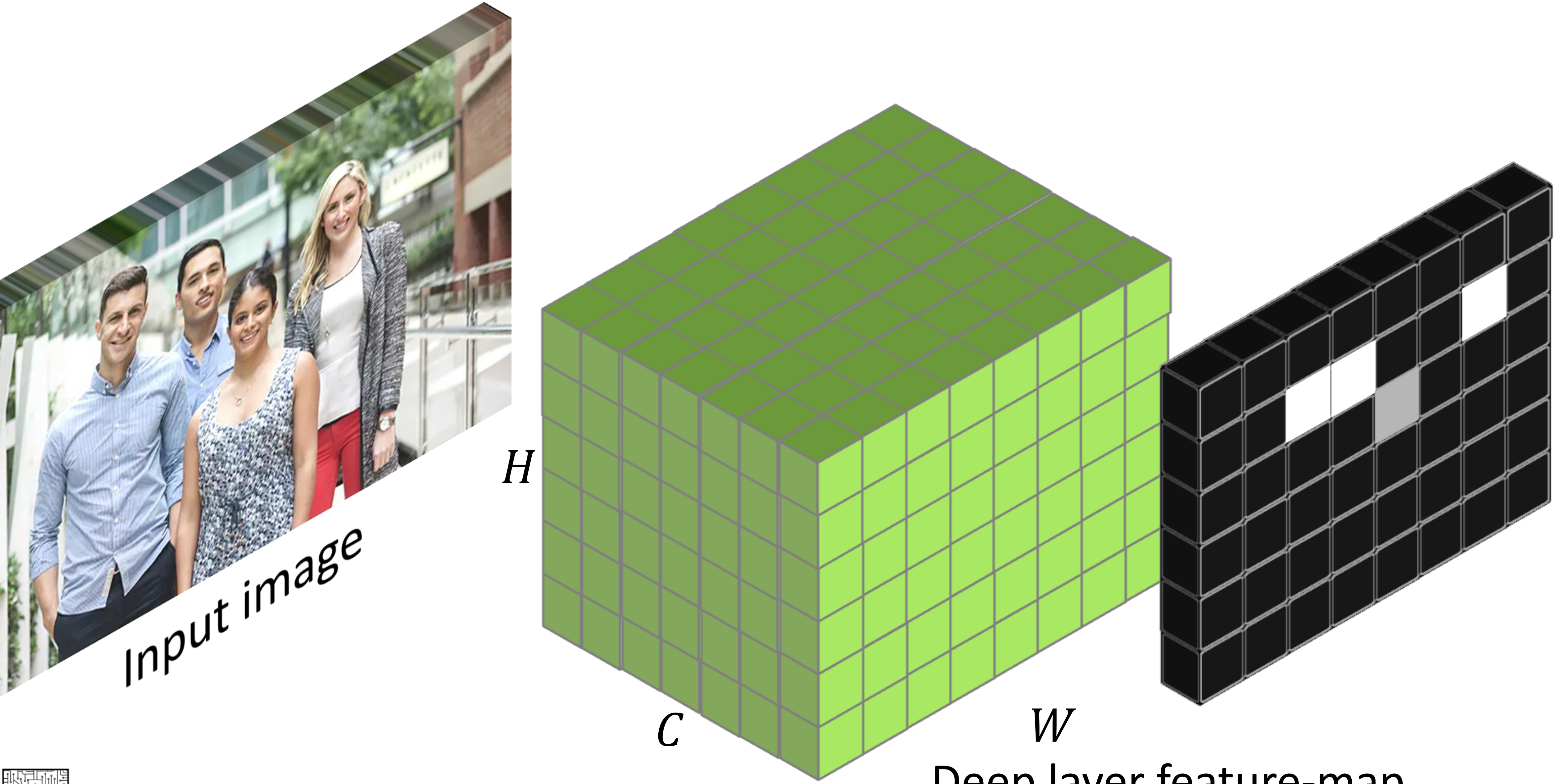
Two important intuitions about feature maps



Two important intuitions about feature maps

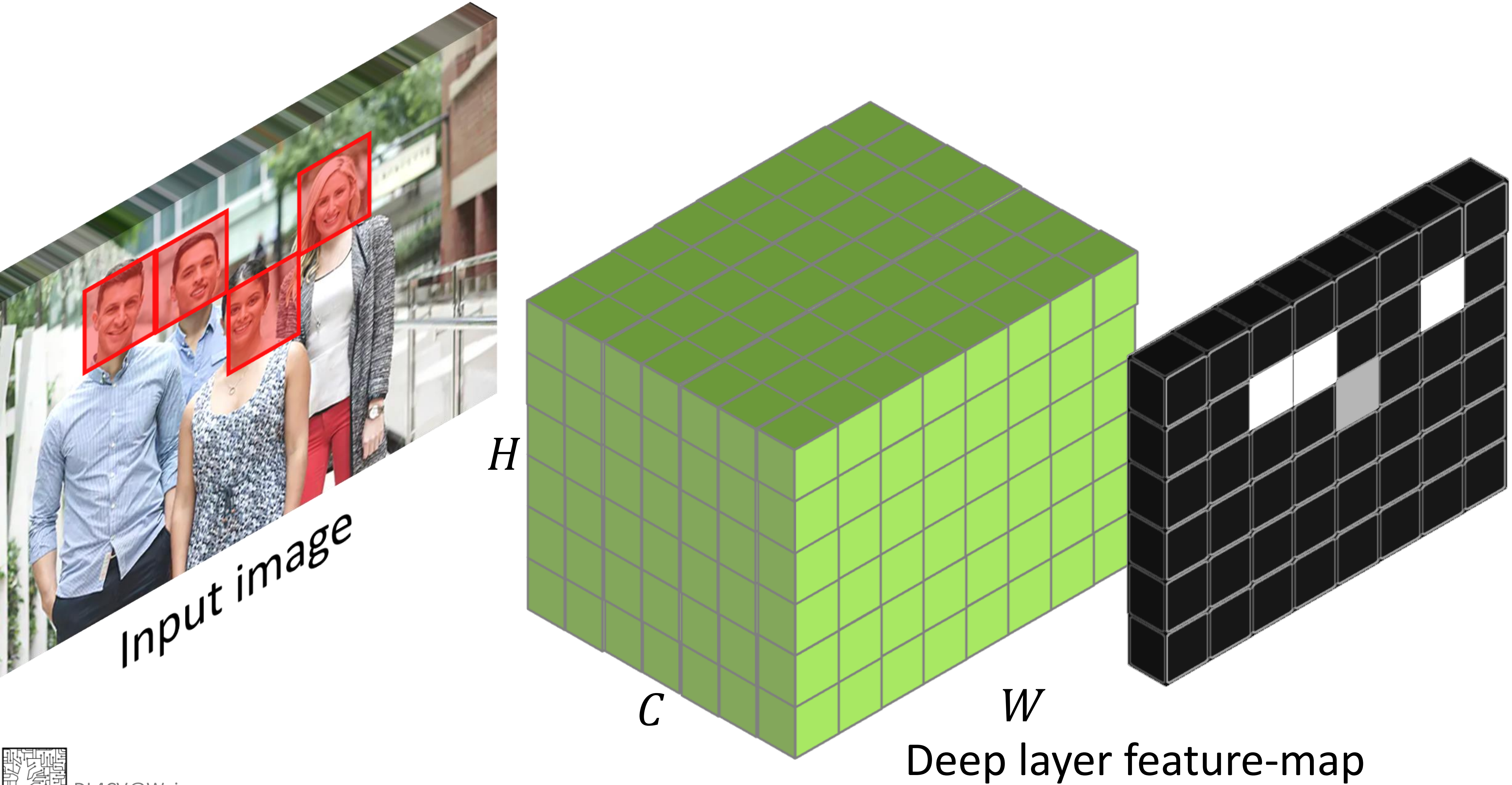


Two important intuitions about feature maps

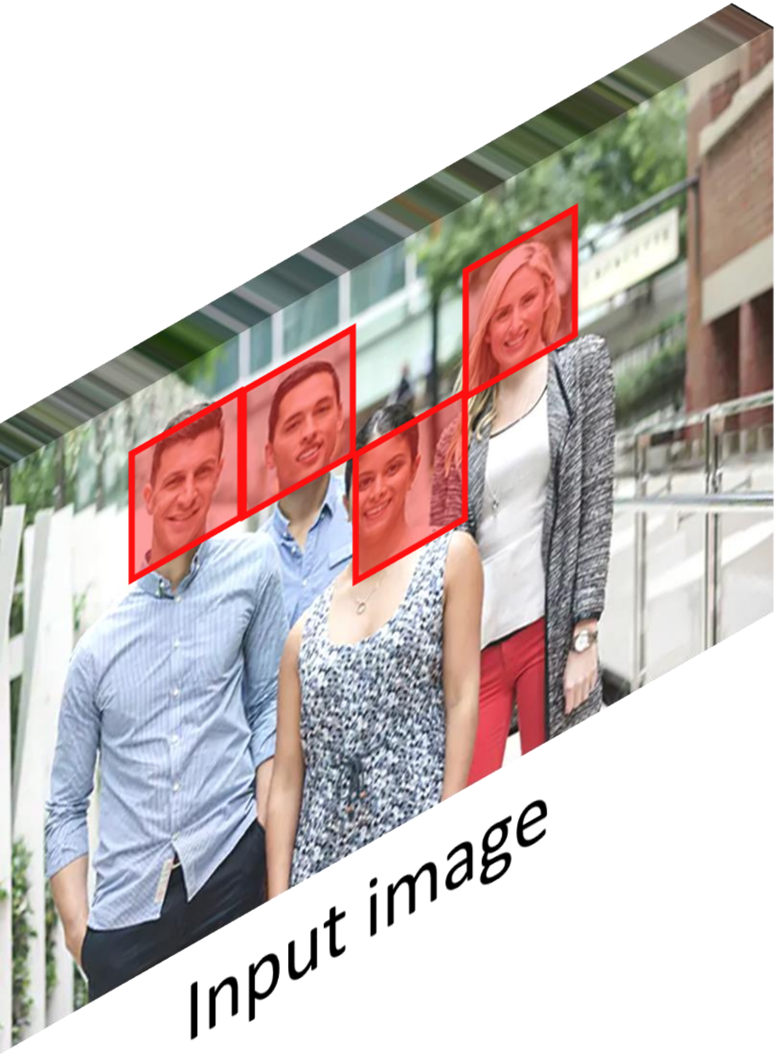


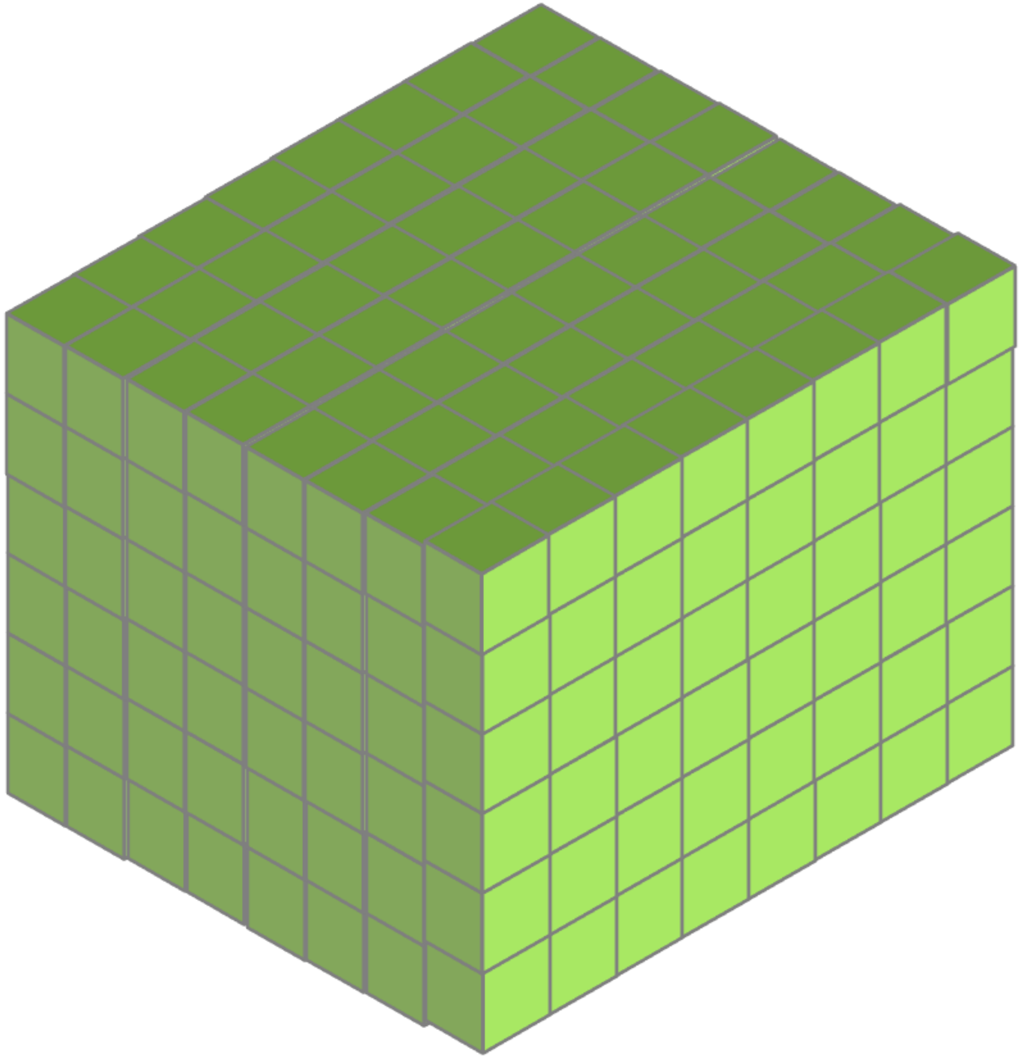
Deep layer feature-map

Two important intuitions about feature maps



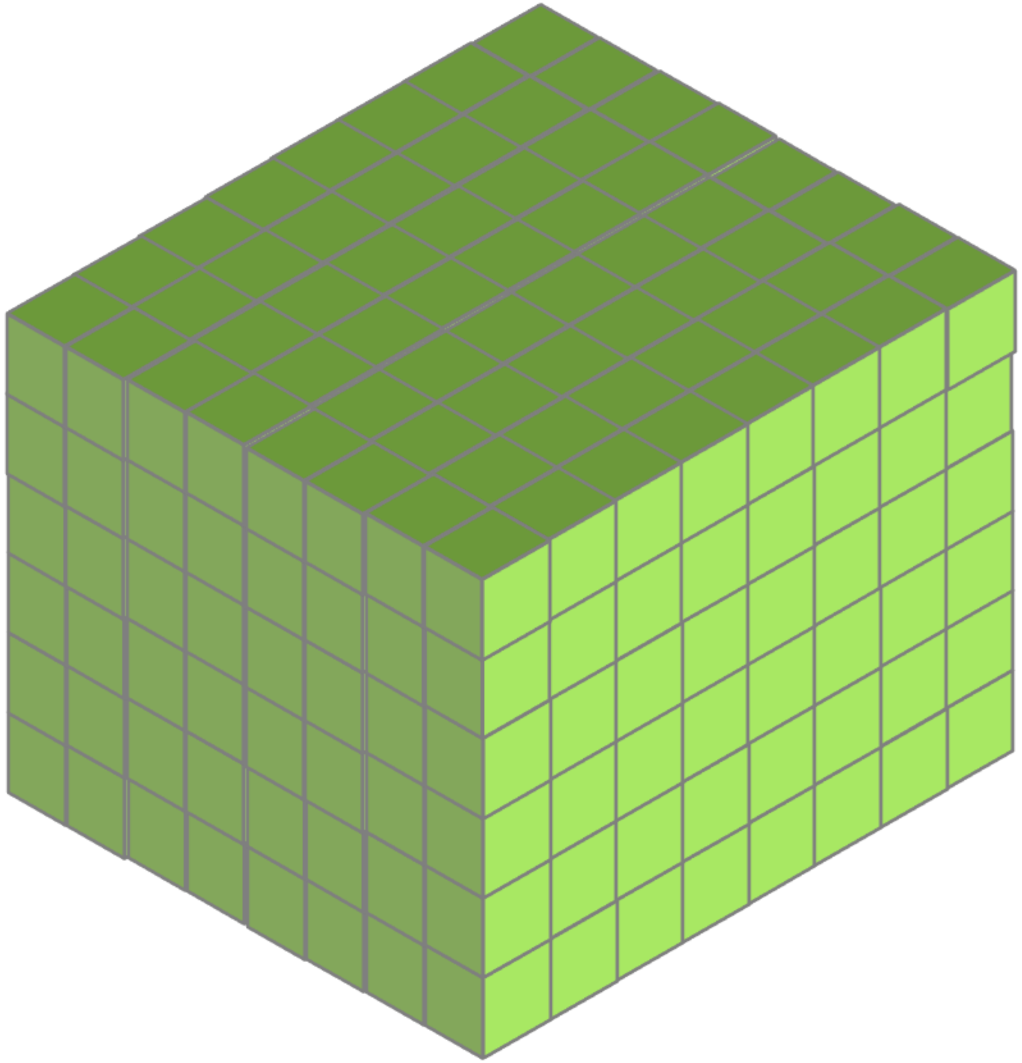
Two important intuitions about feature maps





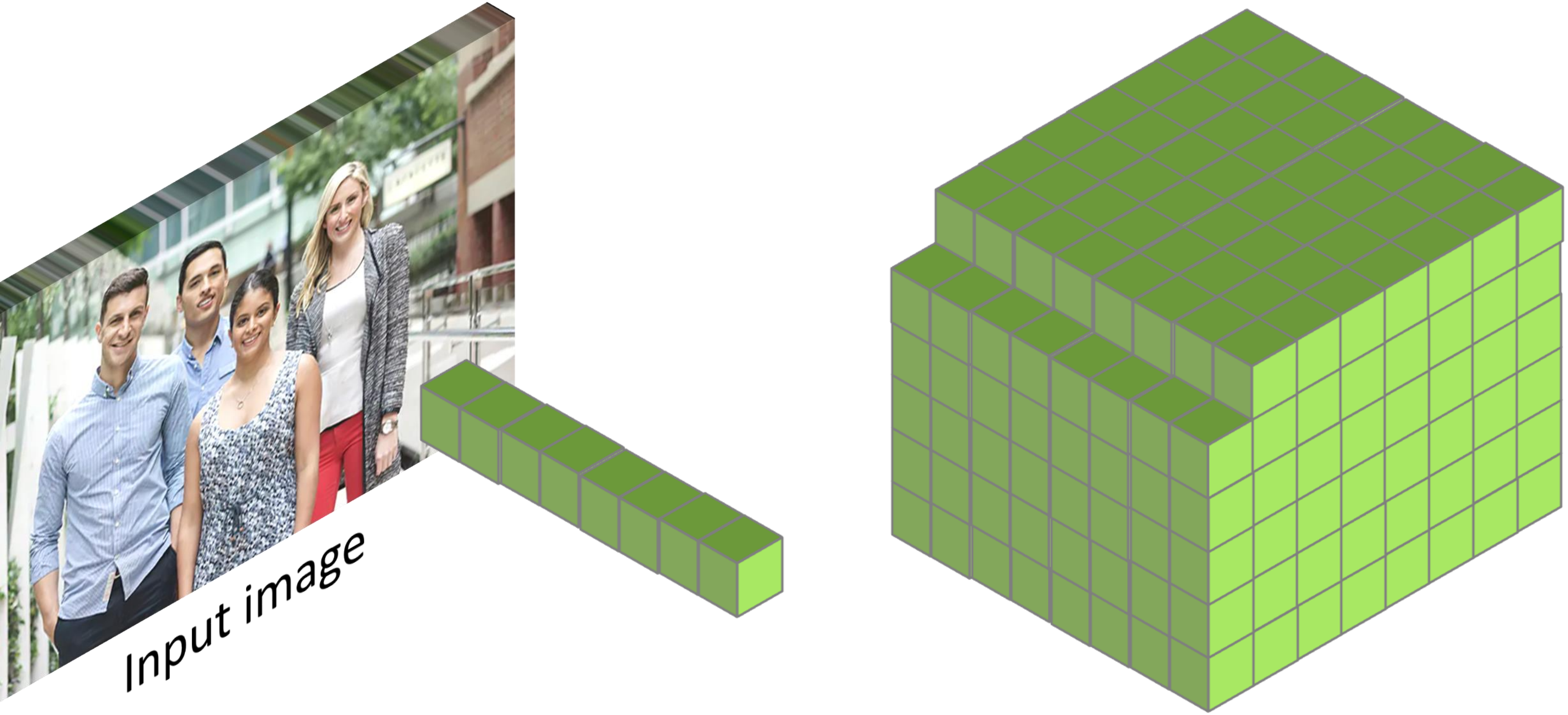
Deep layer feature-map

Two important intuitions about feature maps



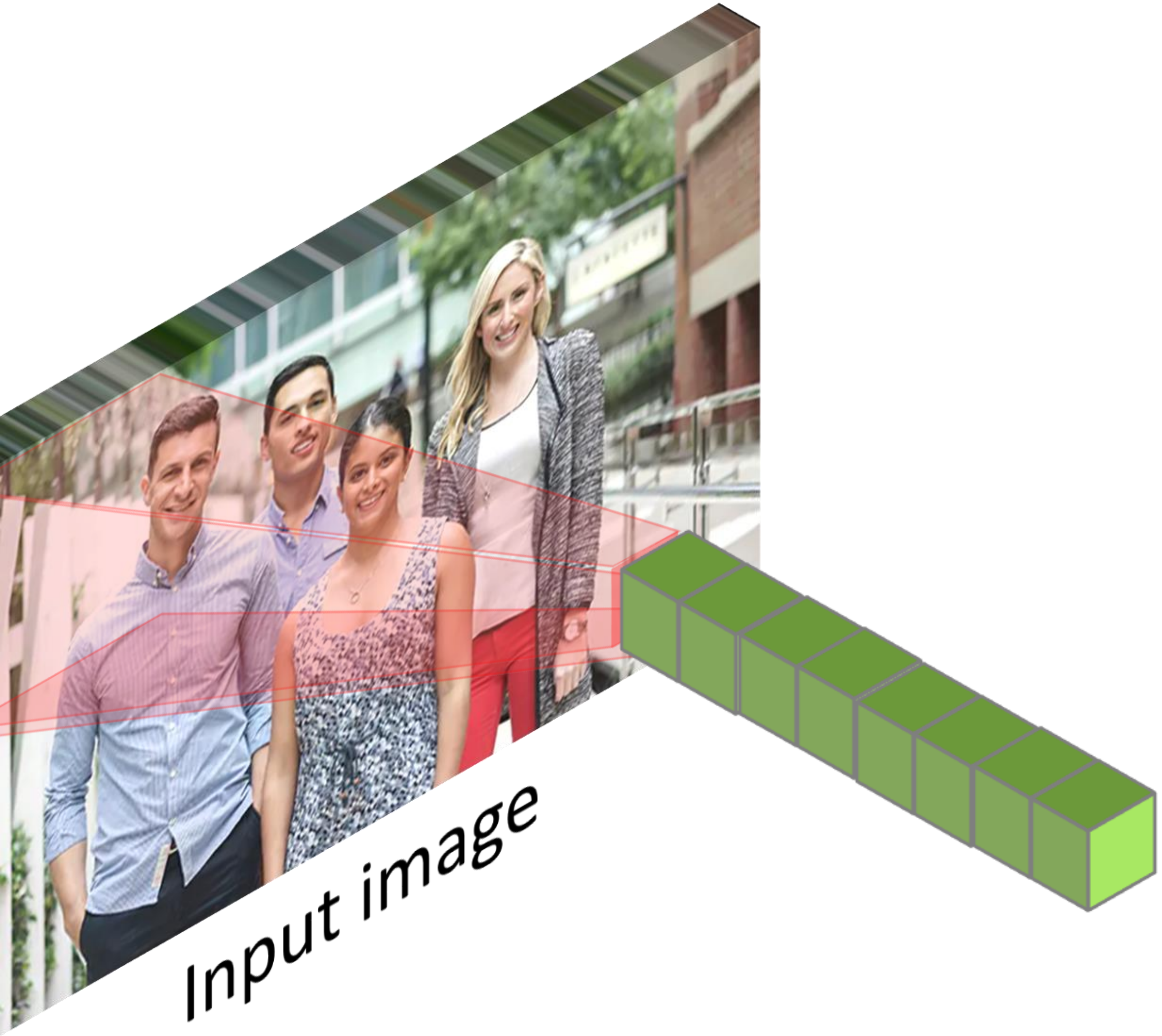
Deep layer feature-map

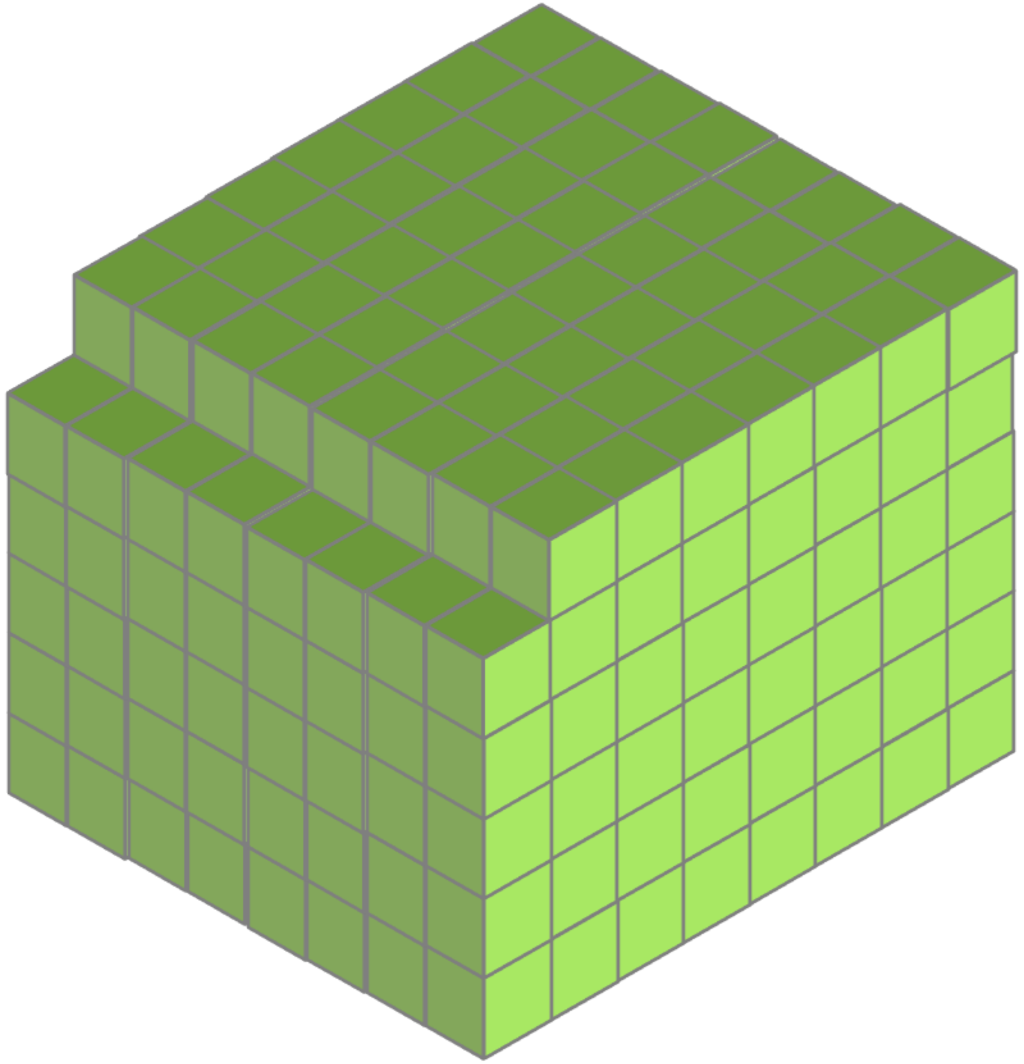
Two important intuitions about feature maps



Deep layer feature-map

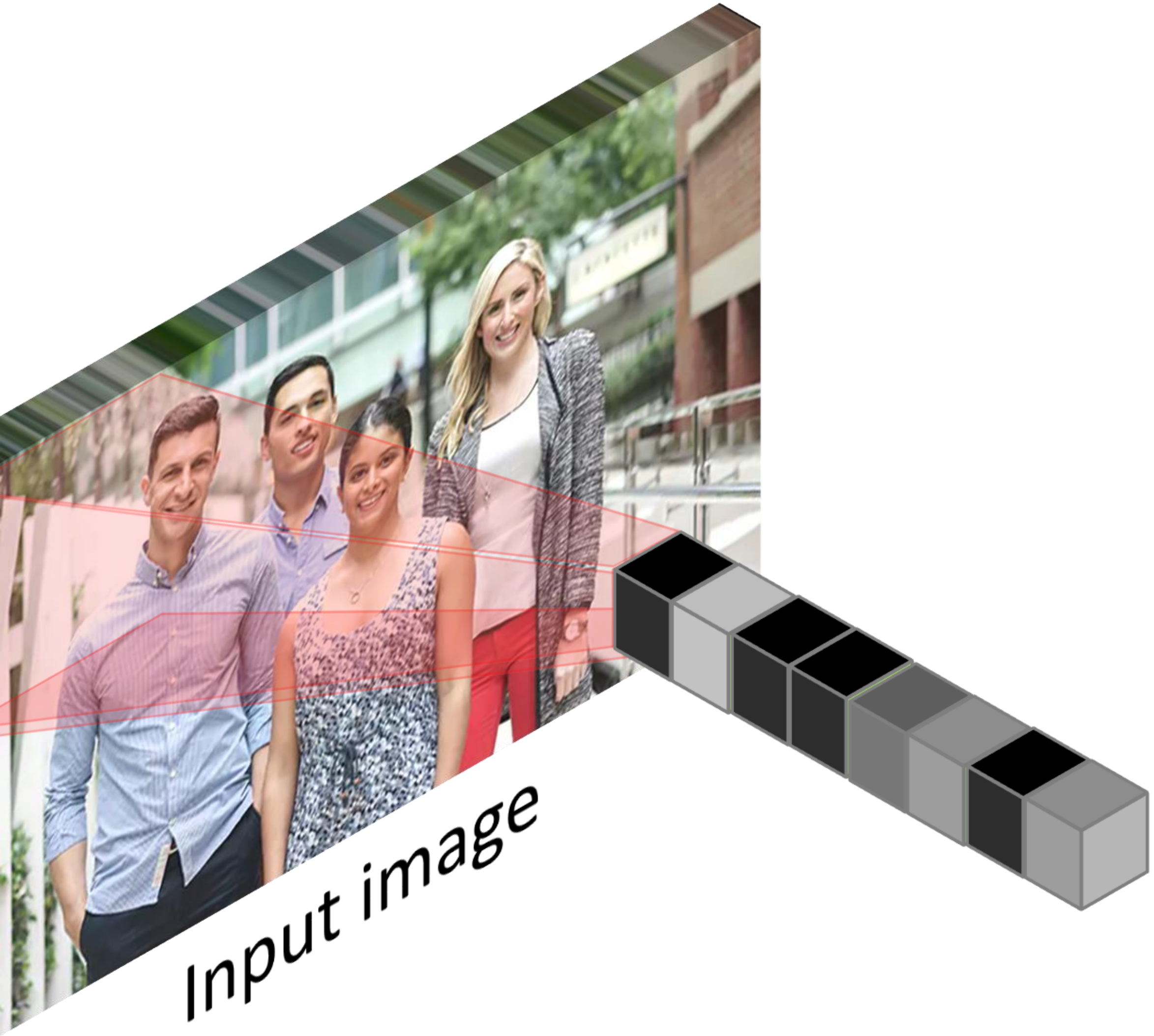
Two important intuitions about feature maps

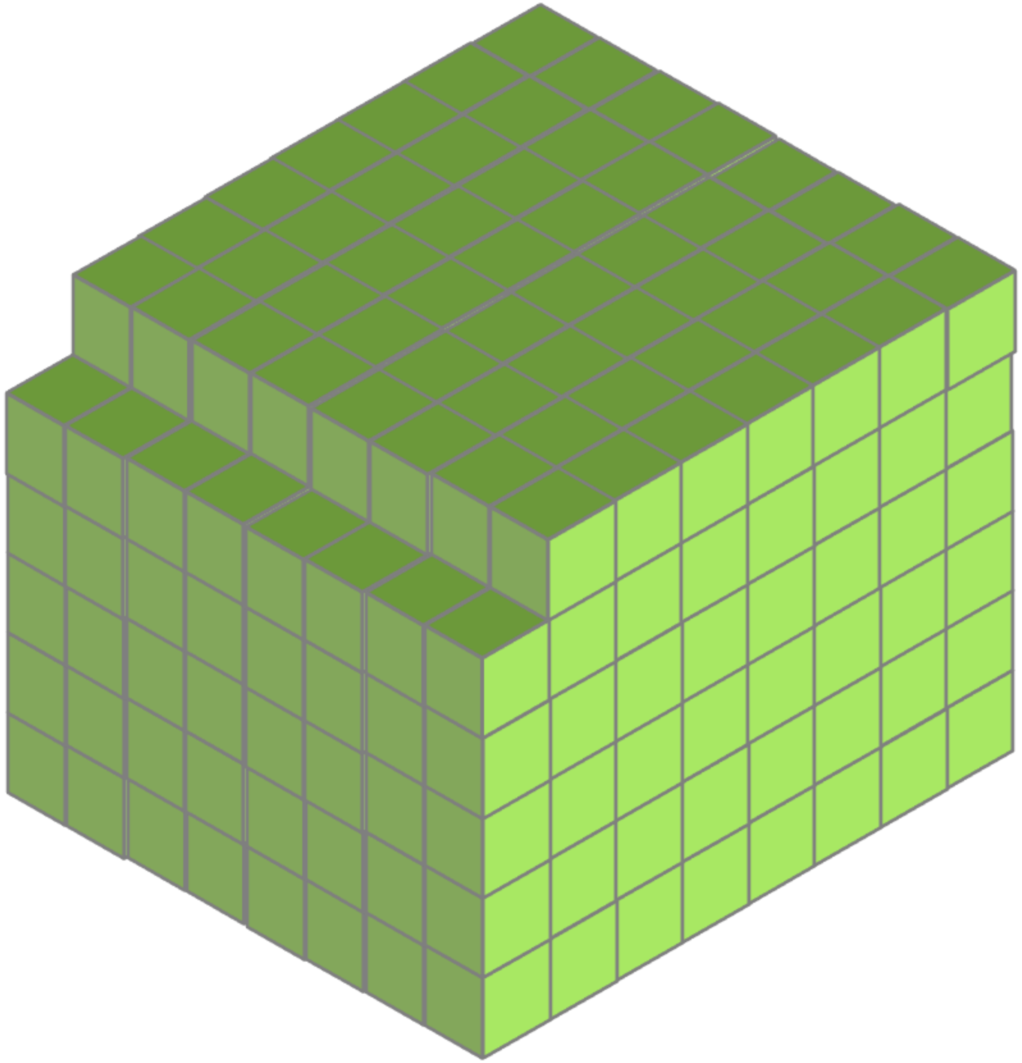




Deep layer feature-map

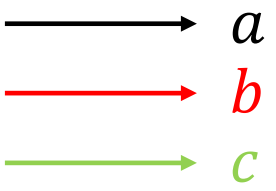
Two important intuitions about feature maps

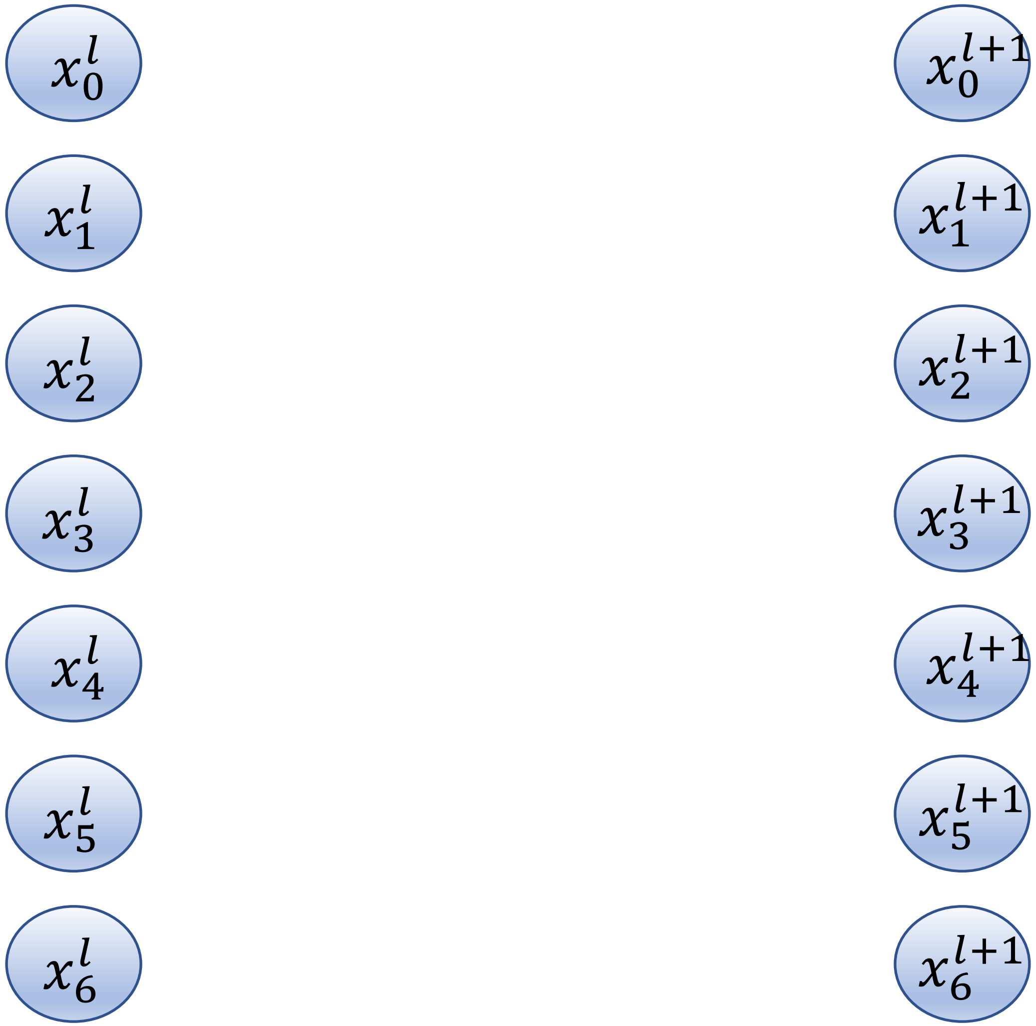




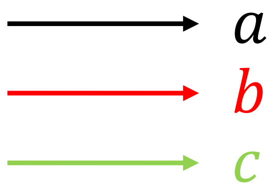
Deep layer feature-map

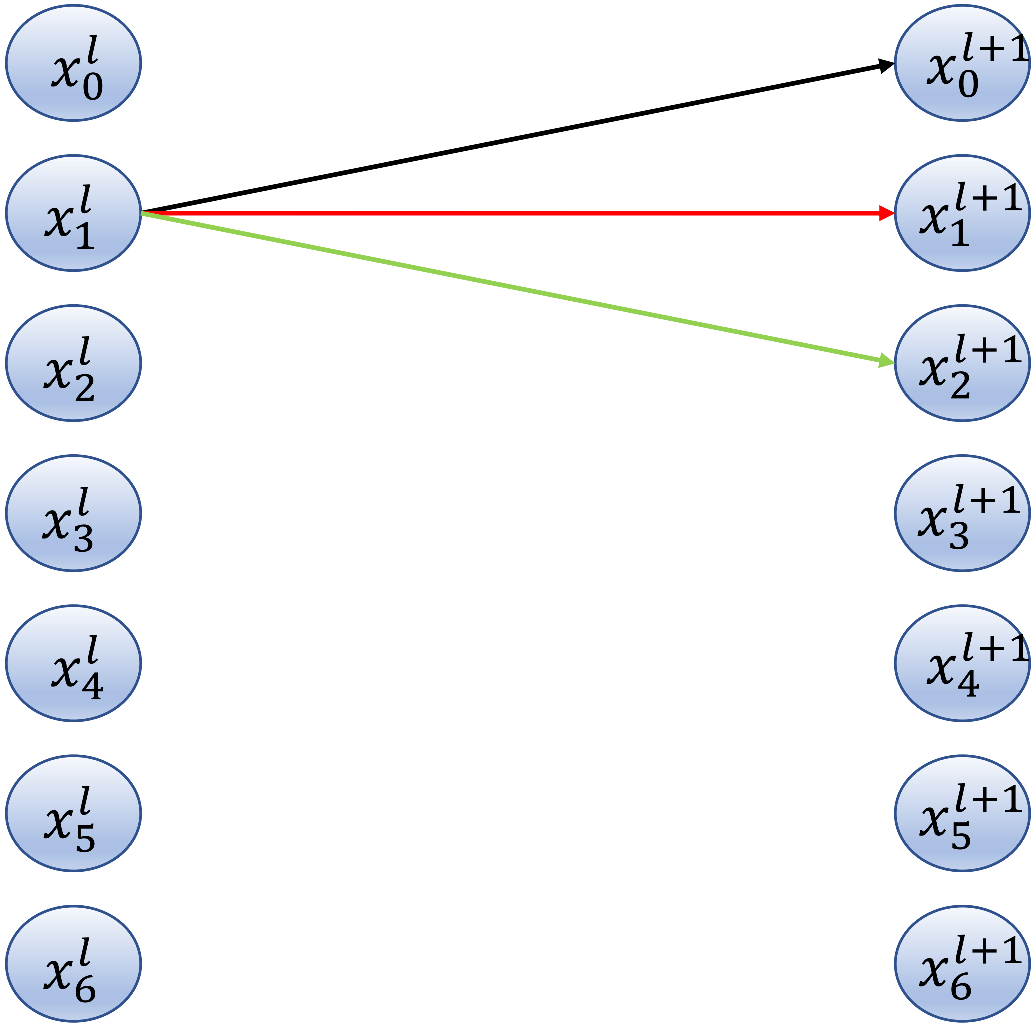
Transposed Convolution



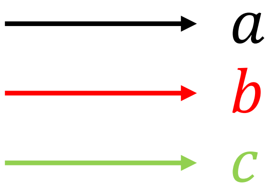


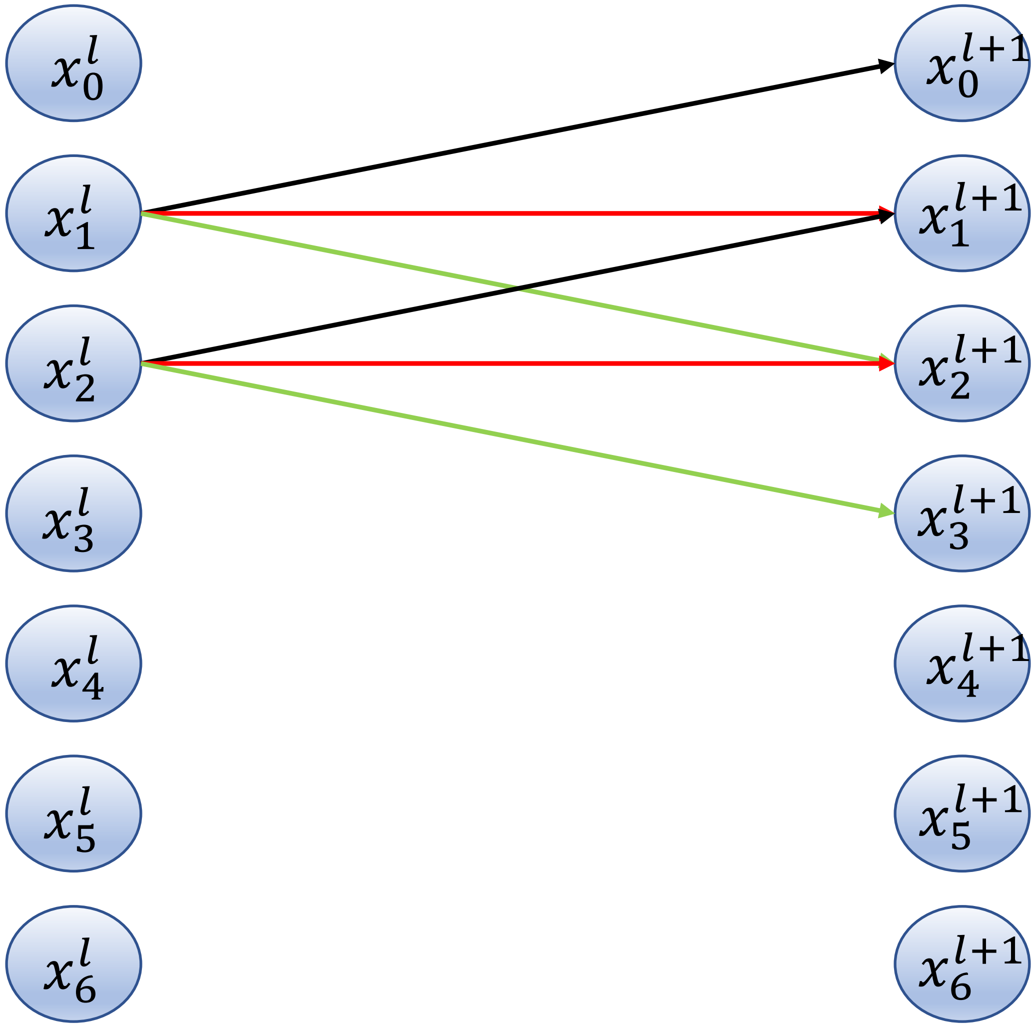
Transposed Convolution



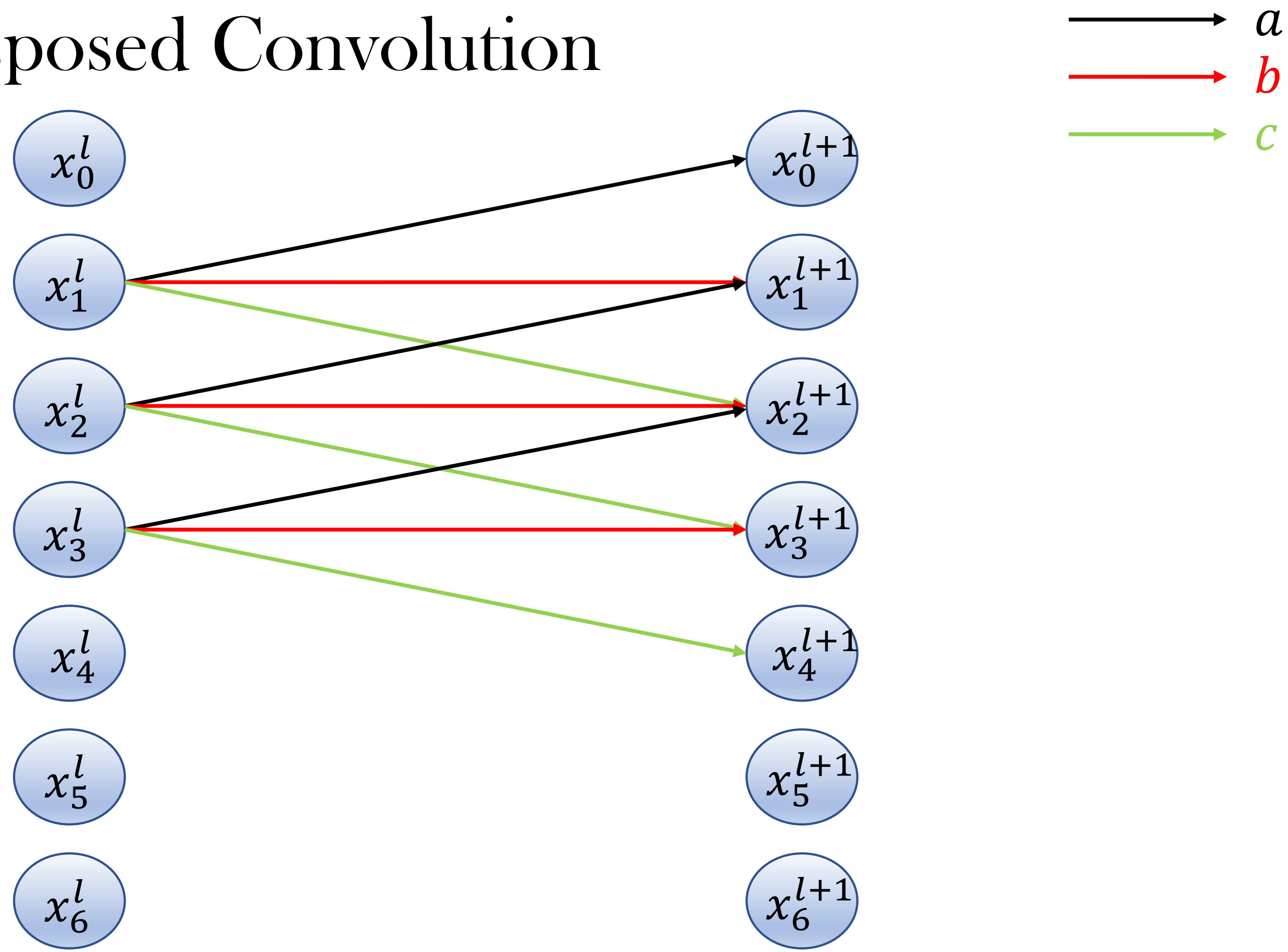


Transposed Convolution

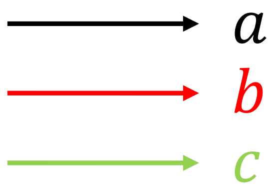


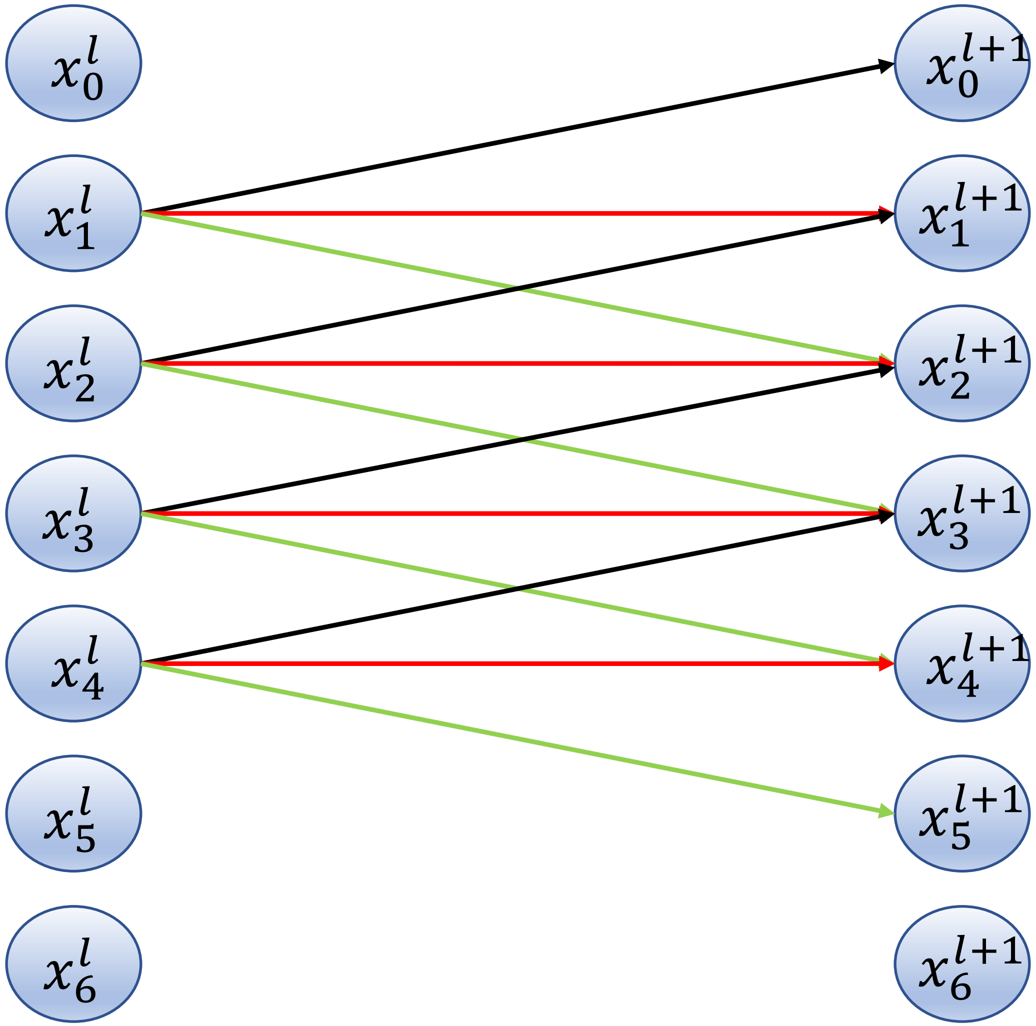


Transposed Convolution

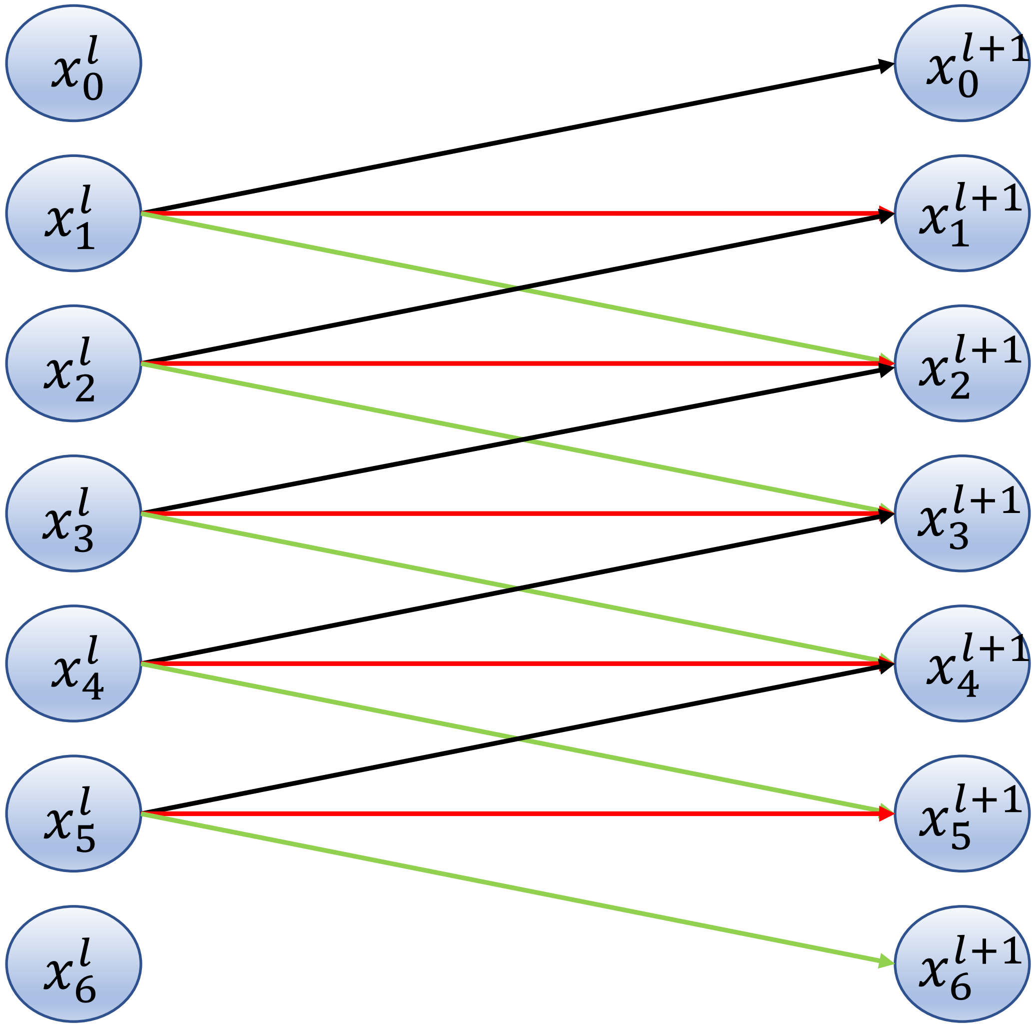


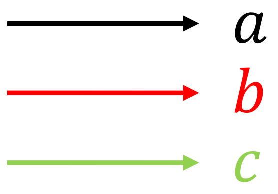
Transposed Convolution



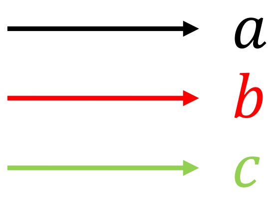


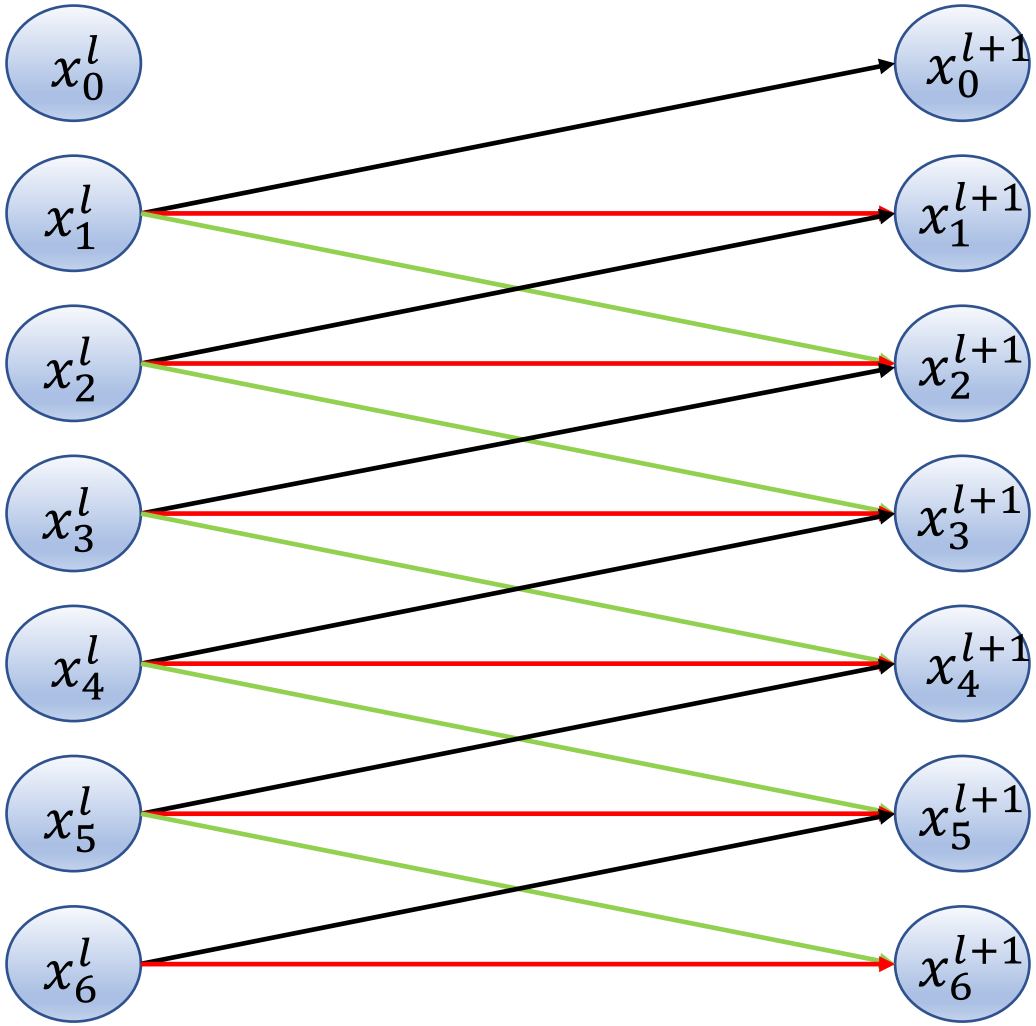
Transposed Convolution



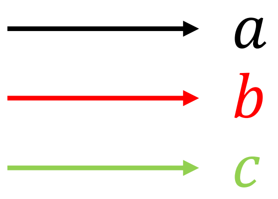


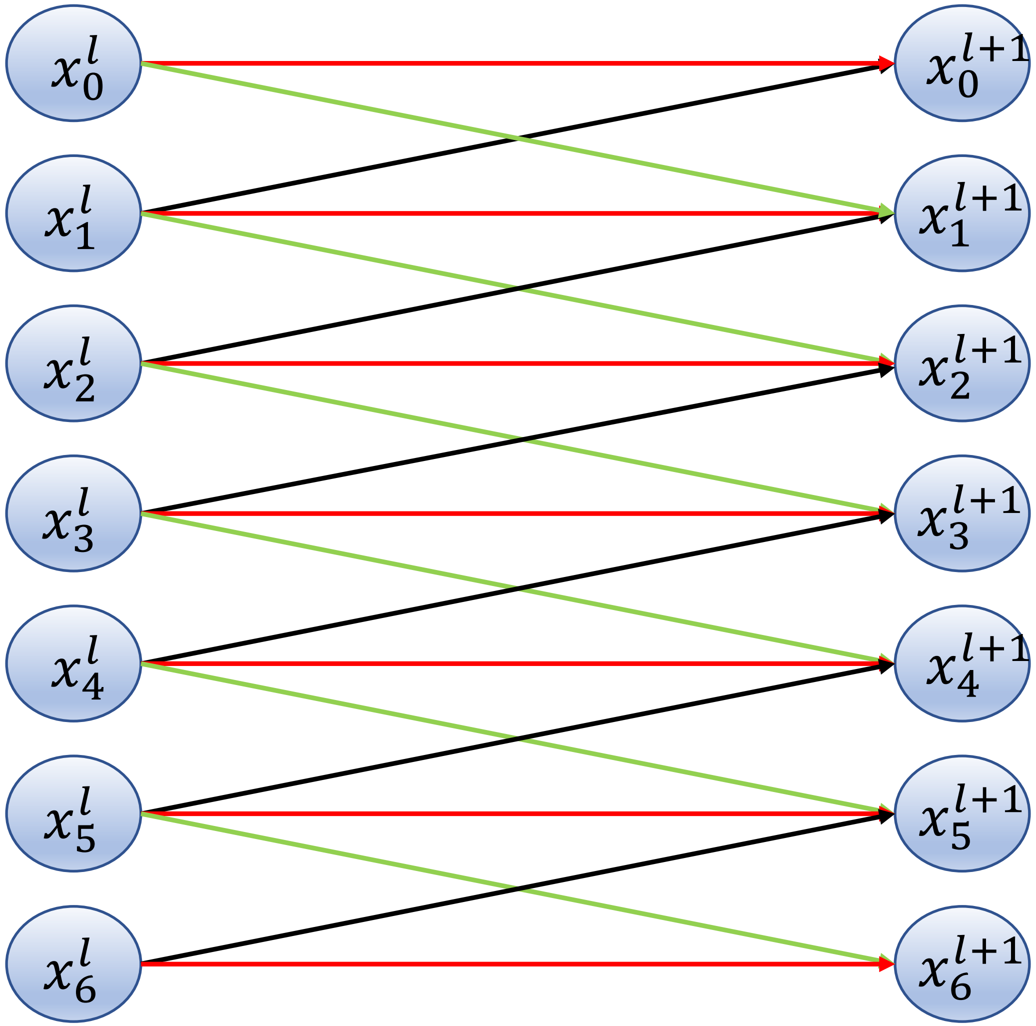
Transposed Convolution



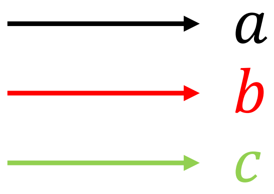


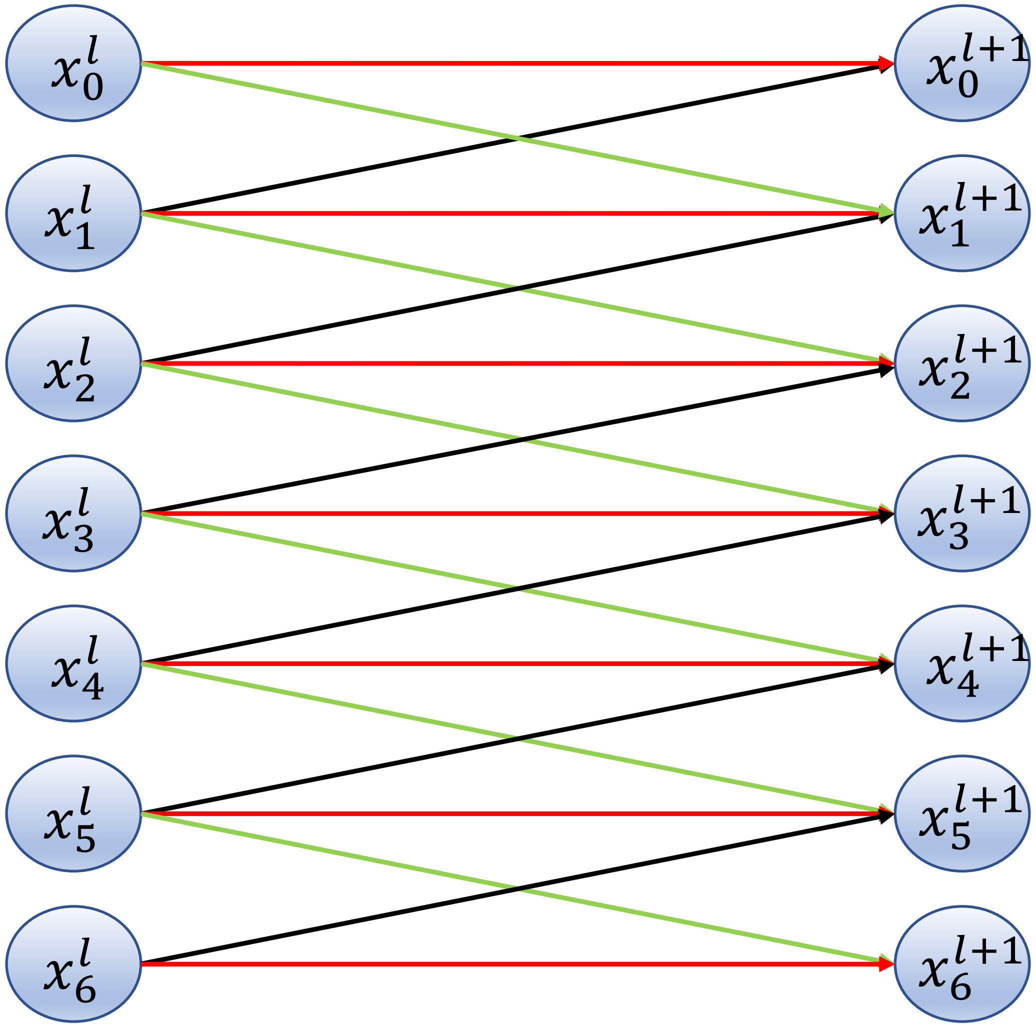
Transposed Convolution





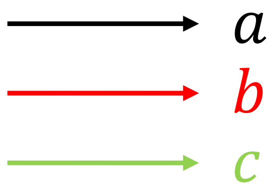
Transposed Convolution

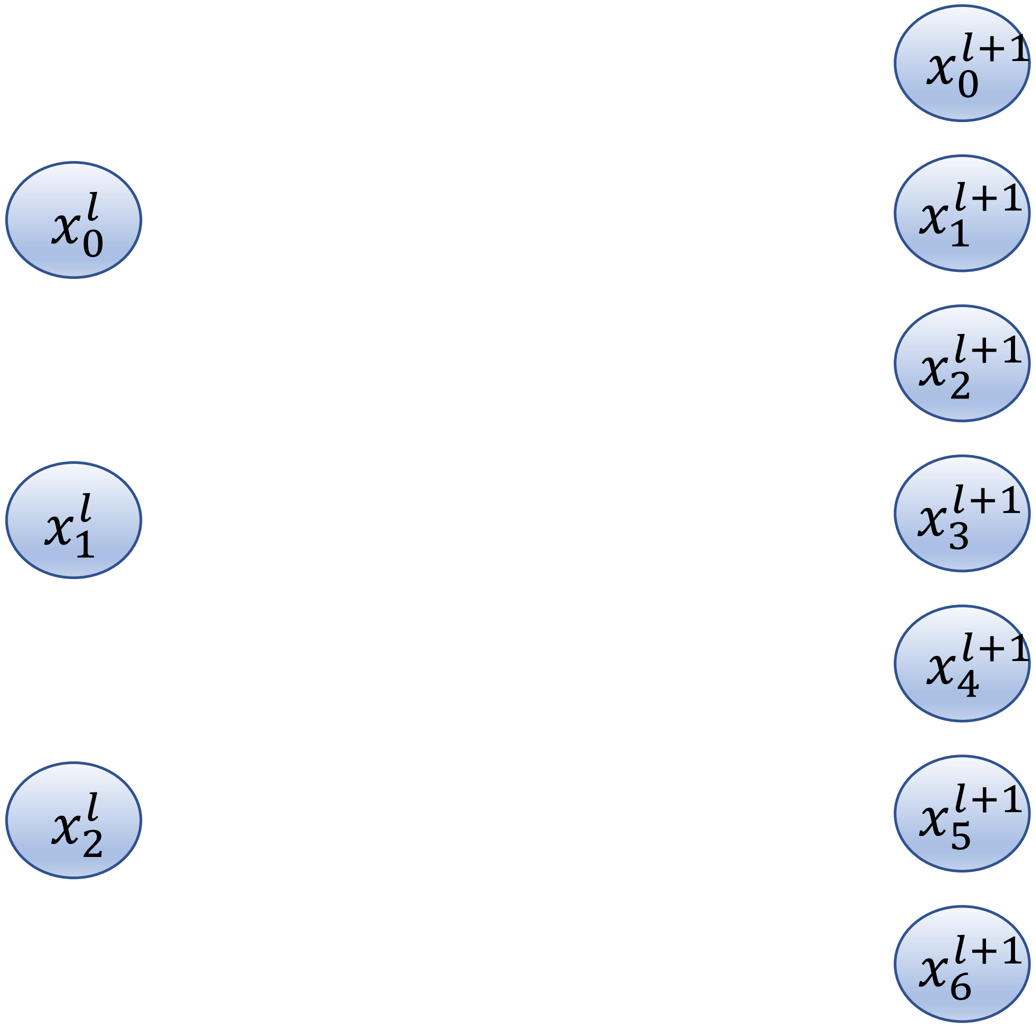




$$\begin{bmatrix} b & a & 0 & 0 & 0 & 0 & 0 \\ c & b & a & 0 & 0 & 0 & 0 \\ 0 & c & b & a & 0 & 0 & 0 \\ 0 & a & c & b & a & 0 & 0 \\ 0 & 0 & a & c & b & a & 0 \\ 0 & 0 & 0 & 0 & c & b & a \\ 0 & 0 & 0 & 0 & 0 & c & b \end{bmatrix}$$

Transposed Convolution with stride





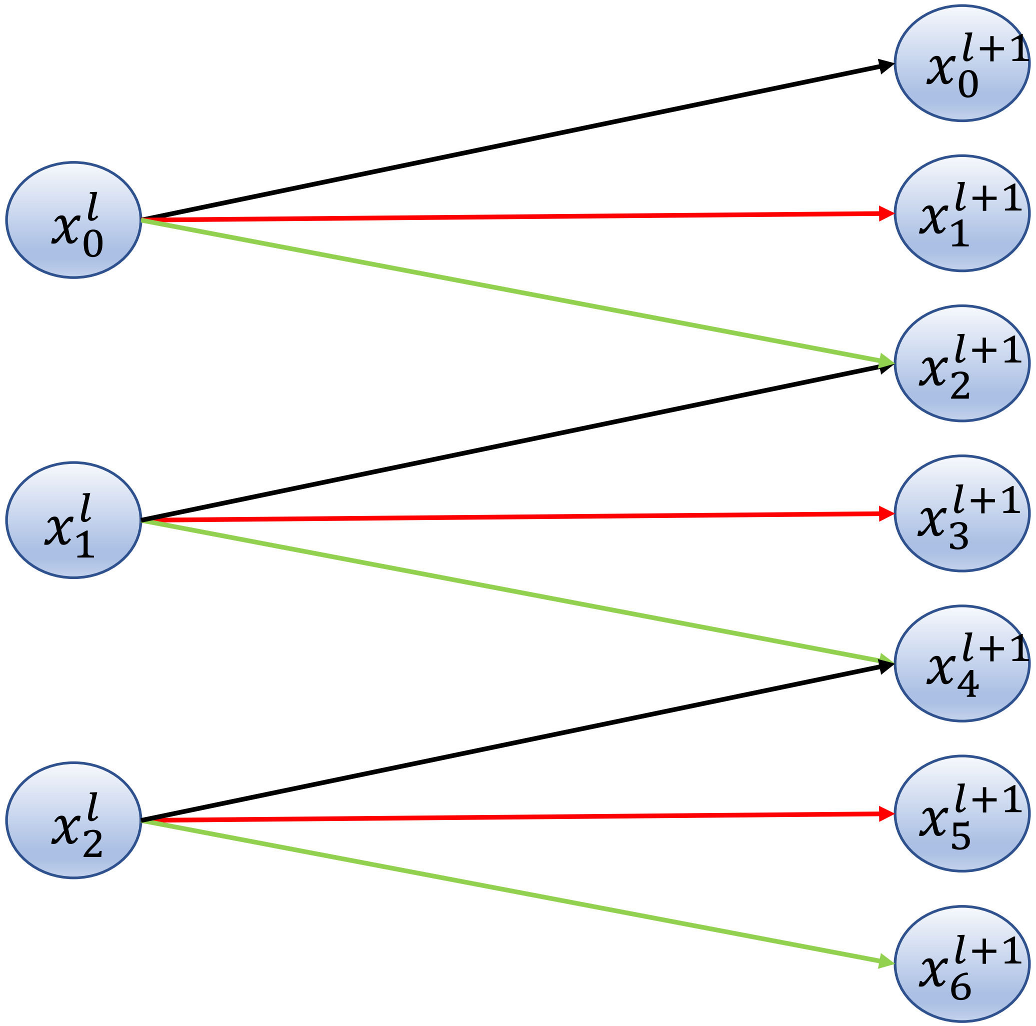
$$\begin{bmatrix} b & a & 0 & 0 & 0 & 0 & 0 \\ c & b & a & 0 & 0 & 0 & 0 \\ 0 & c & b & a & 0 & 0 & 0 \\ 0 & a & c & b & a & 0 & 0 \\ 0 & 0 & a & c & b & a & 0 \\ 0 & 0 & 0 & 0 & c & b & a \\ 0 & 0 & 0 & 0 & 0 & c & b \end{bmatrix}$$

Transposed Convolution with stride

$\longrightarrow a$

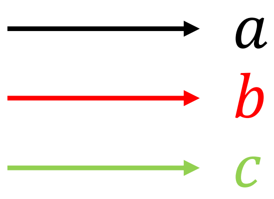
$\longrightarrow b$

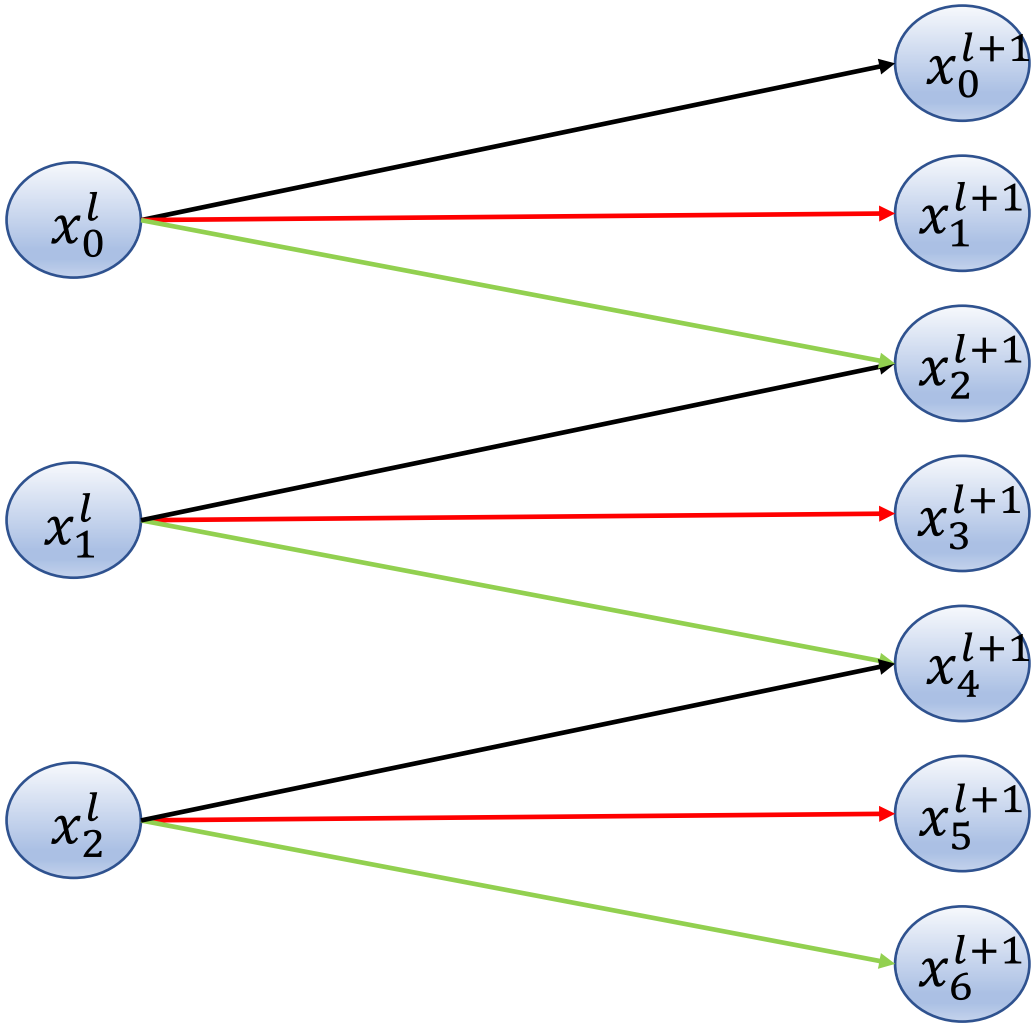
$\longrightarrow c$



$$\begin{bmatrix} b & a & 0 & 0 & 0 & 0 & 0 \\ c & b & a & 0 & 0 & 0 & 0 \\ 0 & c & b & a & 0 & 0 & 0 \\ 0 & a & c & b & a & 0 & 0 \\ 0 & 0 & a & c & b & a & 0 \\ 0 & 0 & 0 & 0 & c & b & a \\ 0 & 0 & 0 & 0 & 0 & c & b \end{bmatrix}$$

Transposed Convolution with stride





$$
\begin{bmatrix}
b & a & 0 & 0 & 0 & 0 & 0 \\
c & b & a & 0 & 0 & 0 & 0 \\
0 & c & b & a & 0 & 0 & 0 \\
0 & a & c & b & a & 0 & 0 \\
0 & 0 & a & c & b & a & 0 \\
0 & 0 & 0 & 0 & c & b & a \\
0 & 0 & 0 & 0 & 0 & c & b
\end{bmatrix}
$$

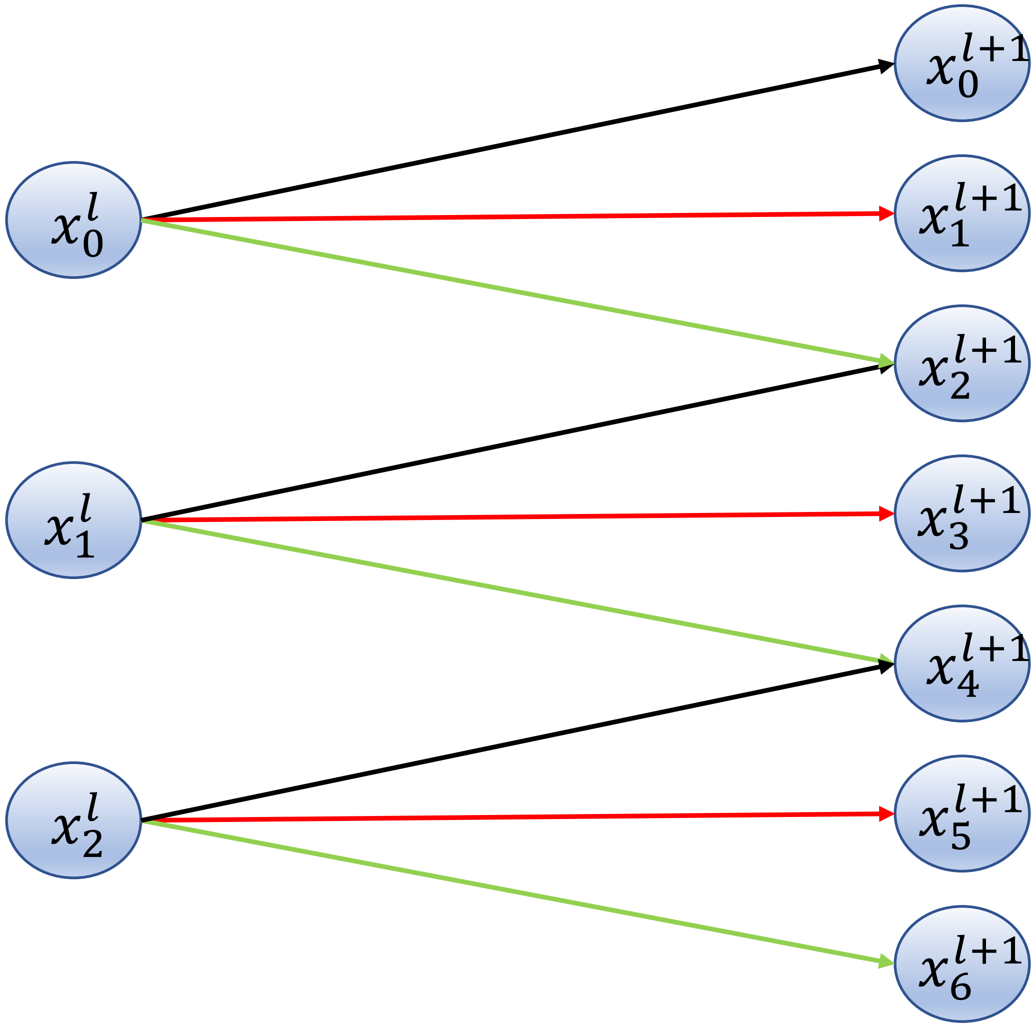
Transposed Convolution with stride

[redacted]

$\rightarrow a$

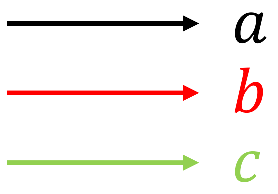
$\rightarrow b$

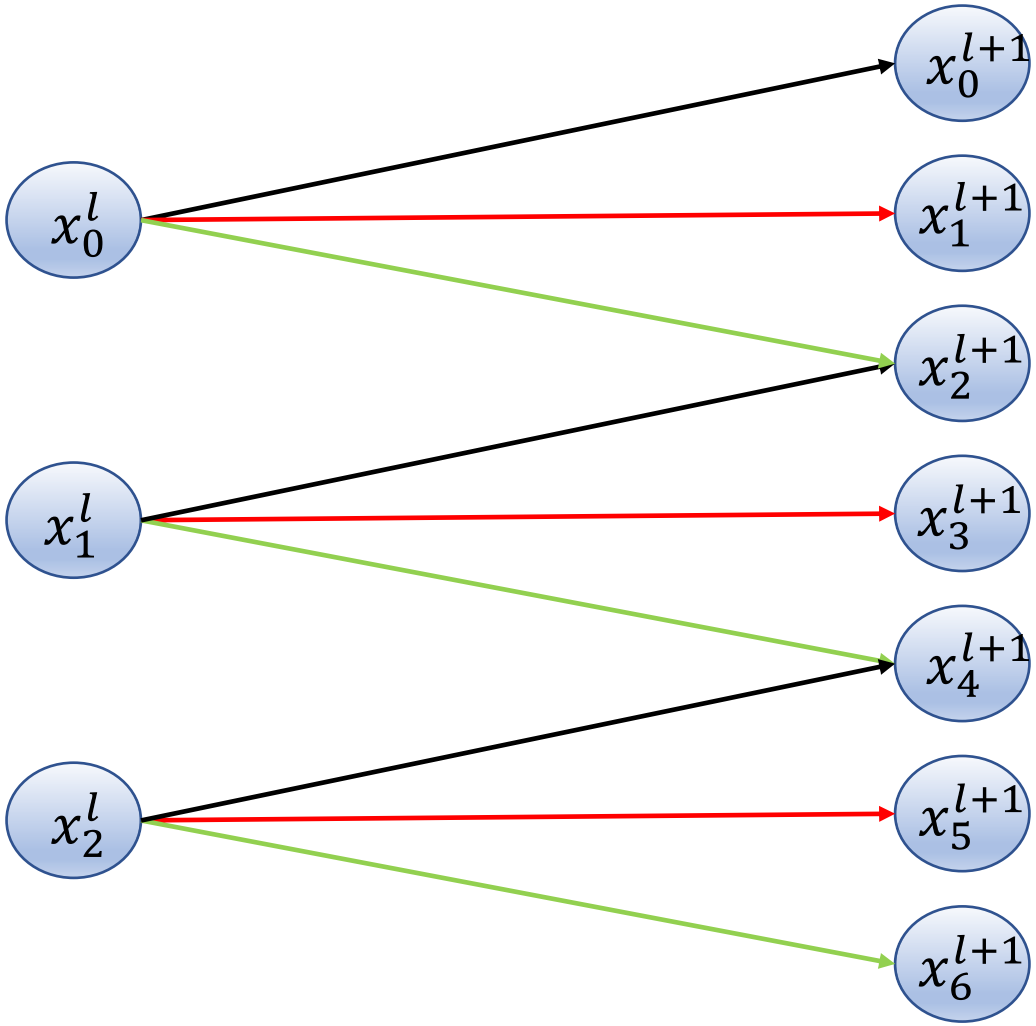
$\rightarrow c$



$$\begin{bmatrix} \boldsymbol{a} & 0 & 0 \\ \boldsymbol{b} & 0 & 0 \\ \boldsymbol{c} & \boldsymbol{a} & 0 \\ a & \boldsymbol{b} & 0 \\ 0 & \boldsymbol{c} & \boldsymbol{a} \\ 0 & 0 & \boldsymbol{b} \\ 0 & 0 & \boldsymbol{c} \end{bmatrix}$$

Transposed Convolution with stride





$$\begin{bmatrix} \boldsymbol{a} & 0 & 0 \\ \boldsymbol{b} & 0 & 0 \\ \boldsymbol{c} & \boldsymbol{a} & 0 \\ a & \boldsymbol{b} & 0 \\ 0 & \boldsymbol{c} & \boldsymbol{a} \\ 0 & 0 & \boldsymbol{b} \\ 0 & 0 & \boldsymbol{c} \end{bmatrix}$$

Recall- stride 2 conv:

$$\begin{bmatrix} \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} \end{bmatrix}$$

Transposed Convolution with stride

$\longrightarrow a$

$\longrightarrow b$

$\longrightarrow c$

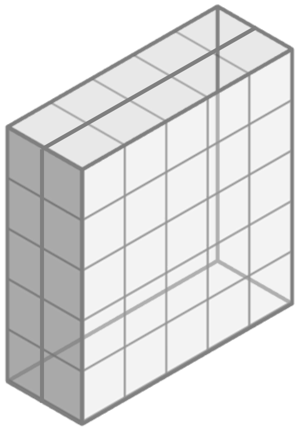
$$\begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & a & 0 \\ a & b & 0 \\ 0 & c & a \\ 0 & 0 & b \\ 0 & 0 & c \end{bmatrix}$$

Recall- stride 2 conv:

$$\begin{bmatrix} a & b & c & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & c & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c \end{bmatrix}$$

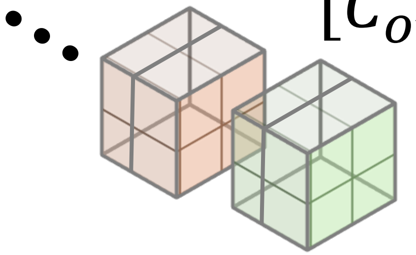
A note about the implementation of conv

A note about the implementation of conv

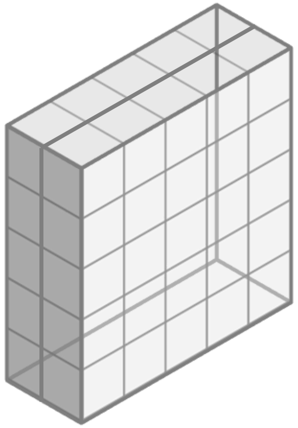


$[N, C_{in}, H, W]$

A note about the implementation of conv



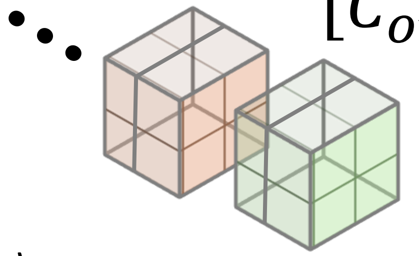
$[C_{out}, C_{in}, h, w]$



$[N, C_{in}, H, W]$

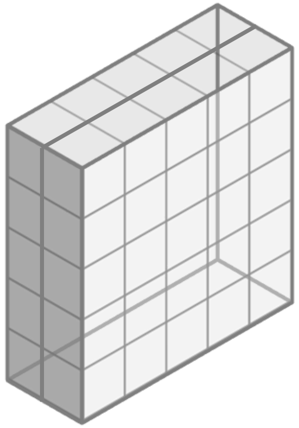
A note about the implementation of conv

$[C_{out}, C_{in}, h, w]$



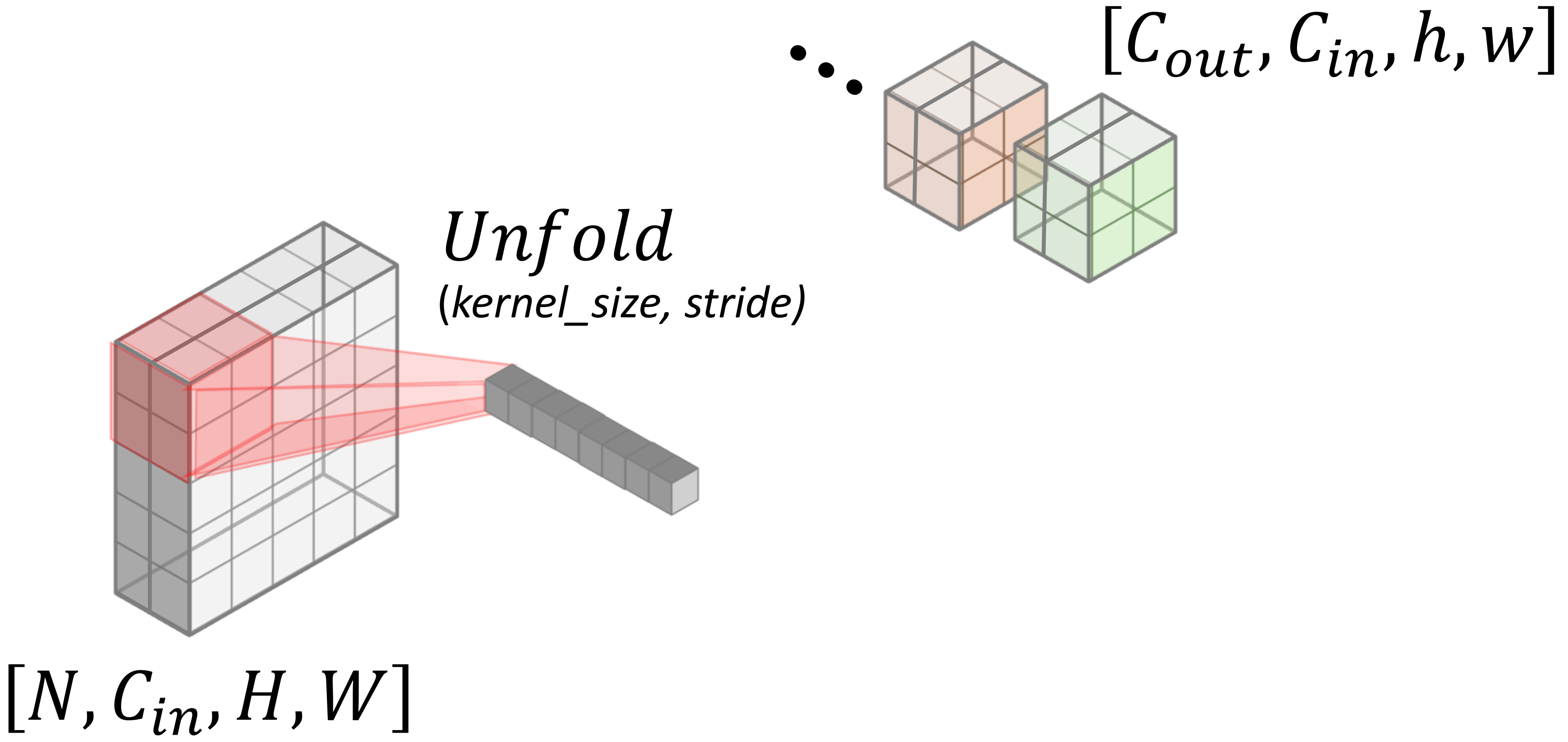
$Unfold$

(*kernel_size, stride)*

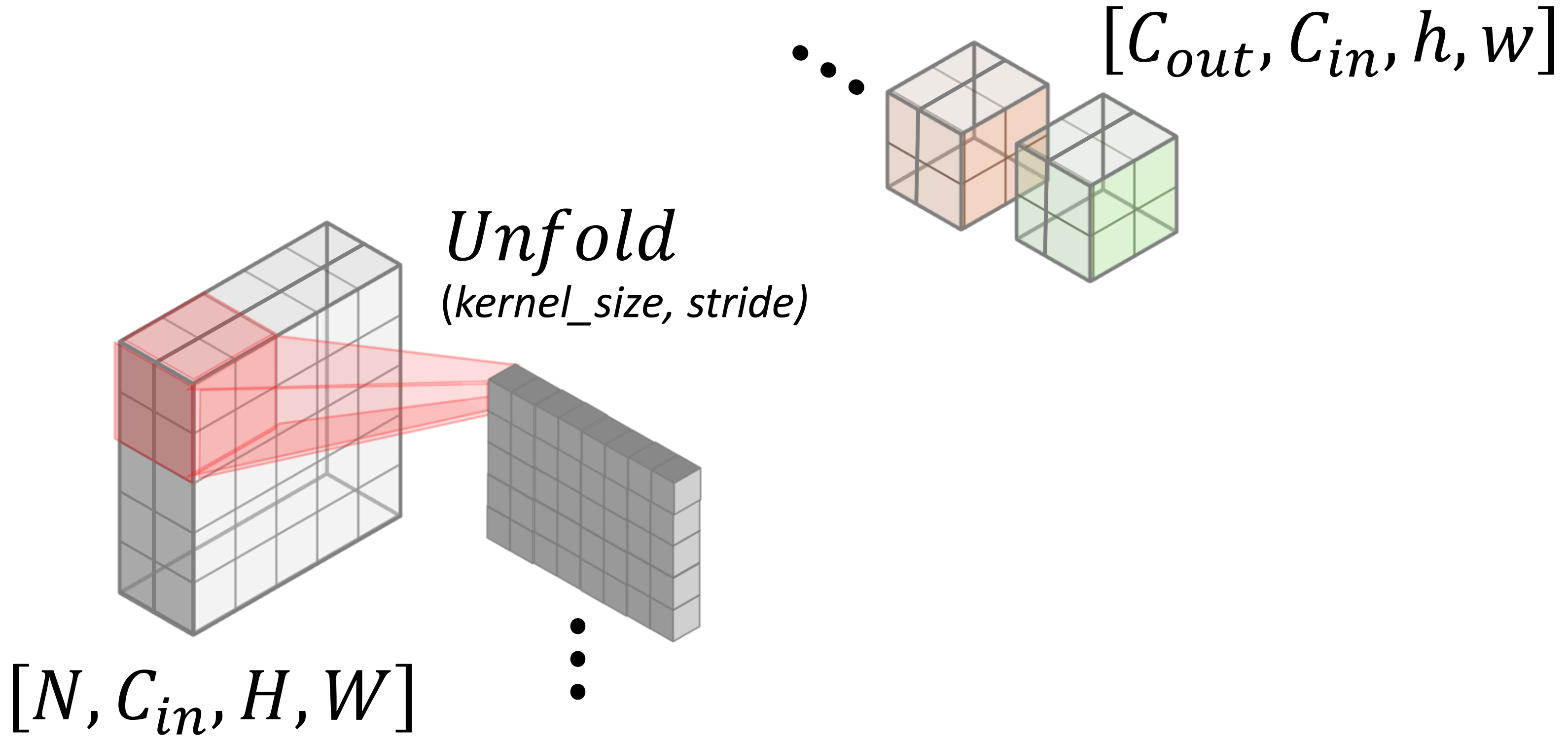


$[N, C_{in}, H, W]$

A note about the implementation of conv



A note about the implementation of conv

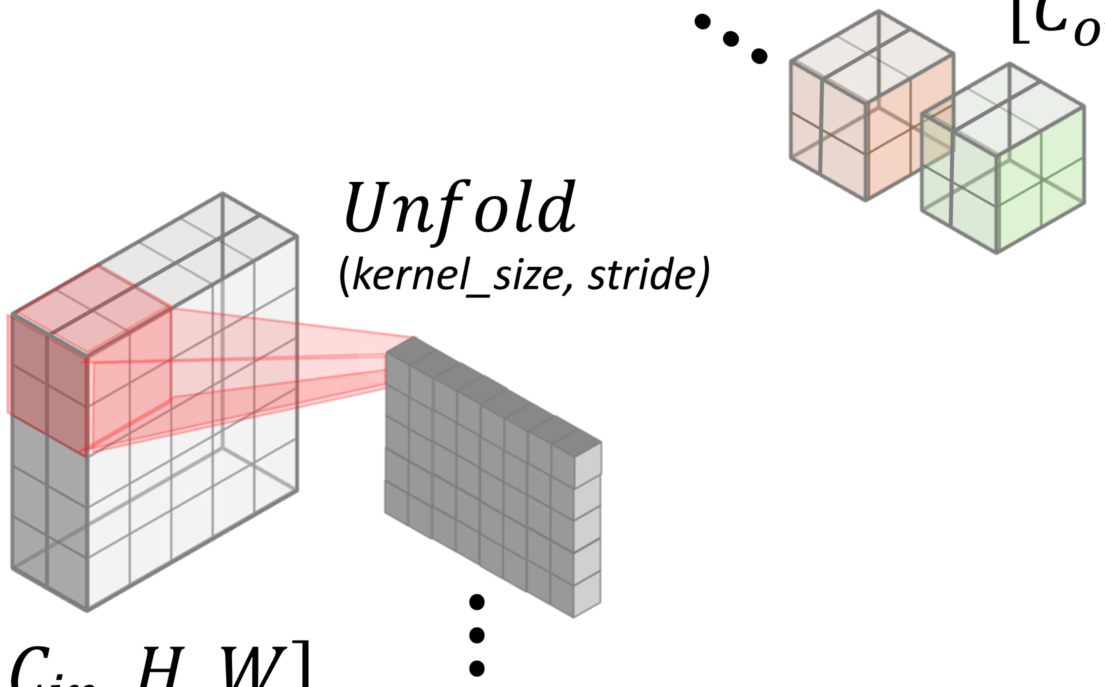


A note about the implementation of conv

$[C_{out}, C_{in}, h, w]$

$Unfold$

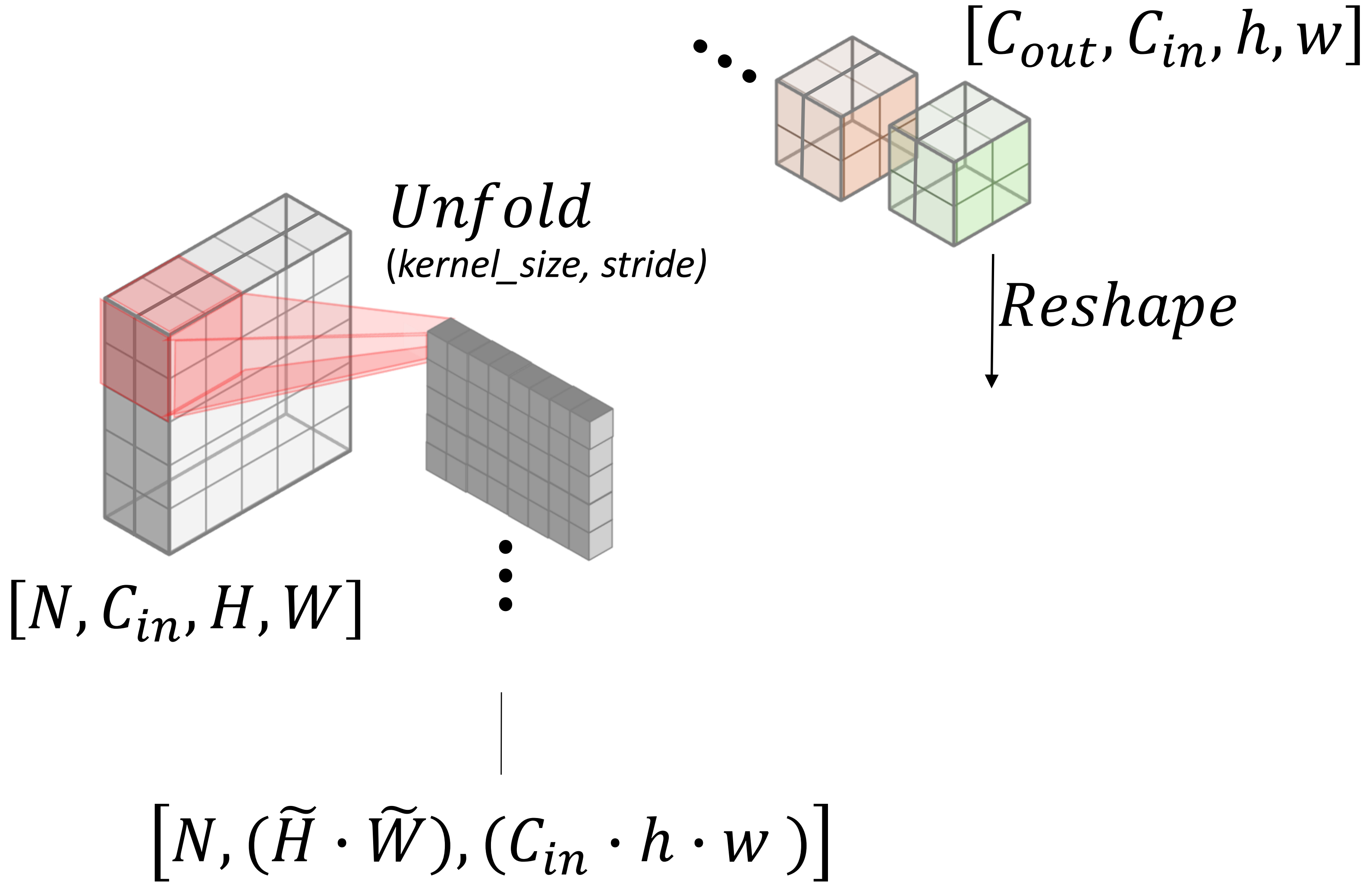
(*kernel_size, stride*)



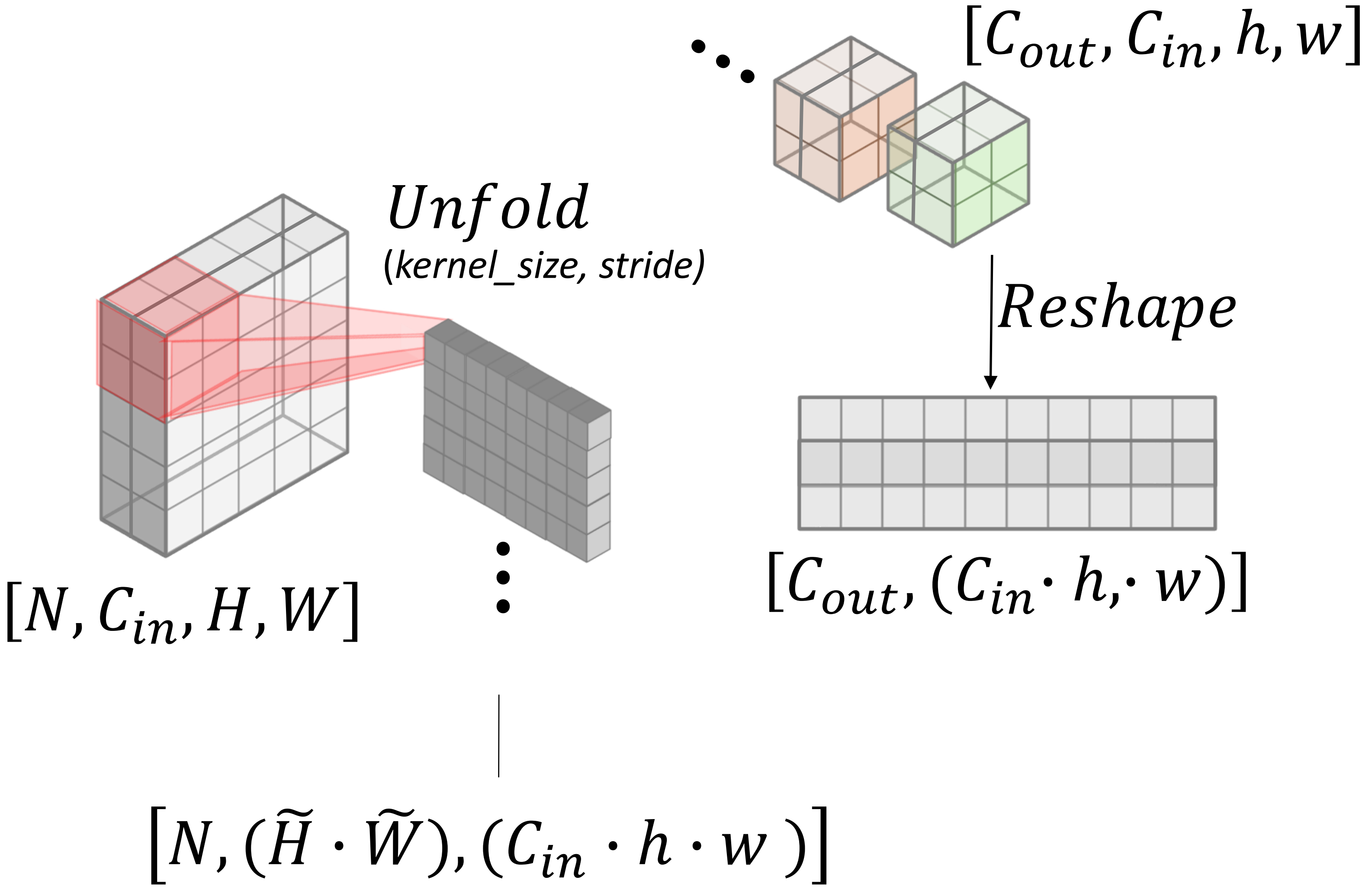
$[N, C_{in}, H, W]$

$[N, (\widetilde{H} \cdot \widetilde{W}), (C_{in} \cdot h \cdot w)]$

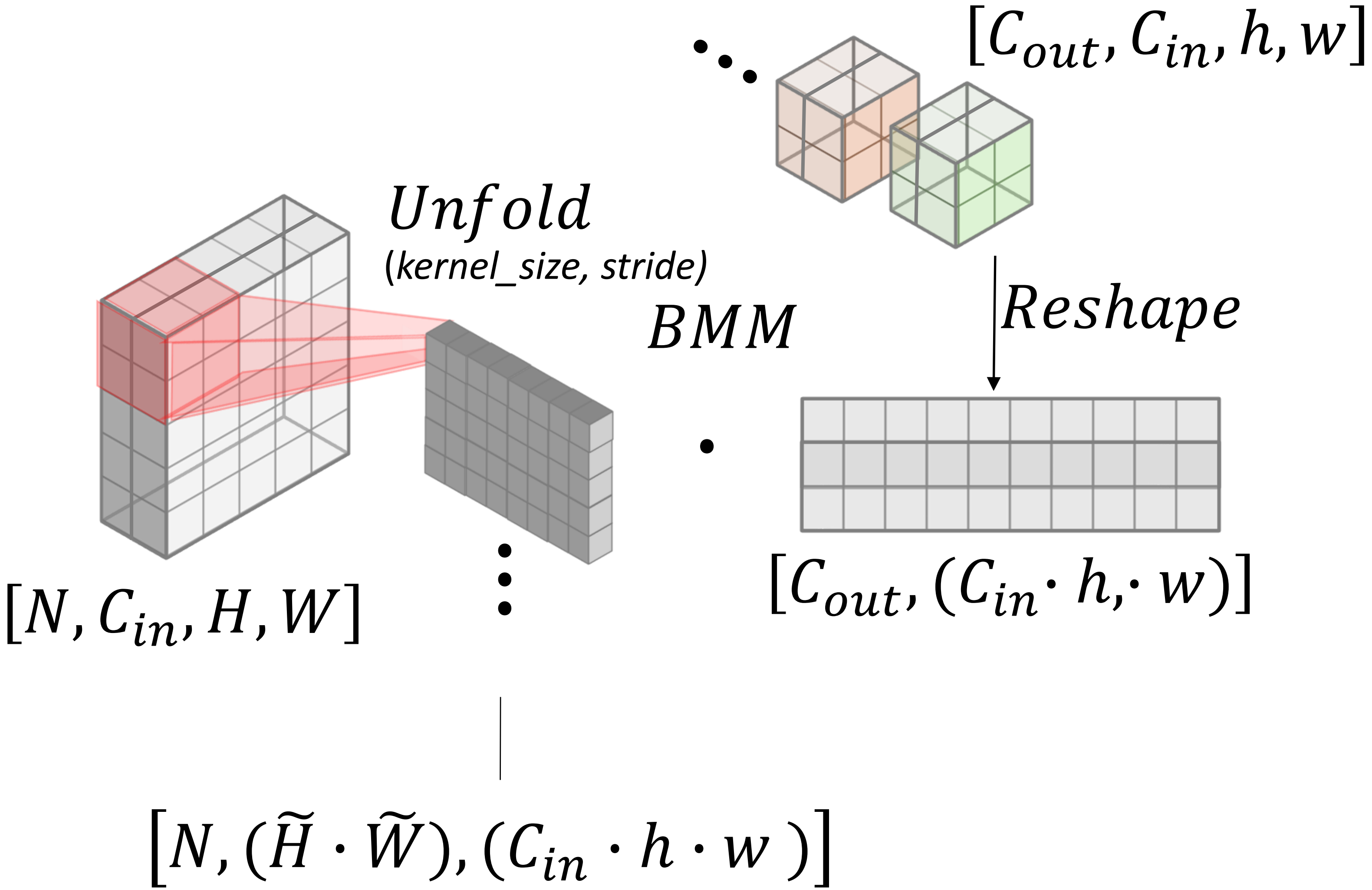
A note about the implementation of conv



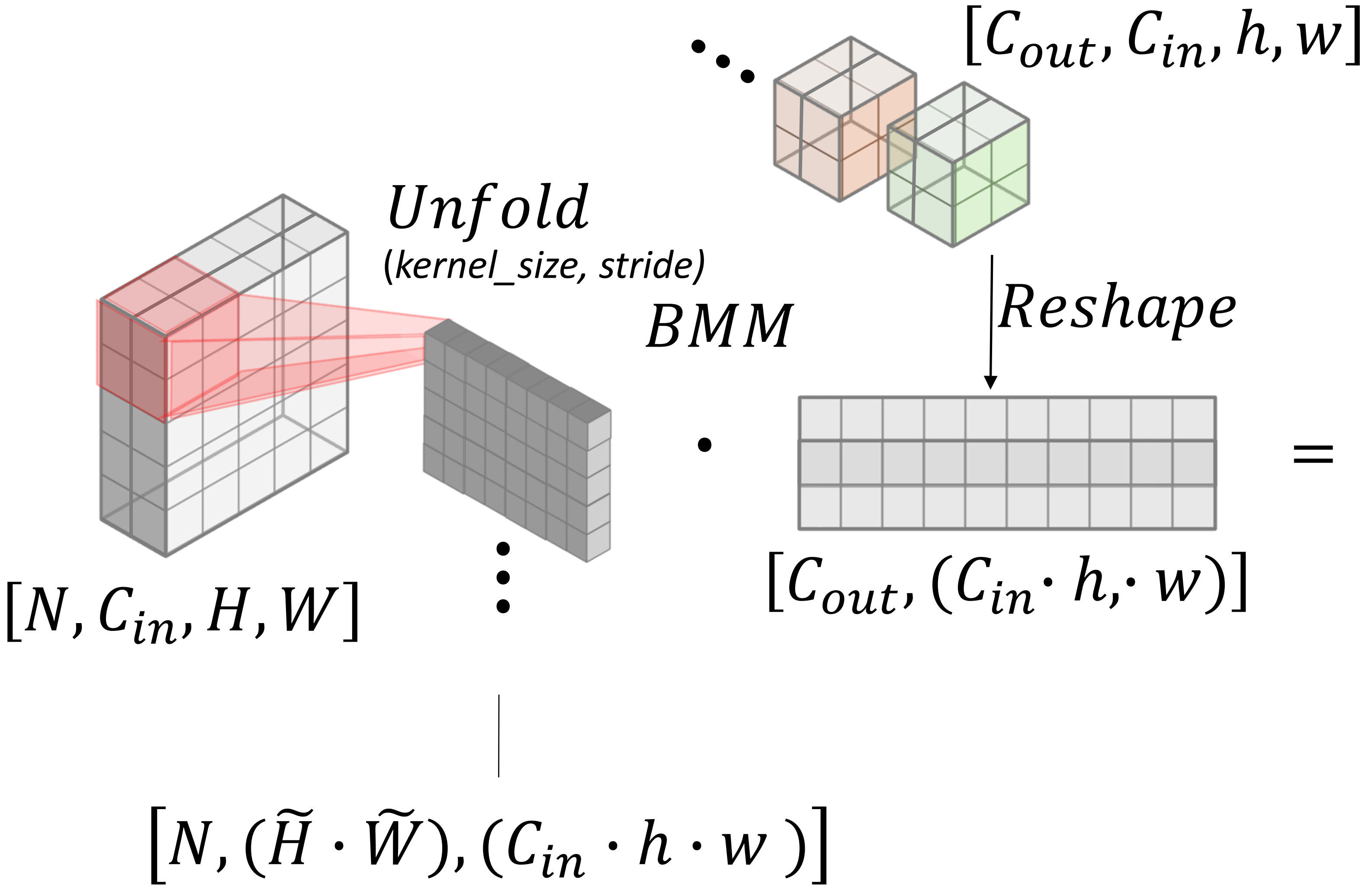
A note about the implementation of conv



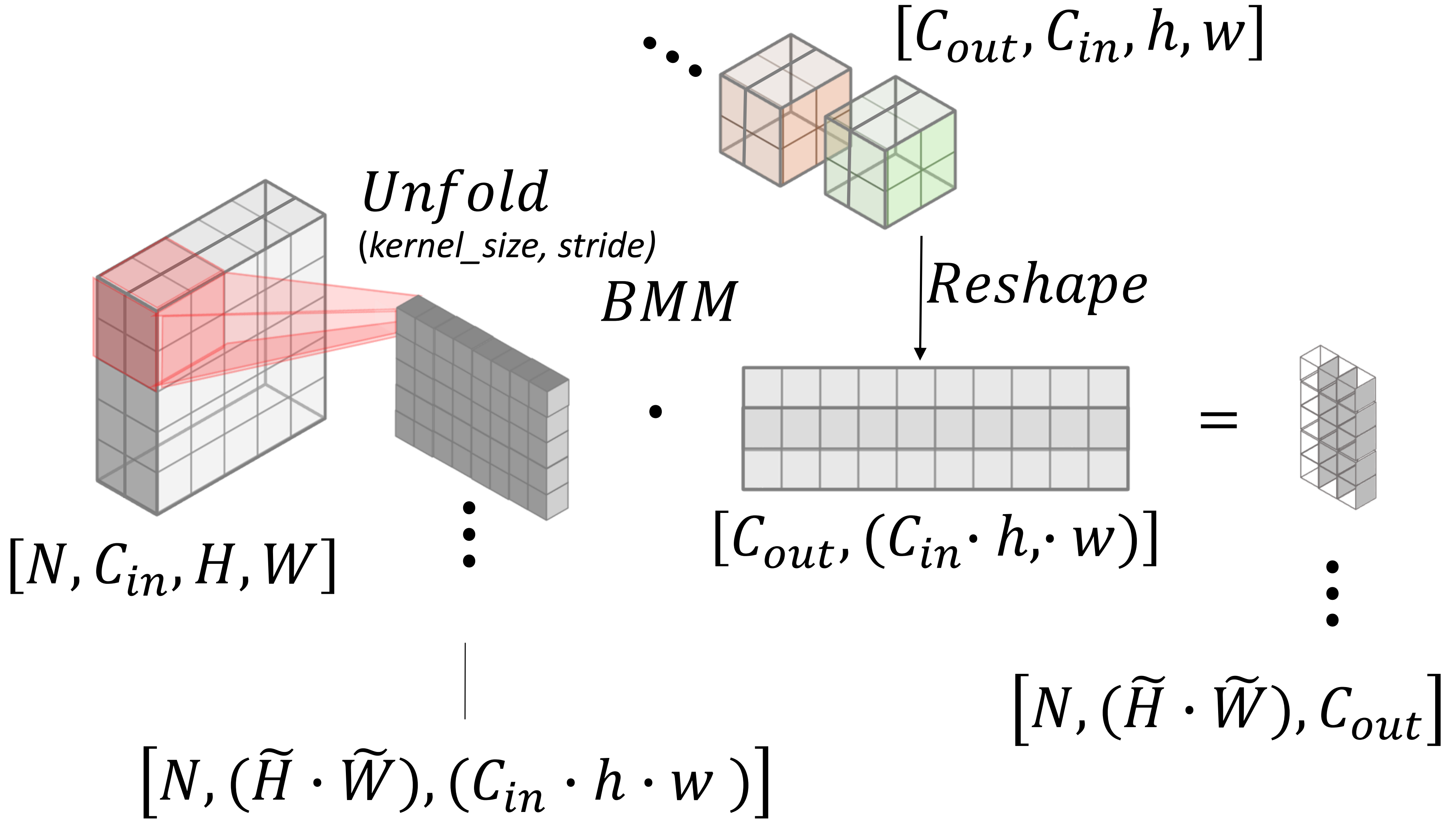
A note about the implementation of conv



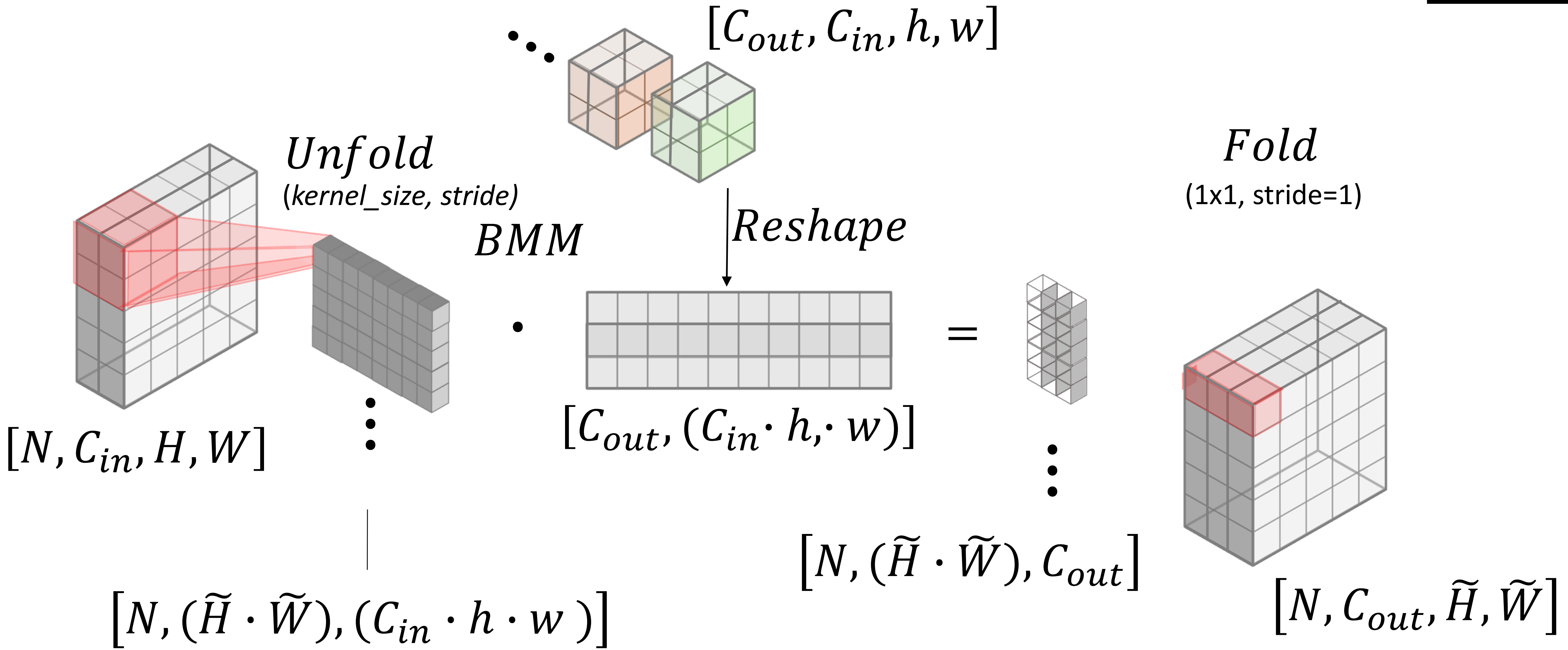
A note about the implementation of conv



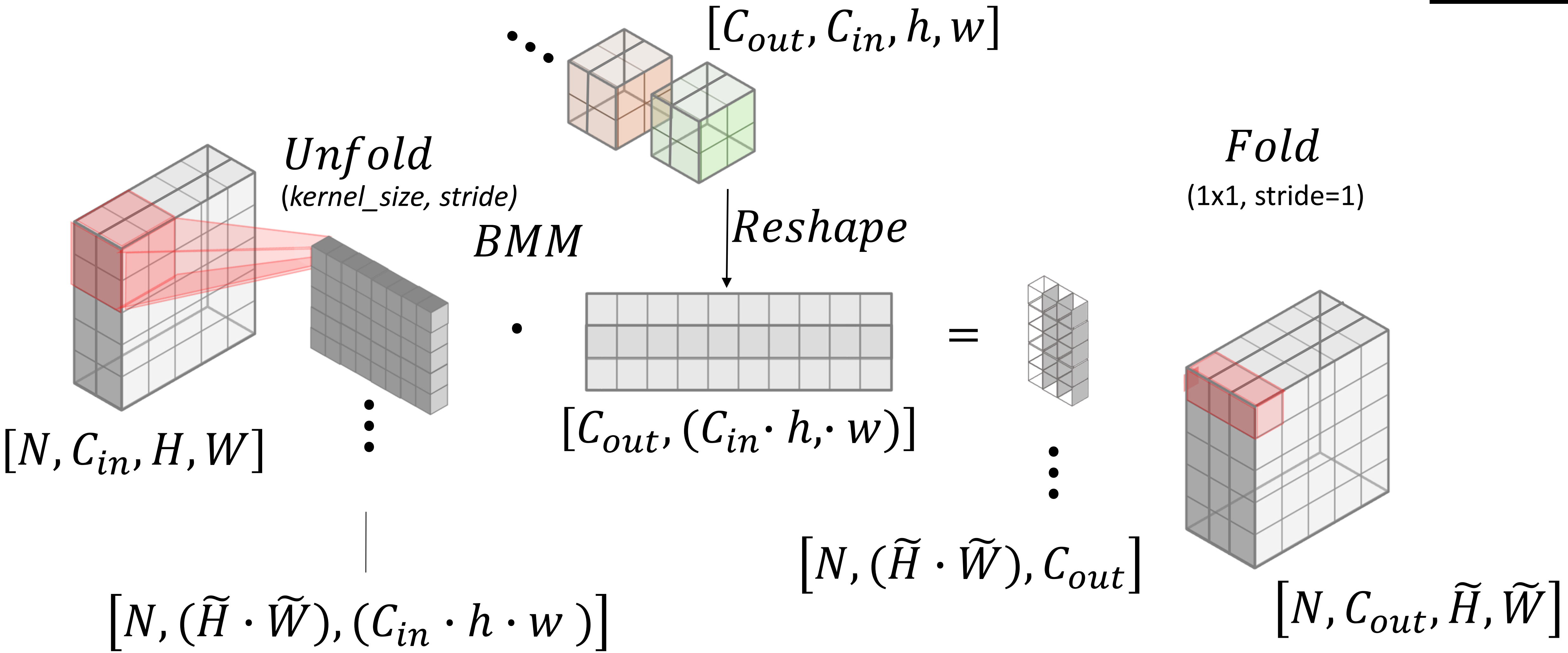
A note about the implementation of conv



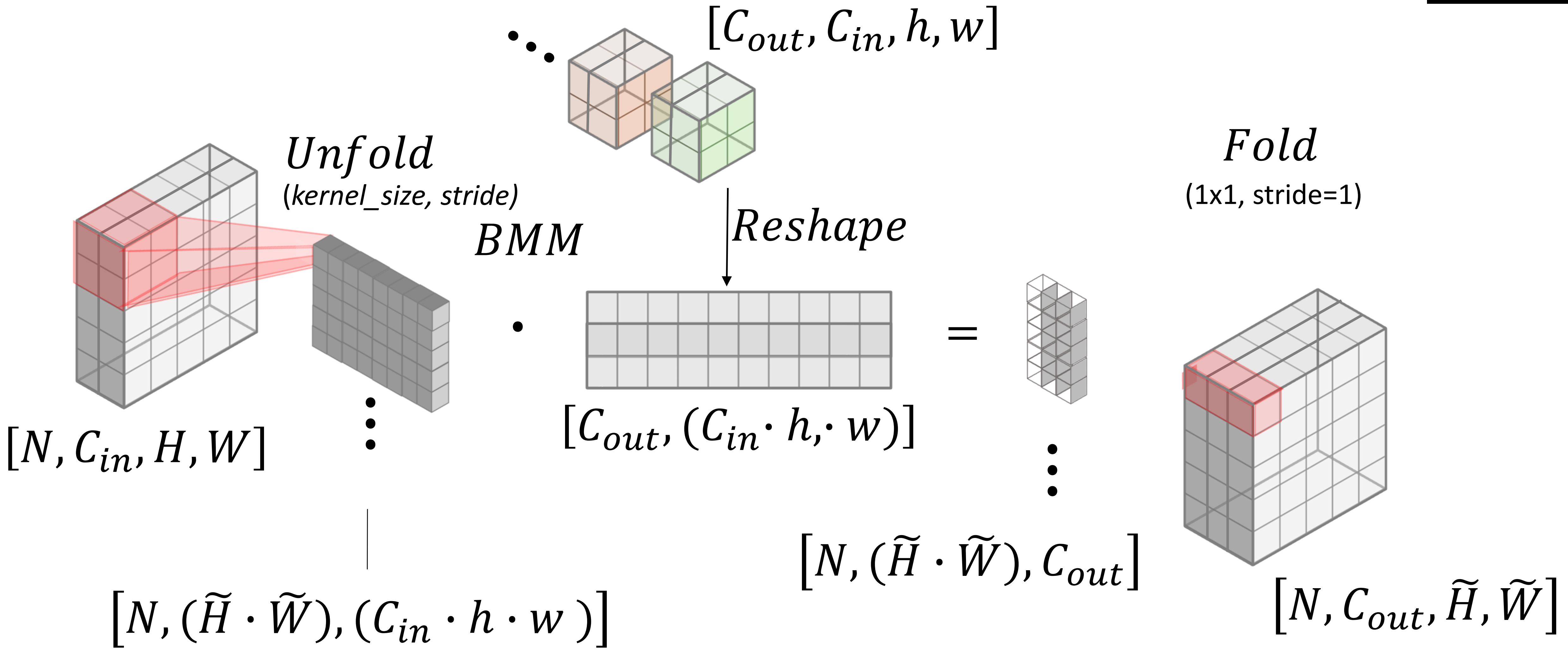
A note about the implementation of conv



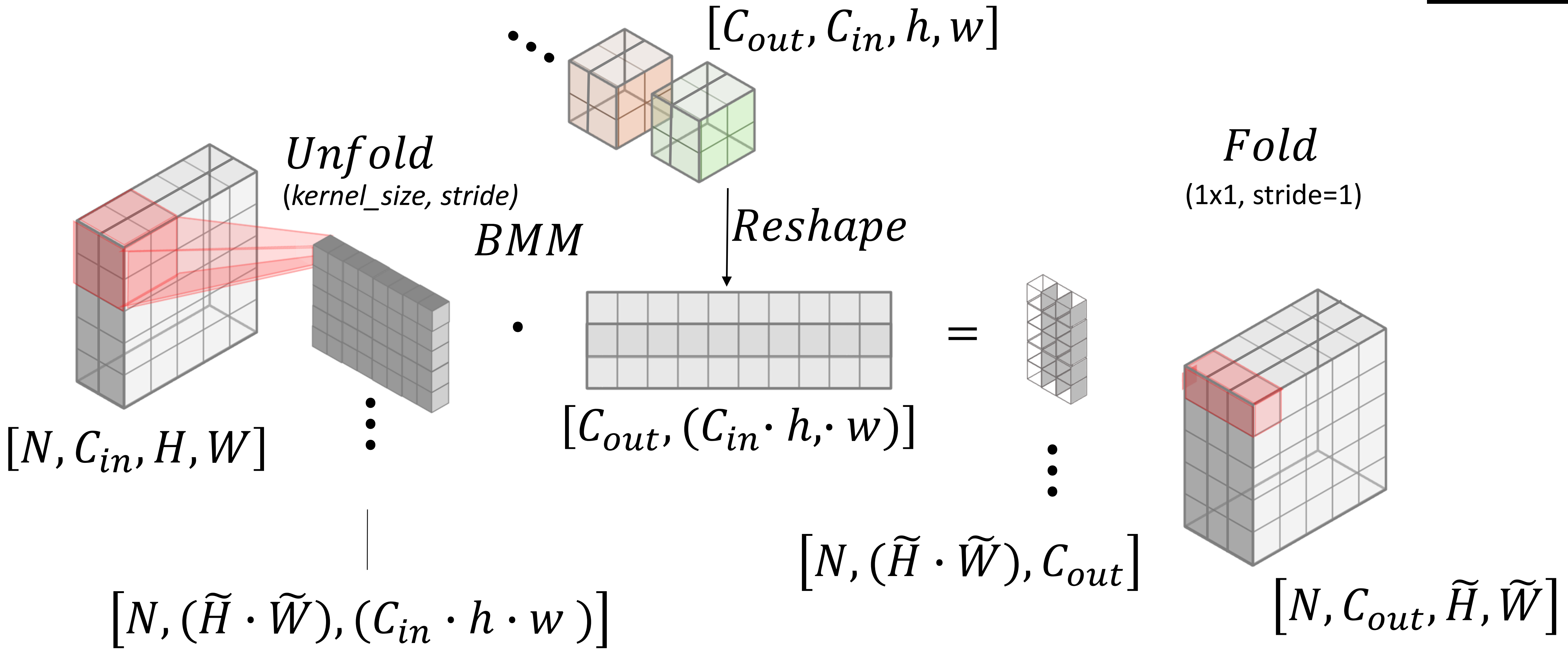
A note about the implementation of conv



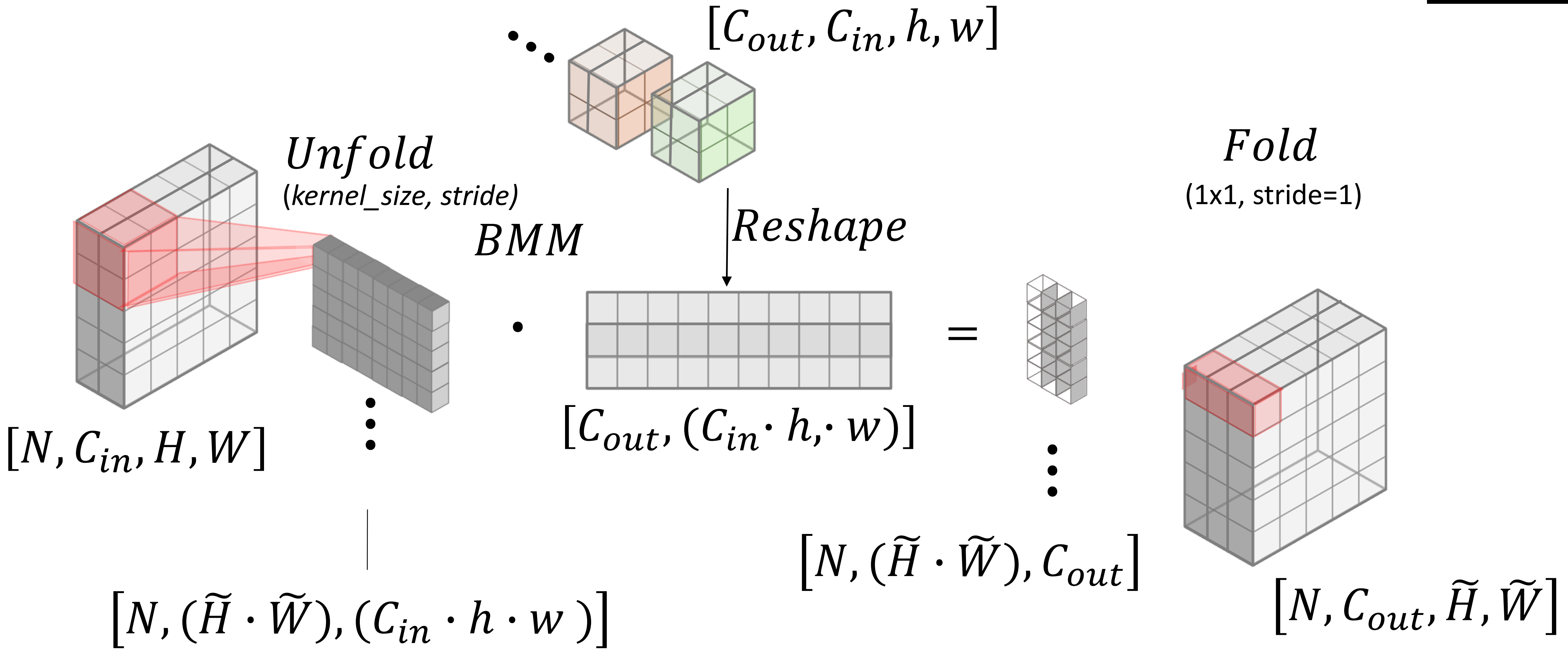
A note about the implementation of conv



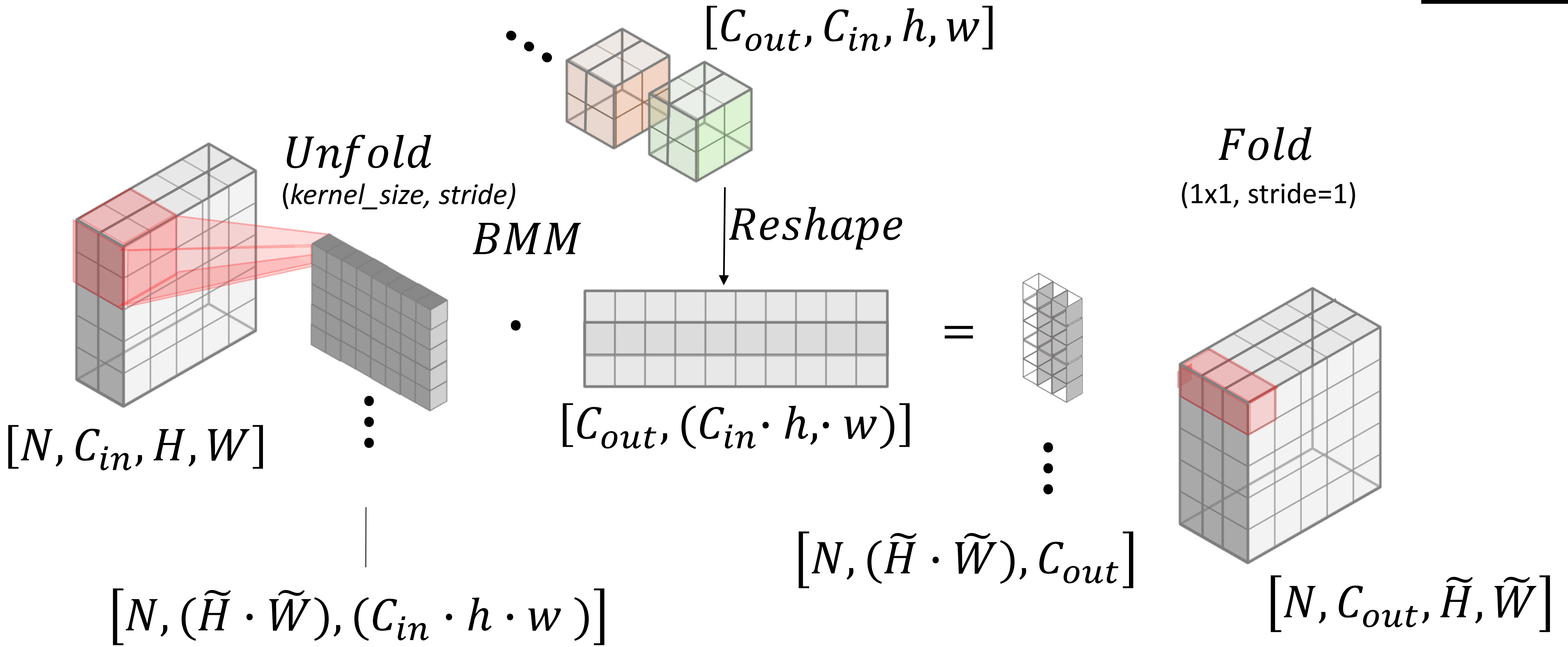
A note about the implementation of conv



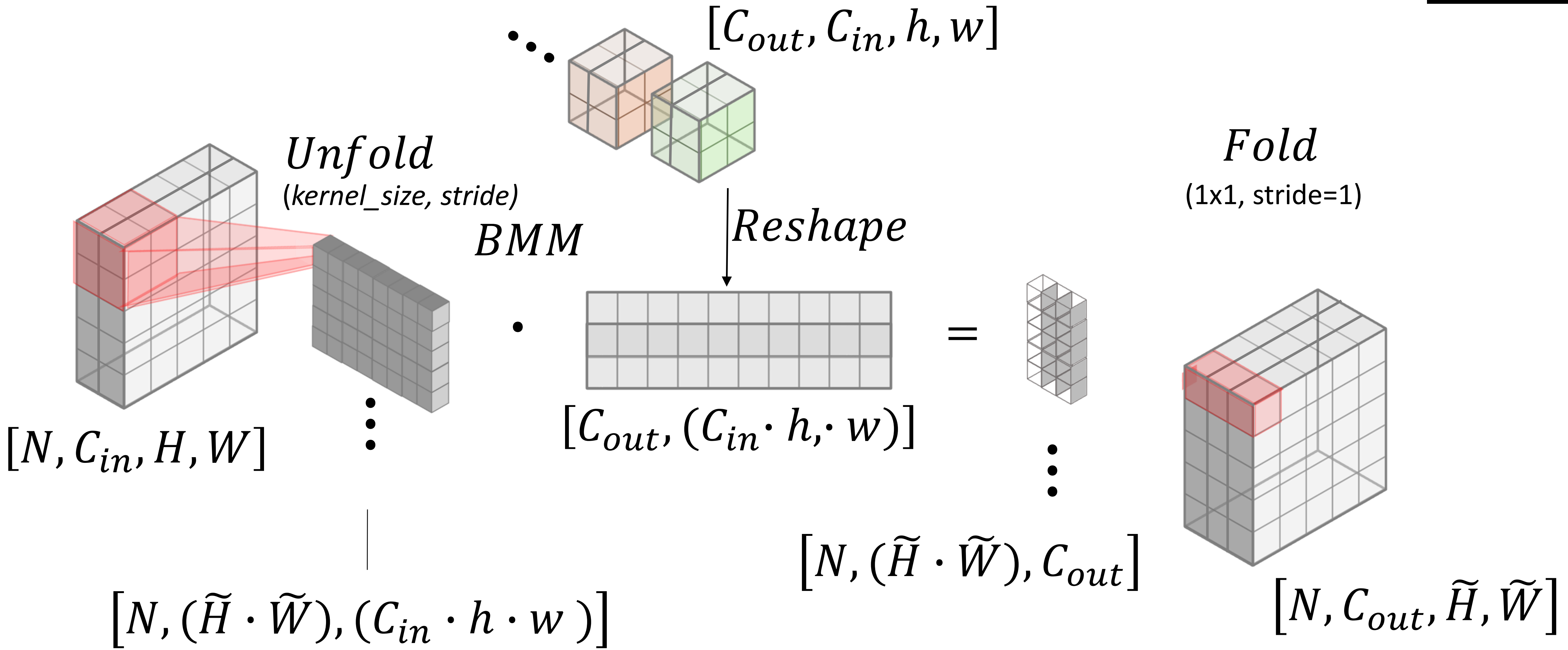
A note about the implementation of conv



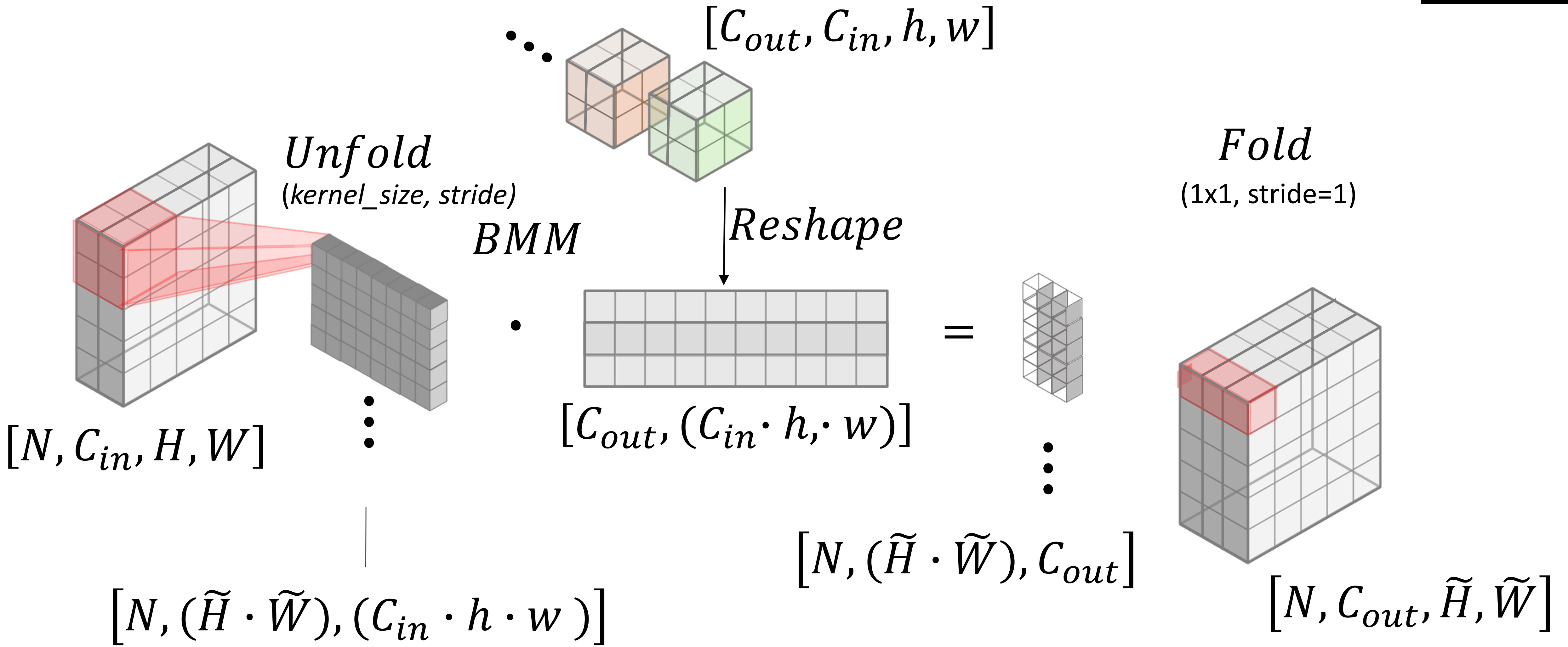
A note about the implementation of conv



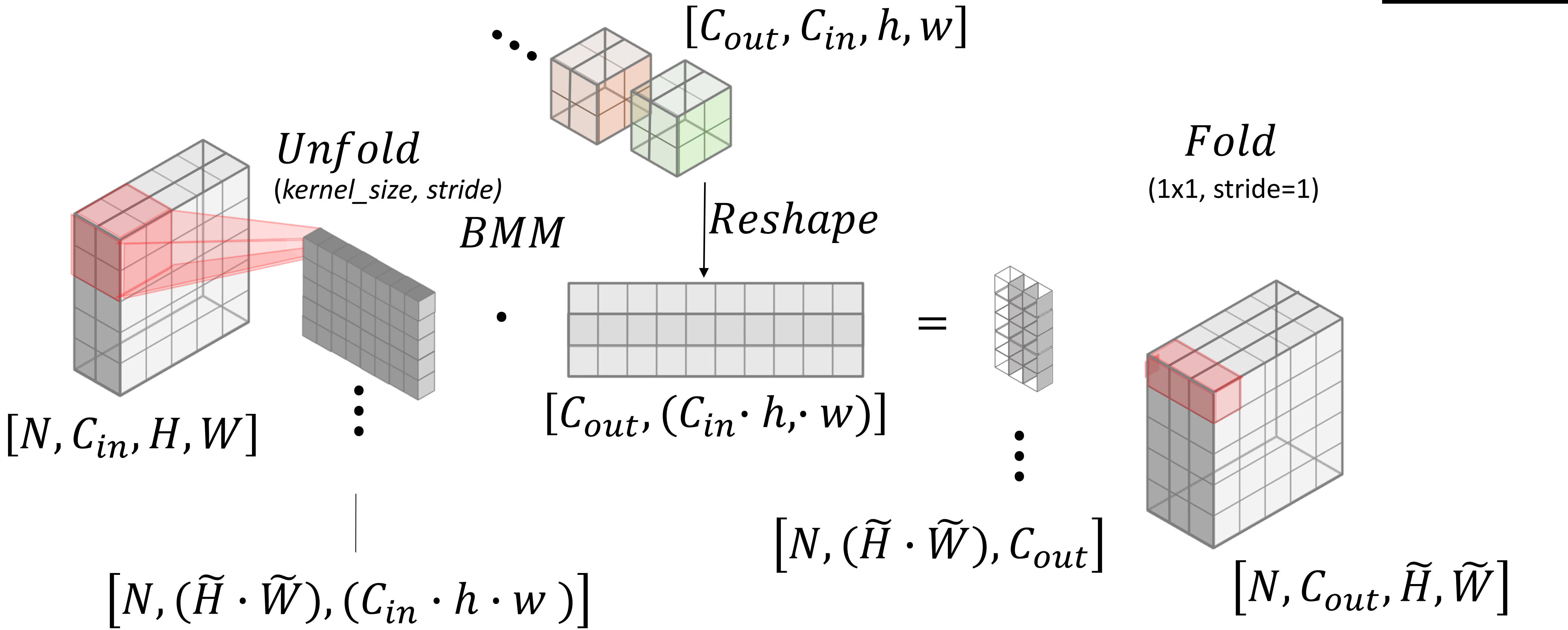
A note about the implementation of conv



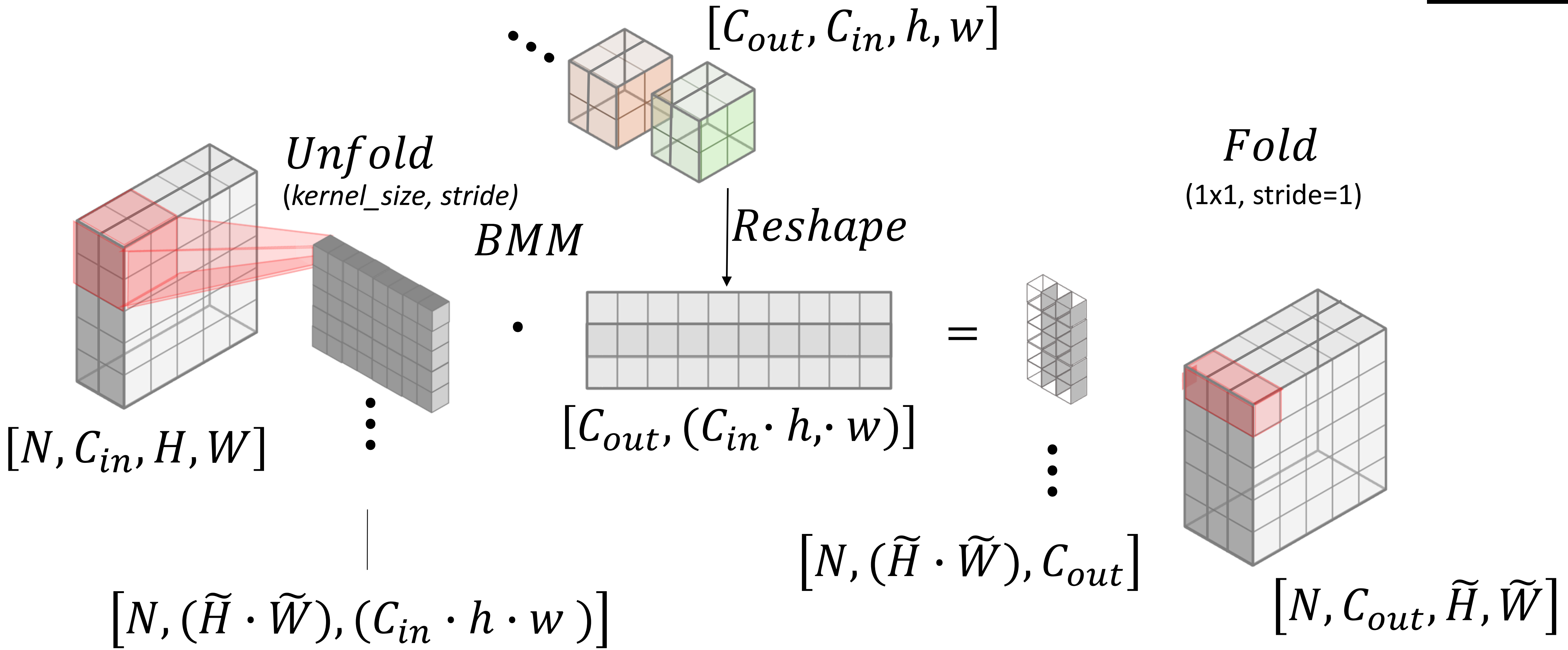
A note about the implementation of conv



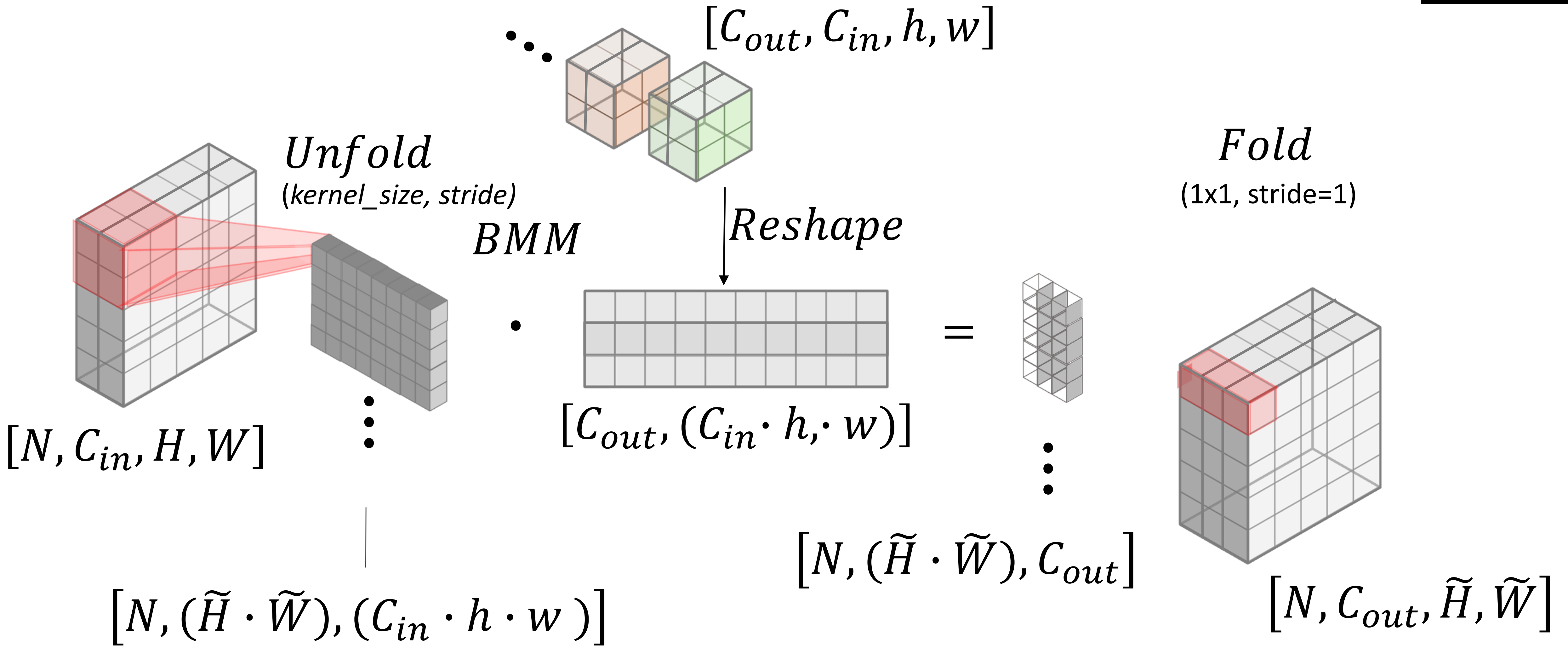
A note about the implementation of conv



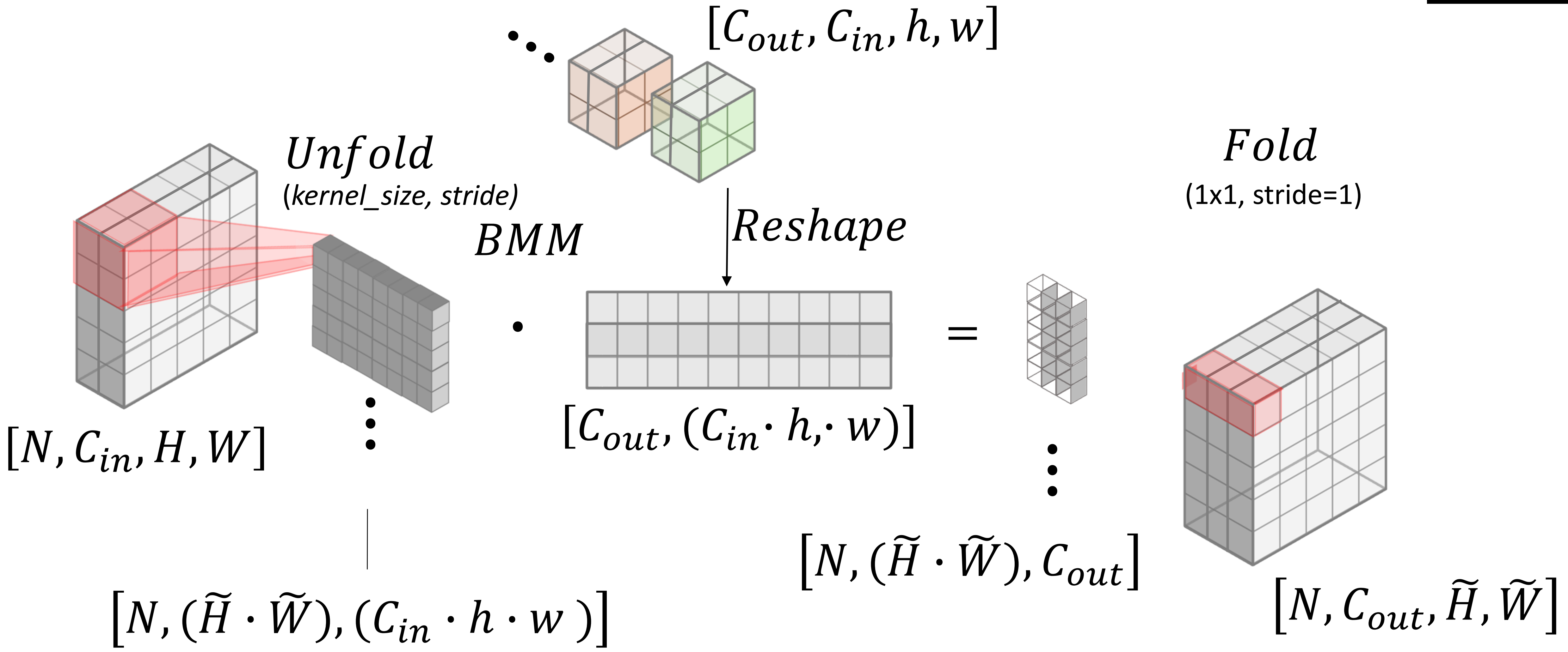
A note about the implementation of conv



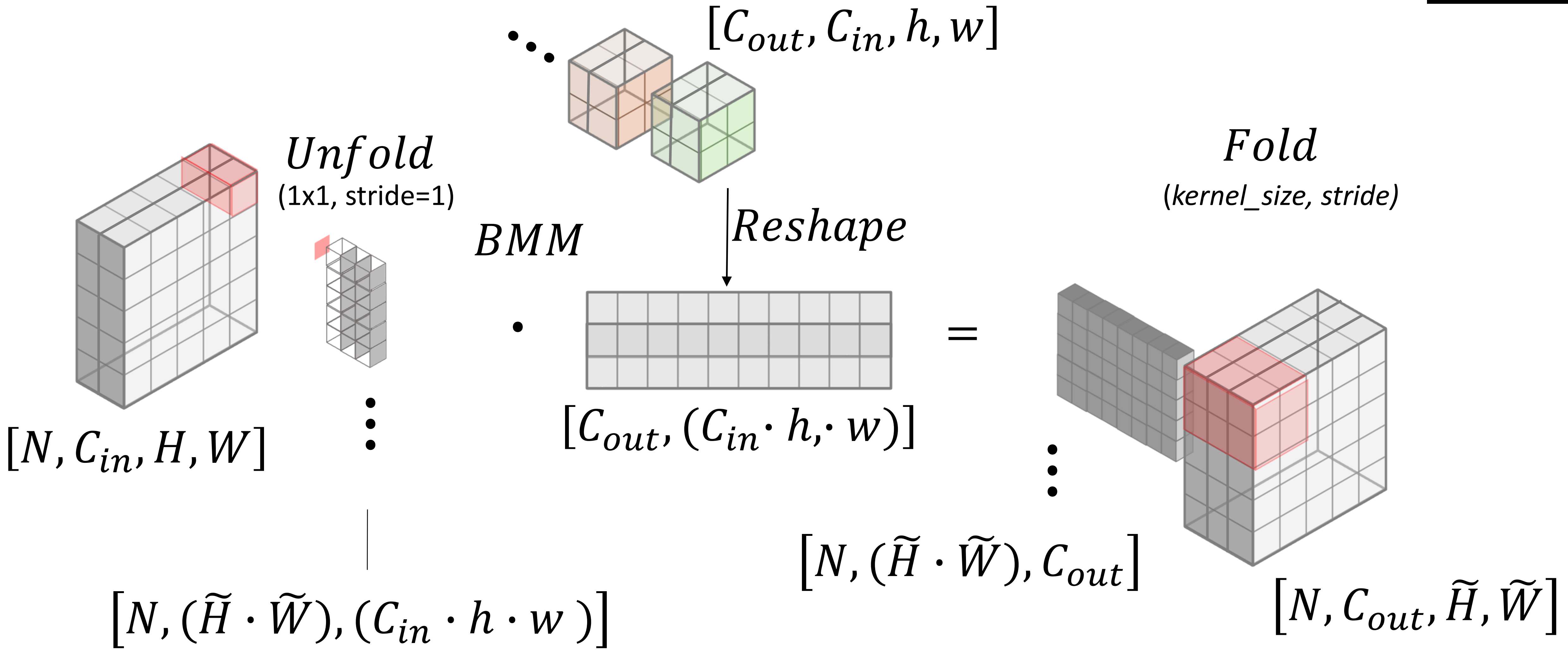
A note about the implementation of conv



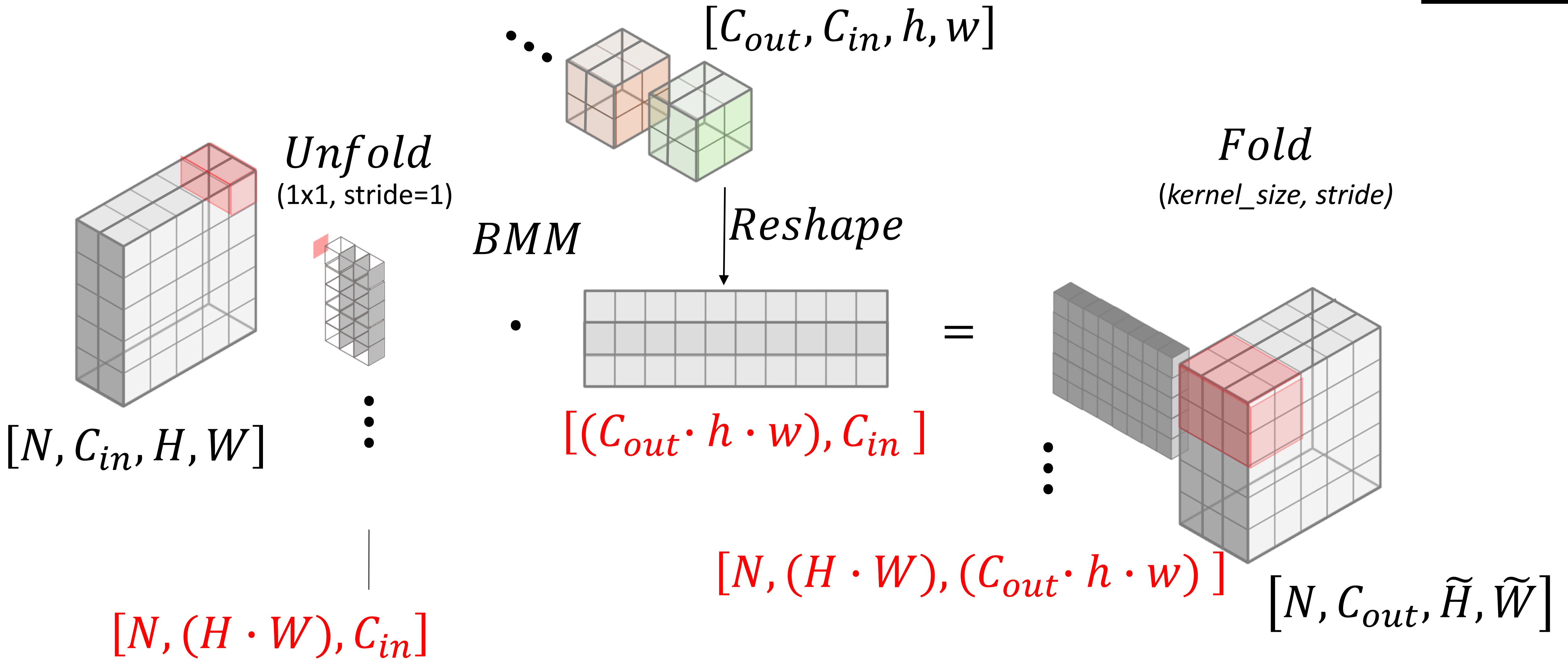
A note about the implementation of conv *TRANSPOSED*



A note about the implementation of conv *TRANSPOSED*



A note about the implementation of conv *TRANSPOSED*



Q: How do you backprop a Conv? (1D, single channel)

Q: How do you backprop a Conv? (1D, single channel)

<u>For derivative of loss w.r.t all neurons</u>:
Transposed Conv with the same filter!

(Conv is a special case of linear)

Q: How do you backprop a Conv? (1D, single channel)

<u>For derivative of loss w.r.t all neurons</u>:
Transposed Conv with the same filter!

(Conv is a special case of linear)

Q: How about Conv2D?

Q: How do you backprop a Conv? (1D, single channel)

For derivative of loss w.r.t all neurons:
Transposed Conv with the same filter!

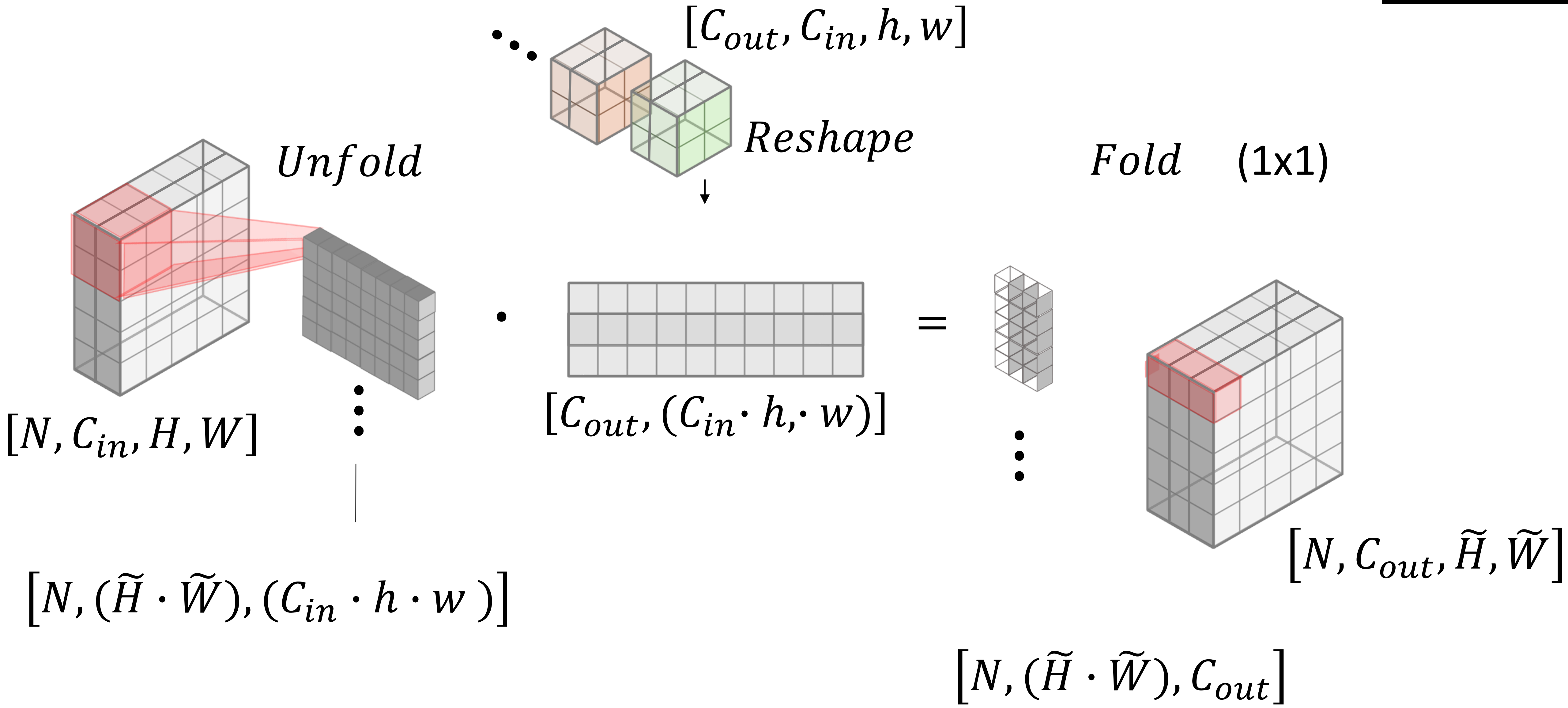
(Conv is a special case of linear)

Q: How about Conv2D?

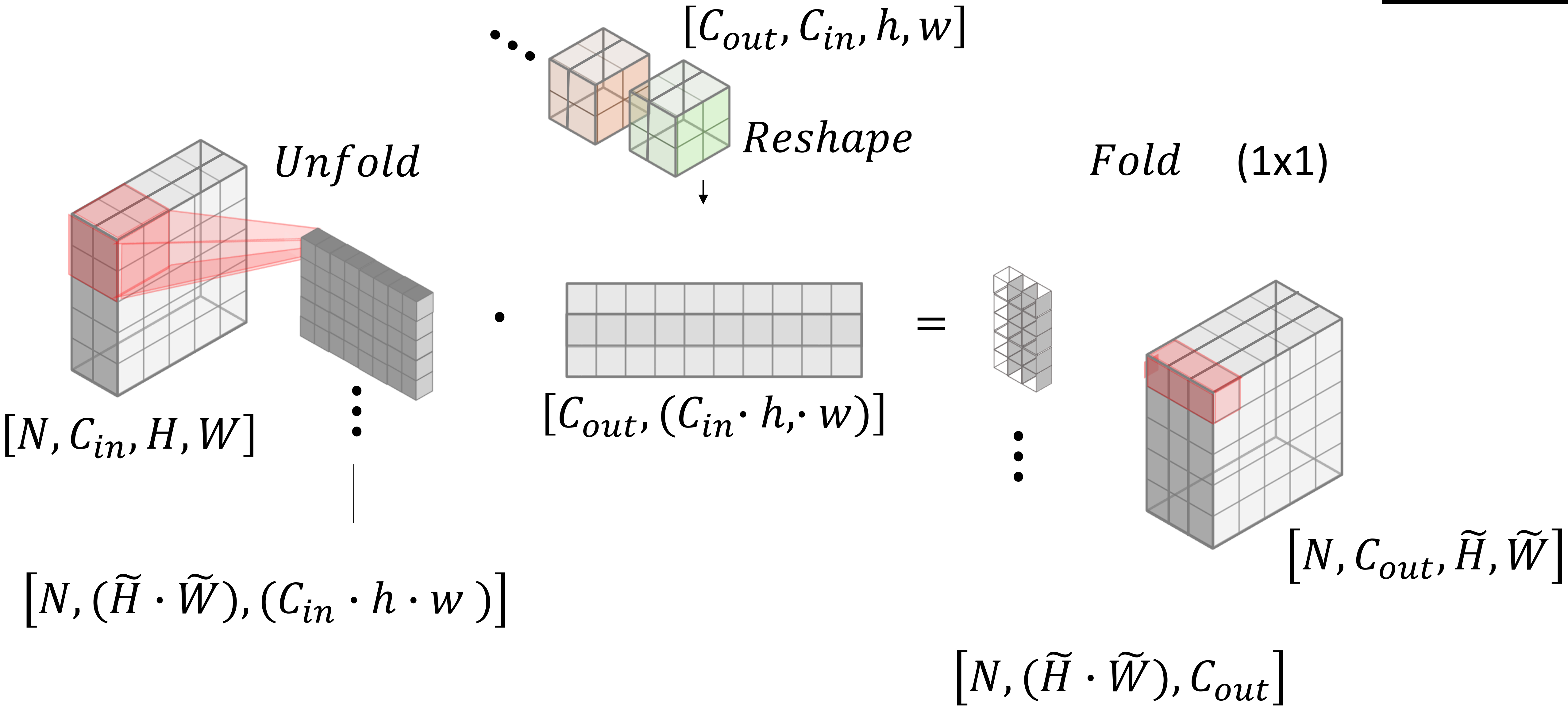
For derivative of loss w.r.t all neurons:
1. Transpose dims of filter c_{in} , c_{out} (as in FC)
2. Transposed Conv2D with the modified filter!

(Think of a conv as a linear layer applied to each patch separately, going from c_{in} to c_{out})

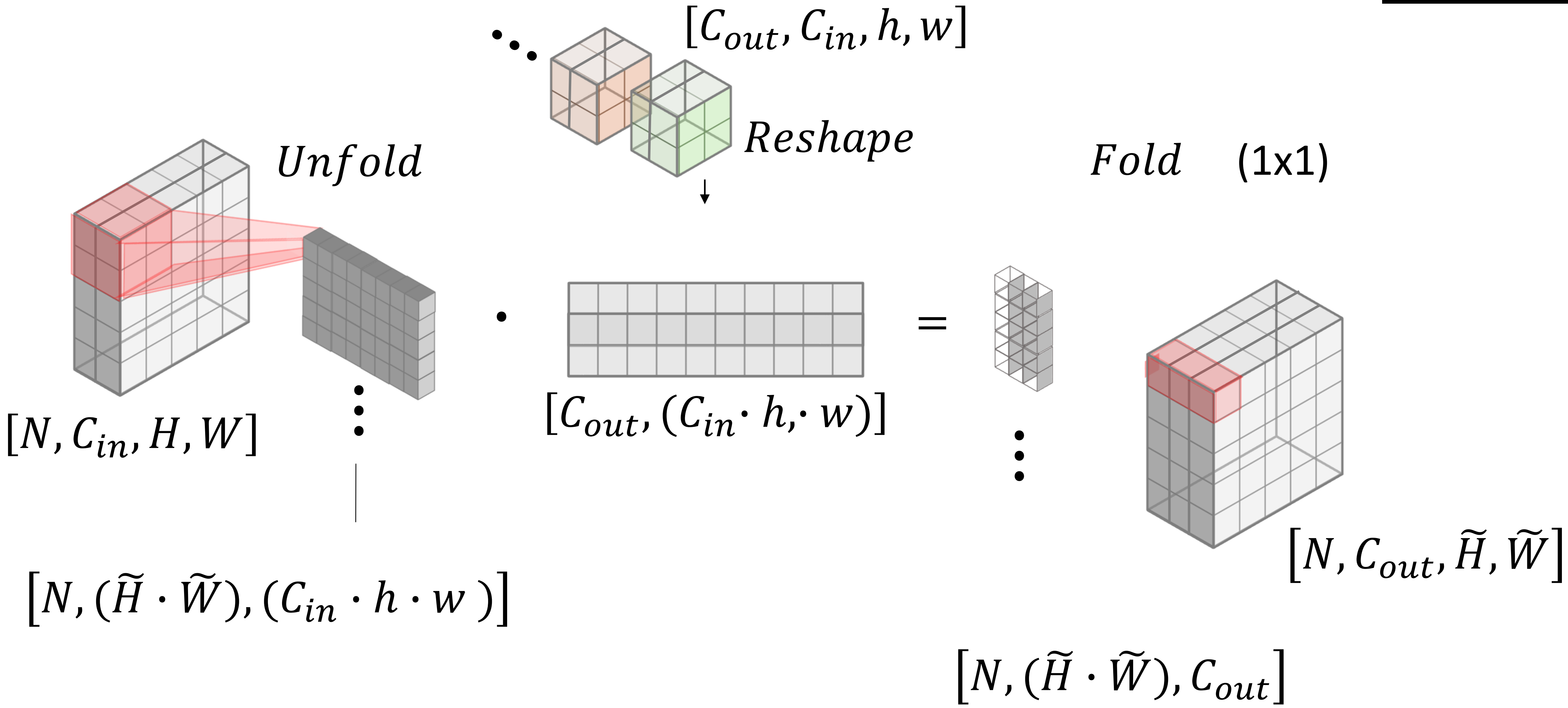
A note about the implementation of conv - BACKWARD



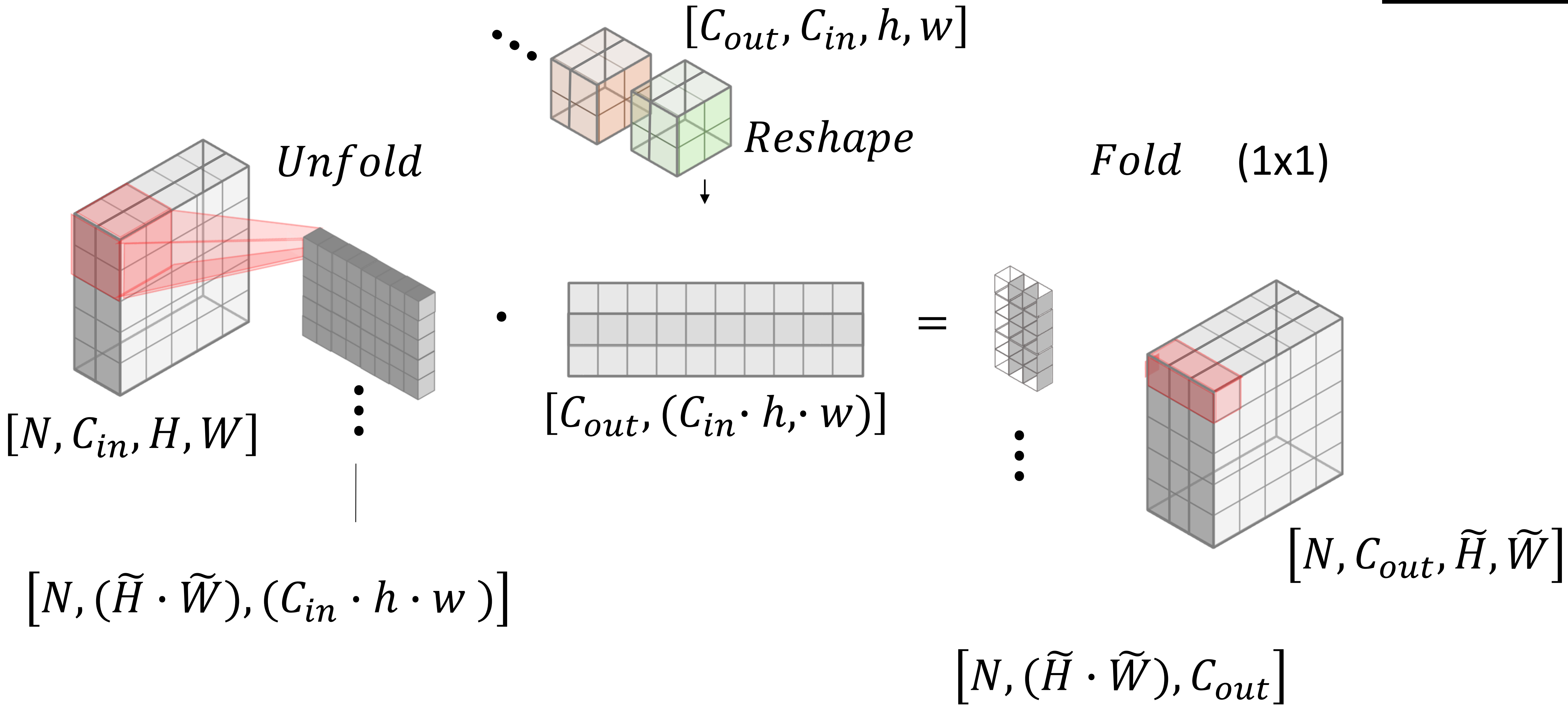
A note about the implementation of conv - BACKWARD



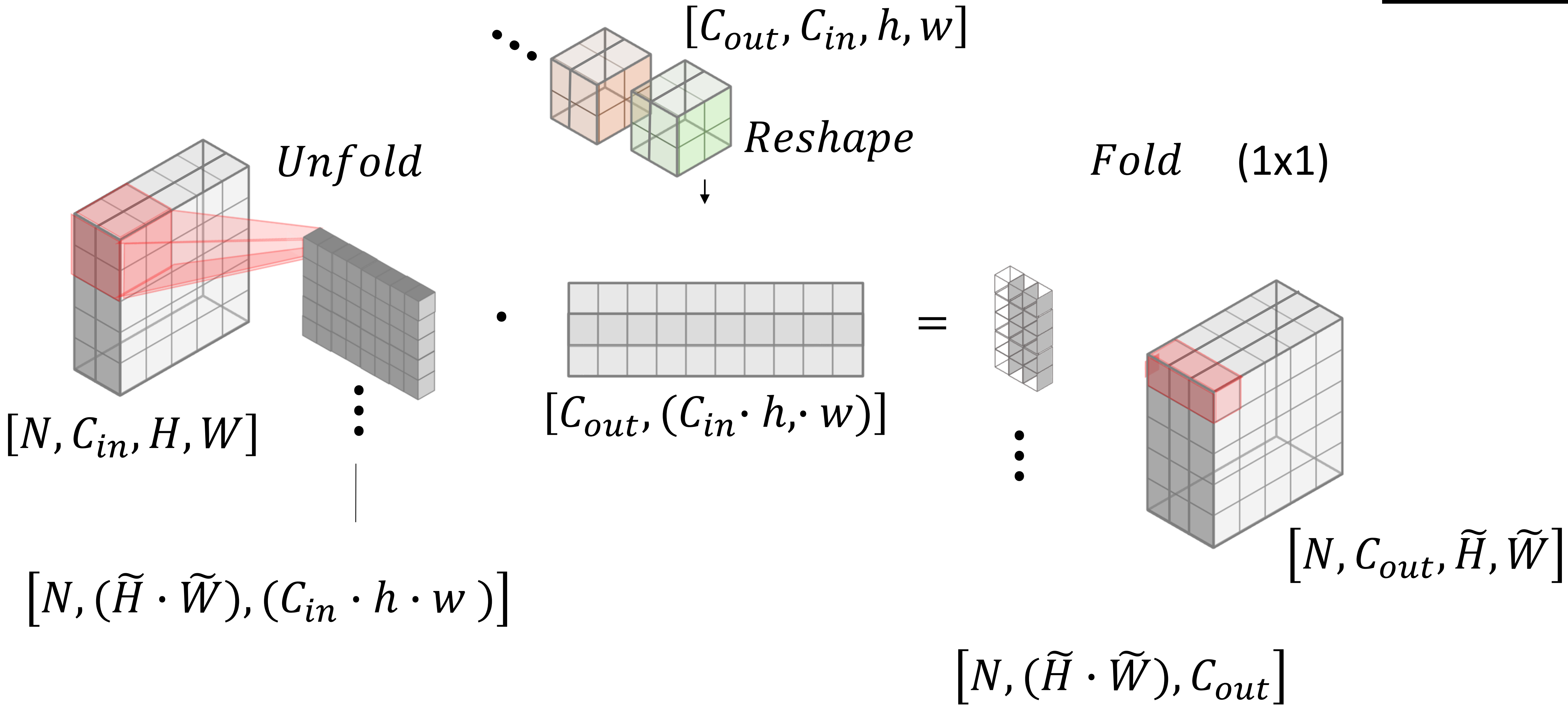
A note about the implementation of conv - BACKWARD



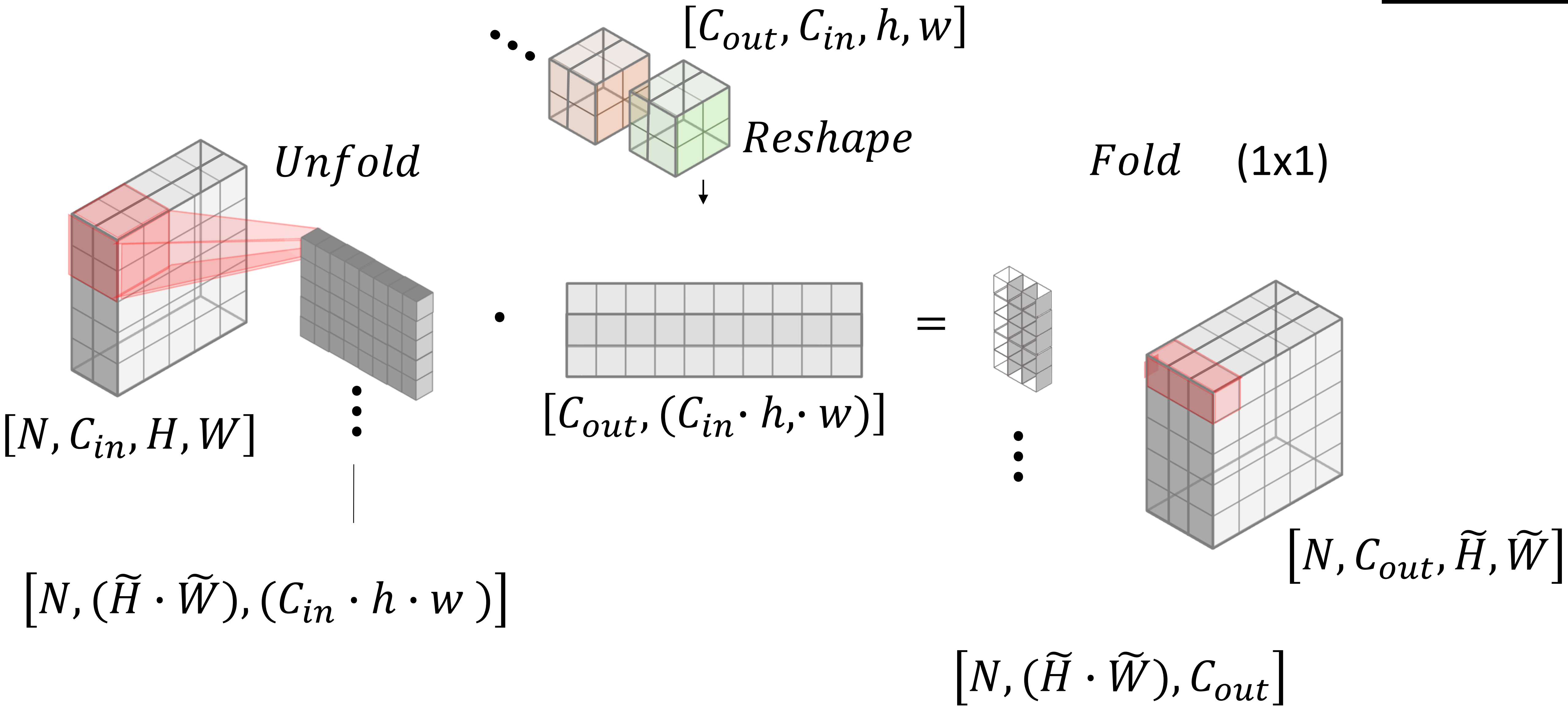
A note about the implementation of conv - BACKWARD



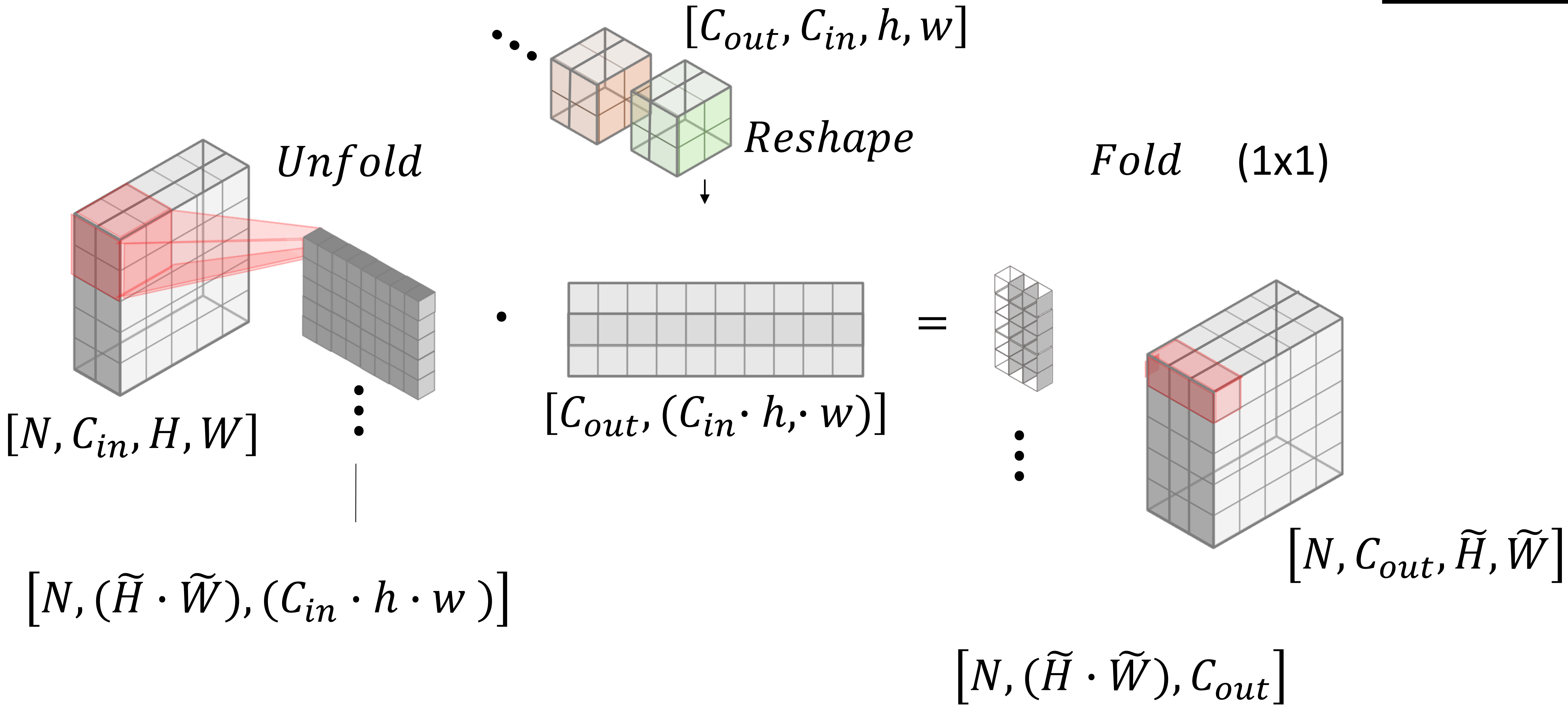
A note about the implementation of conv - BACKWARD



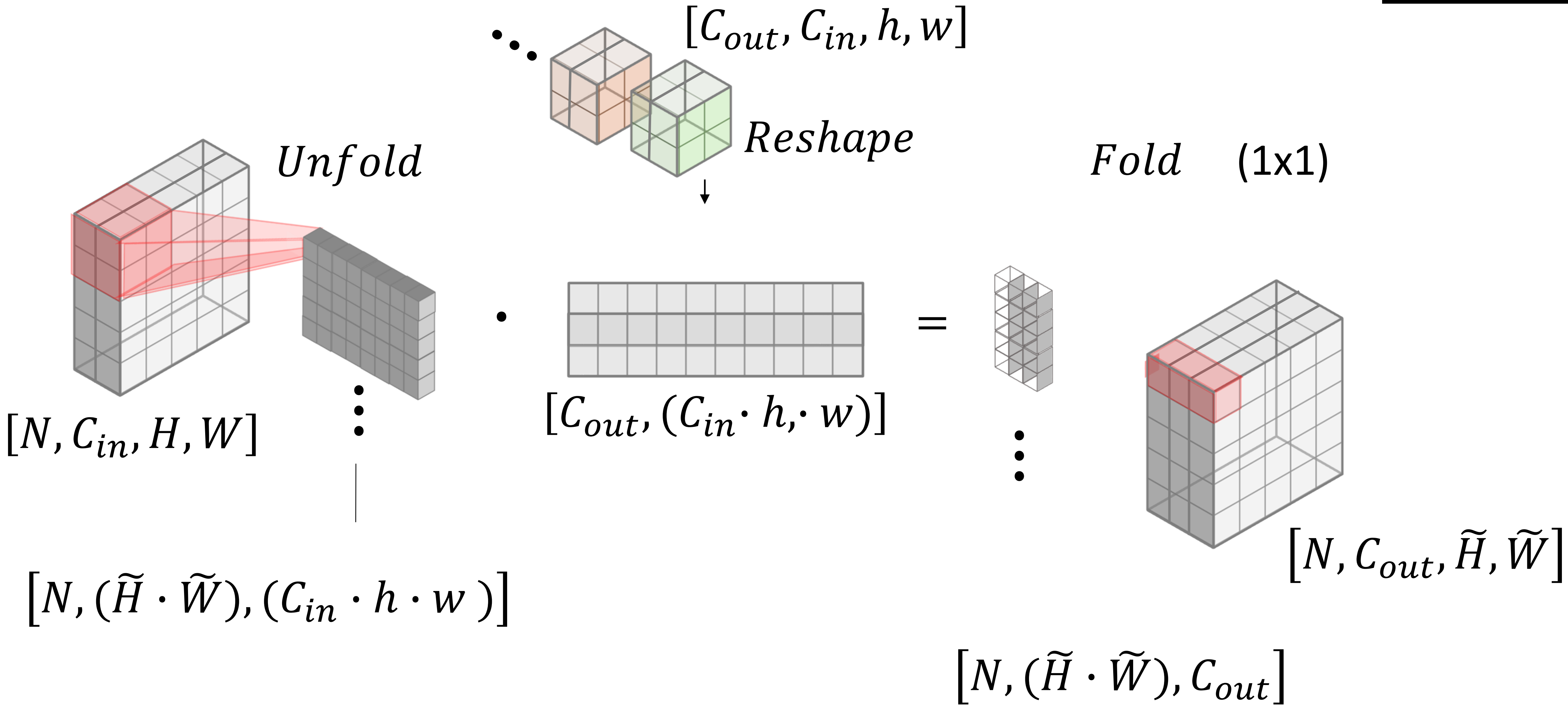
A note about the implementation of conv - BACKWARD



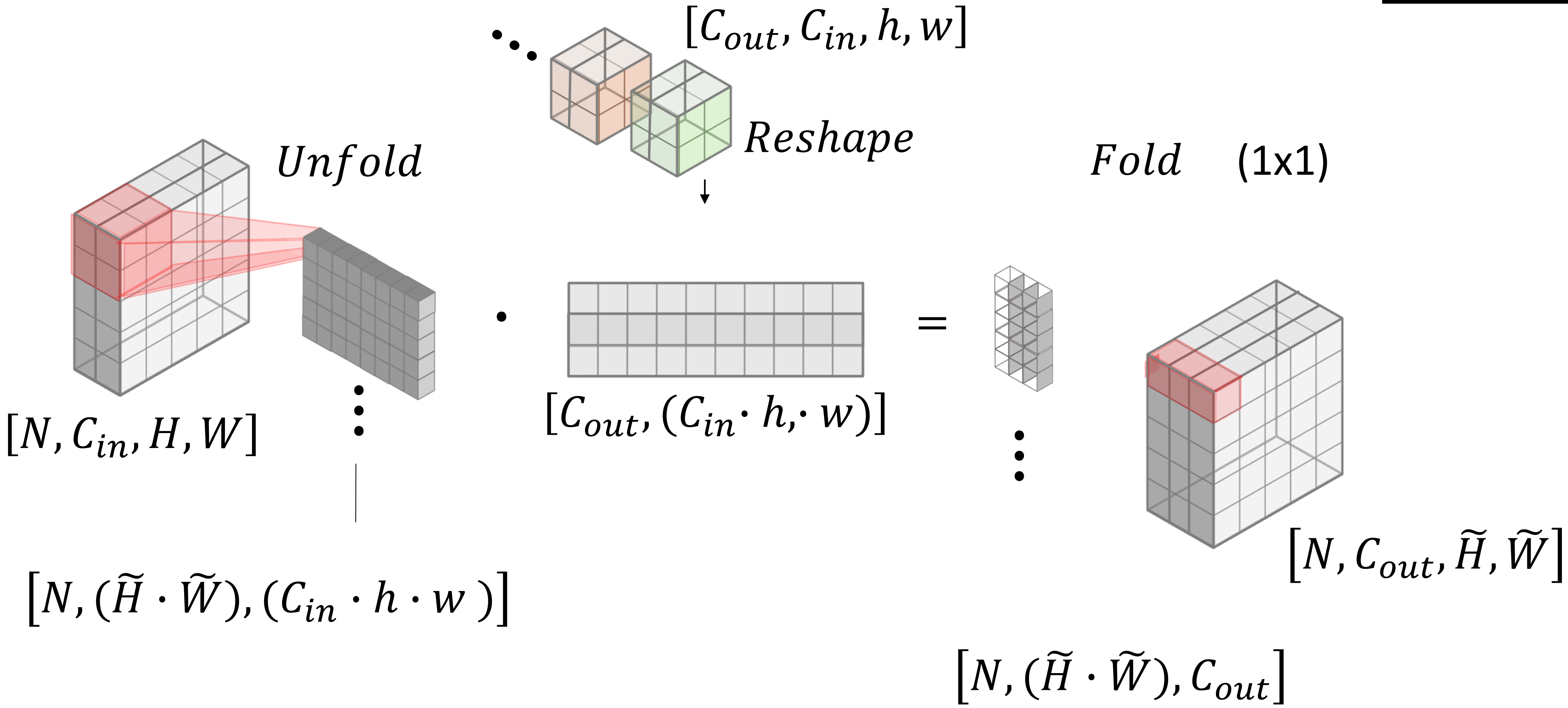
A note about the implementation of conv - BACKWARD



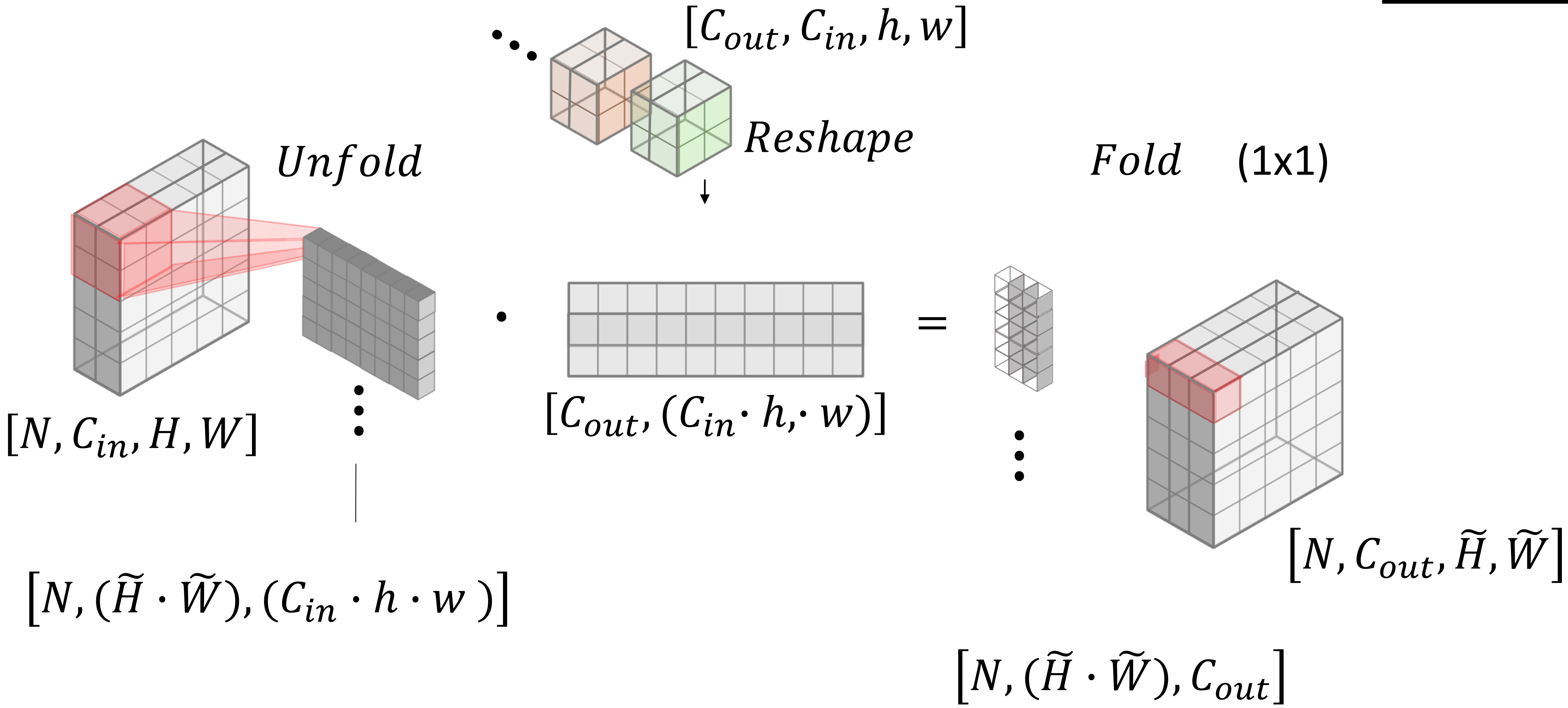
A note about the implementation of conv - BACKWARD



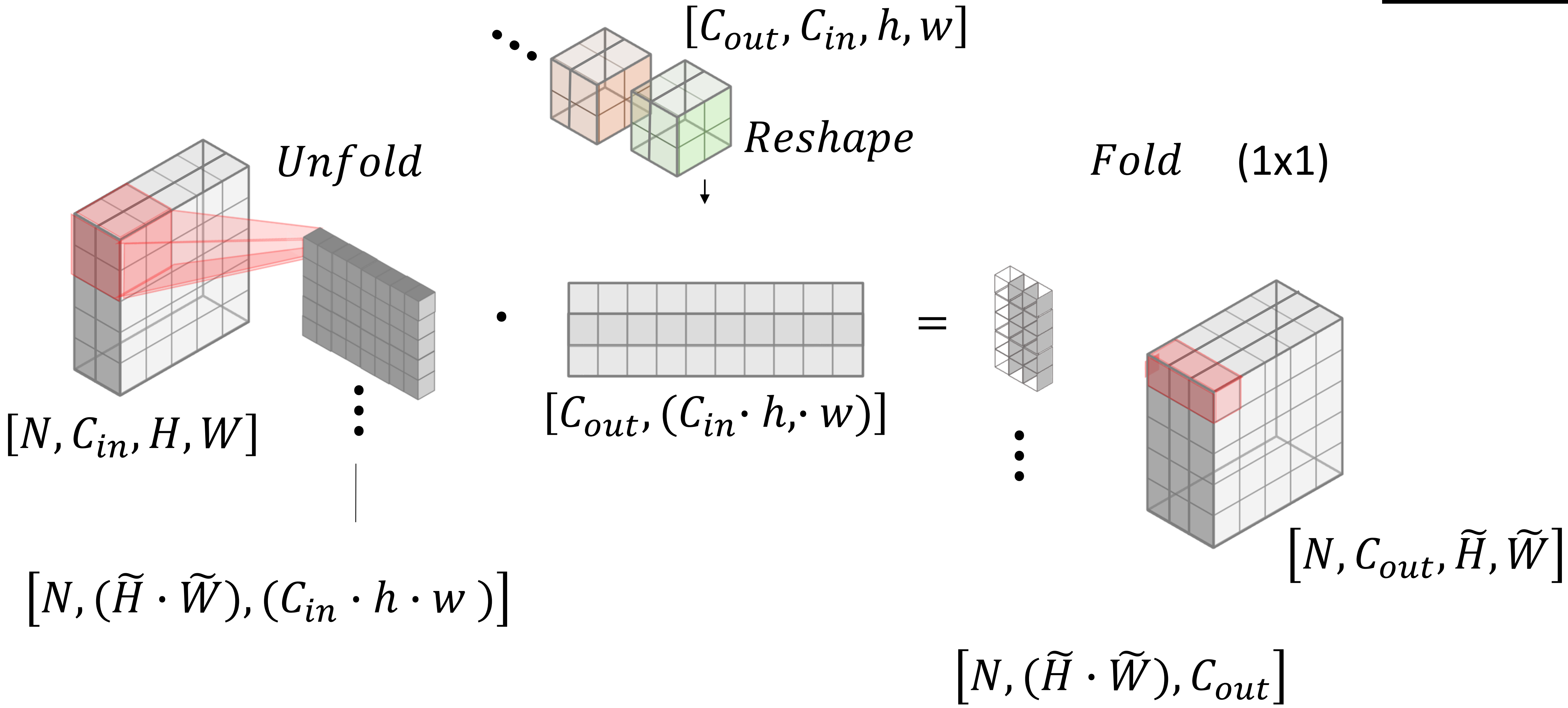
A note about the implementation of conv - BACKWARD



A note about the implementation of conv - BACKWARD

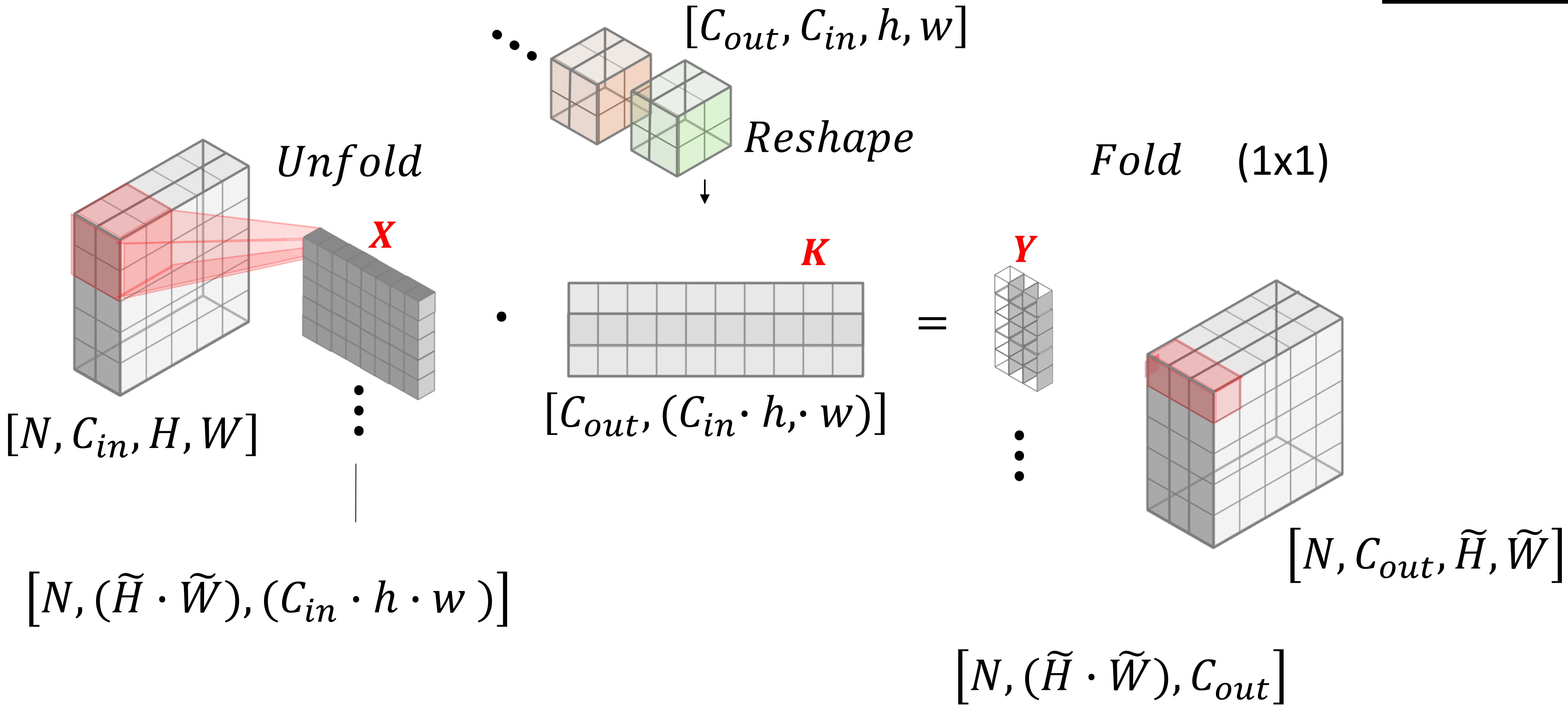


A note about the implementation of conv - BACKWARD

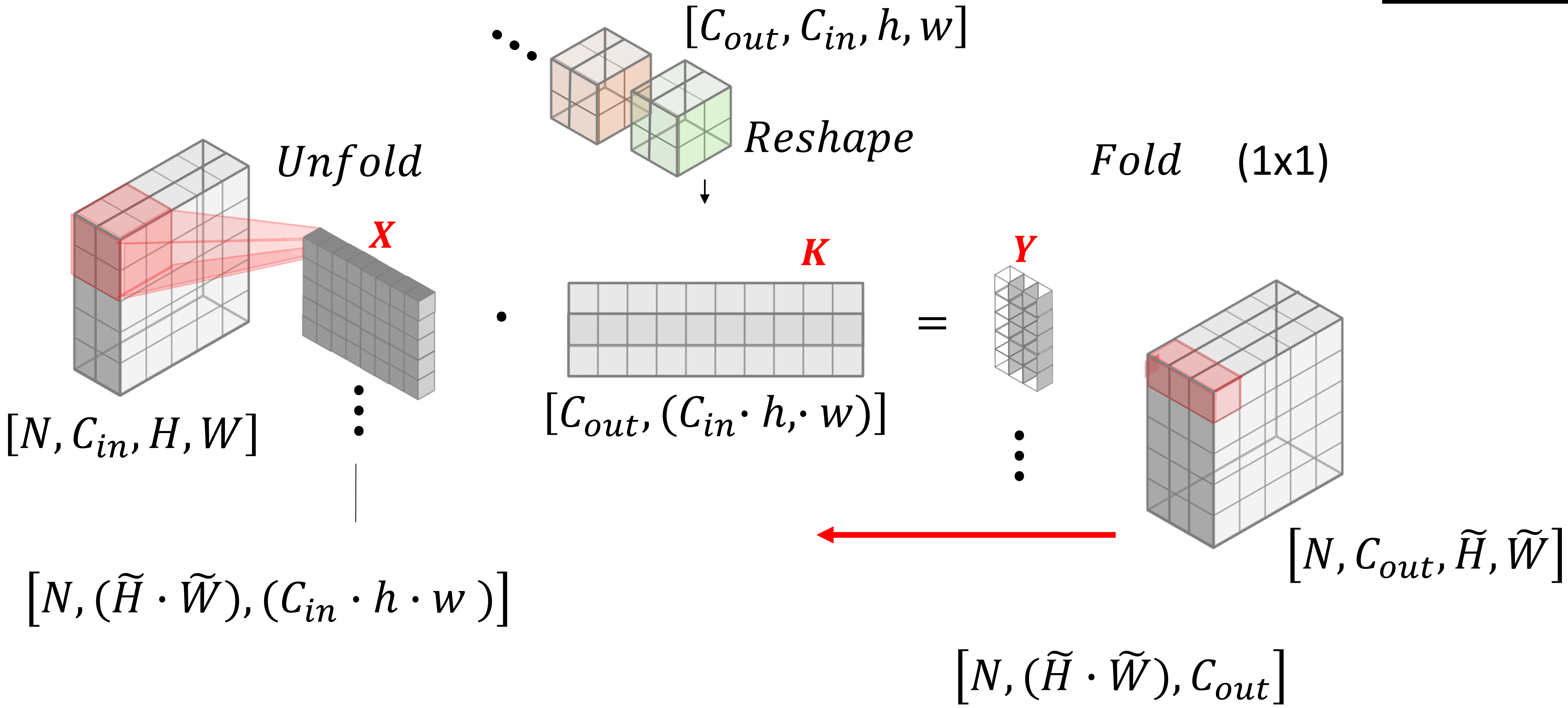


A note about the implementation of conv - BACKWARD

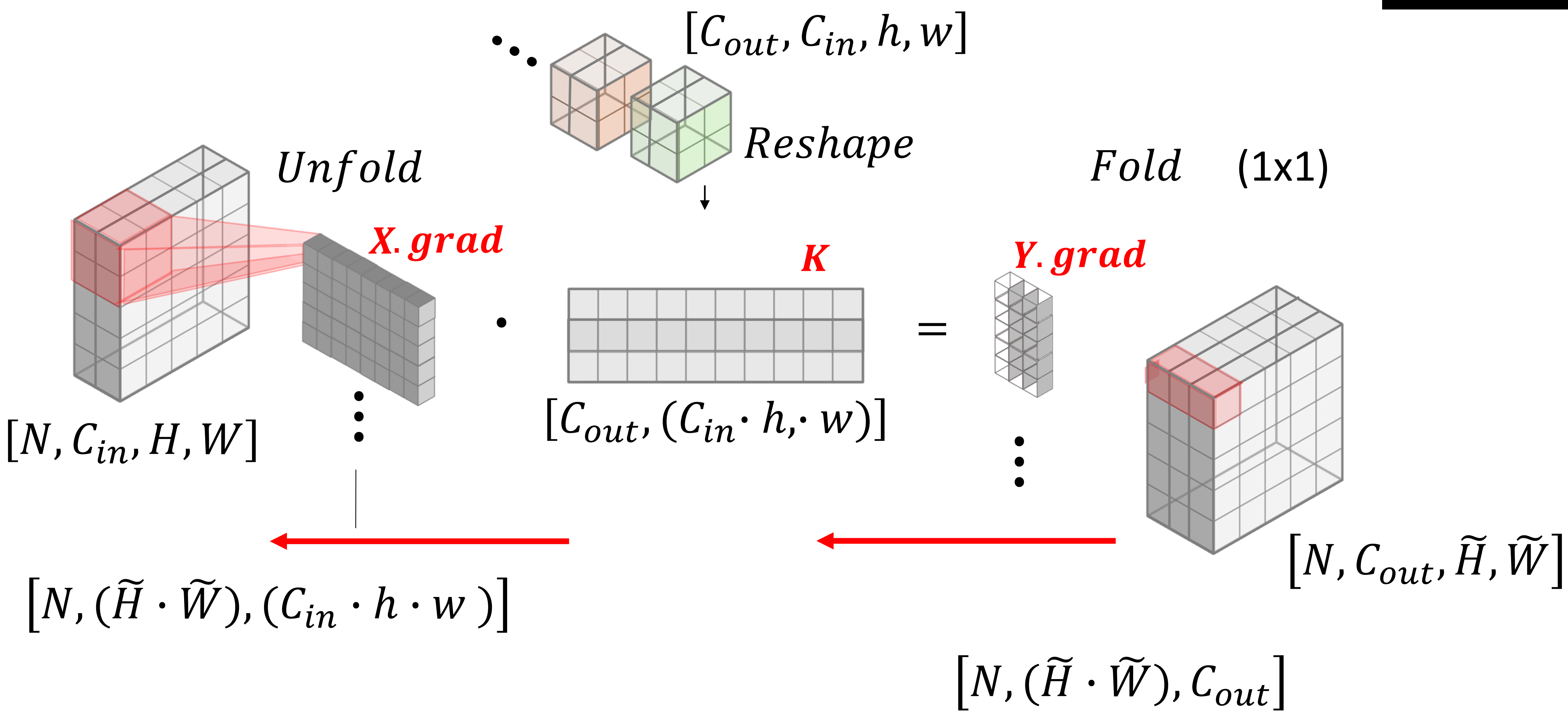
[redacted]



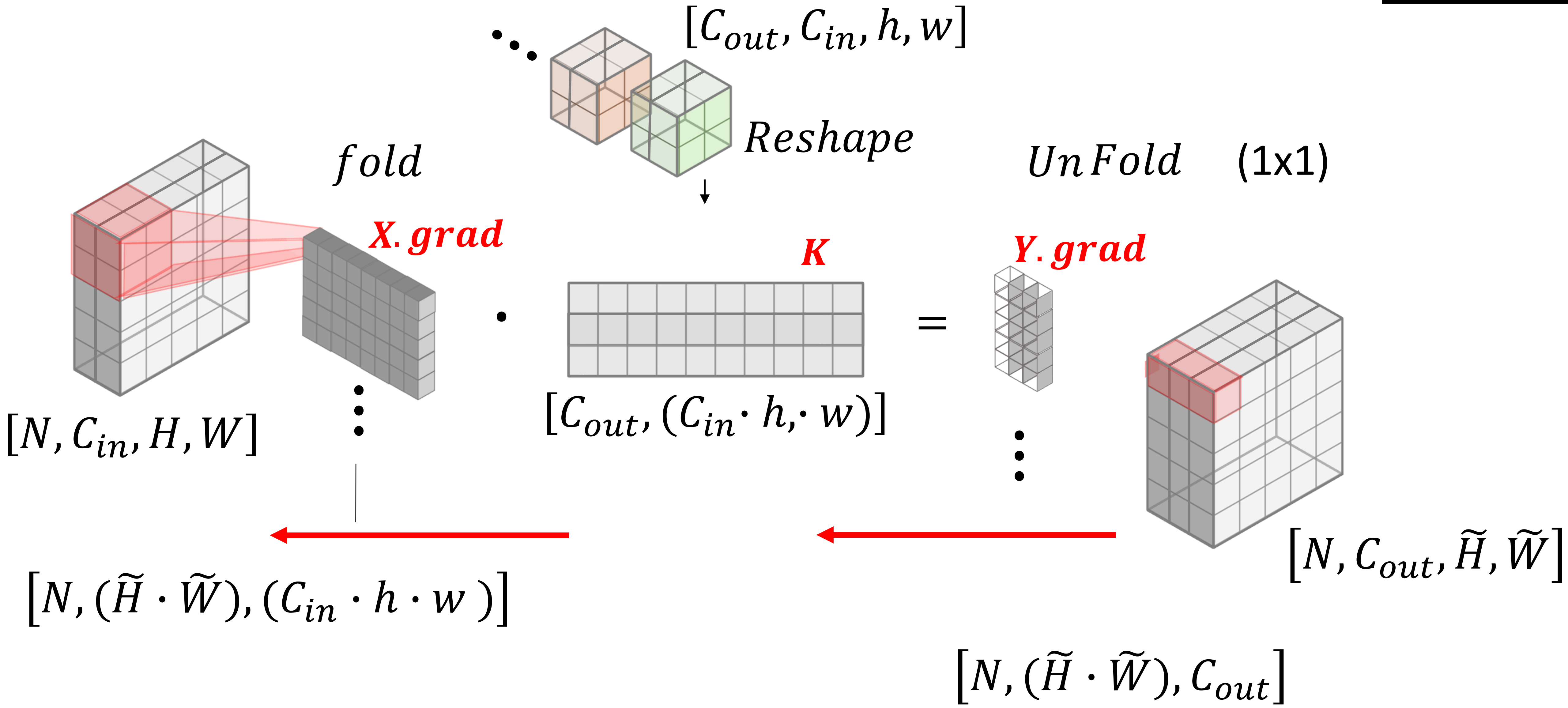
A note about the implementation of conv - BACKWARD



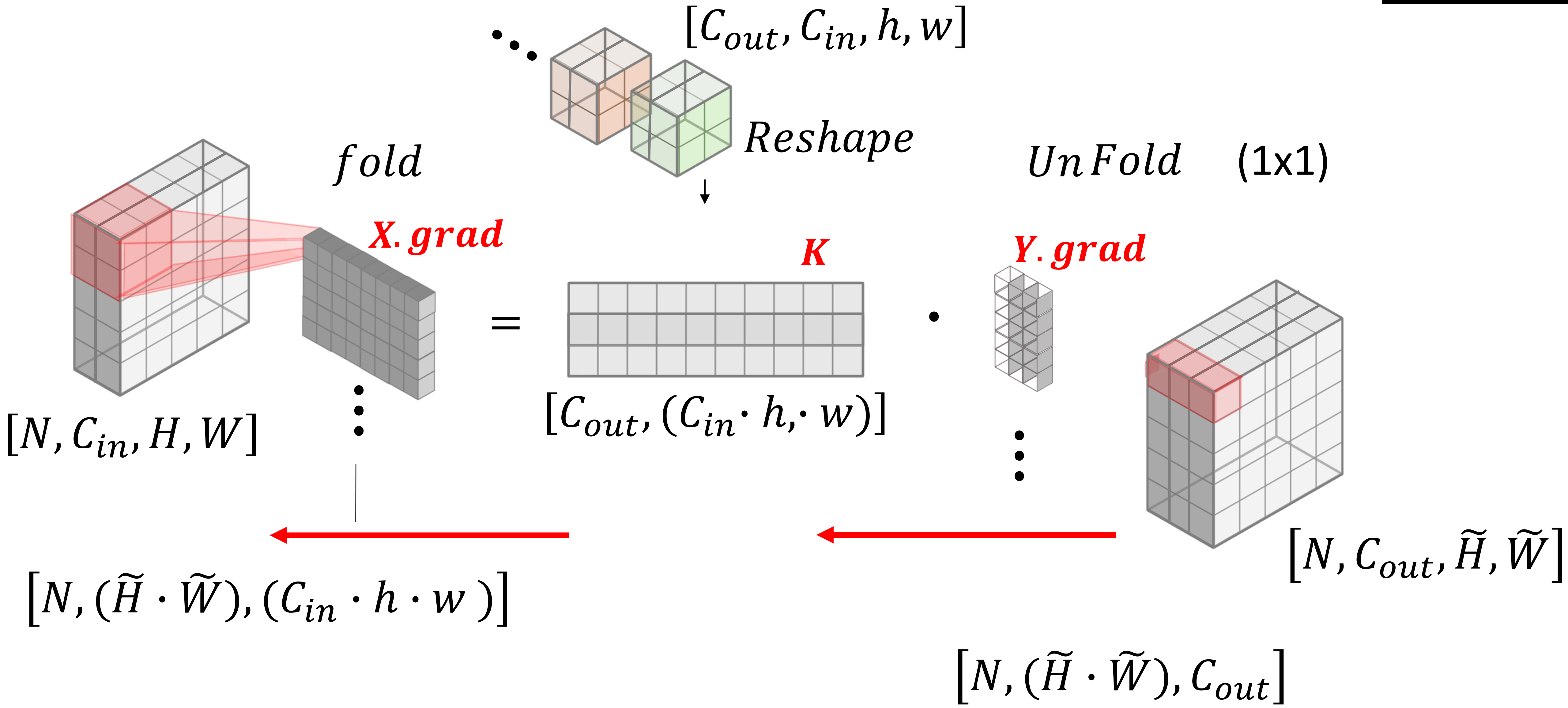
A note about the implementation of conv - BACKWARD



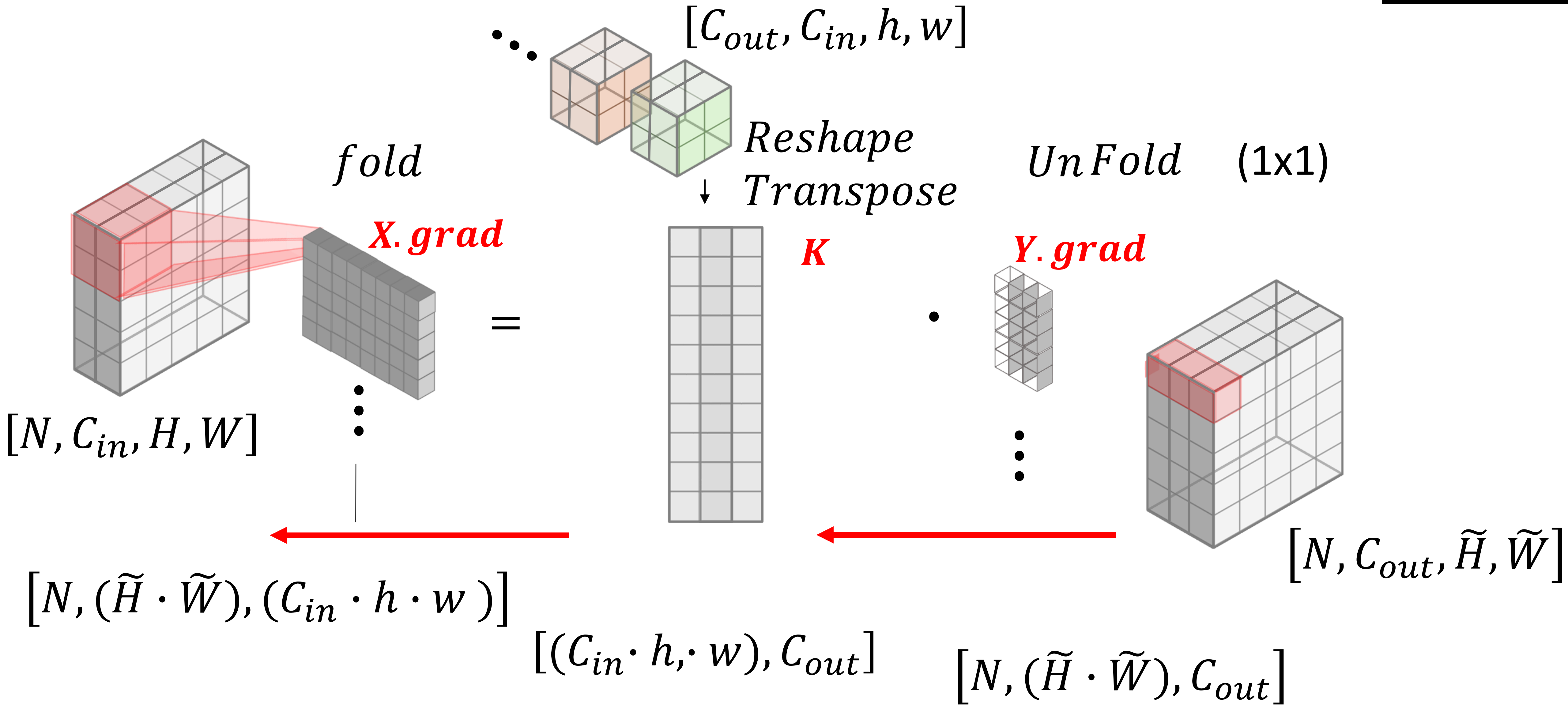
A note about the implementation of conv - BACKWARD



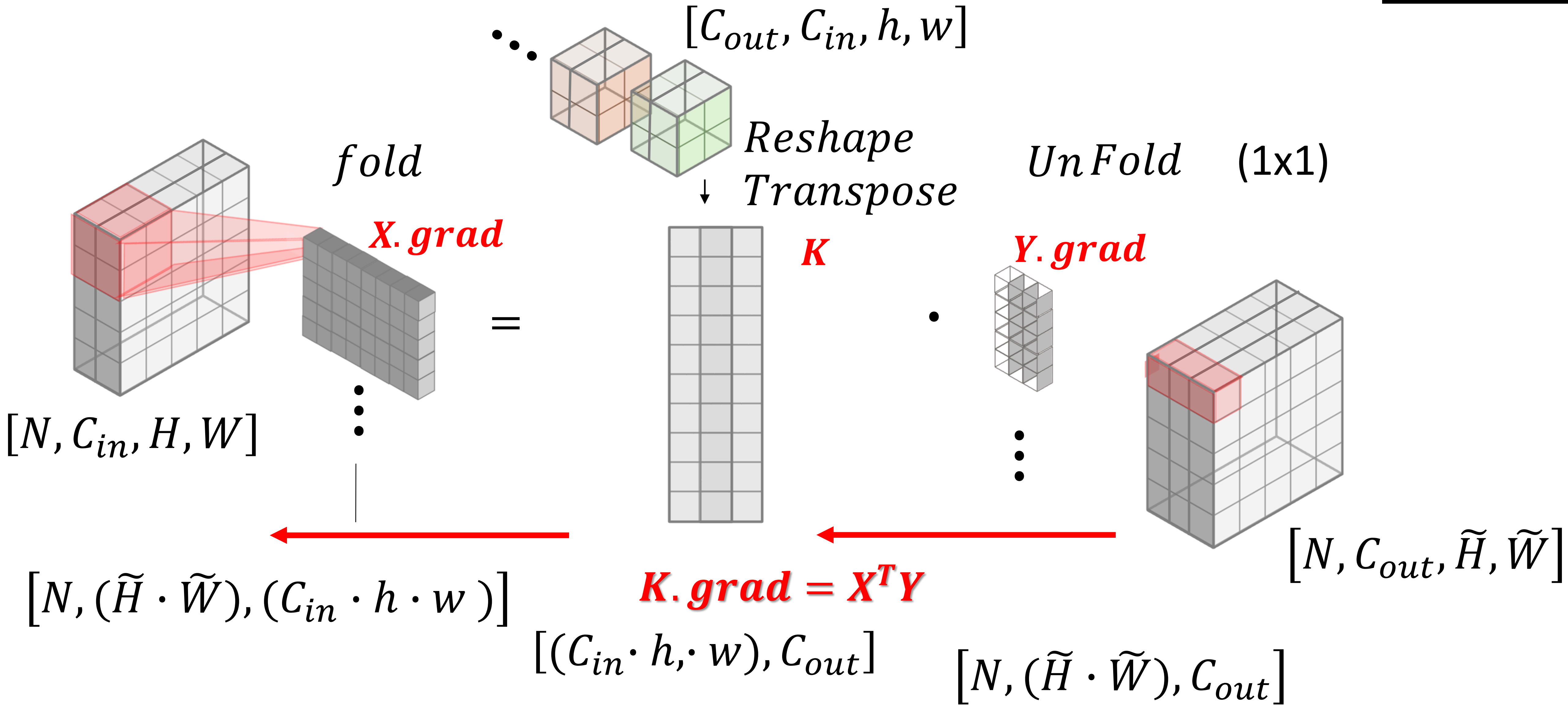
A note about the implementation of conv - BACKWARD



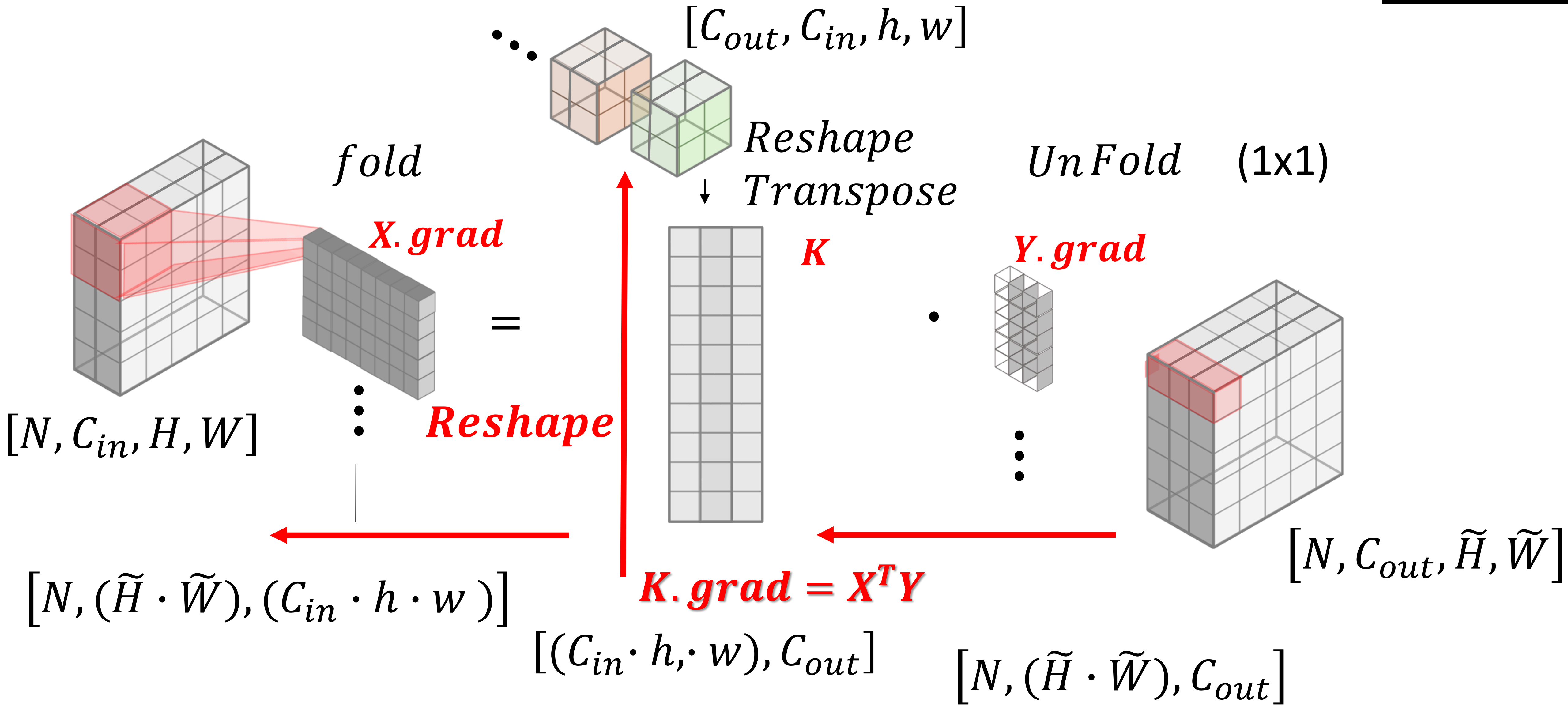
A note about the implementation of conv - BACKWARD



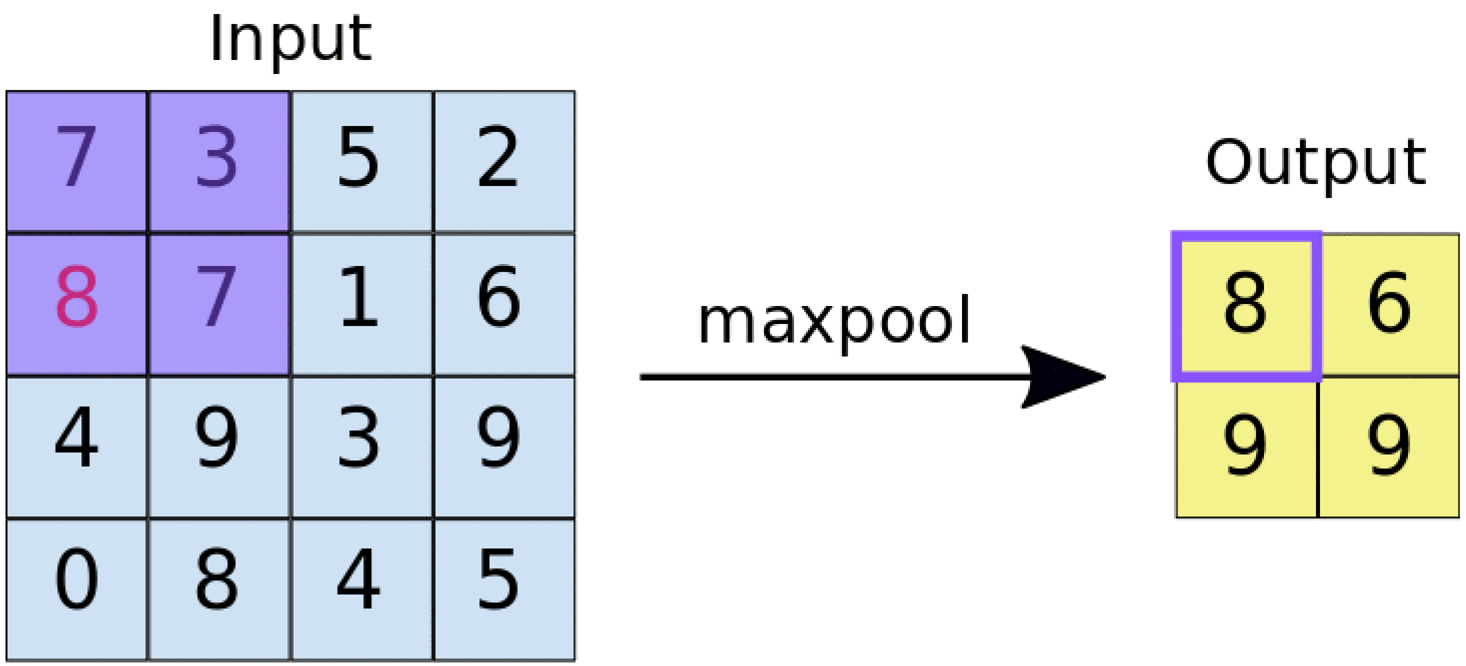
A note about the implementation of conv - BACKWARD



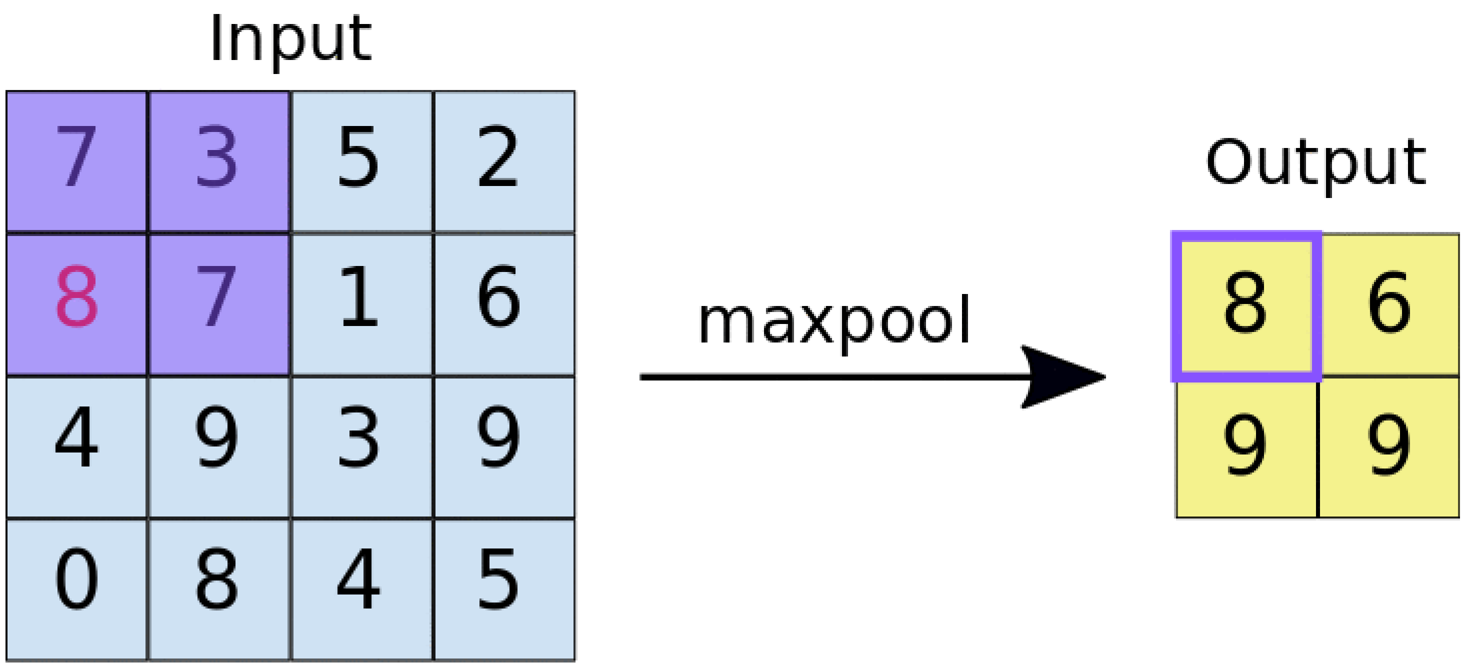
A note about the implementation of conv - BACKWARD



Max Pooling

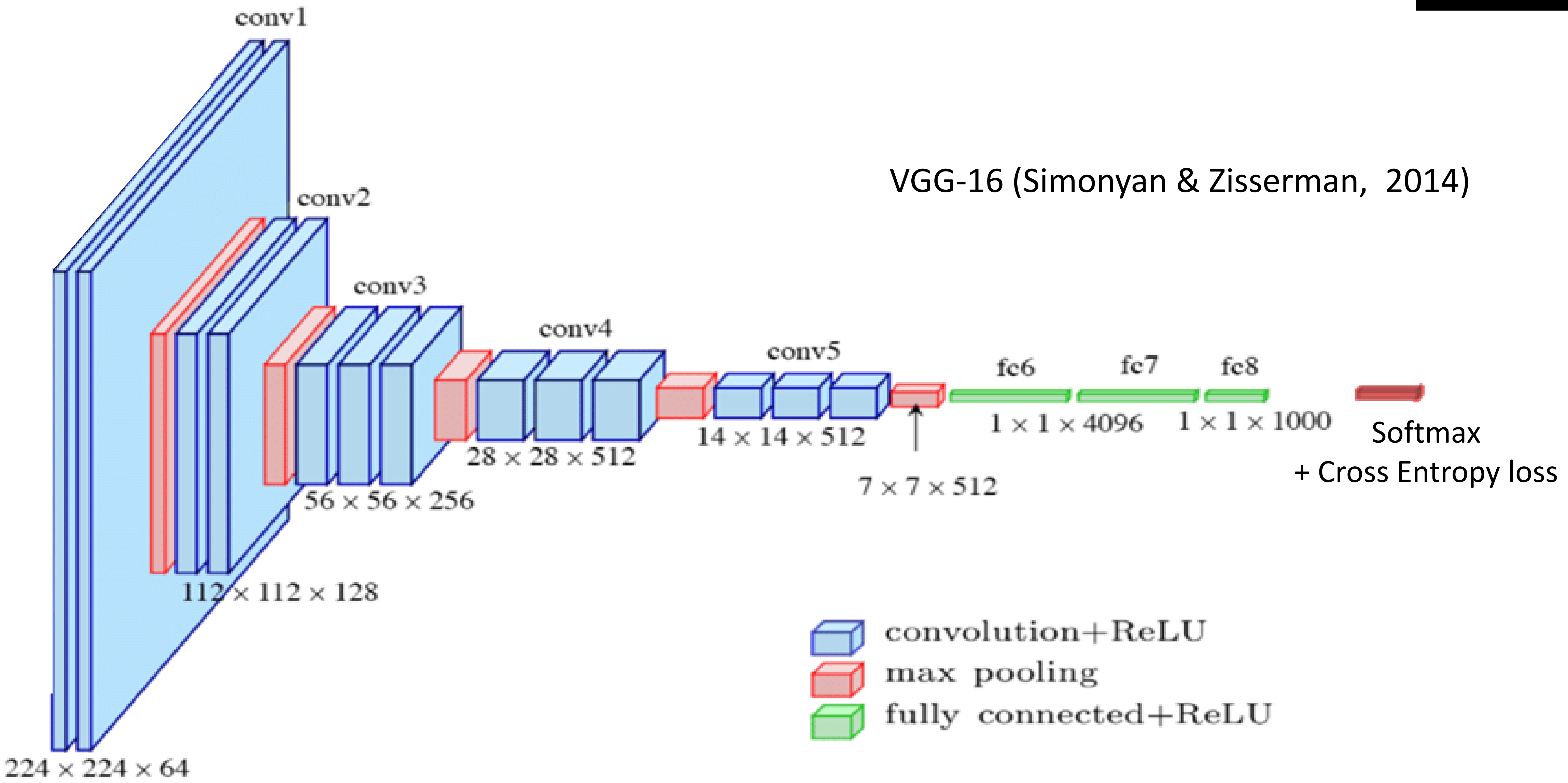


Max Pooling



- Usually stride=win-size, but not always.
- Each channel separately.

ConvNet Example

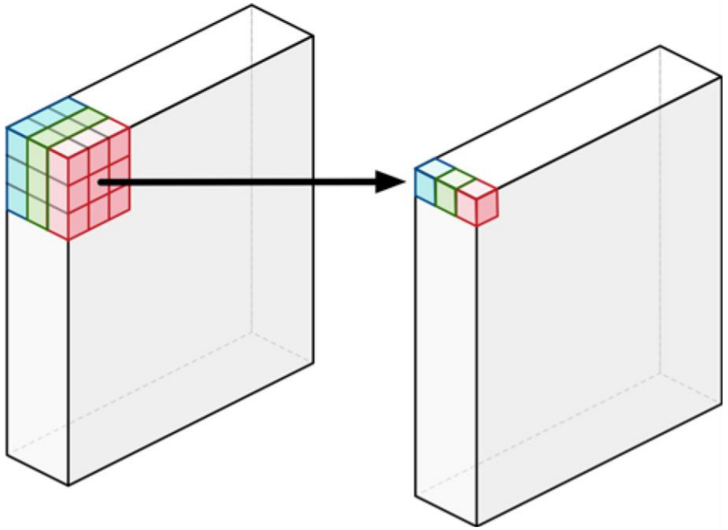


VGG-16 (Simonyan & Zisserman, 2014)

More special Convs!

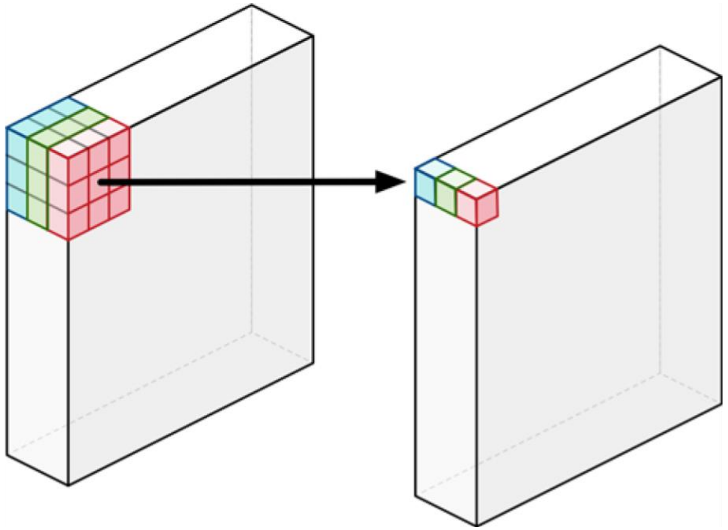
More special Convs!

Depthwise Conv

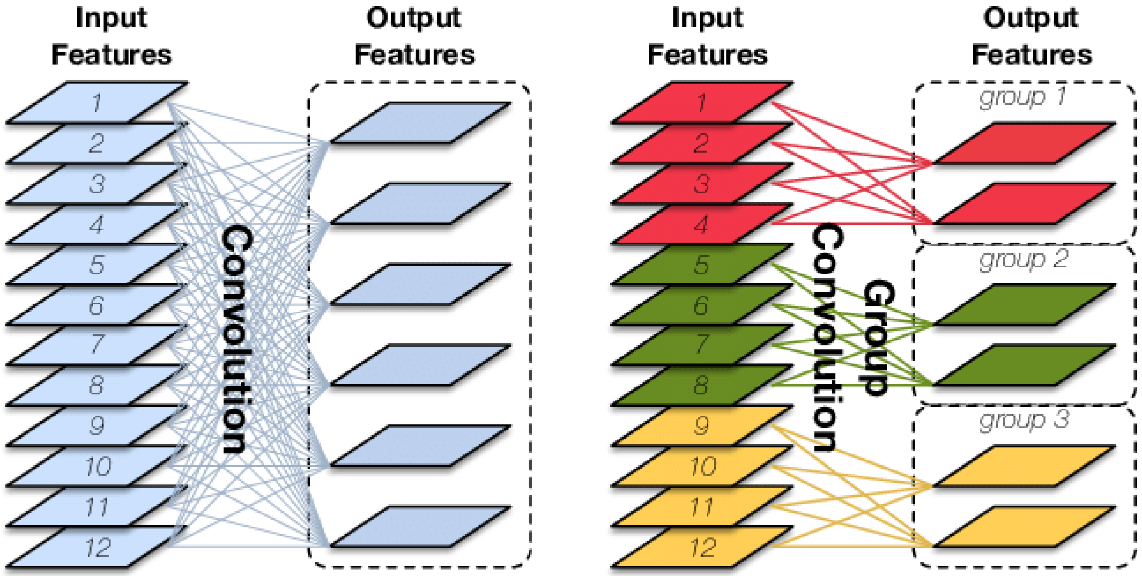


More special Convs!

Depthwise Conv

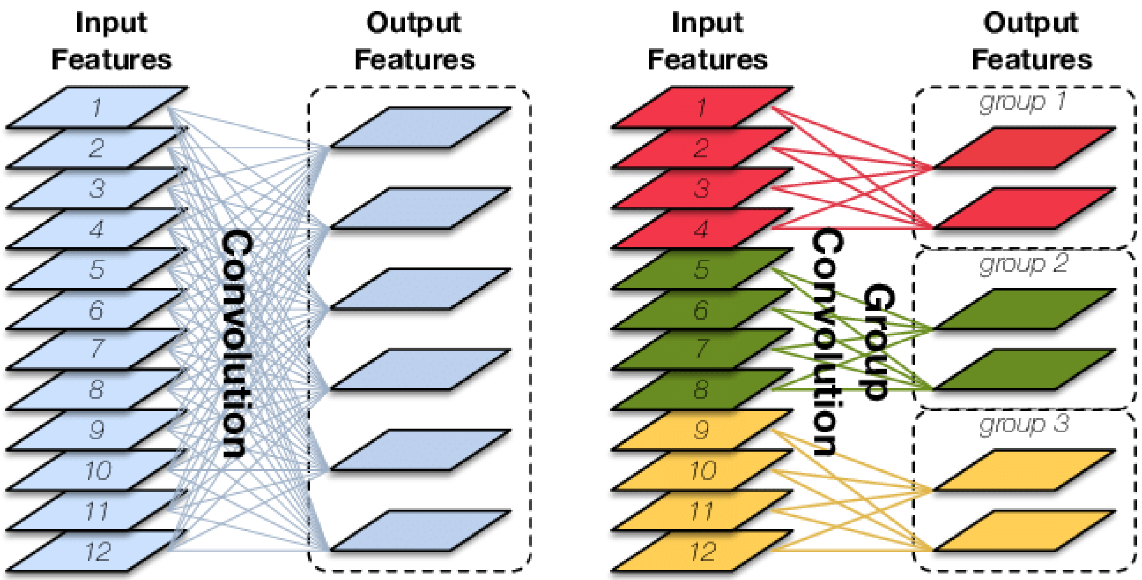


Group Conv (Krihzevsky 2012)

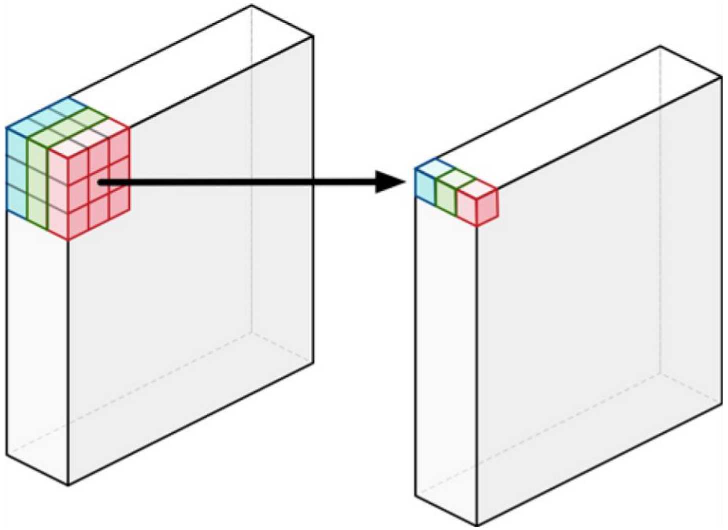


More special Convs!

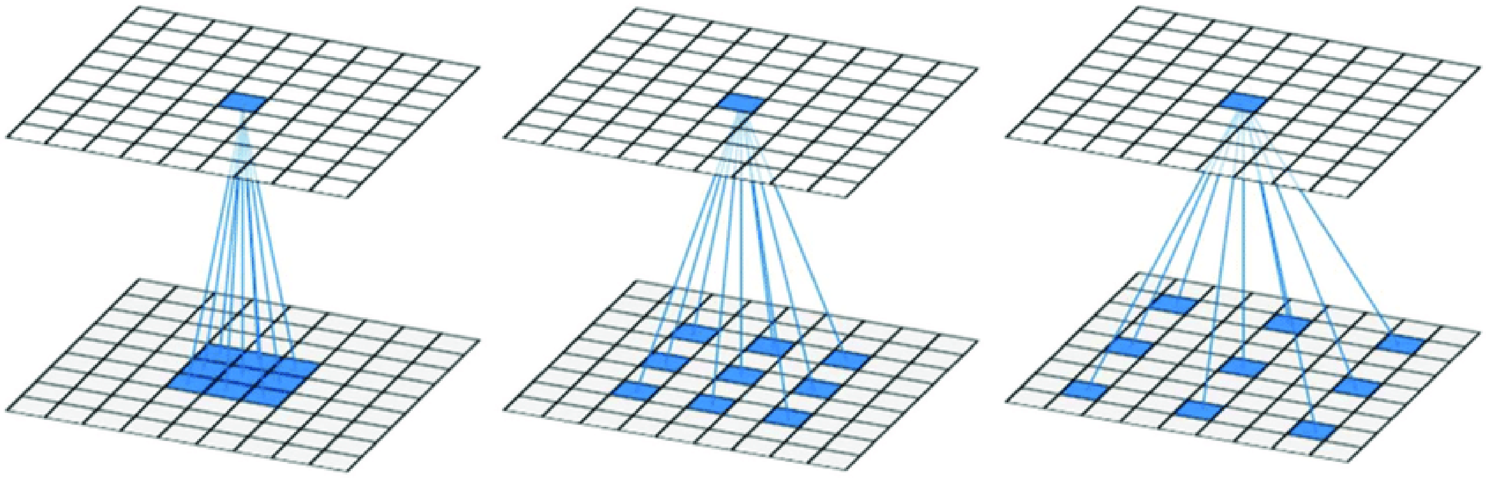
Group Conv (Krihzevsky 2012)



Depthwise Conv



Dilated Conv (Yu&Koltun 2016)



dilation=1 dilation=2 dilation=3

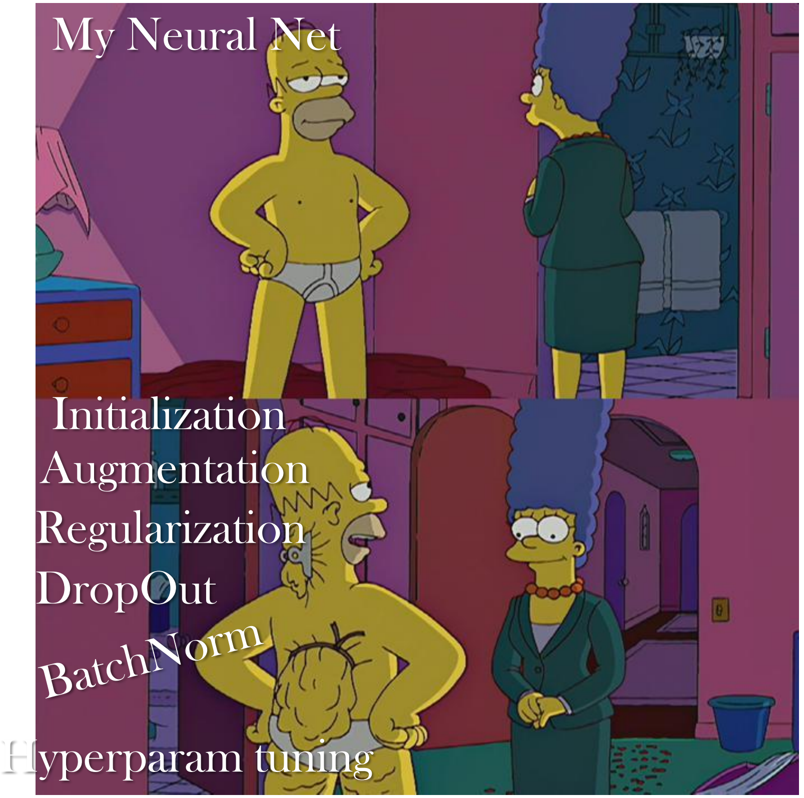
No Tutorial this week! (sigd)

No Tutorial this week! (sigd)

Next week's lecture:

Shai Bagon

Practical Training

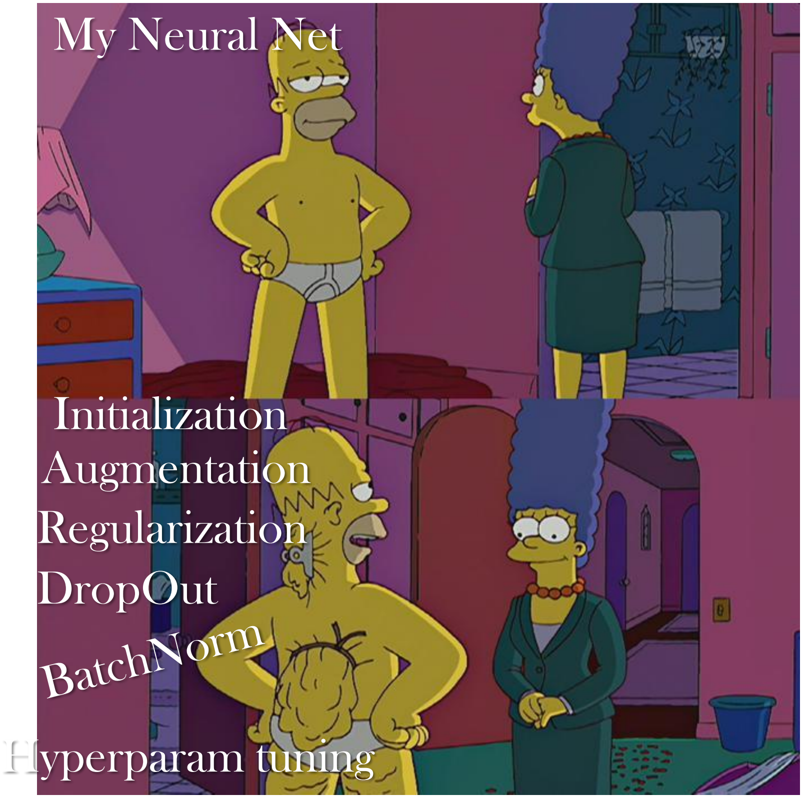


No Tutorial this week! (sigd)

Next week's lecture:

Shai Bagon

Practical Training



Dror Moran

Next week's tutorial:
CNN Architectures