CPEN 455: Deep Learning

Lecture 5: Convolutional Neural Networks I

Renjie Liao

University of British Columbia

Winter, Term 2, 2024

Outline

- Invariance & Equivariance
- Convolution
  - 1D Convolution
  - Matrix Multiplication Views
  - Translation Equivariance
  - 2D Convolution
- Convolution Variants
  - Transposed Convolution
  - Dilated Convolution
  - Grouped Convolution
  - Separable Convolution
- Pooling
- Example Architectures

Outline

- **Invariance & Equivariance**
- Convolution
  - 1D Convolution
  - Matrix Multiplication Views
  - Translation Equivariance
  - 2D Convolution
- Convolution Variants
  - Transposed Convolution
  - Dilated Convolution
  - Grouped Convolution
  - Separable Convolution
- Pooling
- Example Architectures

Invariance & Equivariance

- Invariance:

A mathematical object (or a class of mathematical objects) remains unchanged after operations or transformations of a certain type are applied to the objects

$$f(X) = f(g(X))$$

Invariance & Equivariance

- Invariance:

A mathematical object (or a class of mathematical objects) remains unchanged after operations or transformations of a certain type are applied to the objects

$$f(X) = f(g(X))$$

- Equivariance:

Applying a transformation and then computing the function produces the same result as computing the function and then applying the transformation

$$g(f(X)) = f(g(X))$$

Convolution is Translation Equivariant! We will see why shortly.

Outline

- Invariance & Equivariance
- Convolution
  - **1D Convolution**
  - Matrix Multiplication Views
  - Translation Equivariance
  - 2D Convolution
- Convolution Variants
  - Transposed Convolution
  - Dilated Convolution
  - Grouped Convolution
  - Separable Convolution
- Pooling
- Example Architectures

1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like

n

Input x

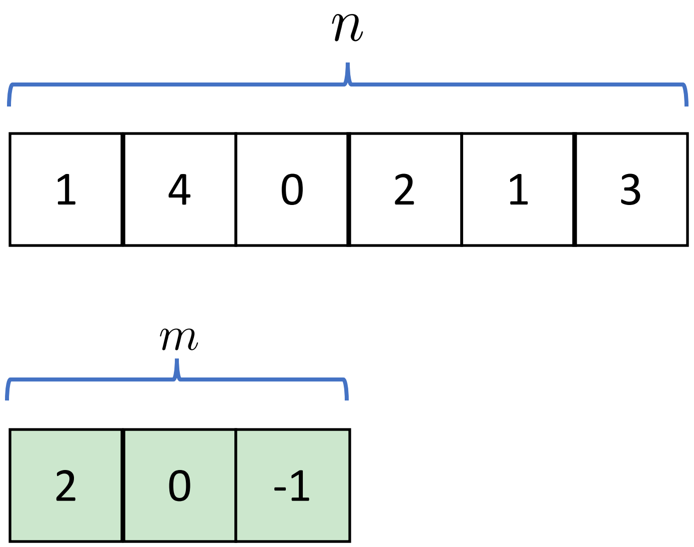
1	4	0	2	1	3

1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like

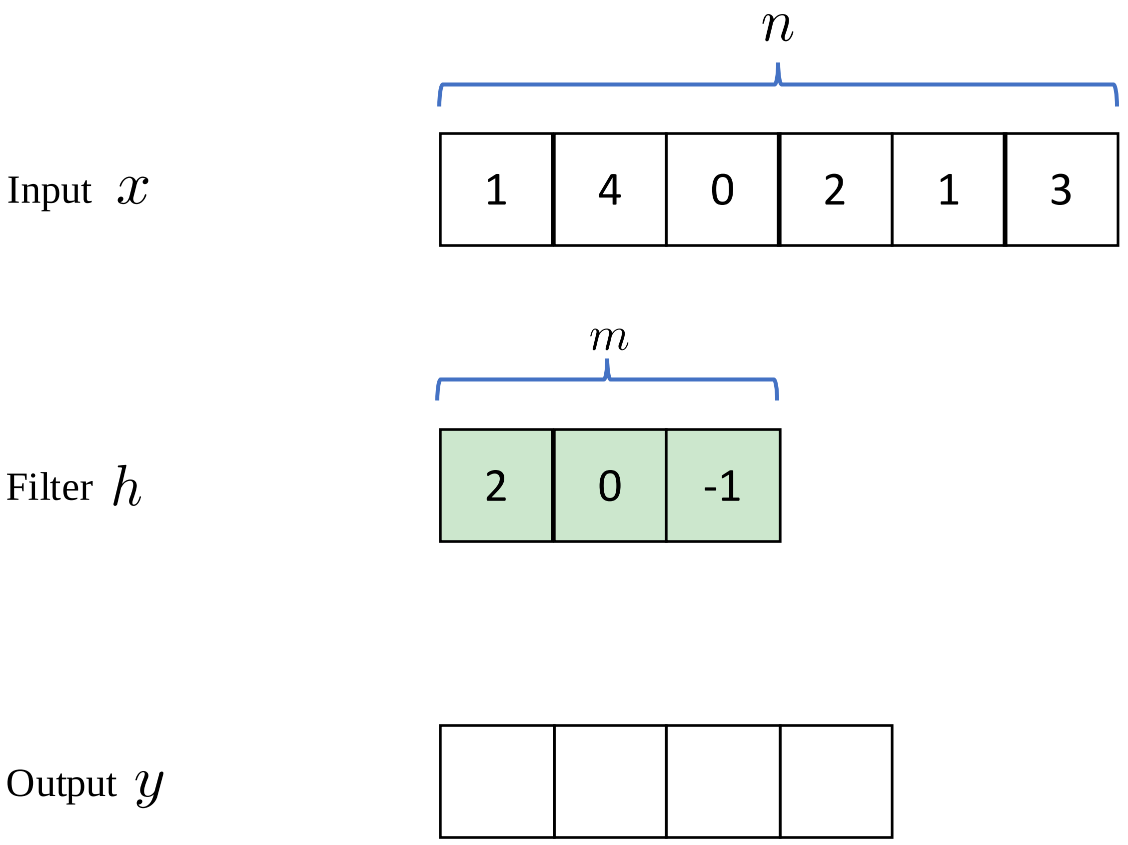
Input x

Filter h



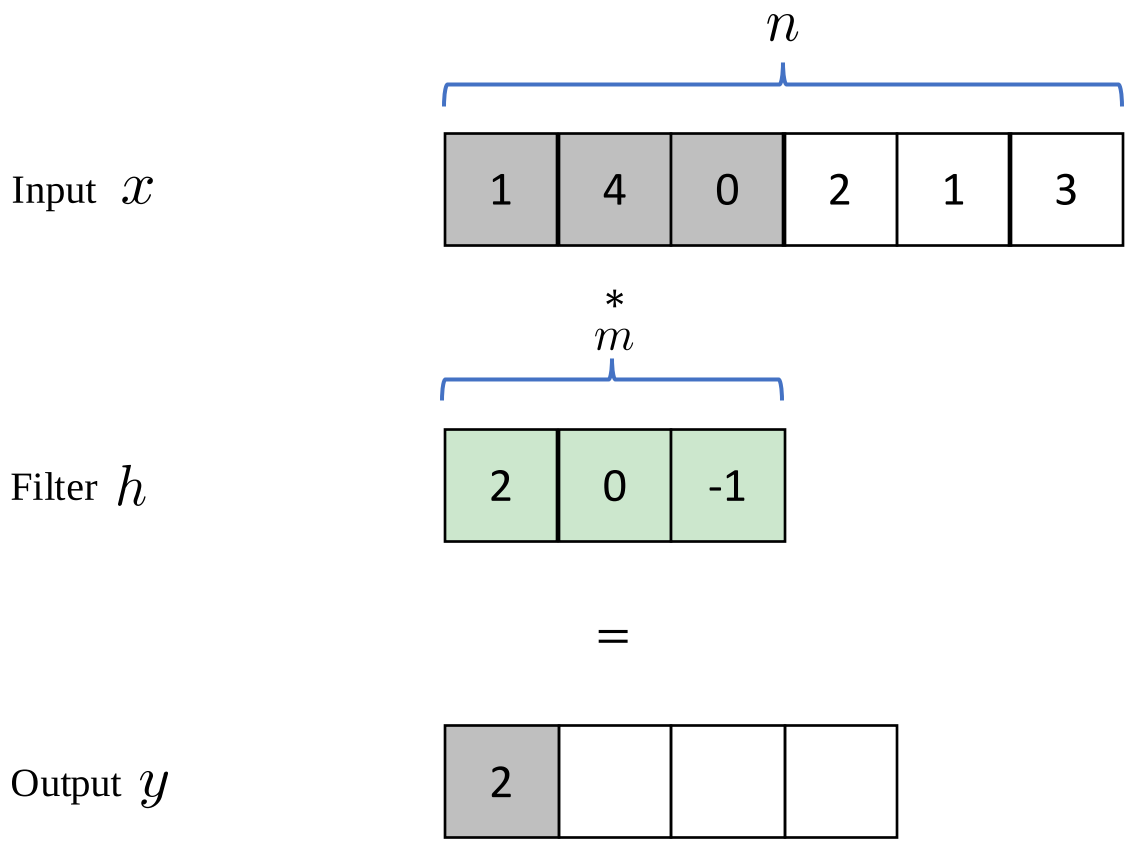
1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like



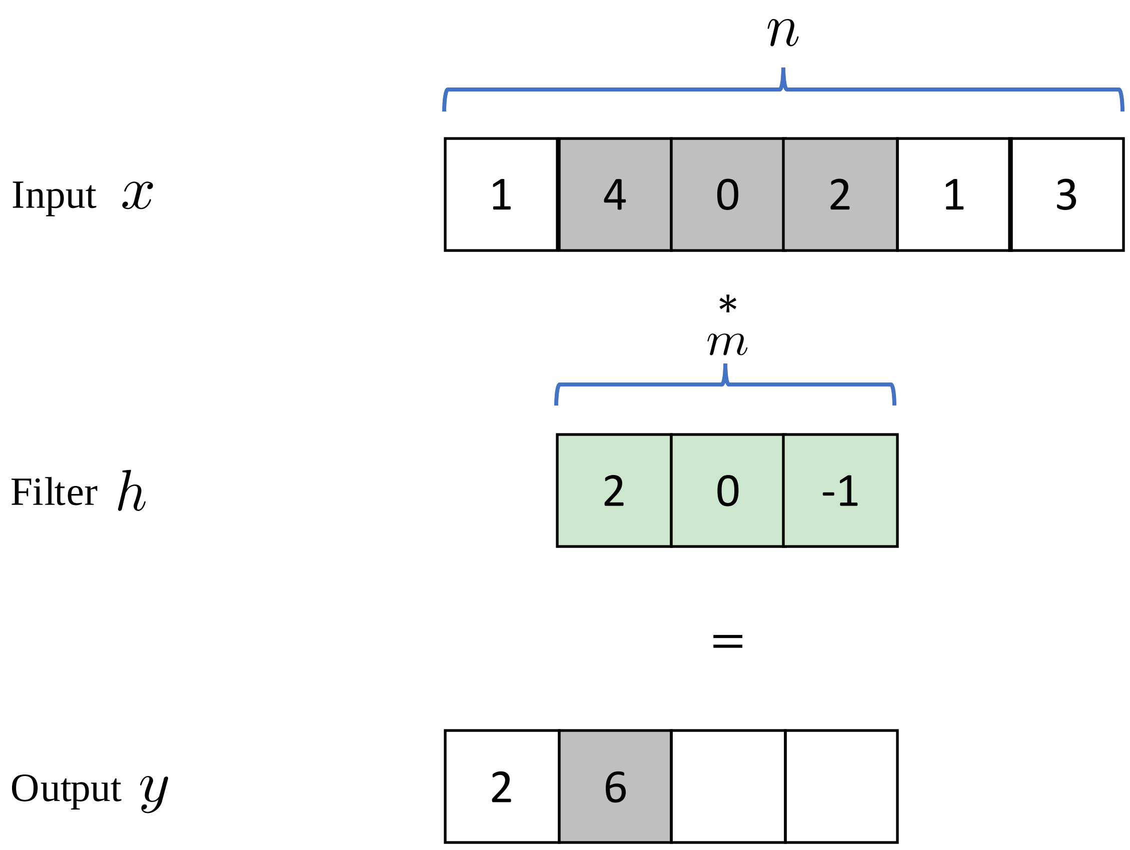
1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like



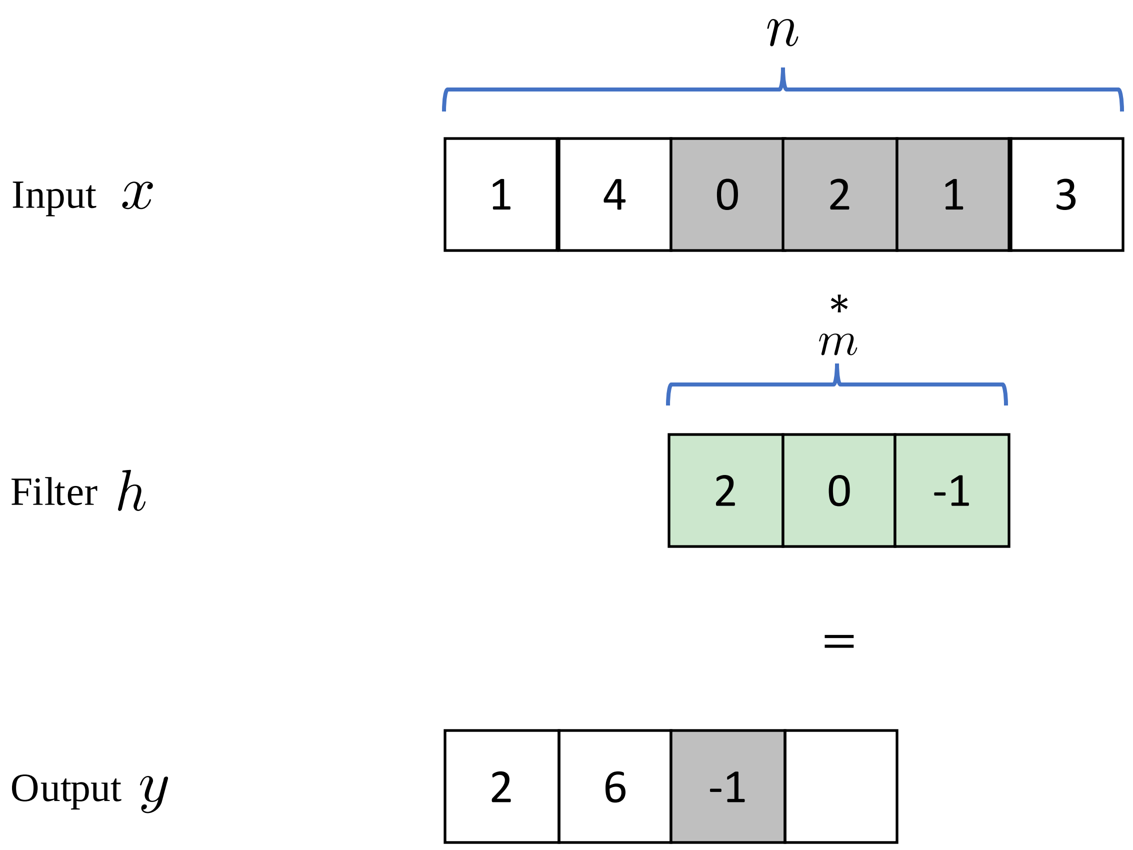
1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like



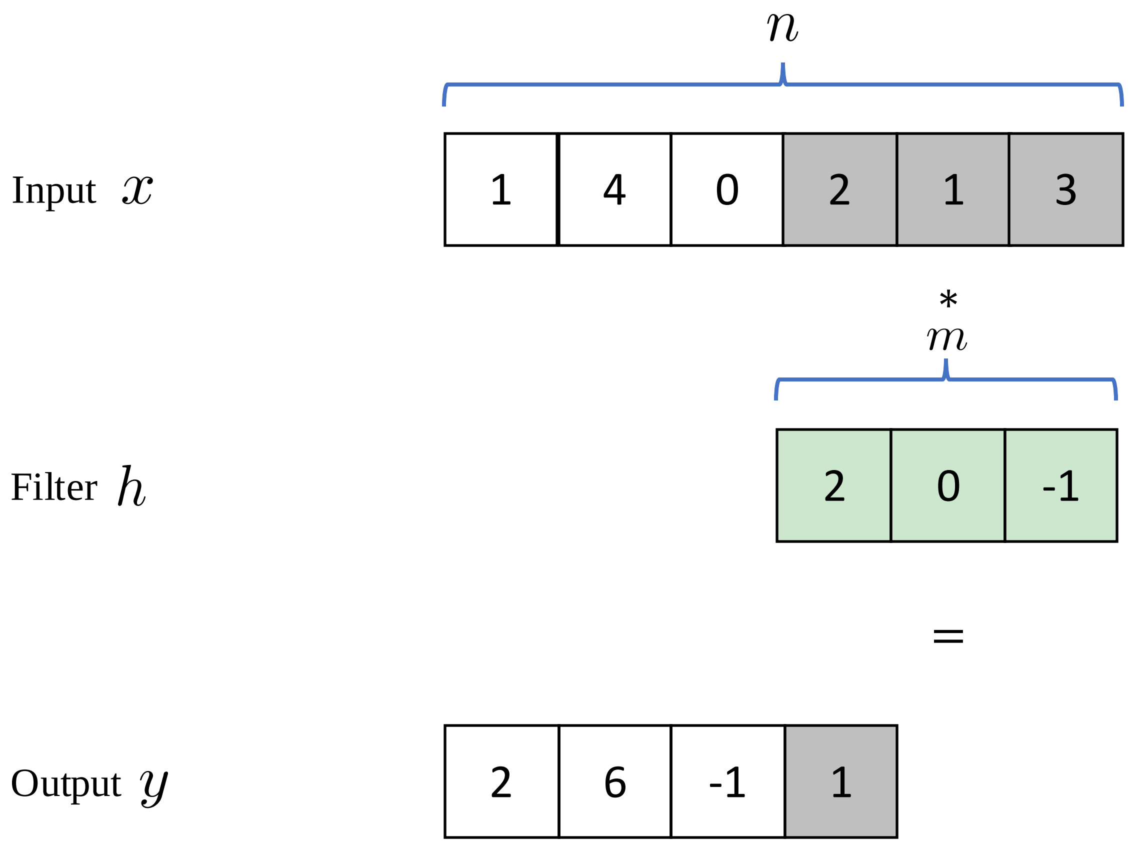
1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like



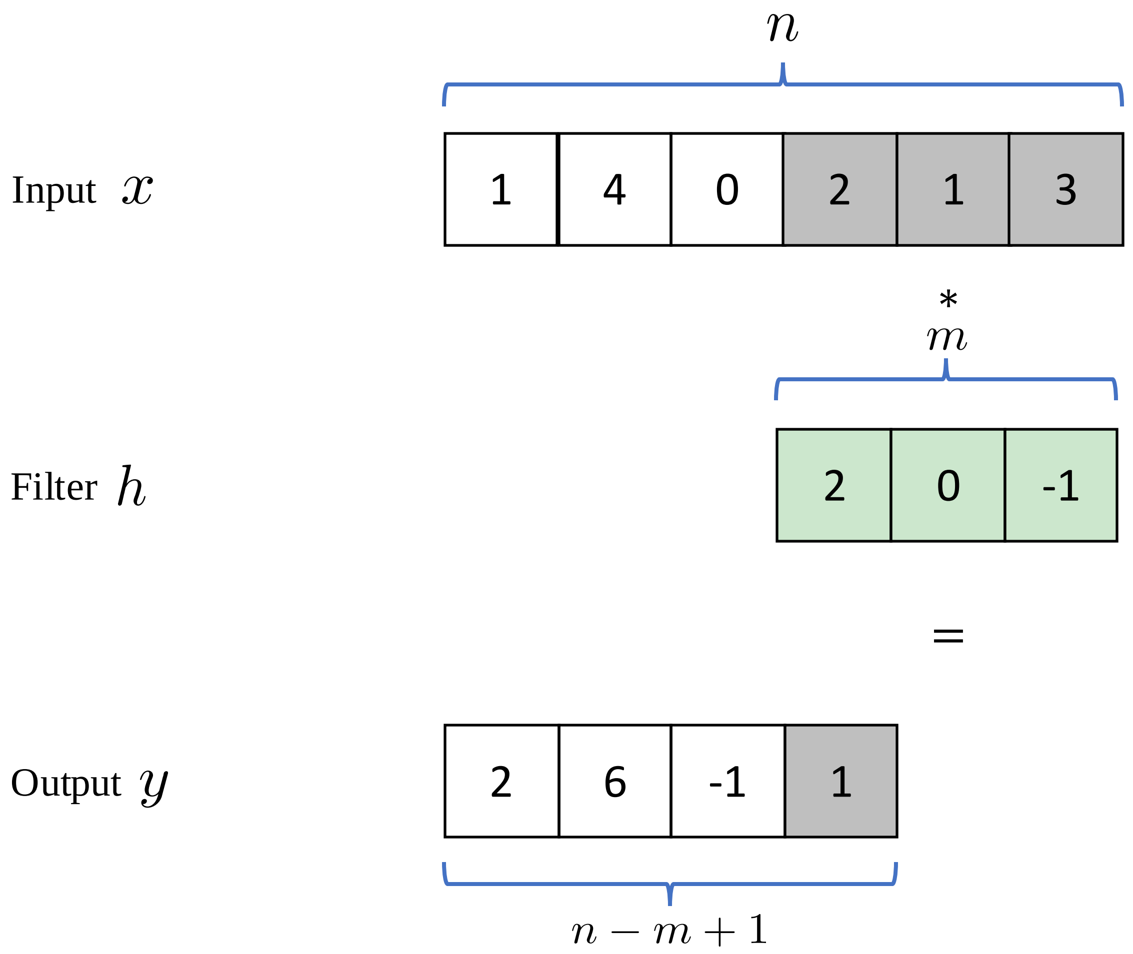
1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like



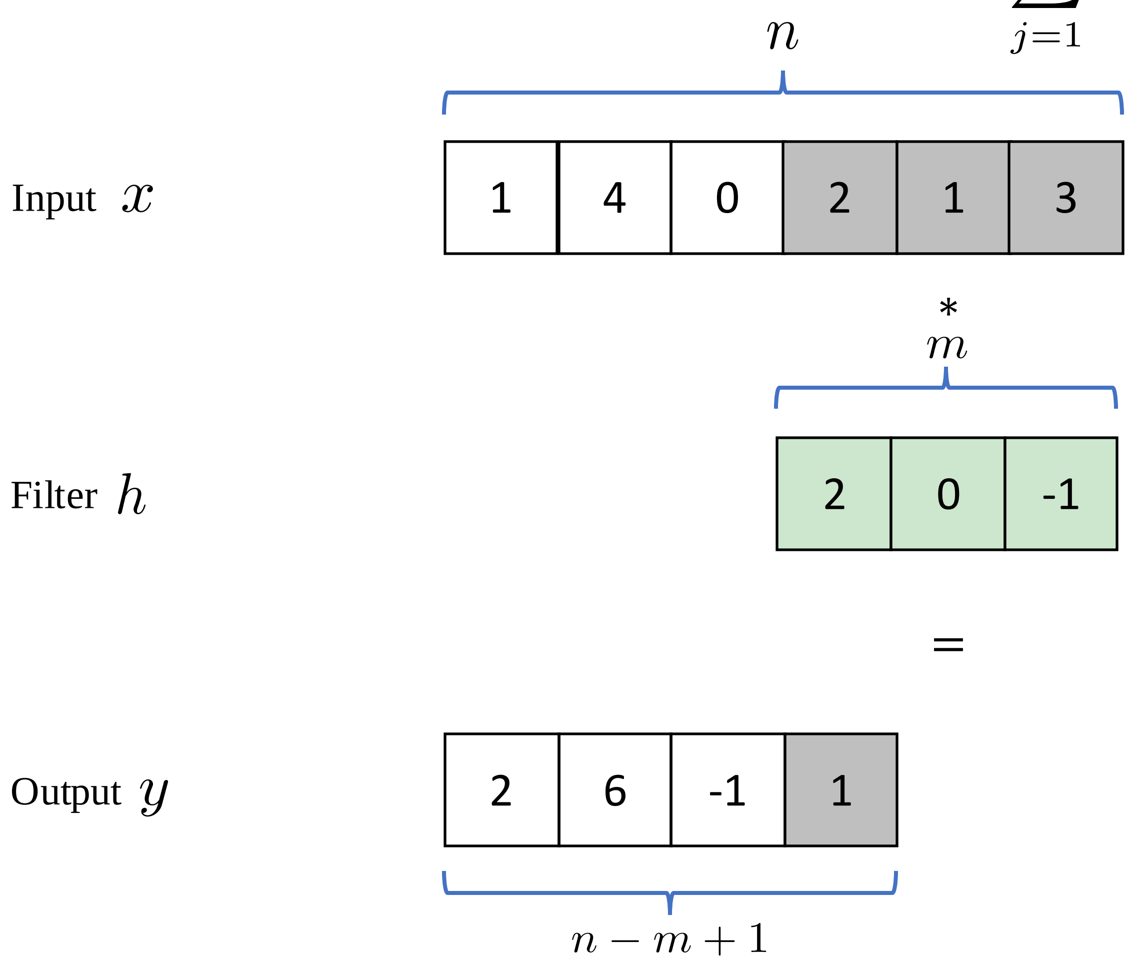
1D (Discrete) Convolutions

Let us see what 1D (Discrete) Convolution looks like



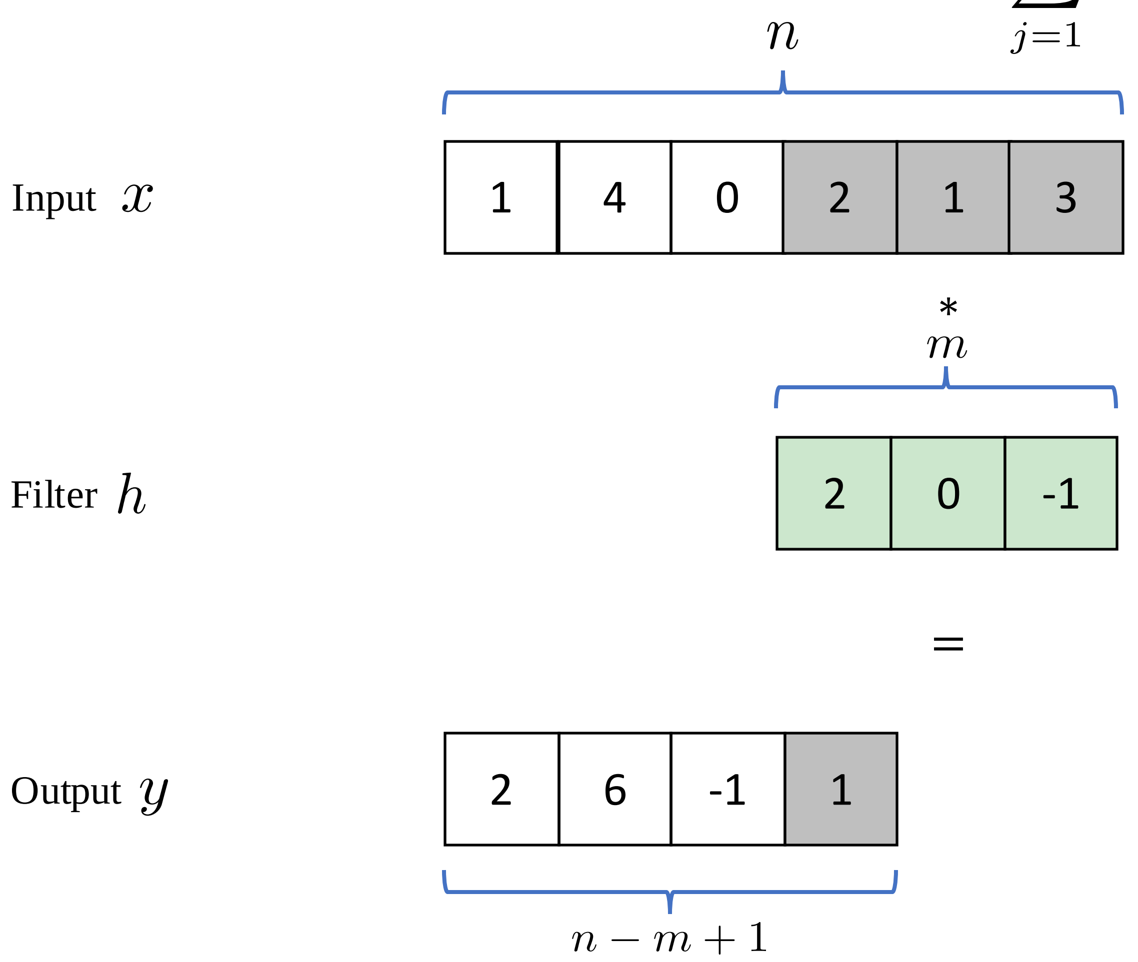
1D (Discrete) Convolutions

Formally, we denote 1D convolution as $y = h * x$ and $y[i] = \sum_{j=1}^{m} h_j x_{i+j-1}$



1D (Discrete) Convolutions

Formally, we denote 1D convolution as $y = h * x$ and $y[i] = \sum_{j=1}^{m} h_j x_{i+j-1}$



Convolution in Deep Learning is actually Cross-Correlation in Math:

$$(h * x)(t) = \int_{-\infty}^{\infty} h(\tau) x(t + \tau) \mathrm{d}\tau$$

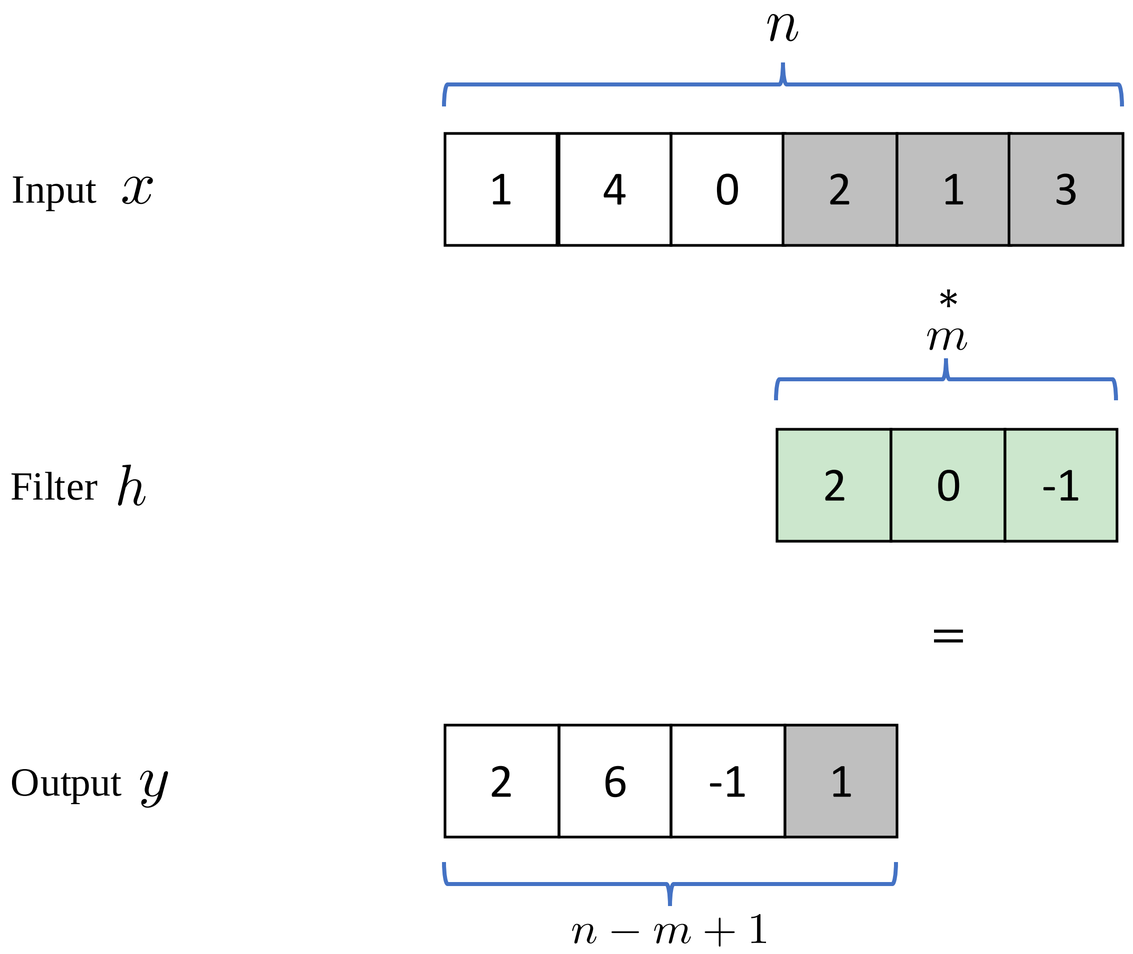
Convolution in Math:

$$(h * x)(t) = \int_{-\infty}^{\infty} h(\tau) x(t - \tau) \mathrm{d}\tau$$

They are equivalent if we reverse the order of the filter. In Deep Learning, this difference does not matter since the filter weights are free to learn!

1D (Discrete) Convolutions

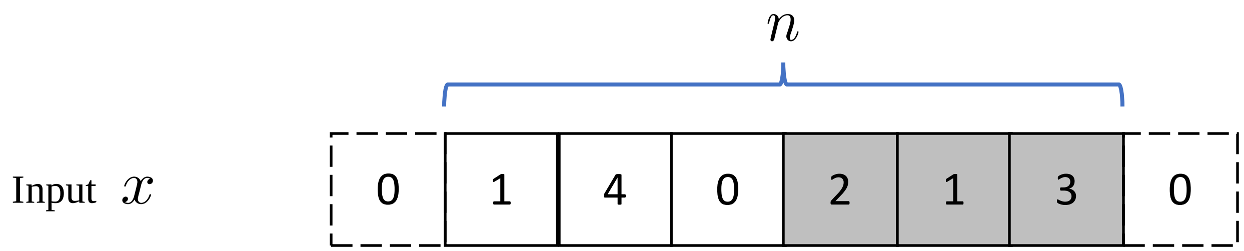
What if we hope the output to have the same shape as input?



1D (Discrete) Convolutions

What if we hope the output to have the same shape as input?

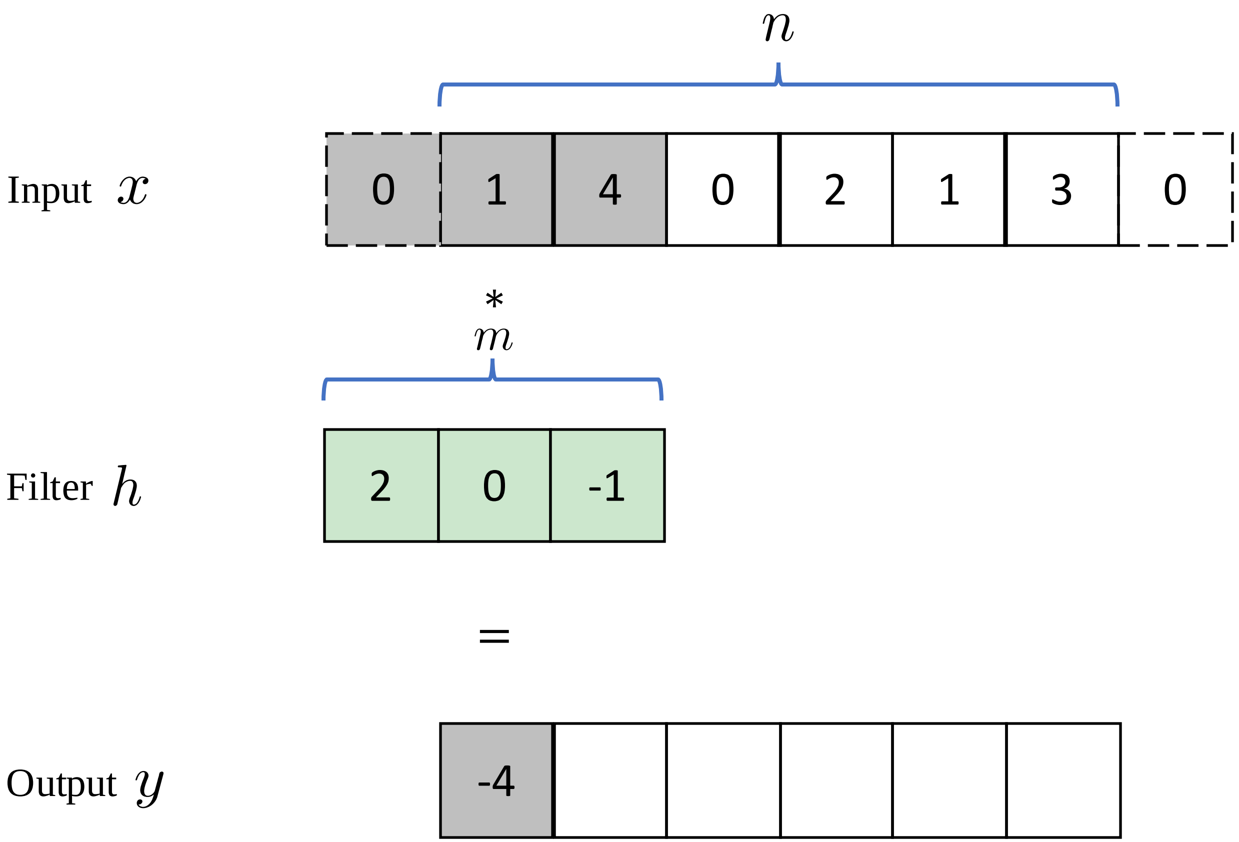
Padding!



1D (Discrete) Convolutions

What if we hope the output to have the same shape as input?

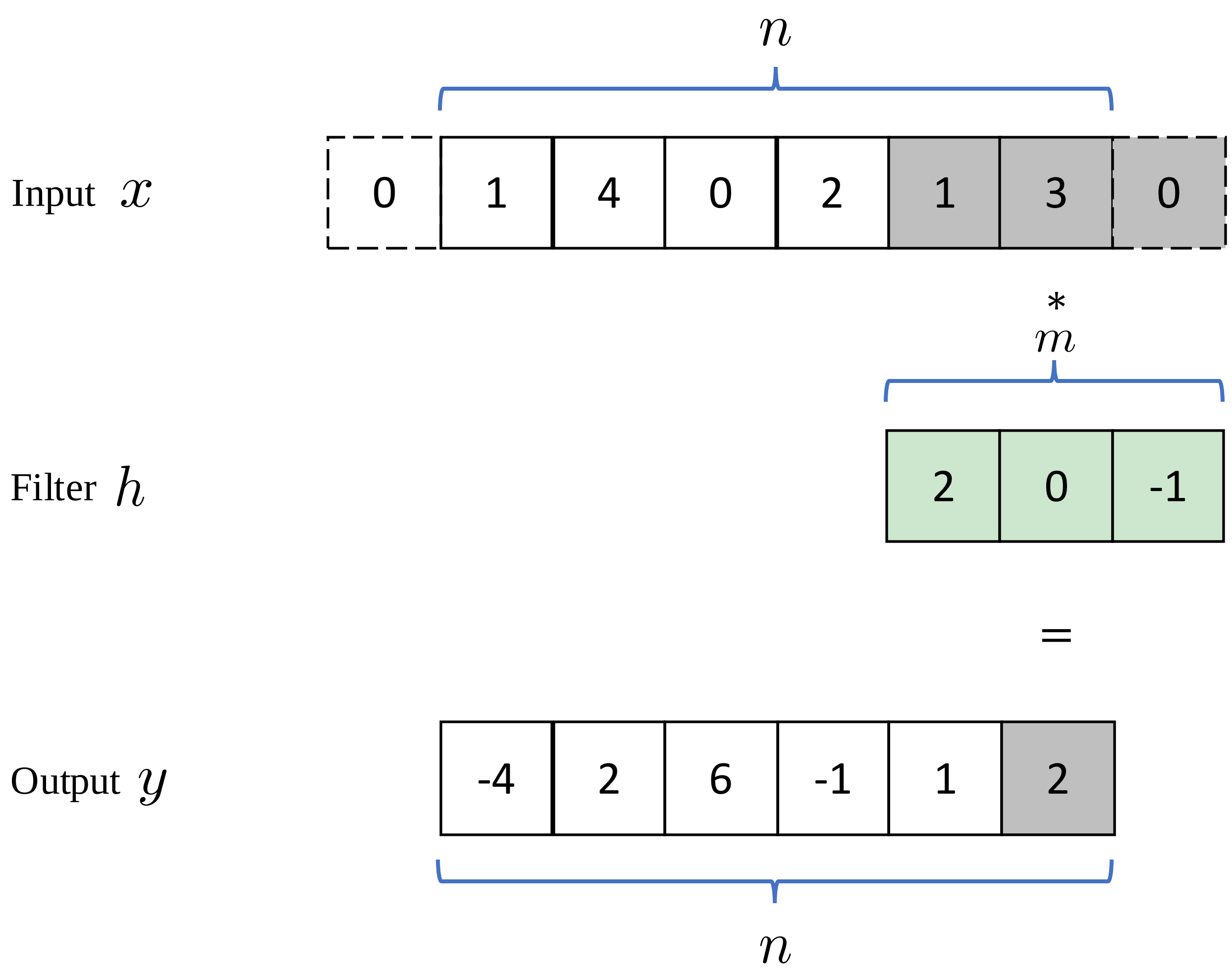
Padding!



1D (Discrete) Convolutions

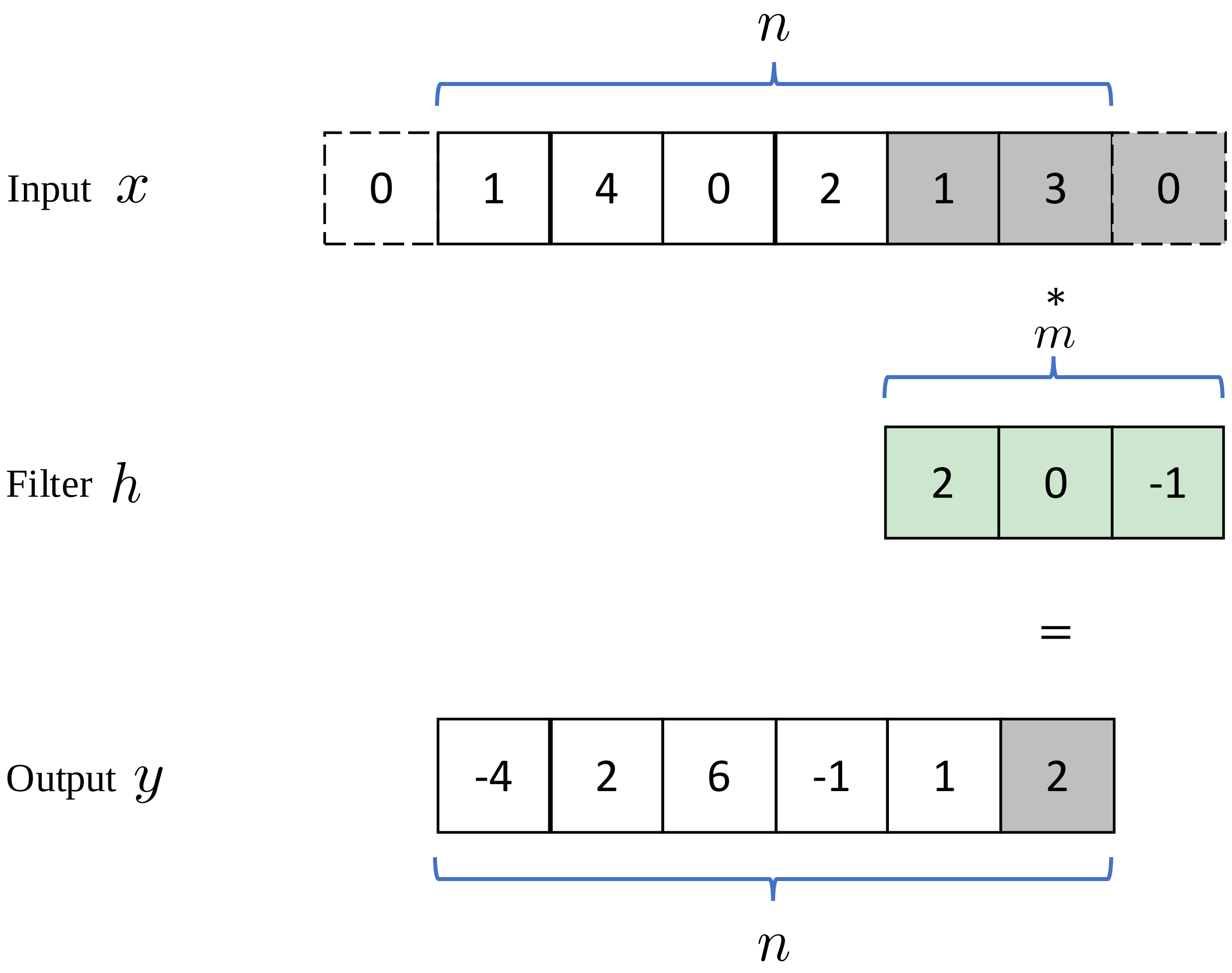
What if we hope the output to have the same shape as input?

Padding!



1D (Discrete) Convolutions

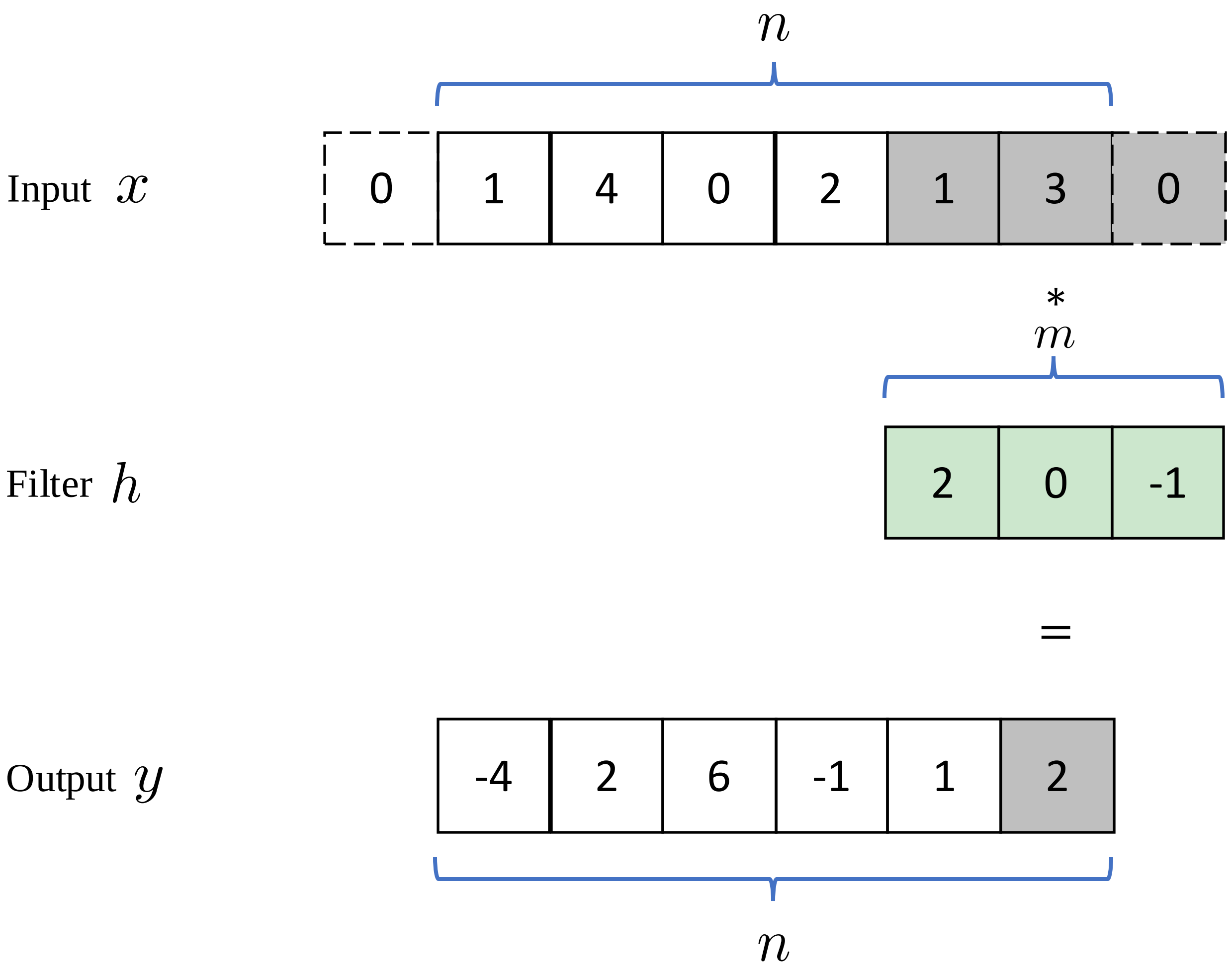
What if we hope the output to have a much smaller size compared to input?



1D (Discrete) Convolutions

What if we hope the output to have a much smaller size compared to input?

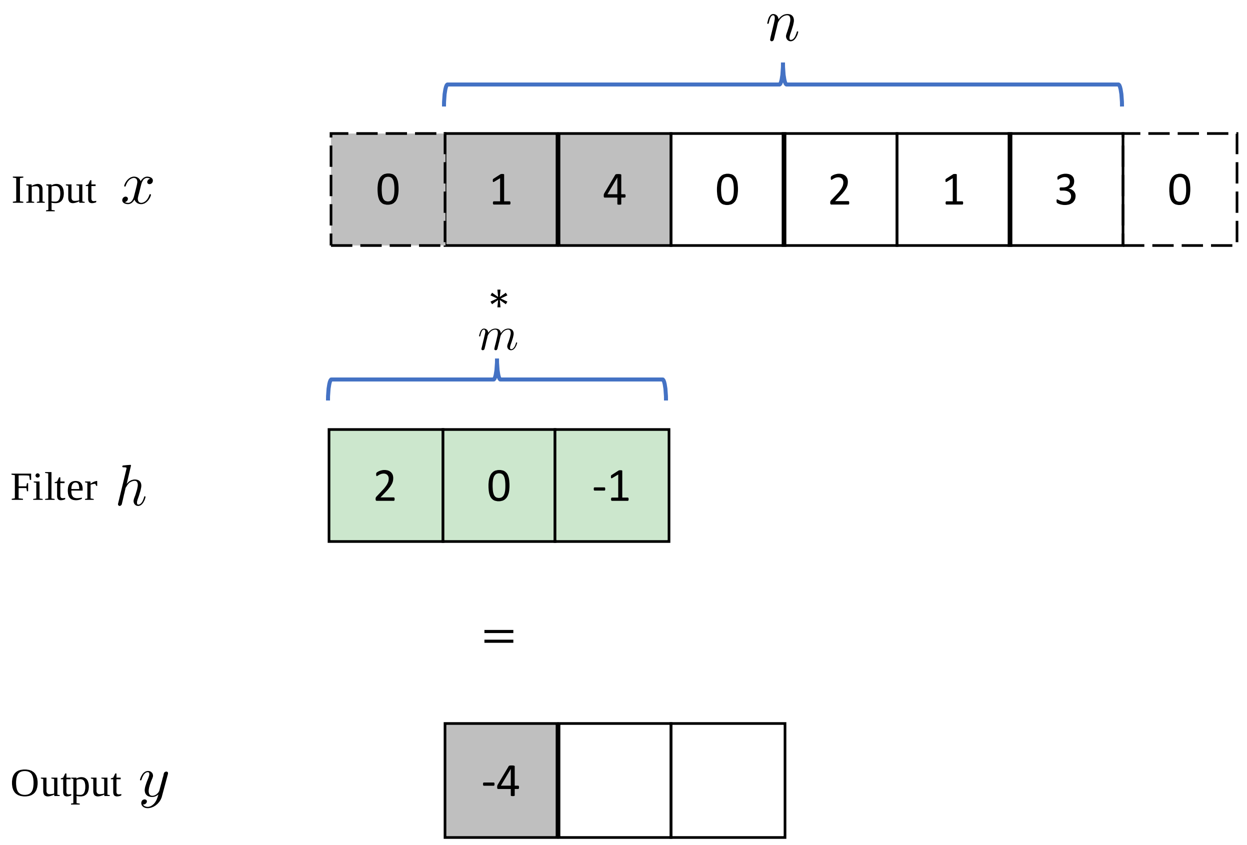
Stride!



1D (Discrete) Convolutions

What if we hope the output to have a much smaller size compared to input?

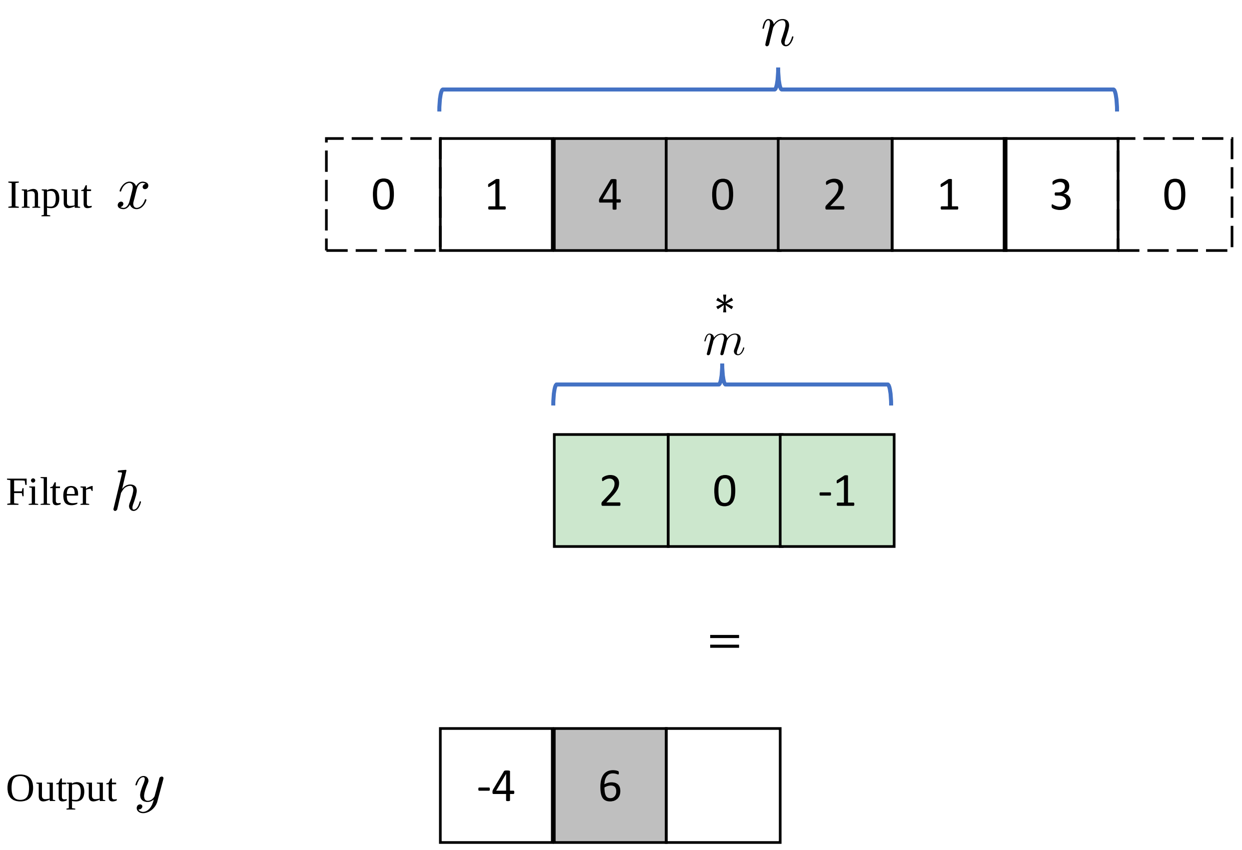
Stride = 2!



1D (Discrete) Convolutions

What if we hope the output to have a much smaller size compared to input?

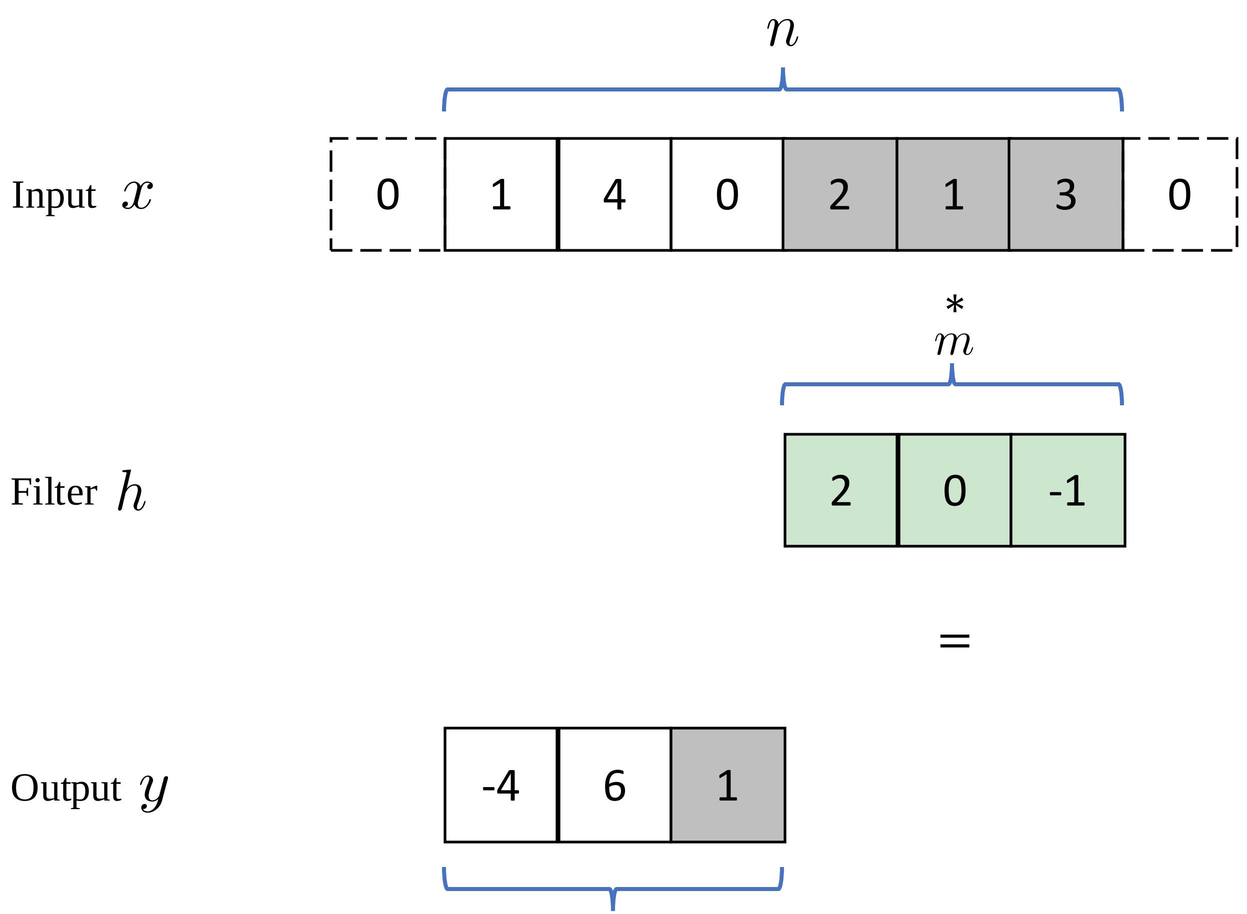
Stride = 2!



1D (Discrete) Convolutions

What if we hope the output to have a much smaller size compared to input?

Stride = 2!



$$\left\lfloor \frac{n + 2p - m}{s} \right\rfloor + 1$$

Stride: s Padding: p

Outline

- Invariance & Equivariance
- Convolution
  - 1D Convolution
  - **Matrix Multiplication Views**
  - Translation Equivariance
  - 2D Convolution
- Convolution Variants
  - Transposed Convolution
  - Dilated Convolution
  - Grouped Convolution
  - Separable Convolution
- Pooling
- Example Architectures

Matrix Multiplication View I

1D Convolution (Discrete) ⇔ Matrix Multiplication

Filter => Toeplitz matrix (diagonal-constant)

Matrix Multiplication View I

1D Convolution (Discrete) ⇔ Matrix Multiplication

Filter => Toeplitz matrix (diagonal-constant)

$$y = h * x = \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_{m-1} & h_m & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & h_1 & h_2 & \cdots & h_m & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+2} & \cdots & h_m & 0 \\ 0 & 0 & \cdots & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ \vdots \\ 0 \end{bmatrix}$$

padding: $\lfloor m/2 \rfloor$

padding: $\lfloor m/2 \rfloor$

Matrix Multiplication View I

1D Convolution (Discrete) ⬄ Matrix Multiplication

Filter => Toeplitz matrix (diagonal-constant)

$$y = h * x = \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_{m-1} & h_m & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & h_1 & h_2 & \cdots & h_m & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+2} & \cdots & h_m & 0 \\ 0 & 0 & \cdots & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ \vdots \\ 0 \end{bmatrix}$$

padding: $\lfloor m/2 \rfloor$

padding: $\lfloor m/2 \rfloor$

Input x

0	…	x_1	…	$x_{m-\lfloor m/2 \rfloor}$	…	…	x_n	…	0

Filter h

h_1	…	$h_{\lfloor m/2 \rfloor+1}$	…	h_m

Matrix Multiplication View I

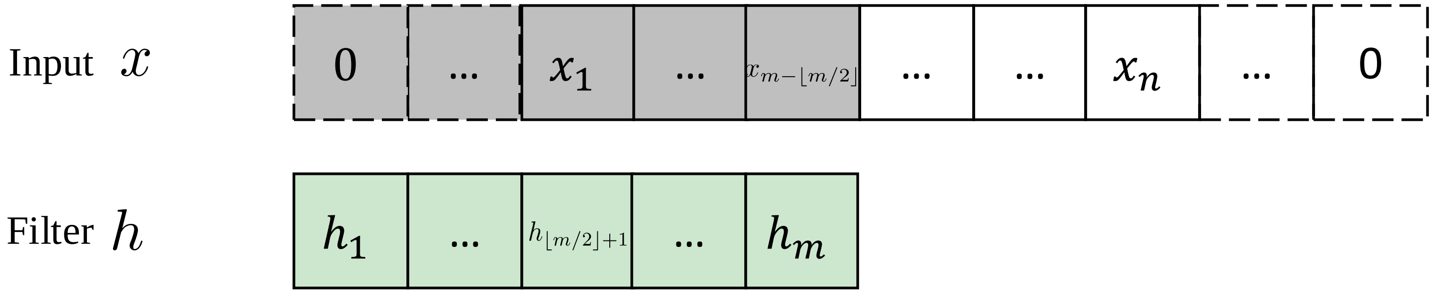
1D Convolution (Discrete) ⇔ Matrix Multiplication

Filter => Toeplitz matrix (diagonal-constant)

$$y = h * x = \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_{m-1} & h_m & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & h_1 & h_2 & \cdots & h_m & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+2} & \cdots & h_m & 0 \\ 0 & 0 & \cdots & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ \vdots \\ 0 \end{bmatrix}$$

padding: $\lfloor m/2 \rfloor$

padding: $\lfloor m/2 \rfloor$



Matrix Multiplication View I

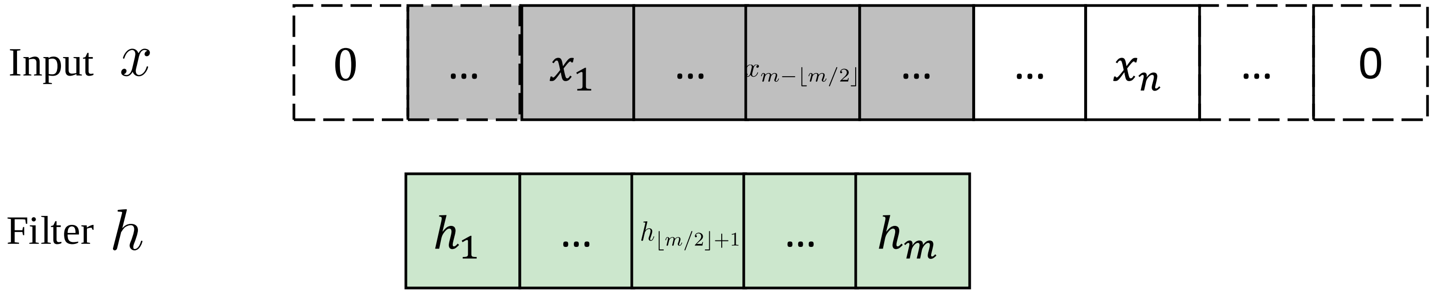
1D Convolution (Discrete) ⬄ Matrix Multiplication

Filter => Toeplitz matrix (diagonal-constant)

$$
y = h * x = \begin{bmatrix}
h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m & 0 & \cdots & \cdots & \cdots & 0 \\
0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_{m-1} & h_m & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & \cdots & \cdots & h_1 & h_2 & \cdots & h_m & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+2} & \cdots & h_m & 0 \\
0 & 0 & \cdots & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m
\end{bmatrix}
\begin{bmatrix} 0 \\ \vdots \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ \vdots \\ 0 \end{bmatrix}
$$

padding: $\lfloor m/2 \rfloor$

padding: $\lfloor m/2 \rfloor$



Matrix Multiplication View I

1D Convolution (Discrete) ⇔ Matrix Multiplication

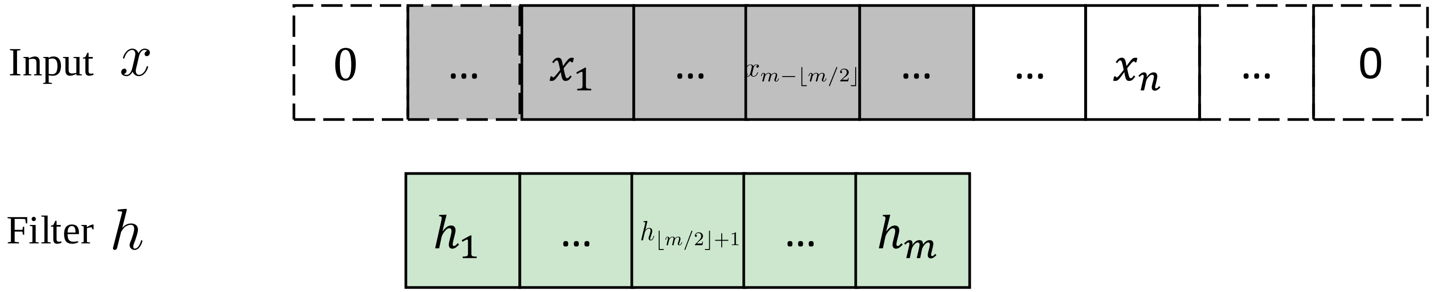
Filter => Toeplitz matrix (diagonal-constant)

It could be very sparse (e.g., when n >> m)!

$$y = h * x = \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor + 1} & \cdots & h_m & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor + 1} & \cdots & h_{m-1} & h_m & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & h_1 & h_2 & \cdots & h_m & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor + 2} & \cdots & h_m & 0 \\ 0 & 0 & \cdots & \cdots & 0 & h_1 & \cdots & h_{\lfloor m/2 \rfloor + 1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ \vdots \\ 0 \end{bmatrix}$$

padding: $\lfloor m/2 \rfloor$

padding: $\lfloor m/2 \rfloor$



Matrix Multiplication View II

1D Convolution (Discrete) ⬄ Matrix Multiplication

Data => Toeplitz matrix (diagonal-constant)

Matrix Multiplication View II

1D Convolution (Discrete) ⬄ Matrix Multiplication

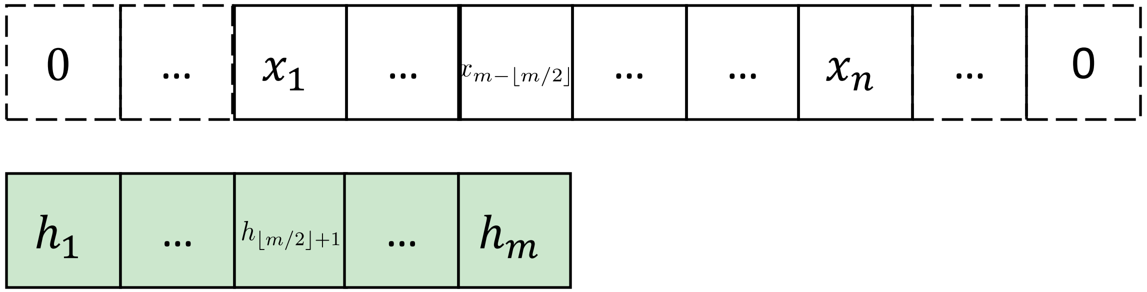
Data => Toeplitz matrix (diagonal-constant)

$$
\begin{aligned}
y^\top &= (h * x)^\top \\
&= \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor + 1} & \cdots & h_m \end{bmatrix}
\begin{bmatrix}
0 & 0 & \cdots & x_1 & \cdots & x_{n-\lfloor m/2 \rfloor - 1} & x_{n-\lfloor m/2 \rfloor} \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\
0 & x_1 & \cdots & x_{\lfloor m/2 \rfloor} & \cdots & x_{n-2} & x_{n-1} \\
x_1 & x_2 & \cdots & x_{\lfloor m/2 \rfloor + 1} & \cdots & x_{n-1} & x_n \\
x_2 & x_3 & \cdots & x_{\lfloor m/2 \rfloor + 2} & \cdots & x_n & 0 \\
\vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\
x_{m-\lfloor m/2 \rfloor} & x_{m-\lfloor m/2 \rfloor + 1} & \cdots & x_m & \cdots & 0 & 0
\end{bmatrix}
\end{aligned}
$$

padding: $\lfloor m/2 \rfloor$

Input x

Filter h



Matrix Multiplication View II

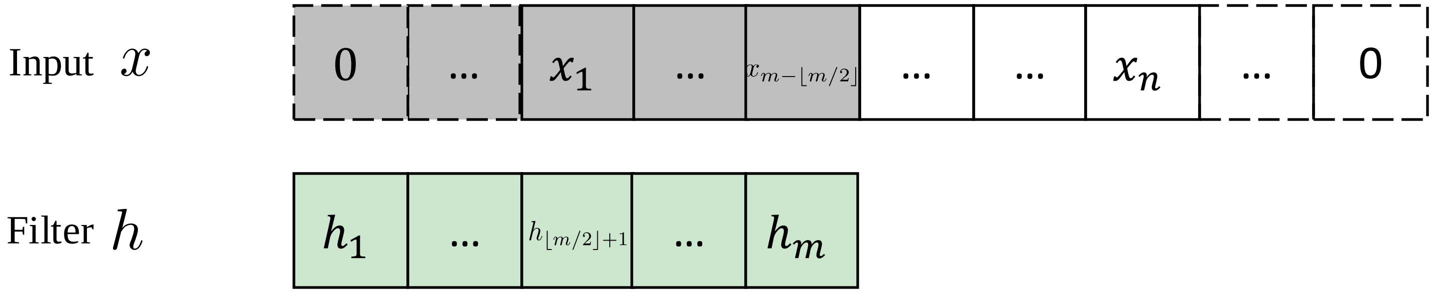
1D Convolution (Discrete) ⇔ Matrix Multiplication

Data => Toeplitz matrix (diagonal-constant)

$$y^\top = (h * x)^\top$$

padding: $\lfloor m/2 \rfloor$

$$= \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 & 0 & \cdots & x_1 & \cdots & x_{n-\lfloor m/2 \rfloor-1} & x_{n-\lfloor m/2 \rfloor} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ 0 & x_1 & \cdots & x_{\lfloor m/2 \rfloor} & \cdots & x_{n-2} & x_{n-1} \\ x_1 & x_2 & \cdots & x_{\lfloor m/2 \rfloor+1} & \cdots & x_{n-1} & x_n \\ x_2 & x_3 & \cdots & x_{\lfloor m/2 \rfloor+2} & \cdots & x_n & 0 \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ x_{m-\lfloor m/2 \rfloor} & x_{m-\lfloor m/2 \rfloor+1} & \cdots & x_m & \cdots & 0 & 0 \end{bmatrix}$$



Matrix Multiplication View II

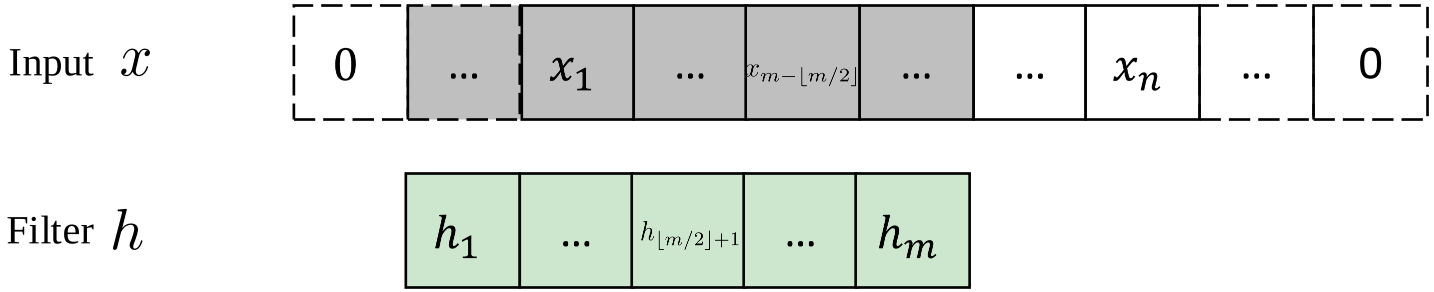
1D Convolution (Discrete) ⬄ Matrix Multiplication

Data => Toeplitz matrix (diagonal-constant)

$$y^\top = (h * x)^\top$$

padding: $\lfloor m/2 \rfloor$

$$= \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor + 1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 & 0 & \cdots & x_1 & \cdots & x_{n-\lfloor m/2 \rfloor - 1} & x_{n-\lfloor m/2 \rfloor} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ 0 & x_1 & \cdots & x_{\lfloor m/2 \rfloor} & \cdots & x_{n-2} & x_{n-1} \\ x_1 & x_2 & \cdots & x_{\lfloor m/2 \rfloor + 1} & \cdots & x_{n-1} & x_n \\ x_2 & x_3 & \cdots & x_{\lfloor m/2 \rfloor + 2} & \cdots & x_n & 0 \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ x_{m-\lfloor m/2 \rfloor} & x_{m-\lfloor m/2 \rfloor + 1} & \cdots & x_m & \cdots & 0 & 0 \end{bmatrix}$$



Matrix Multiplication View II

1D Convolution (Discrete) ⇔ Matrix Multiplication

Data => Toeplitz matrix (diagonal-constant)

It could be dense (e.g., when n >> m)!

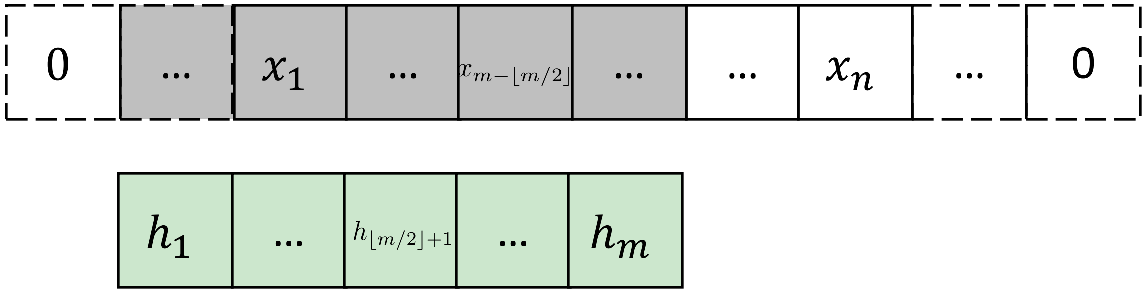
$$y^\top = (h * x)^\top$$

padding: $\lfloor m/2 \rfloor$

$$= \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor + 1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 & 0 & \cdots & x_1 & \cdots & x_{n-\lfloor m/2 \rfloor -1} & x_{n-\lfloor m/2 \rfloor} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ 0 & x_1 & \cdots & x_{\lfloor m/2 \rfloor} & \cdots & x_{n-2} & x_{n-1} \\ x_1 & x_2 & \cdots & x_{\lfloor m/2 \rfloor + 1} & \cdots & x_{n-1} & x_n \\ x_2 & x_3 & \cdots & x_{\lfloor m/2 \rfloor + 2} & \cdots & x_n & 0 \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ x_{m-\lfloor m/2 \rfloor} & x_{m-\lfloor m/2 \rfloor + 1} & \cdots & x_m & \cdots & 0 & 0 \end{bmatrix}$$

Input x

Filter h



Matrix Multiplication View II

1D Convolution (Discrete) ⬄ Matrix Multiplication

Data => Toeplitz matrix (diagonal-constant)

It could be dense (e.g., when n >> m)!

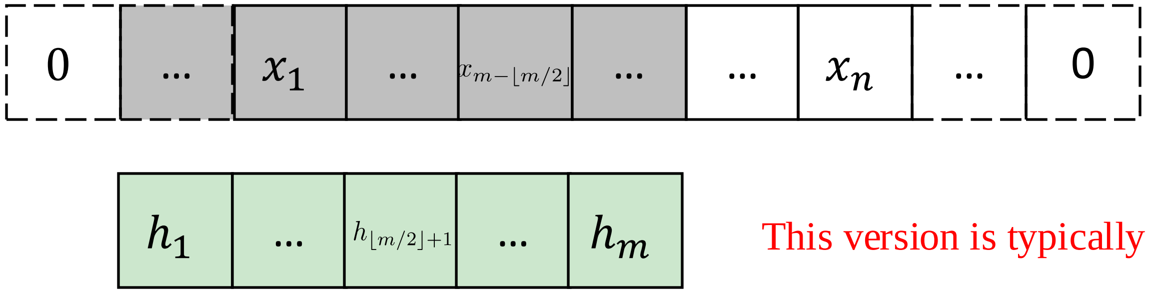
$$y^\top = (h * x)^\top$$

padding: $\lfloor m/2 \rfloor$

$$= \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor+1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 & 0 & \cdots & x_1 & \cdots & x_{n-\lfloor m/2 \rfloor-1} & x_{n-\lfloor m/2 \rfloor} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ 0 & x_1 & \cdots & x_{\lfloor m/2 \rfloor} & \cdots & x_{n-2} & x_{n-1} \\ x_1 & x_2 & \cdots & x_{\lfloor m/2 \rfloor+1} & \cdots & x_{n-1} & x_n \\ x_2 & x_3 & \cdots & x_{\lfloor m/2 \rfloor+2} & \cdots & x_n & 0 \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ x_{m-\lfloor m/2 \rfloor} & x_{m-\lfloor m/2 \rfloor+1} & \cdots & x_m & \cdots & 0 & 0 \end{bmatrix}$$

Input x

Filter h



This version is typically implemented on GPUs!

Matrix Multiplication View II

1D Convolution (Discrete) ⬄ Matrix Multiplication

Data => Toeplitz matrix (diagonal-constant)

This equivalence holds for 2D and other higher-order convolutions!

It could be dense (e.g., when n >> m)!

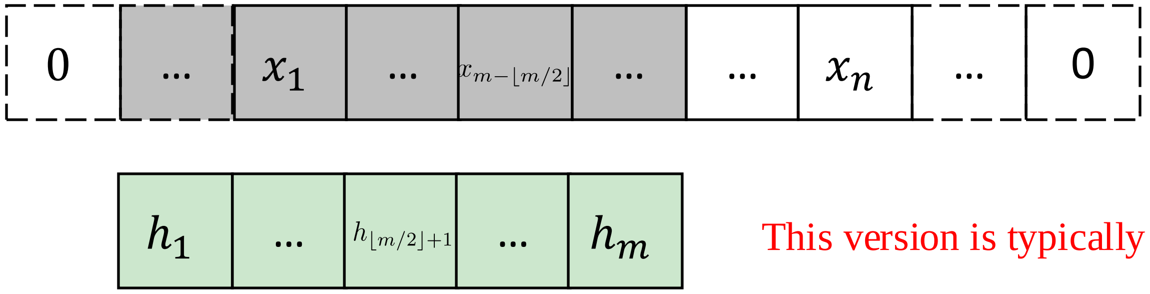
$$y^\top = (h * x)^\top$$

padding: $\lfloor m/2 \rfloor$

$$= \begin{bmatrix} h_1 & \cdots & h_{\lfloor m/2 \rfloor + 1} & \cdots & h_m \end{bmatrix} \begin{bmatrix} 0 & 0 & \cdots & x_1 & \cdots & x_{n-\lfloor m/2 \rfloor - 1} & x_{n-\lfloor m/2 \rfloor} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ 0 & x_1 & \cdots & x_{\lfloor m/2 \rfloor} & \cdots & x_{n-2} & x_{n-1} \\ x_1 & x_2 & \cdots & x_{\lfloor m/2 \rfloor + 1} & \cdots & x_{n-1} & x_n \\ x_2 & x_3 & \cdots & x_{\lfloor m/2 \rfloor + 2} & \cdots & x_n & 0 \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \vdots \\ x_{m-\lfloor m/2 \rfloor} & x_{m-\lfloor m/2 \rfloor + 1} & \cdots & x_m & \cdots & 0 & 0 \end{bmatrix}$$

Input x

Filter h



This version is typically implemented on GPUs!

Outline

- Invariance & Equivariance
- Convolution
  - 1D Convolution
  - Matrix Multiplication Views
  - **Translation Equivariance**
  - 2D Convolution
- Convolution Variants
  - Transposed Convolution
  - Dilated Convolution
  - Grouped Convolution
  - Separable Convolution
- Pooling
- Example Architectures

1D (Discrete) Convolutions

Matrix multiplication view (Filter => Toeplitz matrix) of 1D convolution:

Consider a special Toeplitz matrix: circulant matrix (must be square!)

Convolution with padding

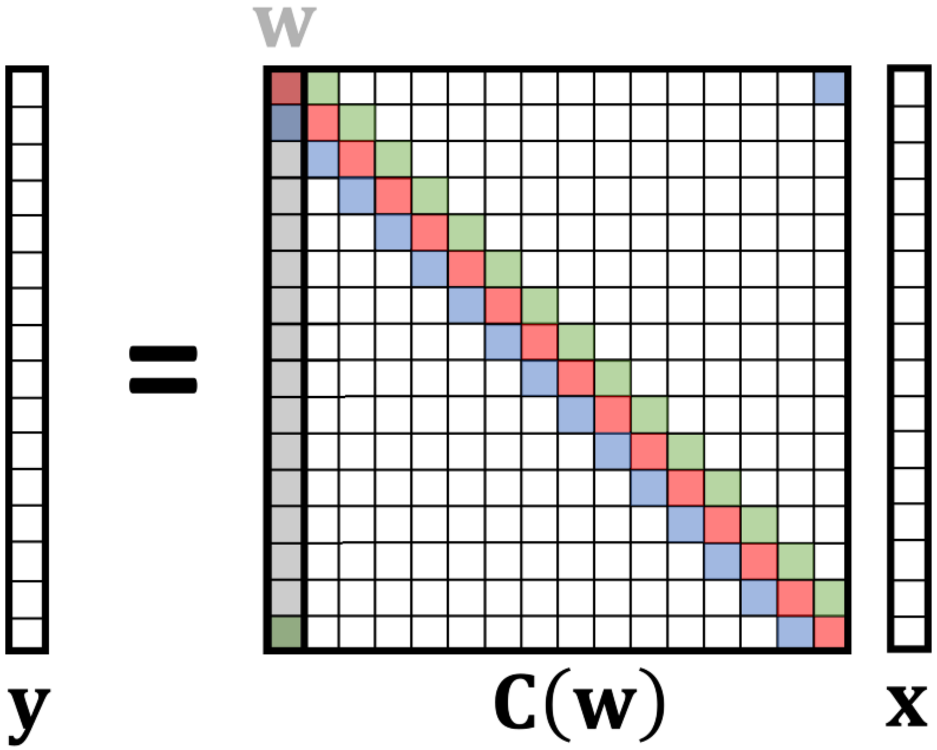


Image Credit: [1]

Translation/Shift Operator

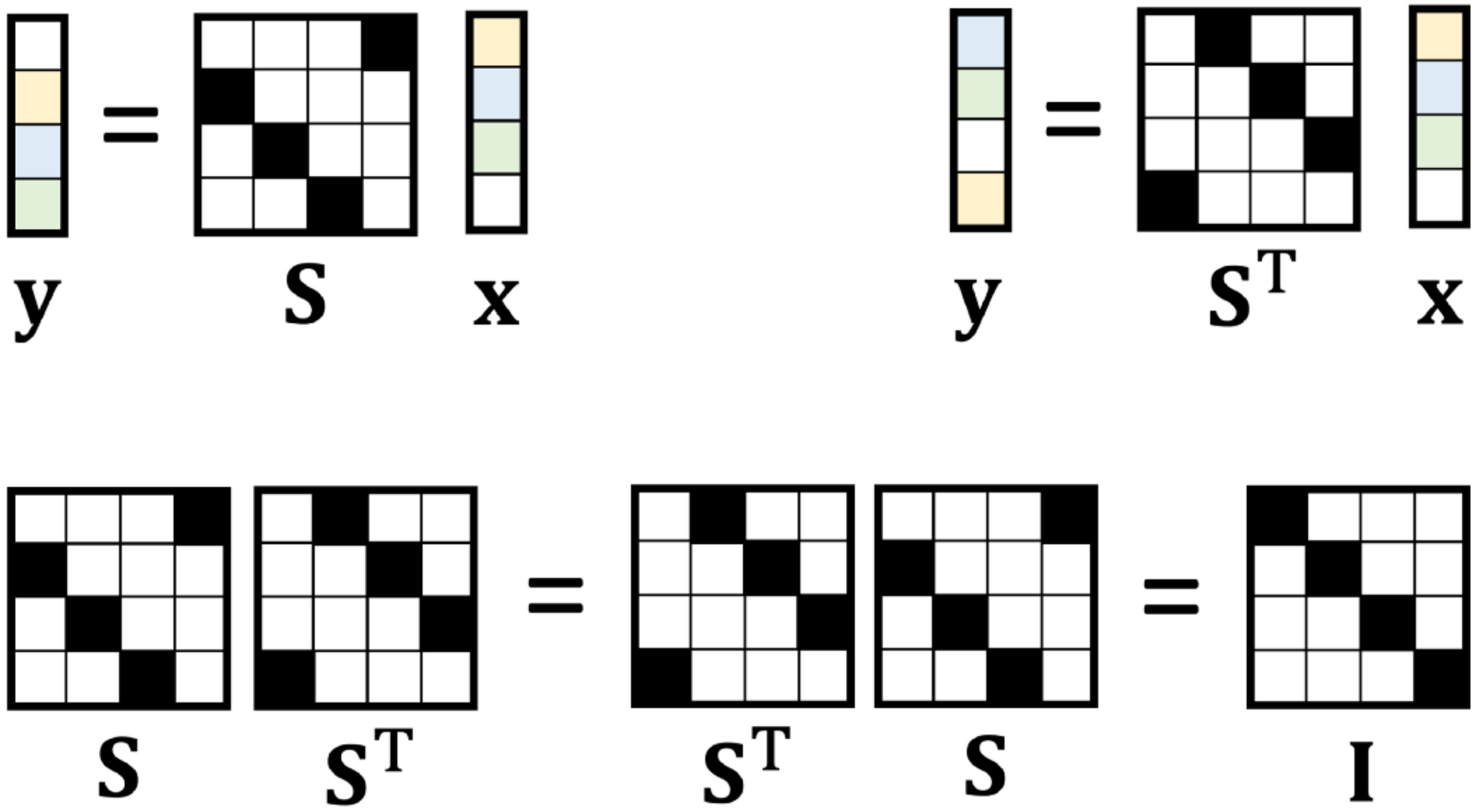


Image Credit: [1]

Translation/Shift Operator

Shift operator is also a circulant matrix!

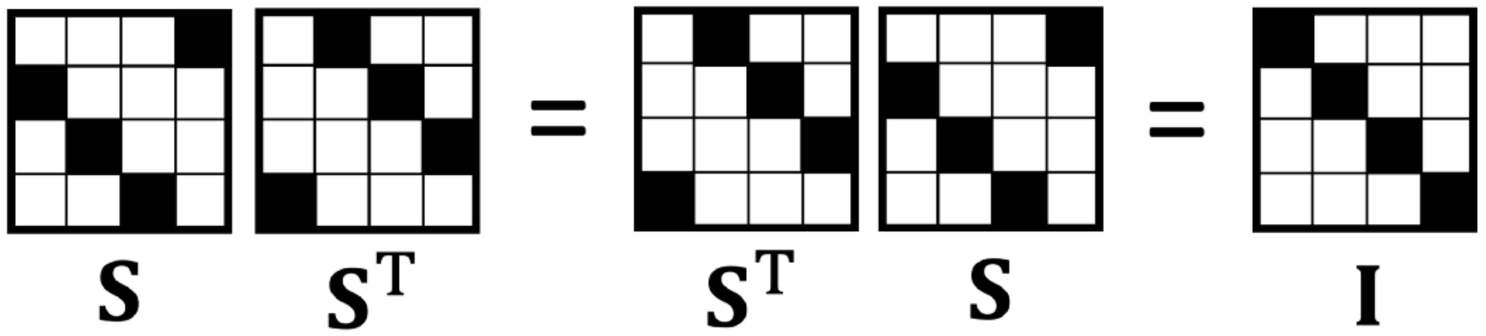


Image Credit: [1]

Translation/Shift Equivariance

Matrix multiplication is non-commutative. But not for circulant matrices!

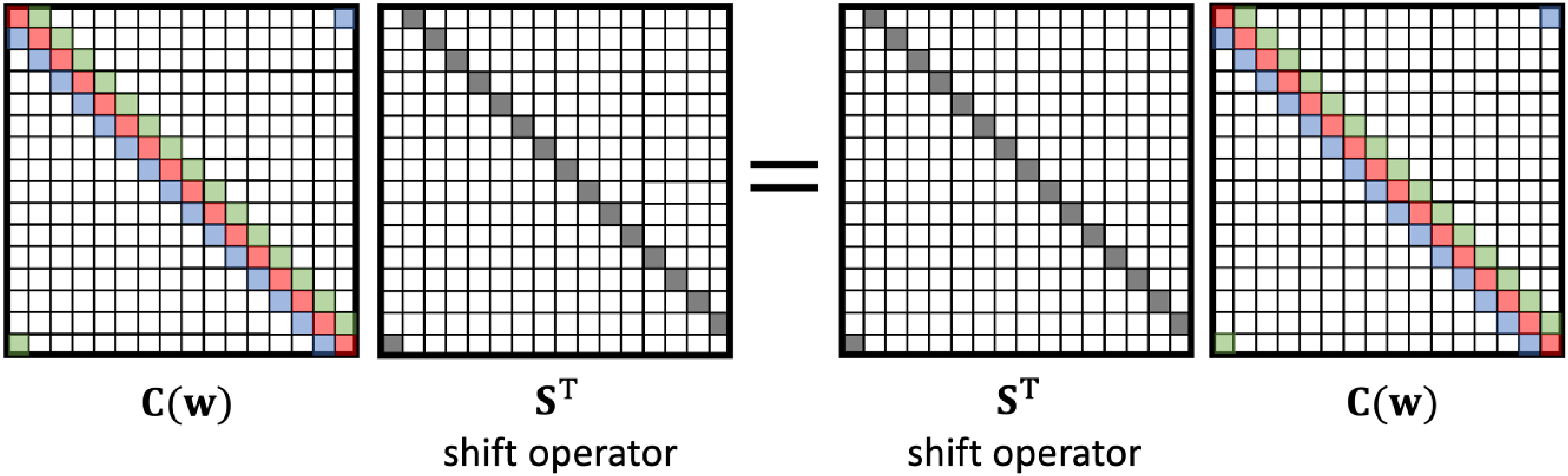
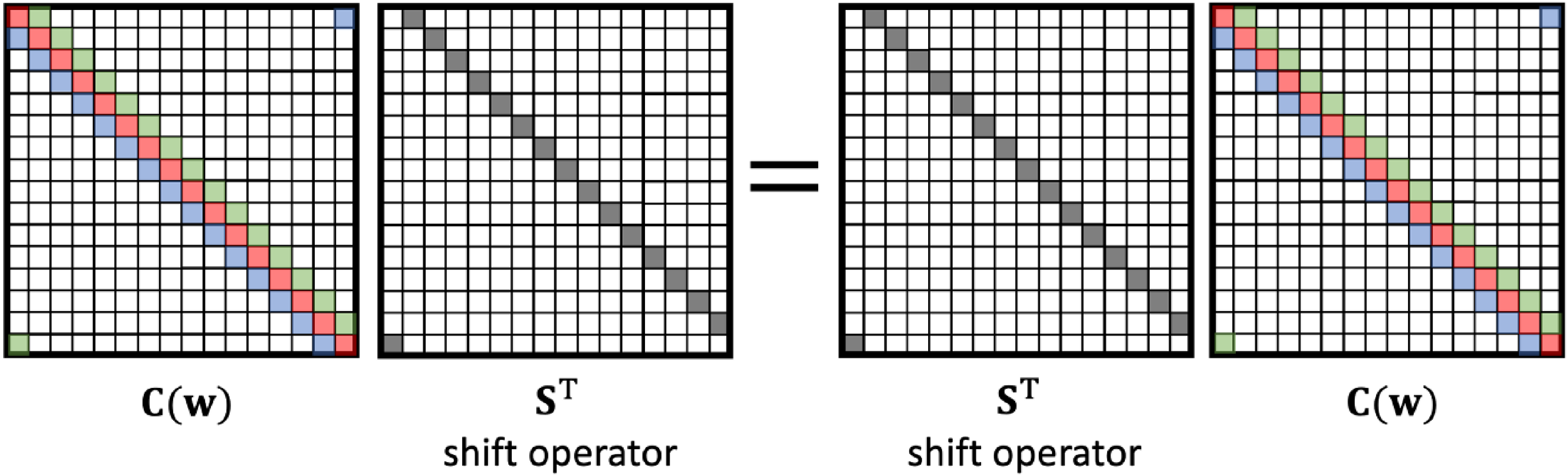


Image Credit: [1]

Translation/Shift Equivariance

Matrix multiplication is non-commutative. But not for circulant matrices!



Convolution is translation equivariant, i.e., Conv(Shift(X)) = Shift(Conv(X))!

This equivariance holds for 2D and higher-order convolutions!

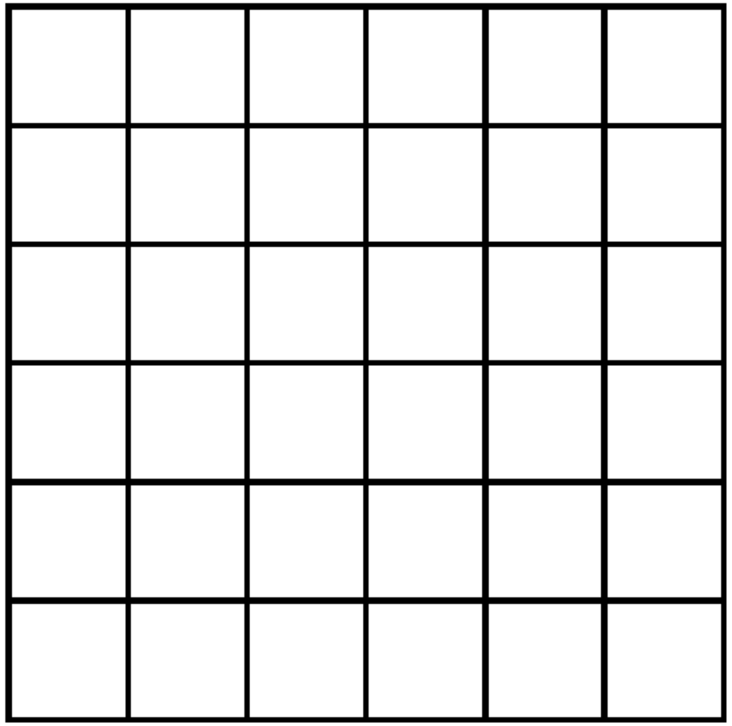
Image Credit: [1]

Outline

- Invariance & Equivariance
- Convolution
    - 1D Convolution
    - Matrix Multiplication Views
    - Translation Equivariance
    - **2D Convolution**
- Convolution Variants
    - Transposed Convolution
    - Dilated Convolution
    - Grouped Convolution
    - Separable Convolution
- Pooling
- Example Architectures

2D (Discrete) Convolution

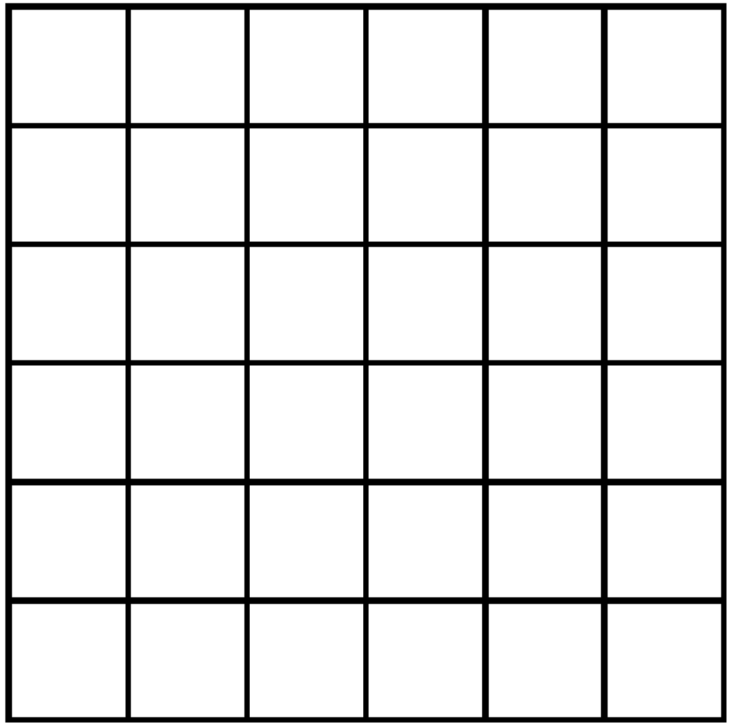
Let us see what convolution is in 2D



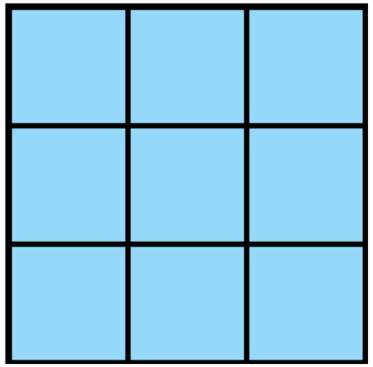
Input $\mathbf{x}$

2D (Discrete) Convolution

Let us see what convolution is in 2D



Input $\mathbf{x}$

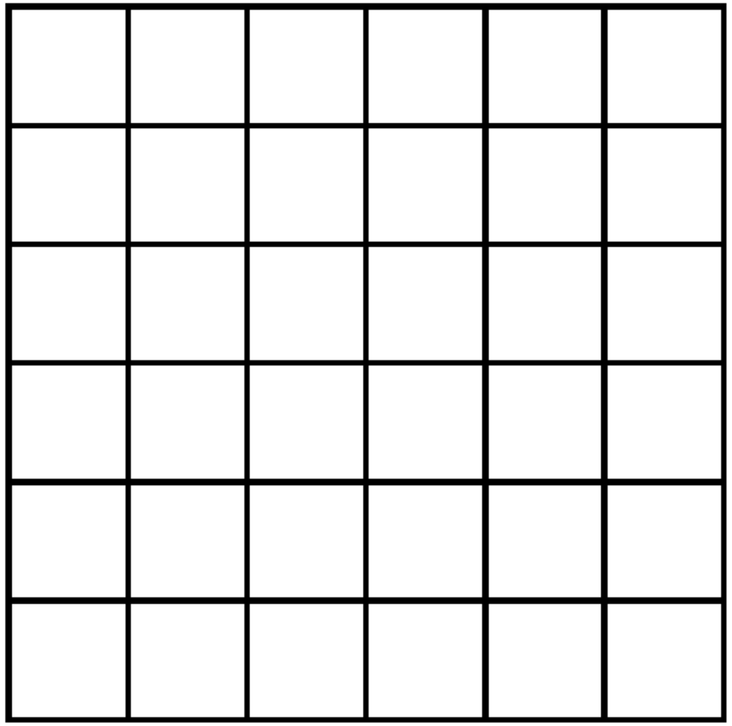


Convolutional Filter

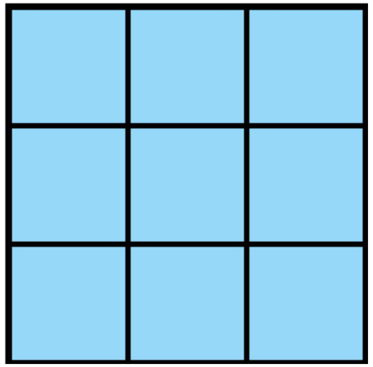
$W \in \mathbb{R}^{K \times K}$

2D (Discrete) Convolution

Let us see what convolution is in 2D

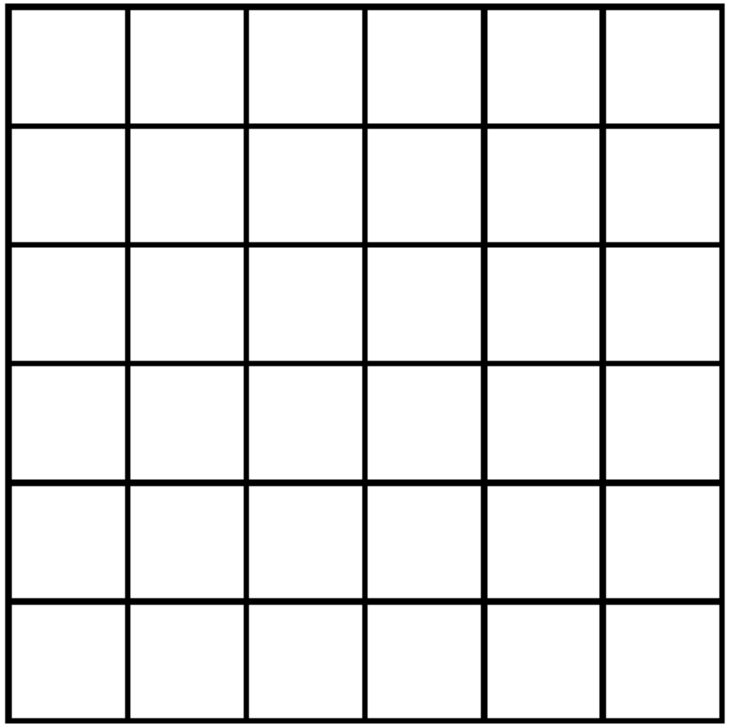


Input $\mathbf{x}$



Convolutional Filter

$$W \in \mathbb{R}^{K \times K}$$

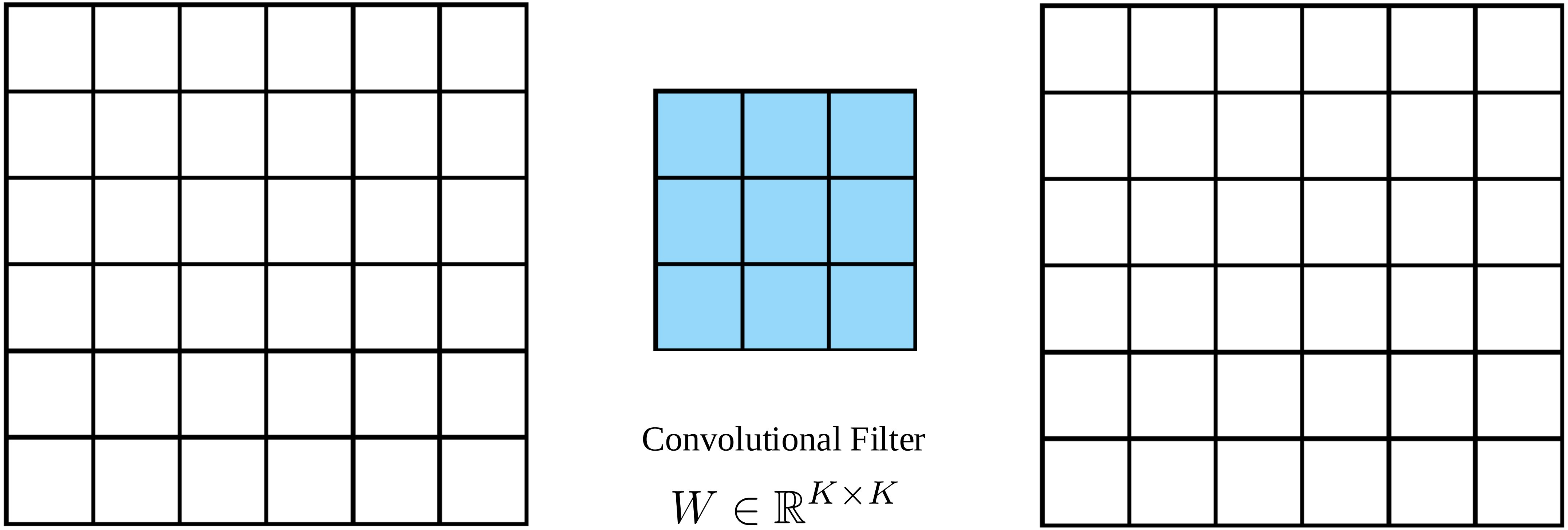


Sliding Window

2D (Discrete) Convolution

Let us see what convolution is in 2D

$$\mathbf{y}_{i,j} = \sum_{m=1}^{K} \sum_{n=1}^{K} W_{m,n} \mathbf{x}_{i+m-\lceil K/2 \rceil, j+n-\lceil K/2 \rceil}$$



Input $\mathbf{x}$

Convolutional Filter

$W \in \mathbb{R}^{K \times K}$

Output $\mathbf{y}$

2D (Discrete) Convolution

2D Convolution with Stride = 1

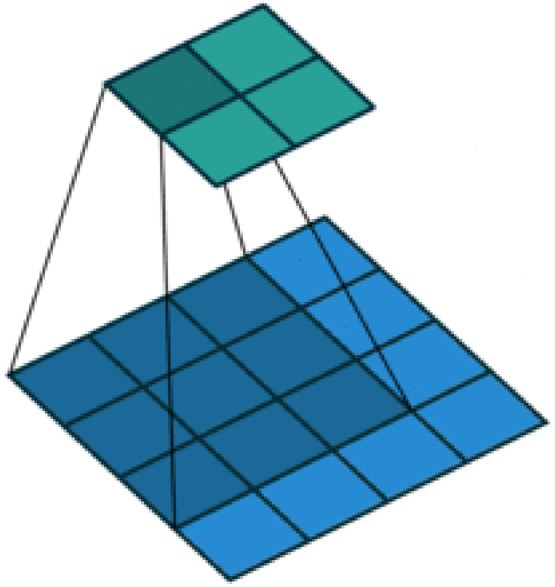


Image Credit: [2]

2D (Discrete) Convolution

2D Convolution with Stride = 1, Half Padding

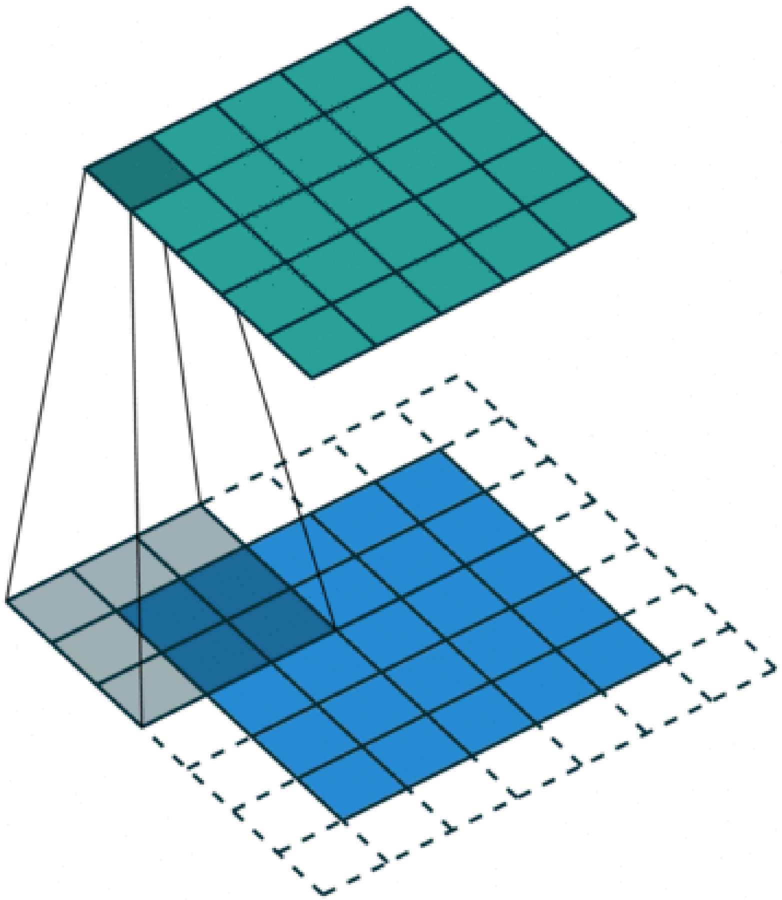


Image Credit: [2]

2D (Discrete) Convolution

2D Convolution with Stride = 2, Half Padding

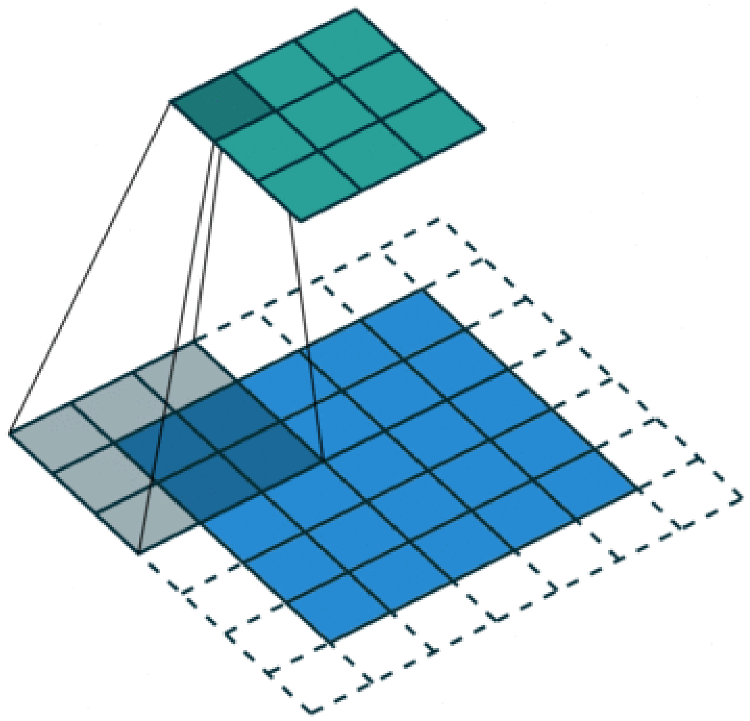


Image Credit: [2]

2D (Discrete) Convolution

2D Convolution with multiple input channels

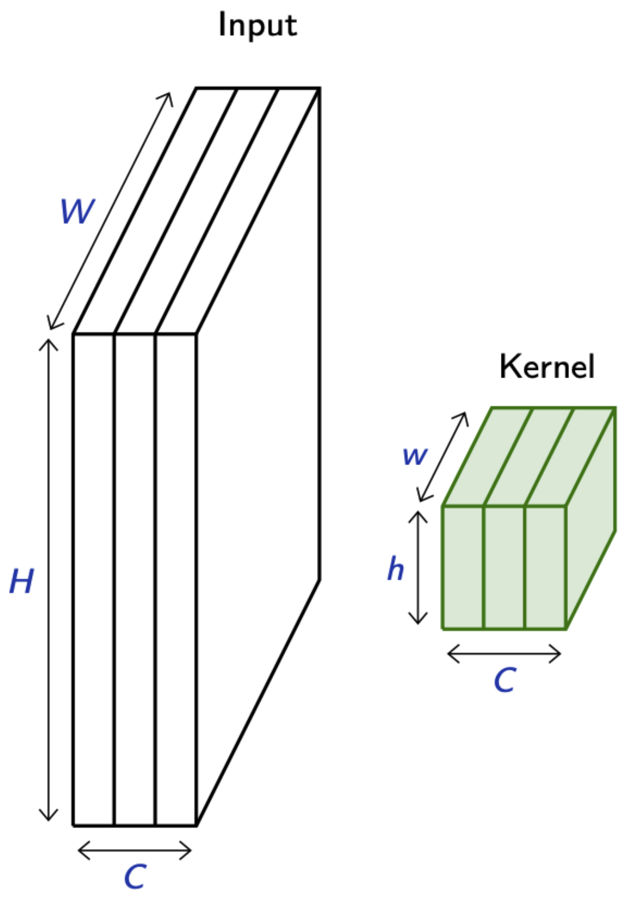


Image Credit: [3]

2D (Discrete) Convolution

2D Convolution with multiple input channels

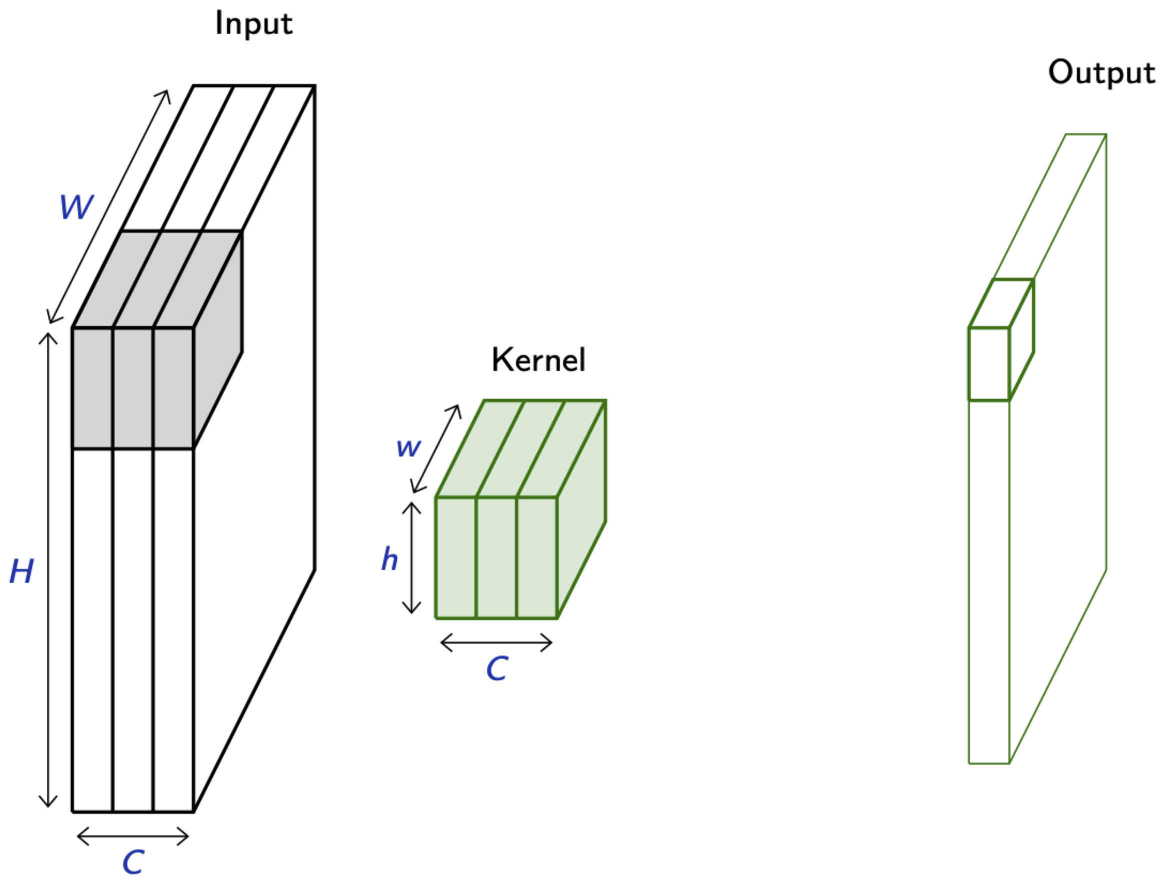


Image Credit: [3]

2D (Discrete) Convolution

2D Convolution with multiple input channels

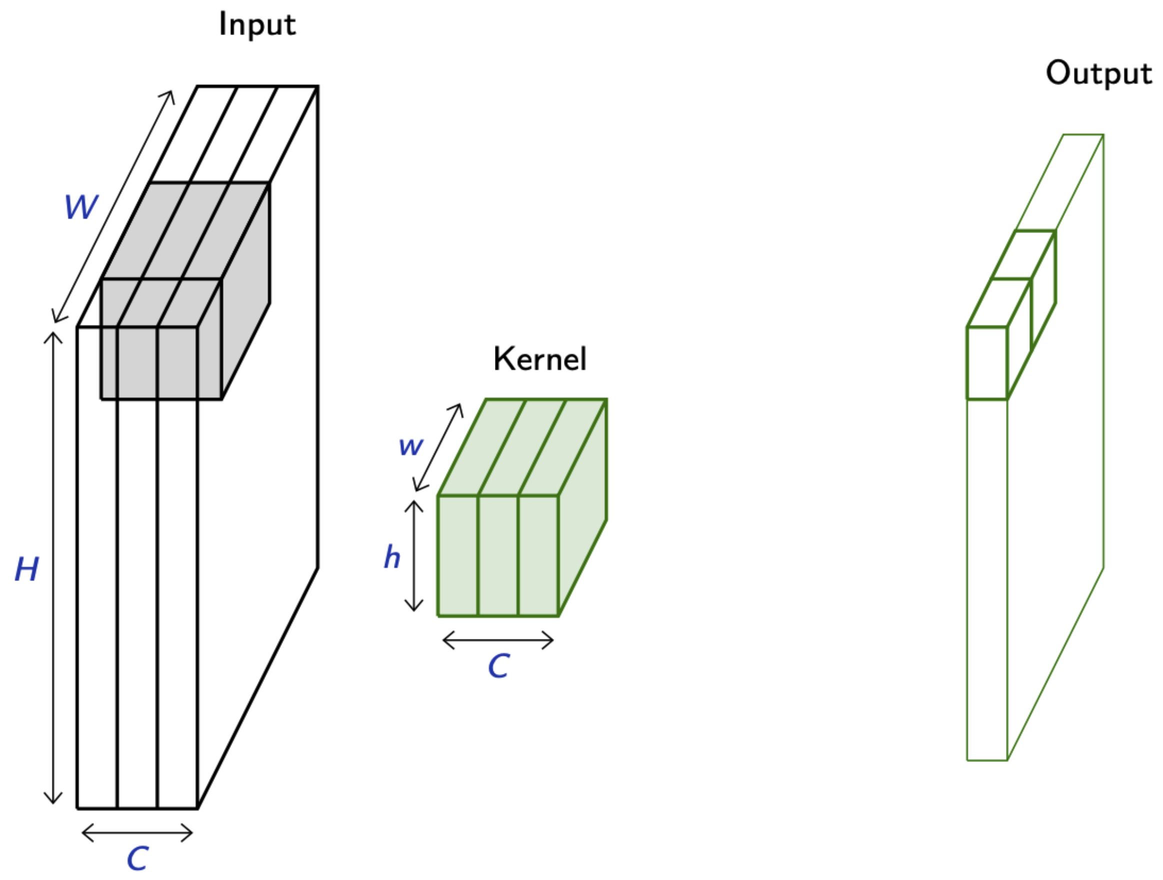


Image Credit: [3]

2D (Discrete) Convolution

2D Convolution with multiple input channels

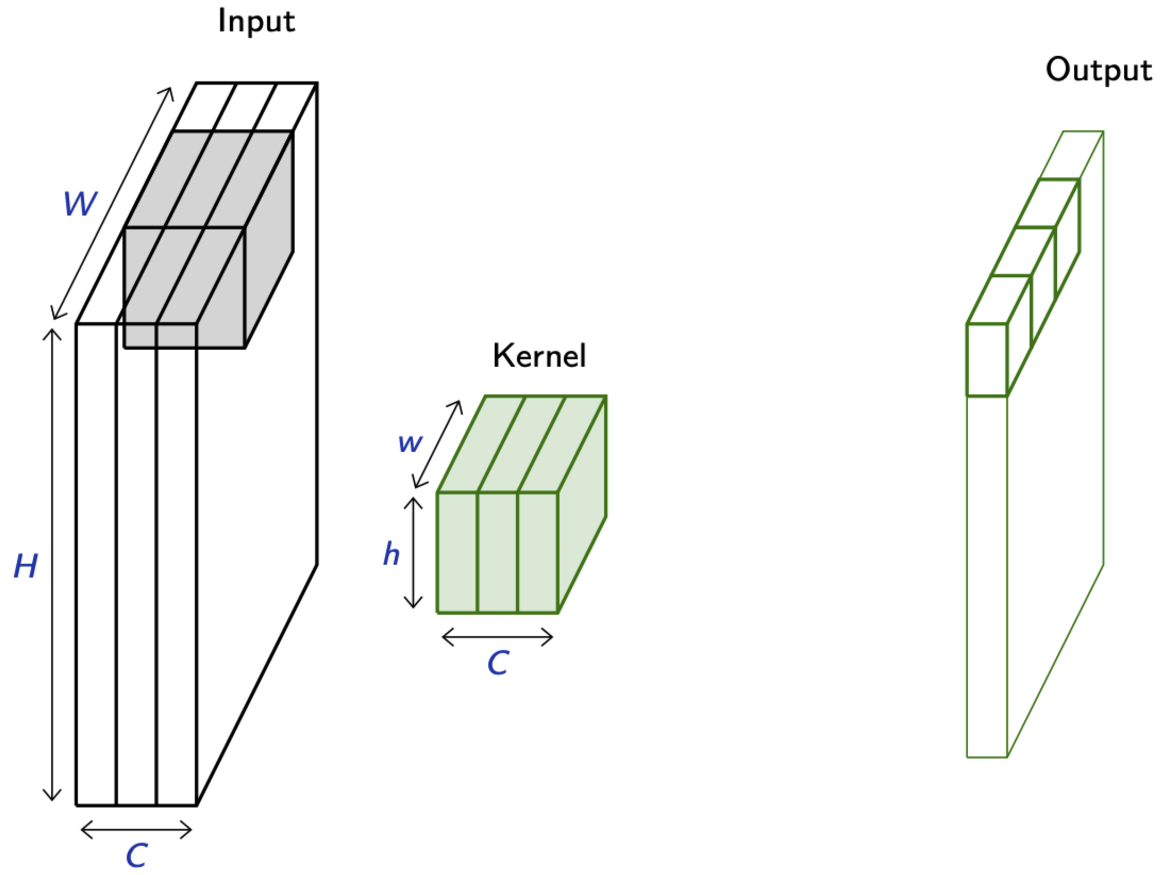


Image Credit: [3]

2D (Discrete) Convolution

2D Convolution with multiple input channels

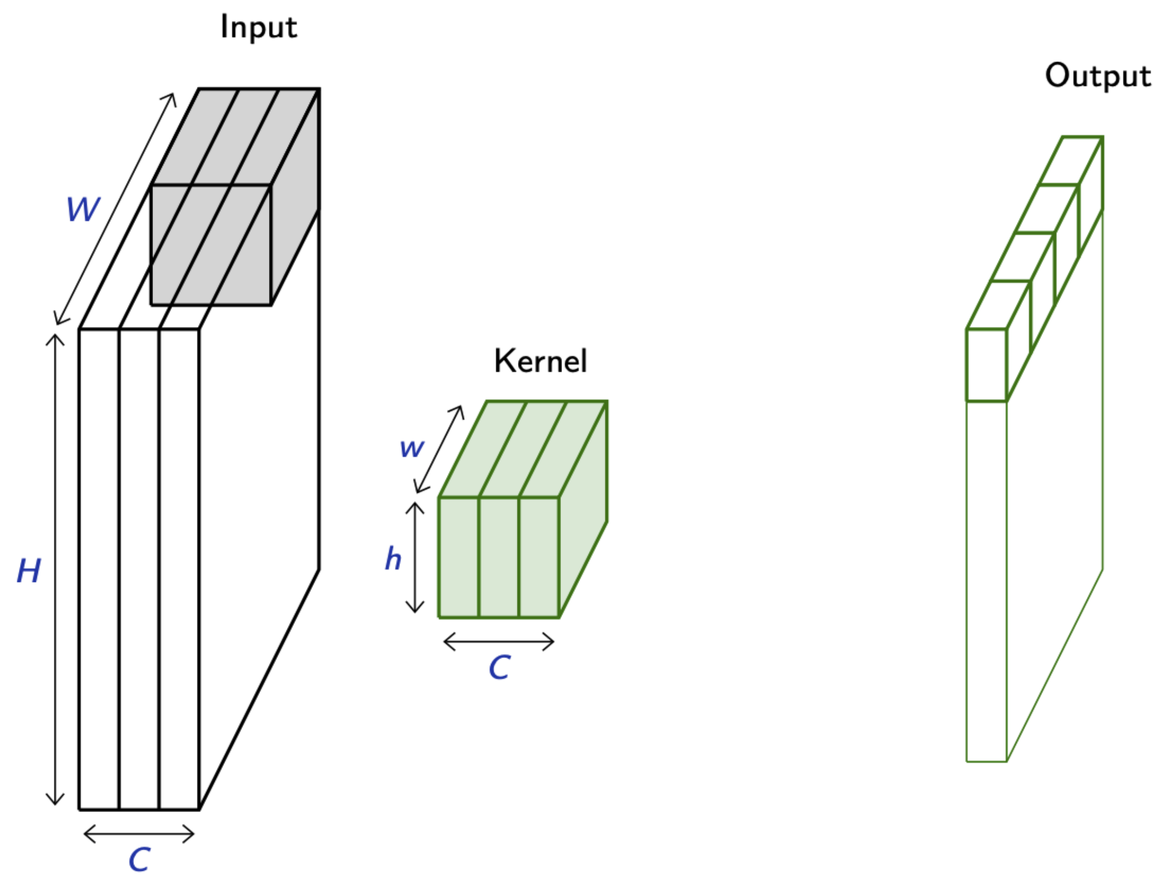


Image Credit: [3]

2D (Discrete) Convolution

2D Convolution with multiple input channels

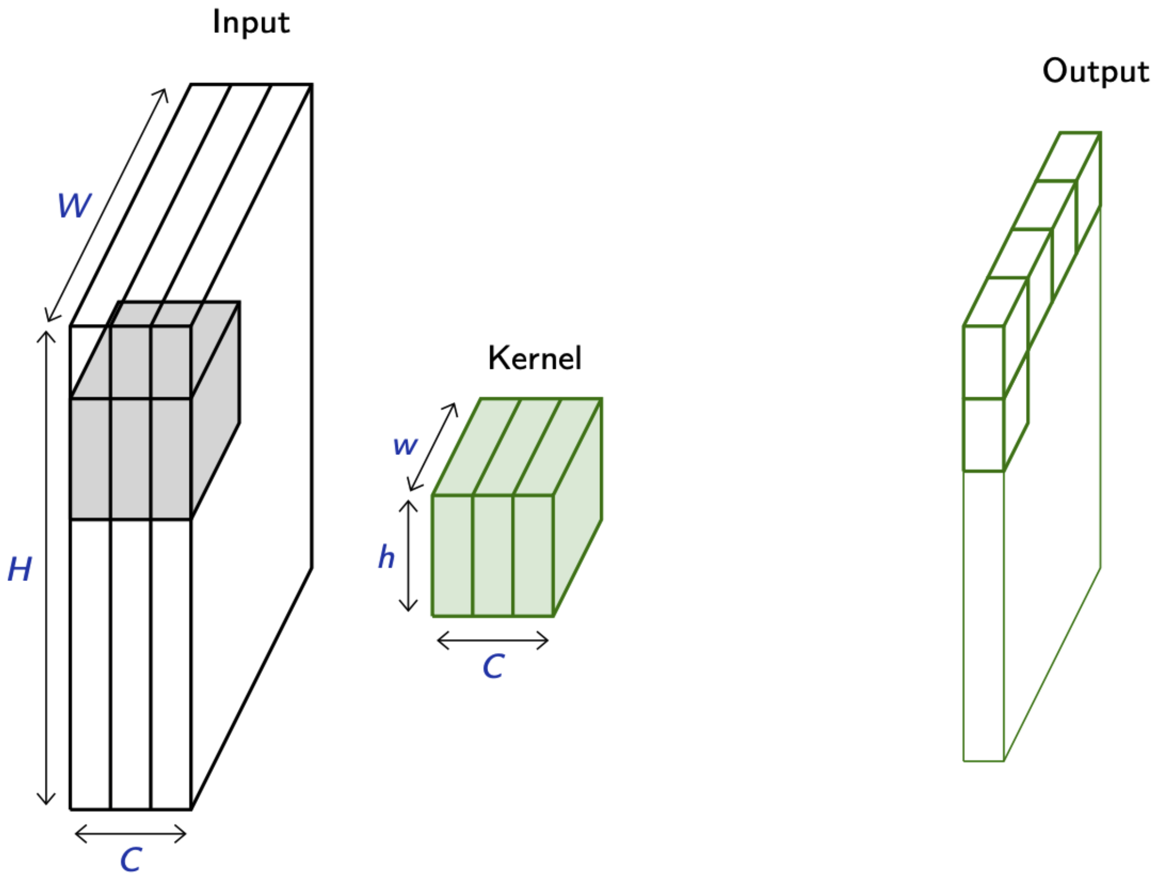


Image Credit: [3]

2D (Discrete) Convolution

2D Convolution with multiple input channels

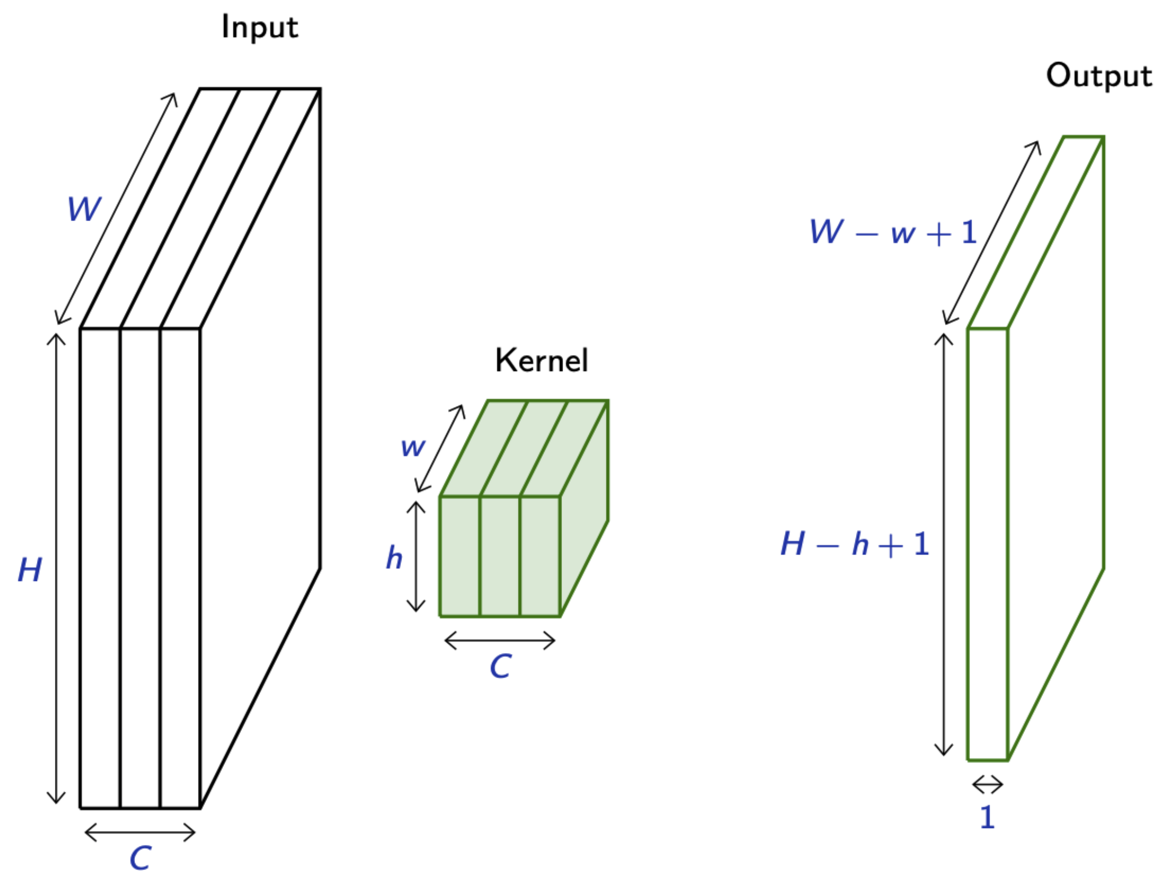


Image Credit: [3]

2D (Discrete) Convolution

2D Convolution with multiple input channels and multiple filters

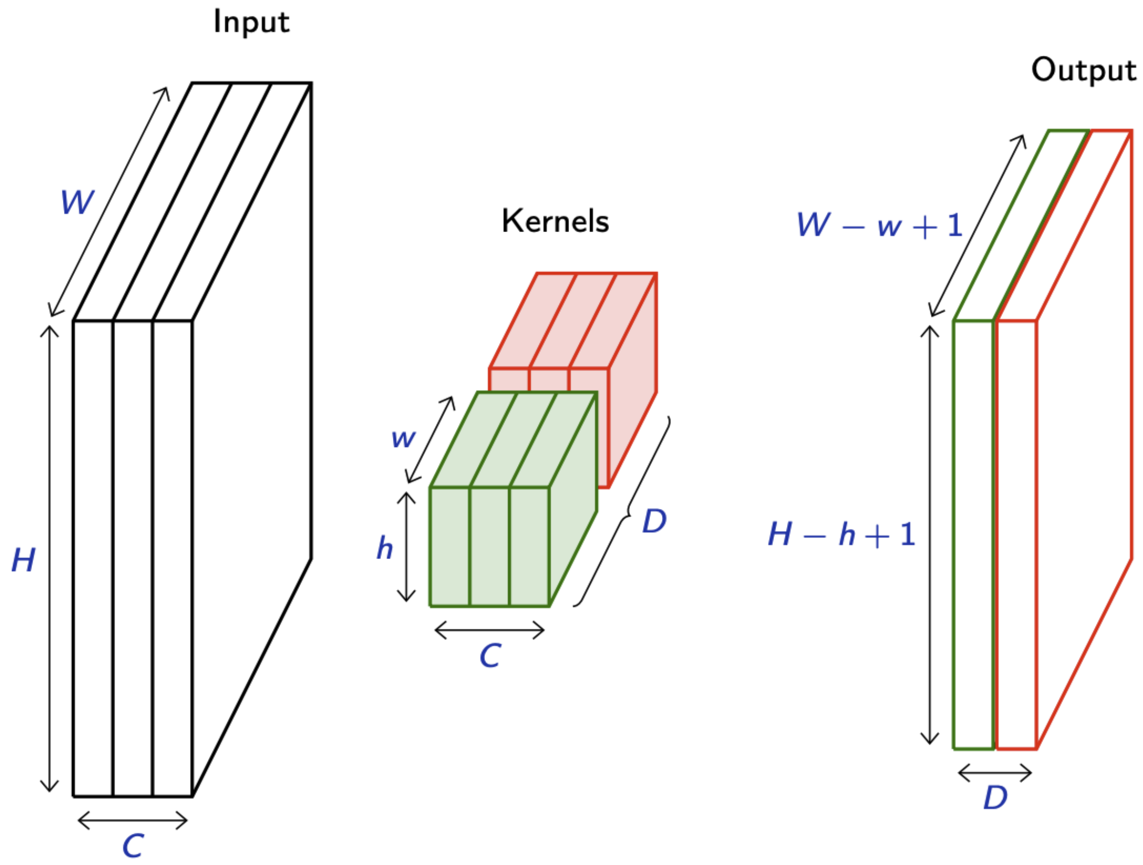


Image Credit: [3]

2D (Discrete) Convolution

Let us see the effect of 2D convolution:

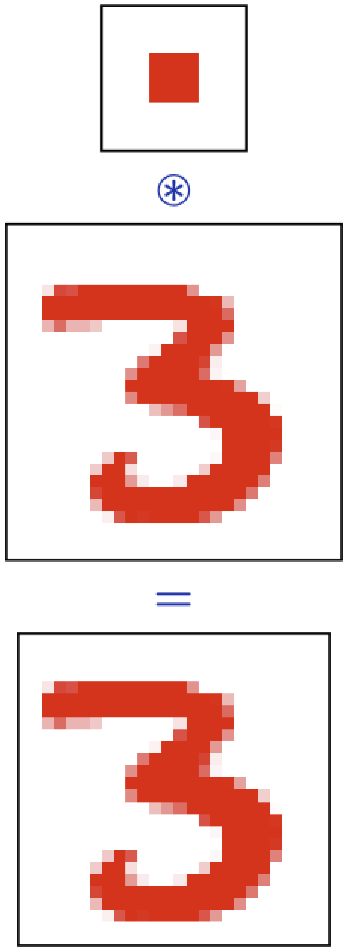


Image Credit: [3]

2D (Discrete) Convolution

Let us see the effect of 2D convolution:

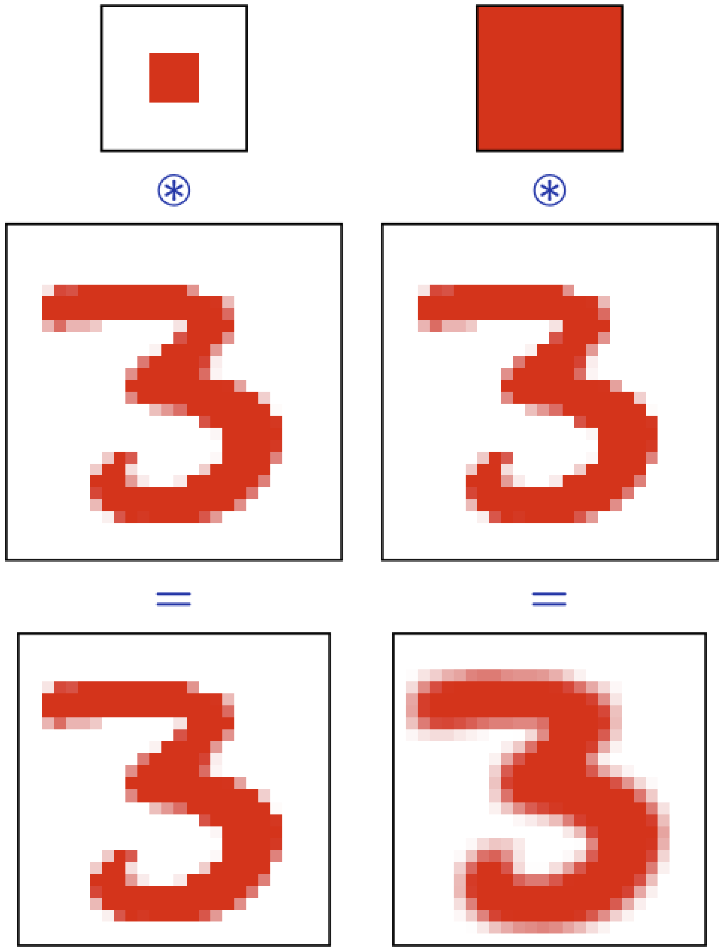


Image Credit: [3]

2D (Discrete) Convolution

Let us see the effect of 2D convolution:

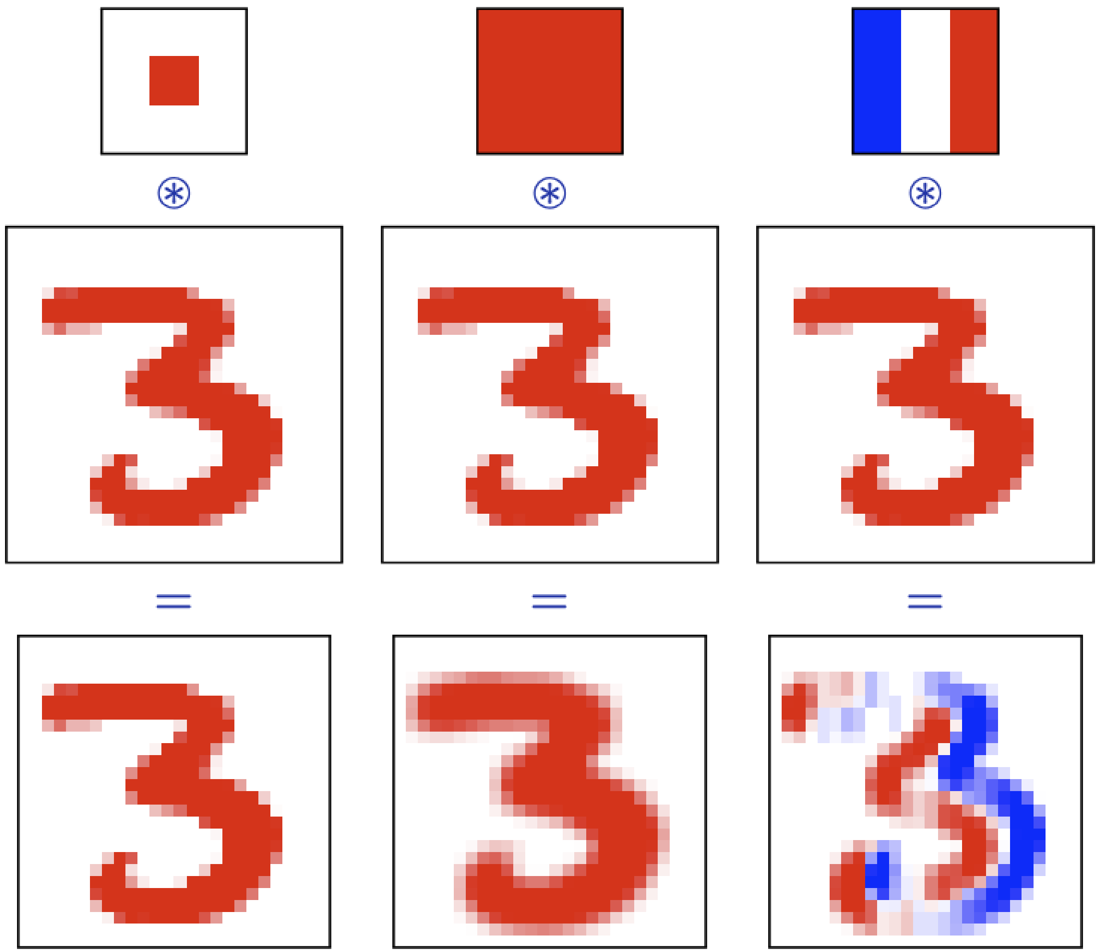


Image Credit: [3]

2D (Discrete) Convolution

Let us see the effect of 2D convolution:

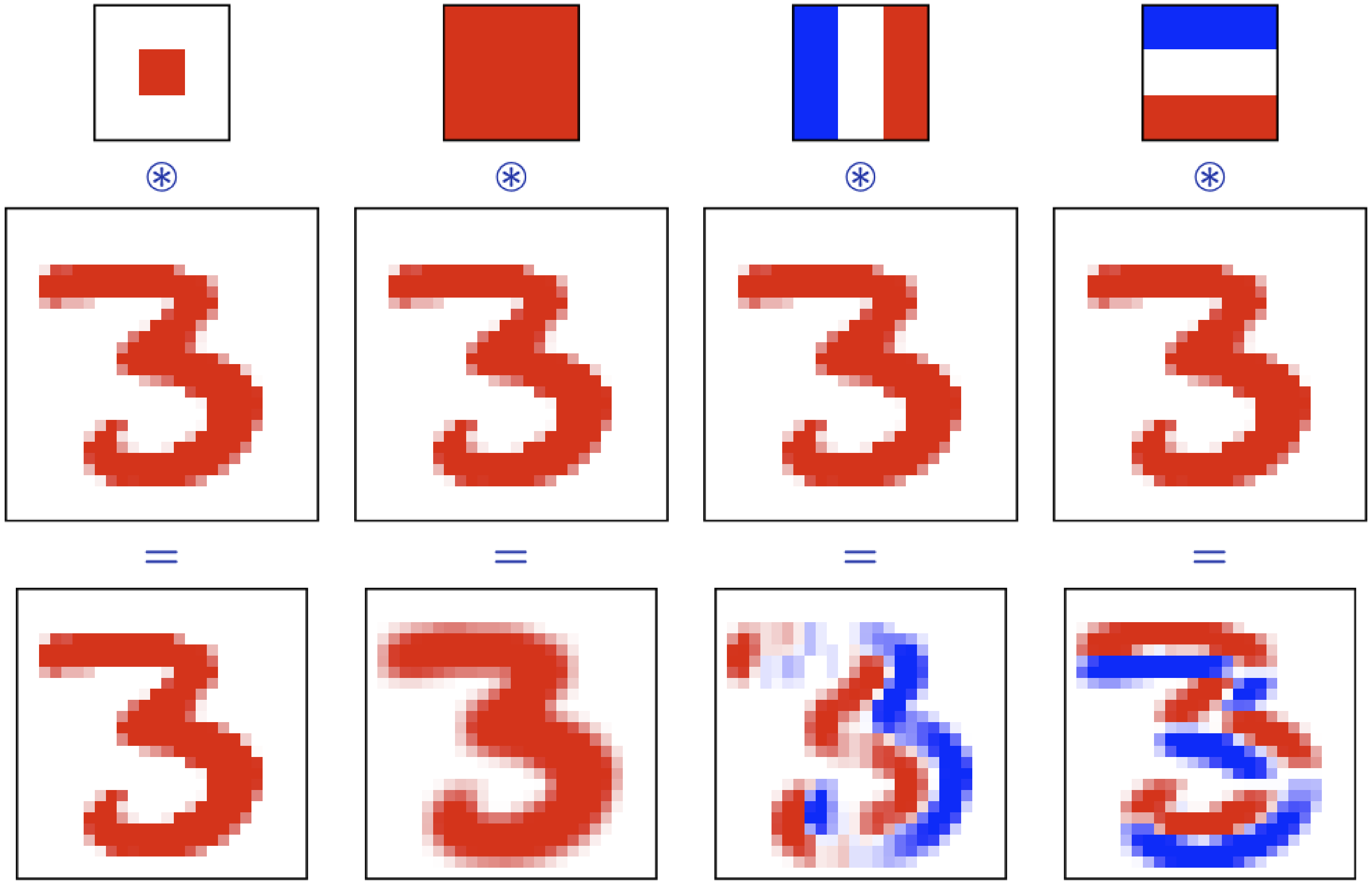


Image Credit: [3]

2D (Discrete) Convolution

Let us see the effect of 2D convolution:

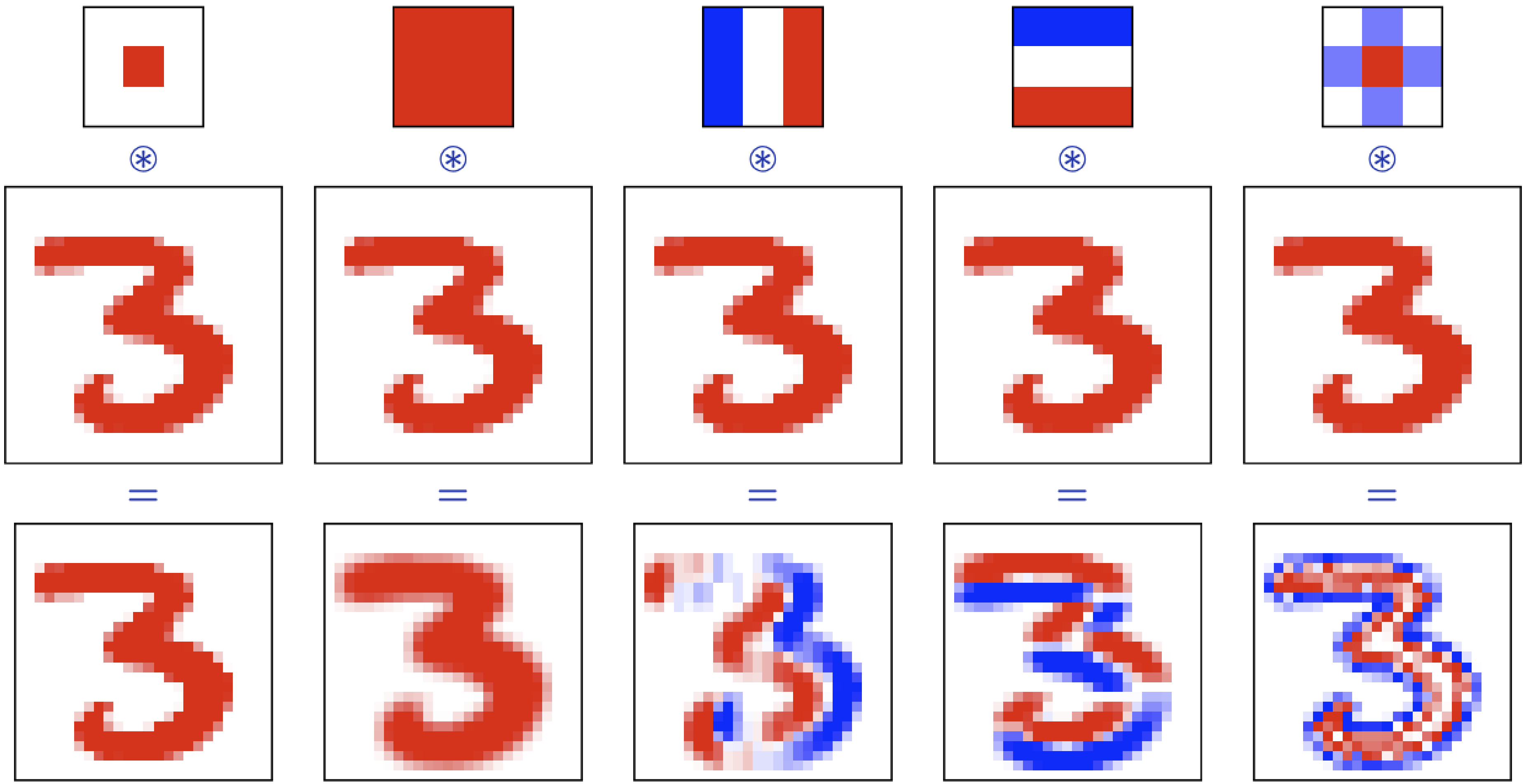


Image Credit: [3]

References

[1] https://towardsdatascience.com/deriving-convolution-from-first-principles-4ff124888028

[2] https://github.com/vdumoulin/conv_arithmetic/blob/master/README.md

[3] https://fleuret.org/dlc/materials/dlc-slides-4-4-convolutions.pdf

Questions?