Lecture 12: Convolutional Neural Networks

Introduction to Machine Learning [25737]

Sajjad Amini

Sharif University of Technology

Contents

References

Except explicitly cited, the reference for the material in slides is:

- Murphy, K. P. (2022). *Probabilistic machine learning: an introduction.* MIT press.

Section 1

Approach Definition

Approach Definition

MLPs

Assume transformation $\boldsymbol{z} = \varphi(\boldsymbol{W}\boldsymbol{x})$. Then j-th element in $\boldsymbol{z}$ can be represented as:

$$z_j = \varphi(\boldsymbol{w}_j^T \boldsymbol{x})$$

We can interpret this equation as computing the similarity between $\boldsymbol{x}$ and $\boldsymbol{w}_j$. When working with 2D images, this structure can lead to sever problems.

Approach Definition

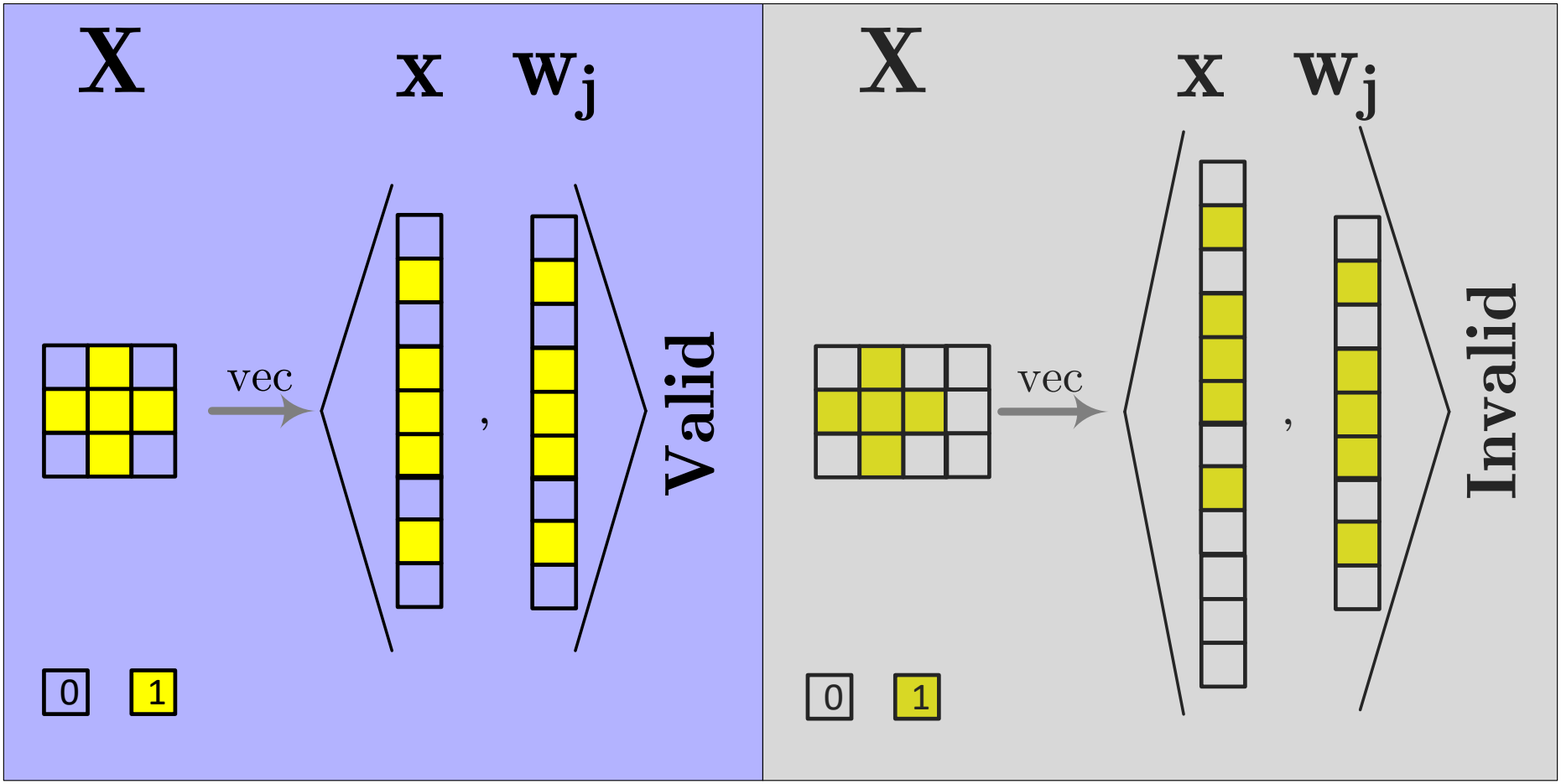


Figure: Not applicable when image size is changed

Approach Definition

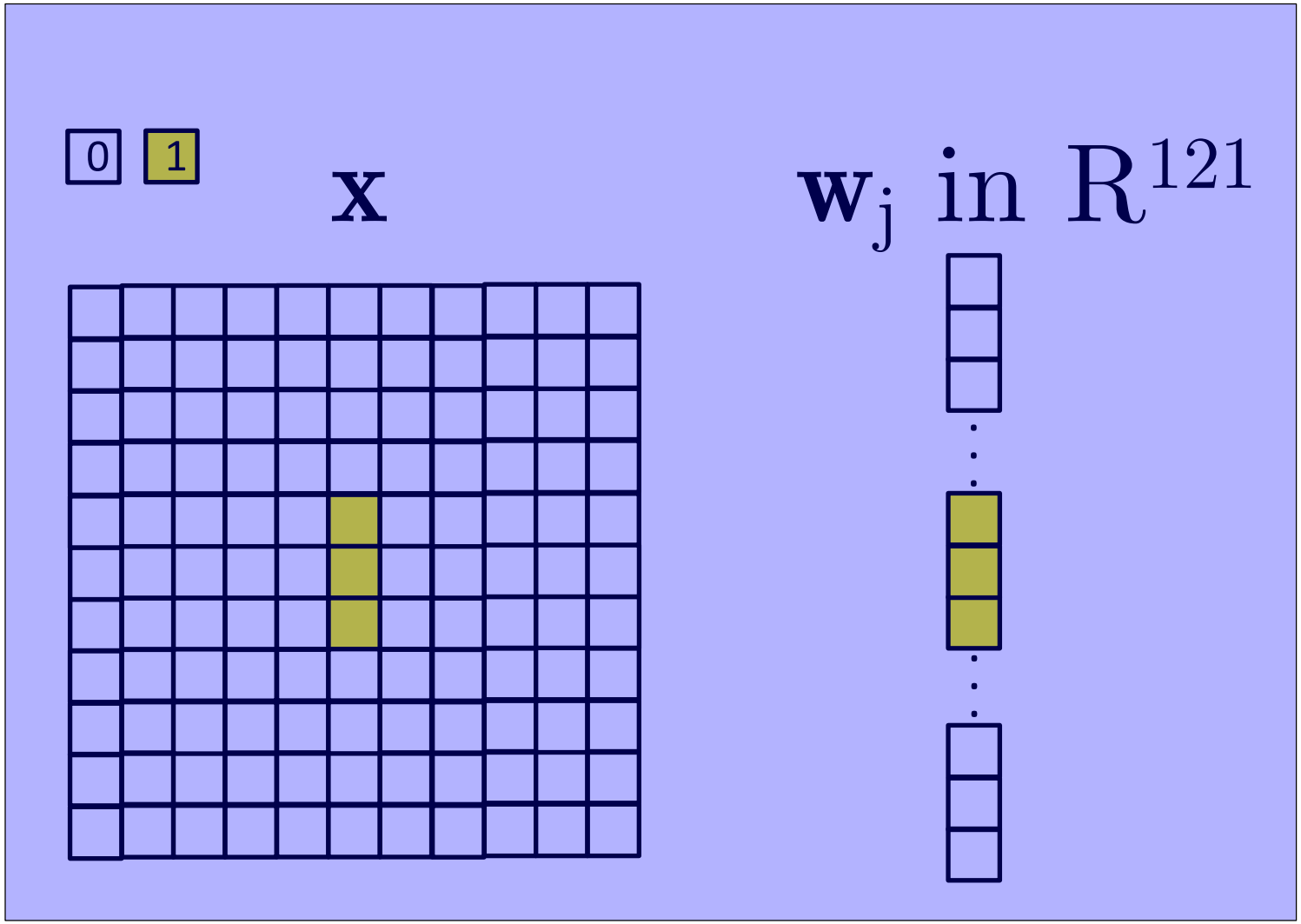


Figure: Highly redundant

Approach Definition

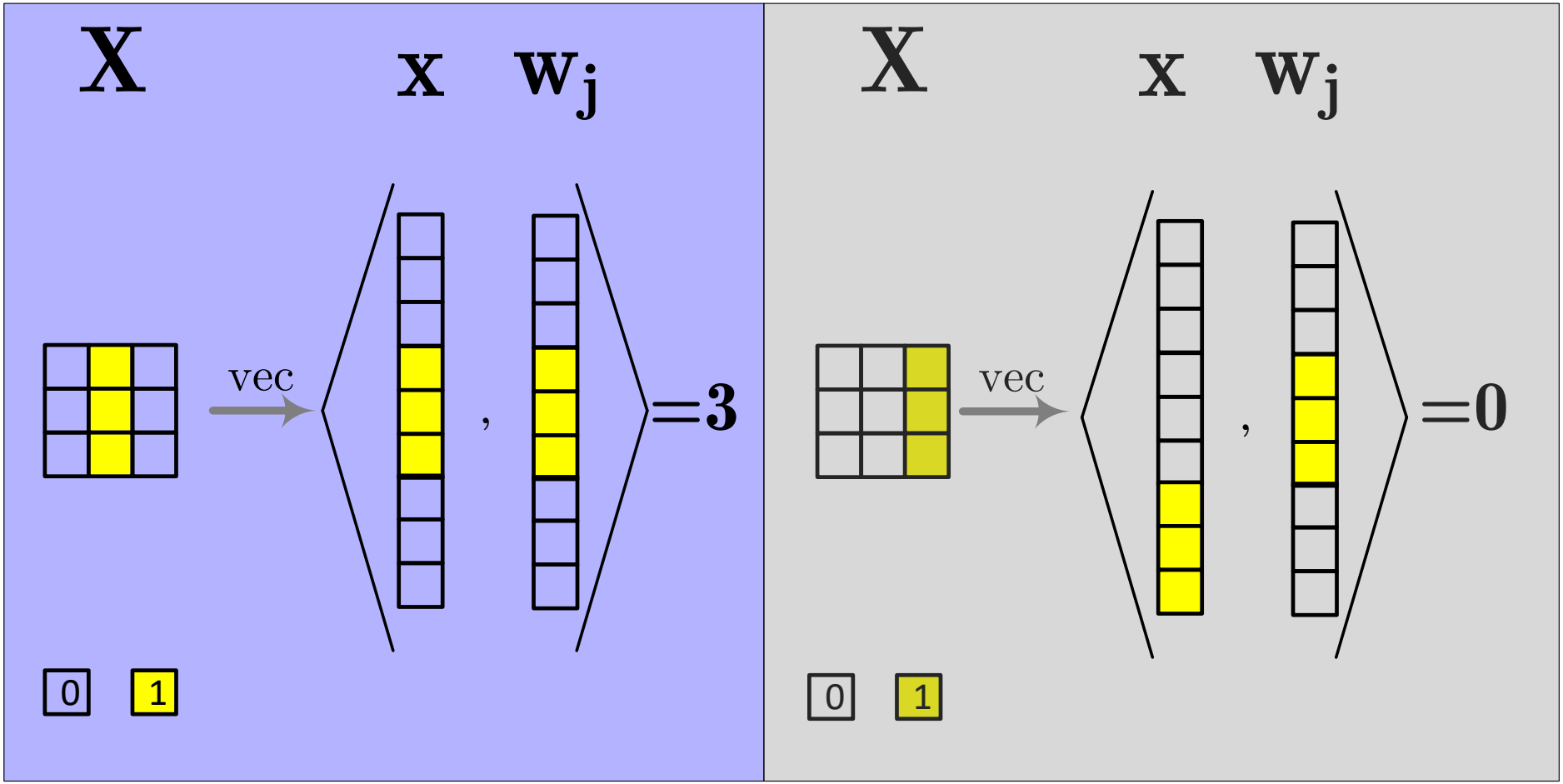


Figure: Not exhibiting translation invariance

Approach Definition

Convolutional Neural Networks (CNN)

To solve the challenges mentioned above, CNNs are introduced where matrix multiplication is replaced with convolution operator.

- We can compute the convolution of different size images with the same filter.
- The size of convolution filter is smaller than the size of input features.
- Convolution is a template matching operator and can present translation invariance.

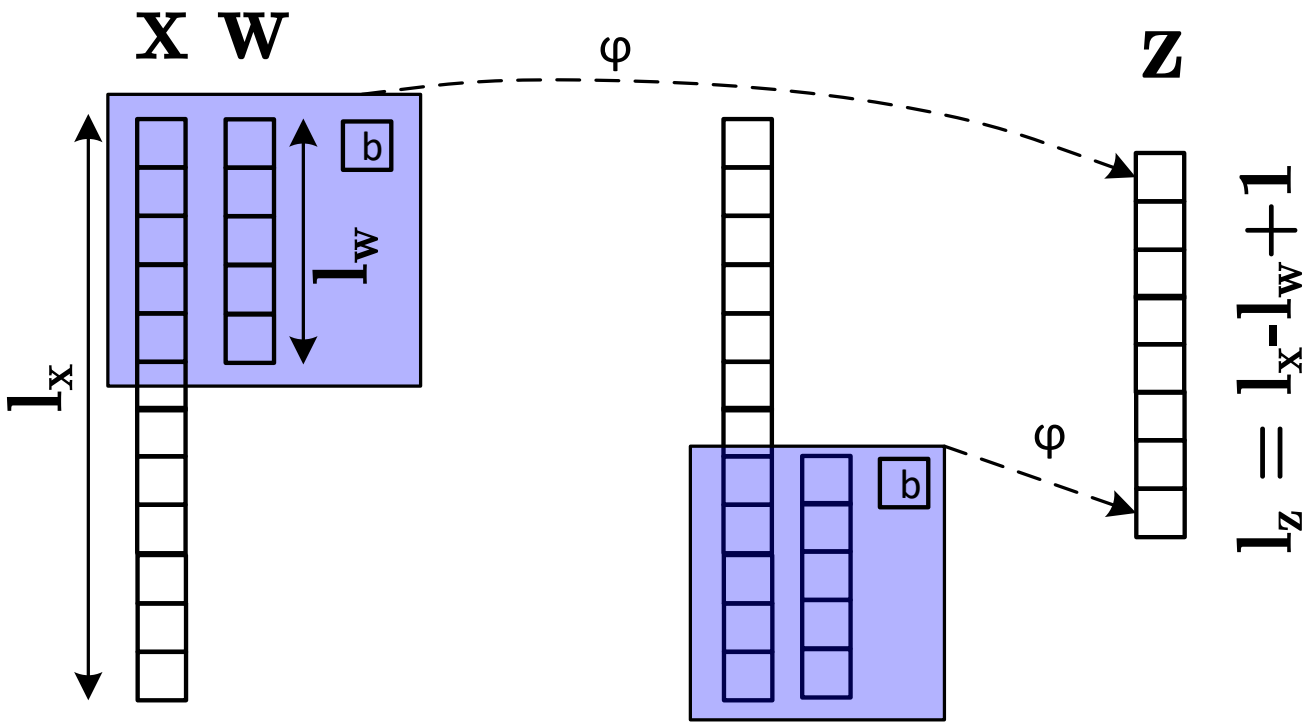
Section 2

Common Layers

1D Convolution Layer

1D convolution (valid)

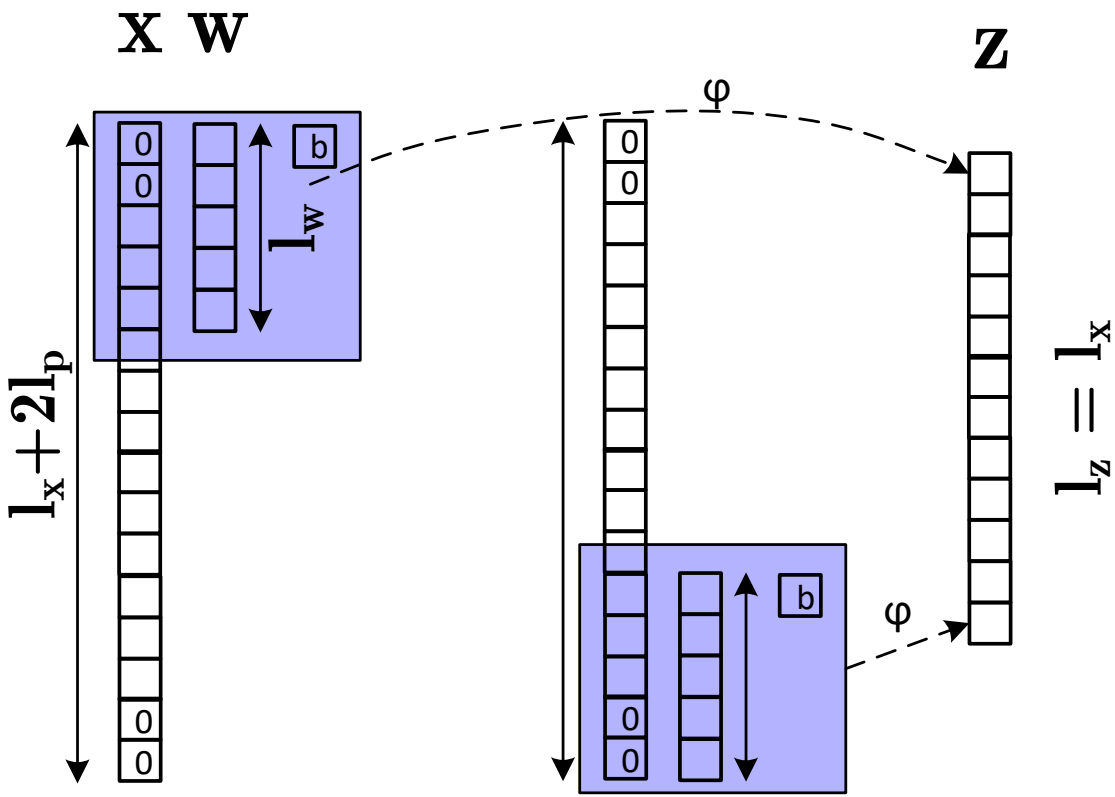
$$\begin{cases} \boldsymbol{x} \in \mathbb{R}^{l_x} \\ \boldsymbol{w} \in \mathbb{R}^{l_w} \end{cases} \Rightarrow z_p = \varphi\left(b + \sum_{j=0}^{K-1} w_j x_{p+j}\right), \; 0 \le p \le l_x - l_w$$



1D Convolution Layer

1D convolution (same)

- If we pad each side of input feature vector $\boldsymbol{x} \in \mathbb{R}^{l_x}$ with p elements and $\boldsymbol{w} \in \mathbb{R}^{l_w}$, then the output size will be $l_x + 2p - l_w + 1$.
- If we select $p = \frac{l_w - 1}{2}$, then input and output sizes are the same.



1D Convolution Layer

Convolution to Matrix Multiplication

Assume we have the following convolution operator:

$$\boldsymbol{z} = \varphi\left(\boldsymbol{x} \circledast \boldsymbol{w} + b\right), \quad \begin{cases} \boldsymbol{x} \in \mathbb{R}^{l_x} \\ \boldsymbol{w} \in \mathbb{R}^3 \\ \boldsymbol{z} \in \mathbb{R}^{l_x - 2} \end{cases}$$

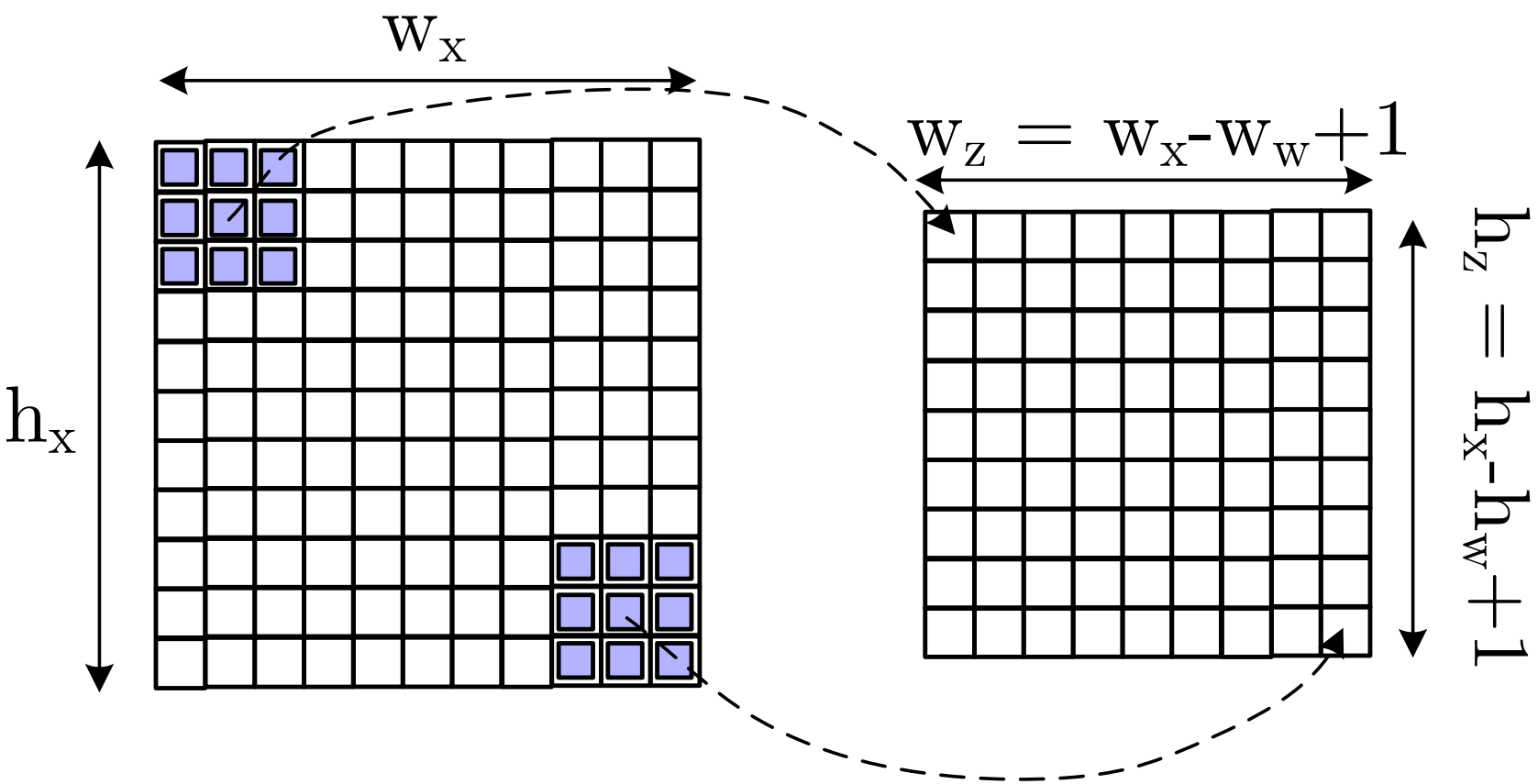
then we can write above mapping in matrix multiplication as:

$$\begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{l_x-4} \\ z_{l_x-3} \end{bmatrix} = \varphi\left(\begin{bmatrix} w_0 & w_1 & w_2 & 0 & \dots & 0 & 0 & 0 & 0 \\ 0 & w_0 & w_1 & w_2 & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & w_0 & w_1 & w_2 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & w_0 & w_1 & w_2 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{l_x-2} \\ x_{l_x-1} \end{bmatrix} + \begin{bmatrix} b \\ b \\ \vdots \\ b \\ b \end{bmatrix} \right)$$

2D Convolution Layer

2D convolution (valid)

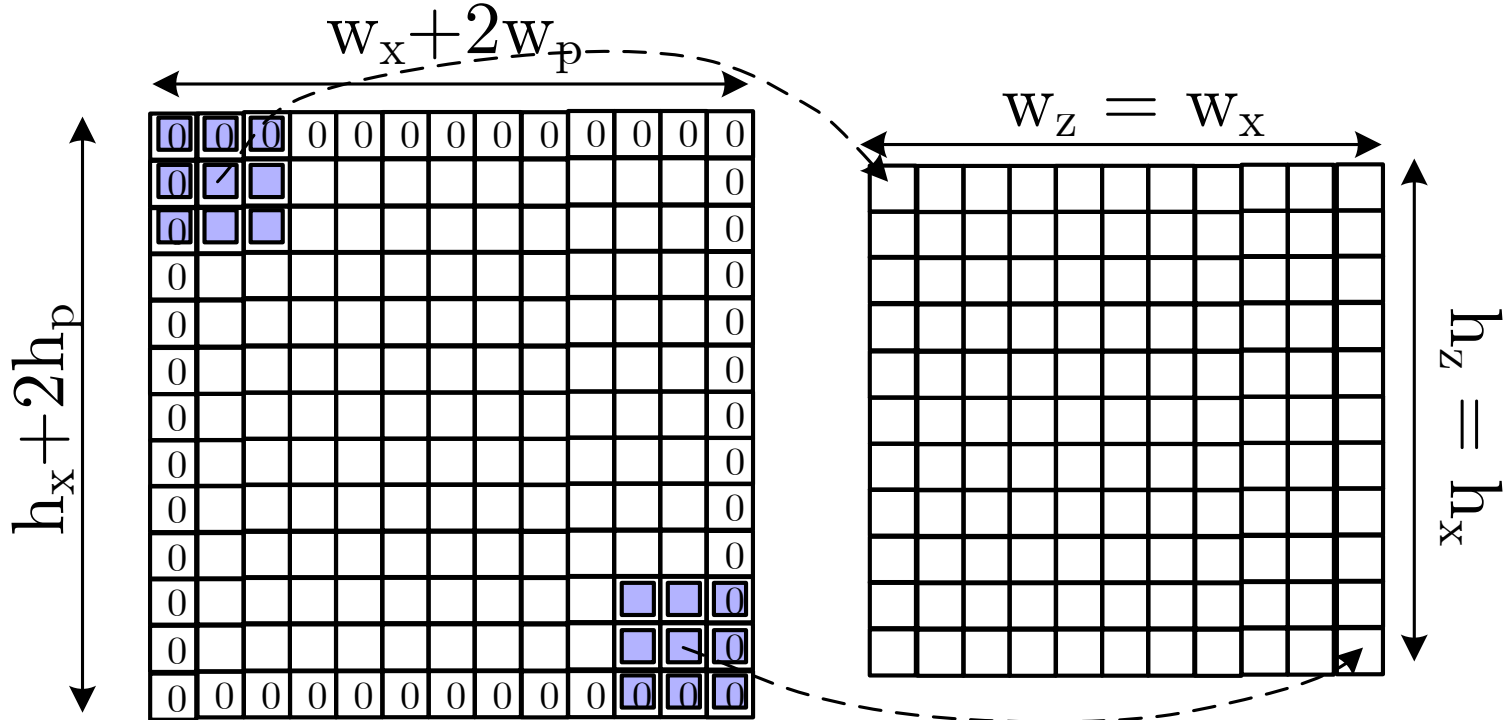
$$\begin{cases} \boldsymbol{X} \in \mathbb{R}^{h_x \times w_x} \\ \boldsymbol{W} \in \mathbb{R}^{h_w \times w_w} \end{cases} \Rightarrow z_{pq} = \varphi\left(b + \sum_{i=0}^{h_w-1} \sum_{j=0}^{w_w-1} w_{ij} x_{(p+i)(q+j)}\right), \quad \begin{cases} 0 \le p \le h_x - h_w \\ 0 \le q \le w_x - w_w \end{cases}$$



2D Convolution Layer

2D convolution (same)

- If we pad each side of input feature matrix $\boldsymbol{X} \in \mathbb{R}^{h_x \times w_x}$ with h_p and w_p elements and $\boldsymbol{w} \in \mathbb{R}^{h_w \times w_w}$, then the output size will be $(h_x + 2h_p - h_w + 1) \times (w_x + 2w_p - w_w + 1)$.
- If we select $h_p = \frac{h_w - 1}{2}$ and $w_p = \frac{w_w - 1}{2}$, then input and output sizes are the same.



2D Convolution Layer

Convolution to Matrix Multiplication

Assume we have the following convolution operator:

$$\boldsymbol{Z} = \varphi\left(\boldsymbol{X} \circledast \boldsymbol{W} + b\right), \quad \begin{cases} \boldsymbol{X} \in \mathbb{R}^{3\times 3} \\ \boldsymbol{W} \in \mathbb{R}^{2\times 2} \\ \boldsymbol{Z} \in \mathbb{R}^{2\times 2} \end{cases}$$

Assume column-wise vectorizing. Then we can write above mapping in matrix multiplication as:

$$\begin{bmatrix} z_{00} \\ z_{10} \\ z_{01} \\ z_{11} \end{bmatrix} = \varphi\left(\begin{bmatrix} w_{00} & w_{10} & 0 & w_{01} & w_{11} & 0 & 0 & 0 & 0 \\ 0 & w_{00} & w_{10} & 0 & w_{01} & w_{11} & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{00} & w_{10} & 0 & w_{01} & w_{11} & 0 \\ 0 & 0 & 0 & 0 & w_{00} & w_{10} & 0 & w_{01} & w_{11} \end{bmatrix} \begin{bmatrix} x_{00} \\ x_{10} \\ x_{20} \\ x_{01} \\ x_{11} \\ x_{21} \\ x_{02} \\ x_{12} \\ x_{22} \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \\ b \end{bmatrix} \right)$$

Receptive Field

For each element in the output features, its receptive field are the elements in the input features that form the it. In the above example, $\{x_{00}, x_{10}, x_{01}, x_{11}\}$ are receptive field of z_{00}.

Strided Convolution

Redundancy in the Output Features

As the receptive field for neighboring features in the output of convolution layer are highly overlapped, there exist redundancy in the output features. Strided convolution is designed to eliminate this redundancy.

Strided Convolution

Strided Convolution is ordinary convolution while we skip every s_h and s_w in vertical and horizontal shitf.

Strided Convolution

Output Dimensions

$$\begin{cases} \boldsymbol{X} \in \mathbb{R}^{h_x \times w_x} \\ \boldsymbol{W} \in \mathbb{R}^{h_w \times w_w} \\ \text{Heigth Padding}: \ h_p \\ \text{Width Padding}: \ w_p \\ \text{Heigth Stride}: \ h_s \\ \text{Width Stride}: \ w_s \end{cases} \Rightarrow \dim(\boldsymbol{Z}): \left\lfloor \frac{h_x + 2h_p - h_w + h_s}{h_s} \right\rfloor \times \left\lfloor \frac{w_x + 2w_p - w_w}{w_s} \right.$$

Strided Convolution

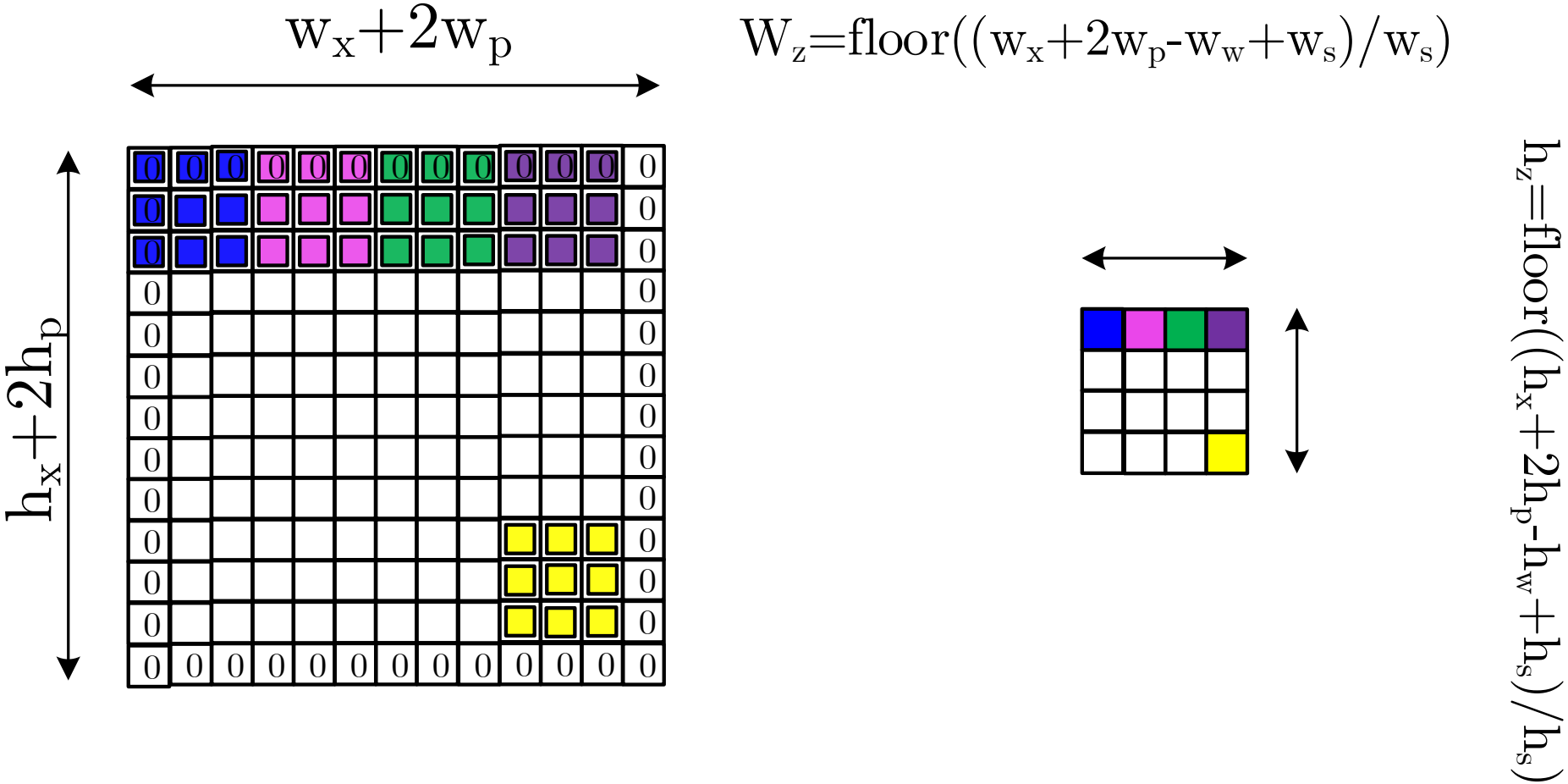


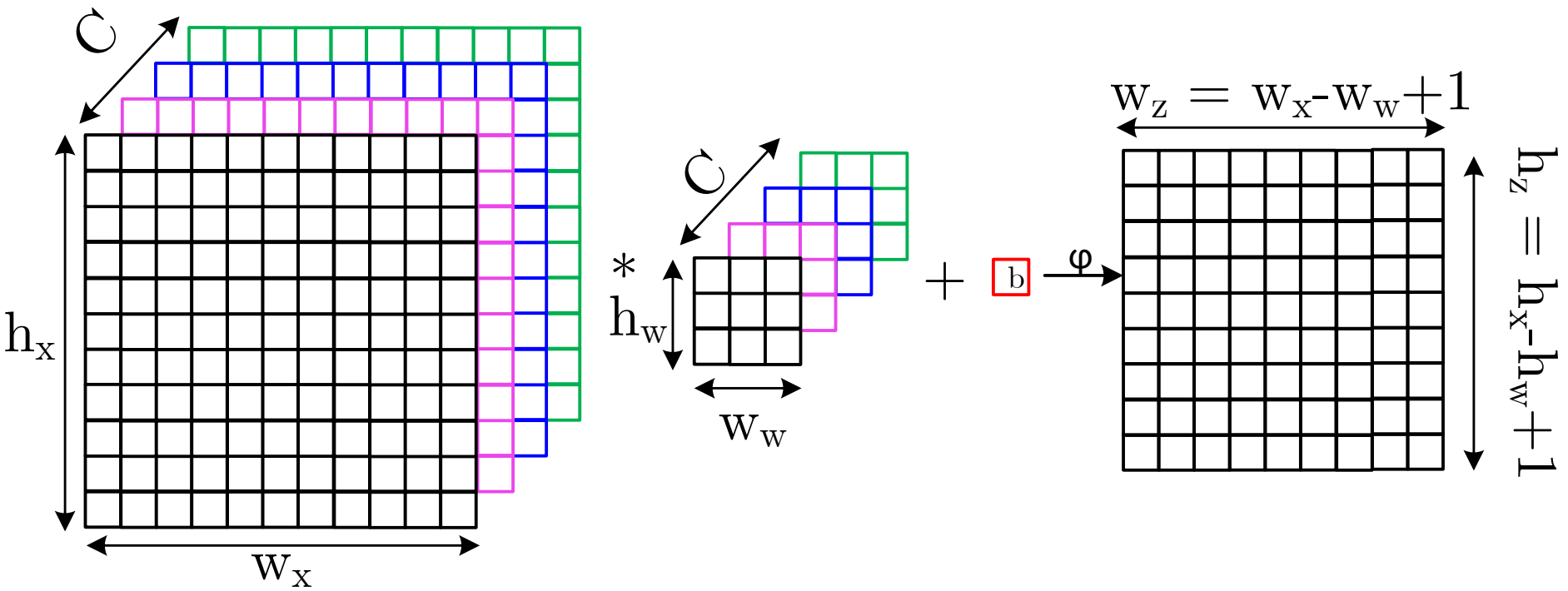
Figure: 2D convolution (same)

Extension 1: Multiple Input Channel

Convolution with multi-channel input

$$\begin{cases} \boldsymbol{X} \in \mathbb{R}^{h_x \times w_x \times C} \\ \boldsymbol{W} \in \mathbb{R}^{h_w \times w_w \times C} \end{cases} \Rightarrow z_{pq} = \varphi\left(b + \sum_{k=0}^{C-1} \sum_{i=0}^{h_w-1} \sum_{j=0}^{w_w-1} w_{ijk} x_{(p+i)(q+j)k}\right), \quad \begin{cases} 0 \le p \le h_x - h_w \\ 0 \le q \le w_x - w_w \end{cases}$$

$$\Rightarrow \boldsymbol{Z} \in \mathbb{R}^{(h_x - h_w + 1) \times (w_x - w_w + 1)}$$



Extension 2: Multiple Output Channel

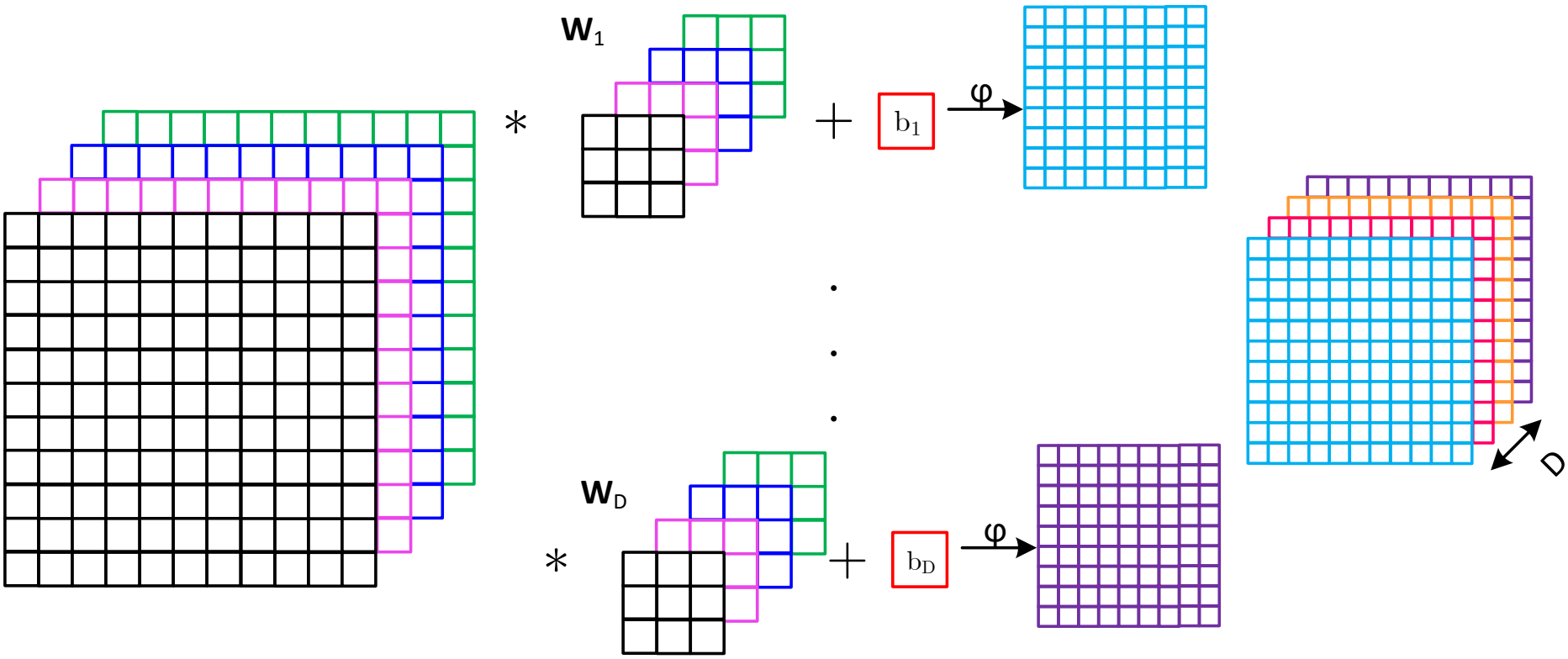
Convolution with multi-channel output

$$\begin{cases} \boldsymbol{X} \in \mathbb{R}^{h_x \times w_x \times C} \\ \{(\boldsymbol{W}_d, b_d)\}_{d=0}^{D-1} \\ \boldsymbol{W}_d \in \mathbb{R}^{h_w \times w_w \times C} \\ b_d \in \mathbb{R} \end{cases} \Rightarrow \begin{cases} z_{pq0} = \varphi\left(b_0 + \sum_{k=0}^{C-1} \sum_{i=0}^{h_w-1} \sum_{j=0}^{w_w-1} w_{ijk0} x_{(p+i)(q+j)k}\right) \\ \vdots \\ z_{pq(D-1)} = \varphi\left(b_{D-1} + \sum_{k=0}^{C-1} \sum_{i=0}^{h_w-1} \sum_{j=0}^{w_w-1} w_{ijk(D-1)} x_{(p+i)(q+j)k}\right) \end{cases}$$

$$\begin{cases} 0 \leq p \leq h_x - h_w \\ 0 \leq q \leq w_x - w_w \end{cases}$$

We can concatenate matrices $\{\boldsymbol{Z}_0, \ldots, \boldsymbol{Z}_{D-1}\}$ which results in $\boldsymbol{Z} \in \mathbb{R}^{(h_x - h_w + 1) \times (w_x - w_w + 1) D}$

Convolution with multi-channel output



Convolutional Layer

All Together

Assume:

$\boldsymbol{X} \in \mathbb{R}^{h_x \times w_x \times C}$	Input feature tensor
$\boldsymbol{W} \in \mathbb{R}^{h_w \times w_w \times C \times D}$	Weight tensor
$\boldsymbol{b} \in \mathbb{R}^{D}$	Bias vector
h_p, w_p	height and width of padding, respectively
h_s, w_s	height and width of stride, respectively

Then the output $\boldsymbol{Z} = \boldsymbol{W} \circledast \boldsymbol{X} + \boldsymbol{b}$ is of the following dimensions:

$$\underbrace{\left\lfloor \frac{h_x + 2h_p - h_w + h_s}{h_s} \right\rfloor}_{h_z} \times \underbrace{\left\lfloor \frac{w_x + 2w_p - w_w + w_s}{w_s} \right\rfloor}_{w_z} \times D$$

and

$$z_{pqd} = \varphi\left(b_c + \sum_{c=0}^{C-1} \sum_{i=0}^{h_w-1} \sum_{j=0}^{w_w-1} w_{ijkc} \widehat{x}_{(h_s \times p+i)(w_s \times q+j)c} \right), \begin{cases} 0 \leq p \leq h_z - 1 \\ 0 \leq q \leq w_z - 1 \\ 0 \leq d \leq D - 1 \end{cases}$$

Pooling Layers

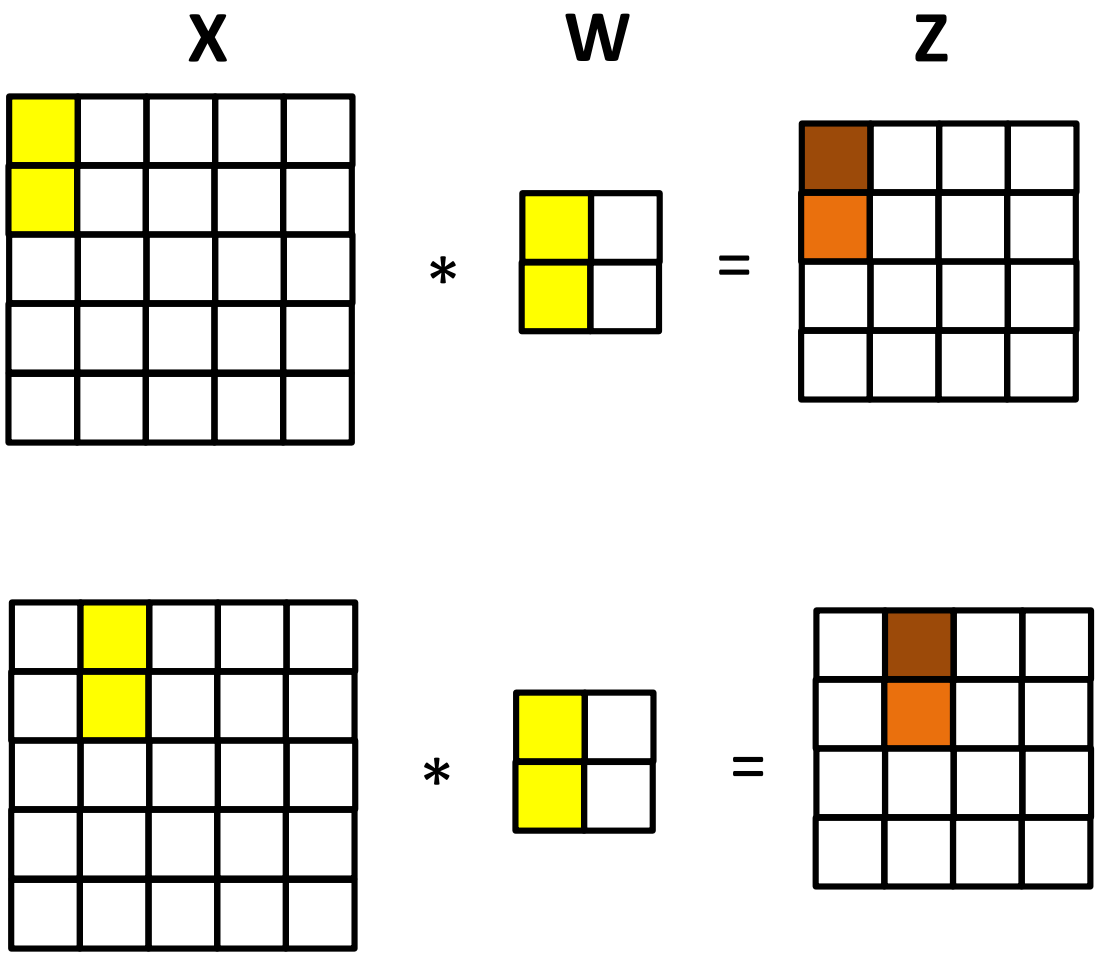


Figure: Output tensor carry information about the location

Pooling Layers

Assume:

$\boldsymbol{X} \in \mathbb{R}^{h_x \times w_x \times C}$	Input feature tensor
h_f, w_f	height and width of pooling, respectively
h_s, w_s	height and width of stride, respectively
$p(\cdot)$	Pooling operator

Then the output of pooling layer is:

$$z_{pqc} = p\left(x_{(h_s \times p : h_s \times p + h_f)(w_s \times q : w_s \times q + w_f)(c)}\right)$$

Pooling Layers

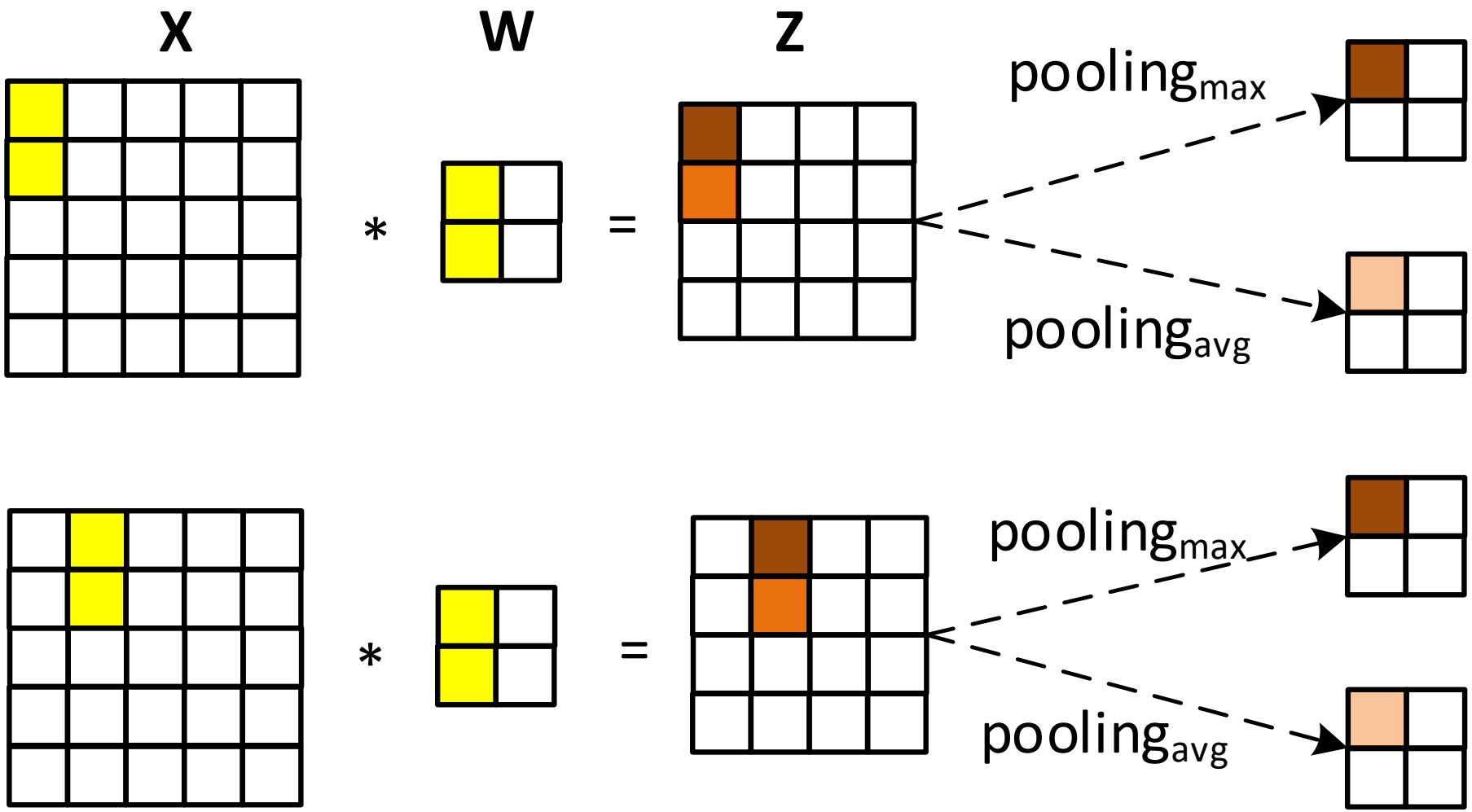


Figure: Pooling layer provide local invariance

Frame Title

Flattening layers are used to reshape the input feature tensor $\boldsymbol{X} \in \mathbb{R}^{h_x \times w_x \times C}$ into output feature vector $\boldsymbol{z} \in \mathbb{R}^{(h_x \times w_x \times C)}$ using vectorizing operator.

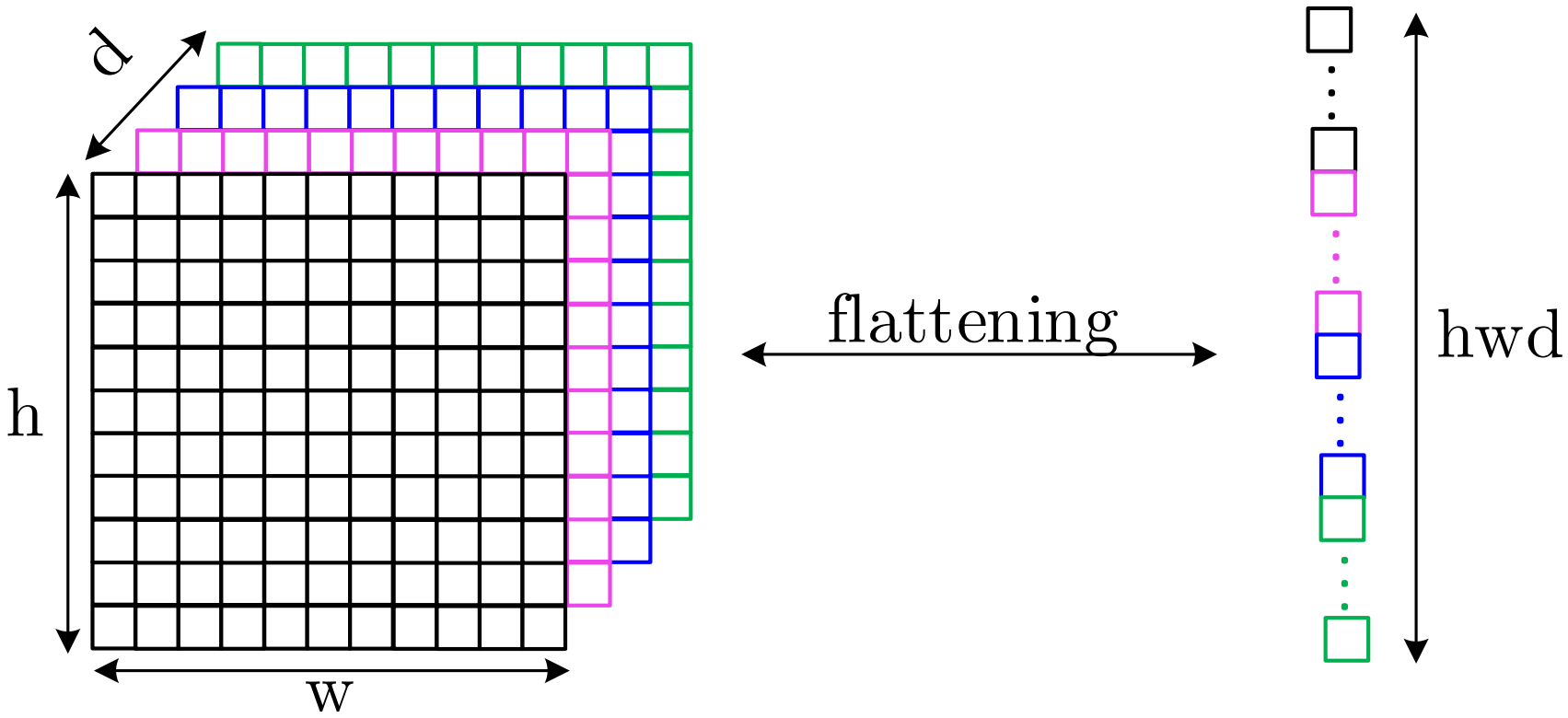


Figure: Flattening Layer

Section 3

All Together

LeNet5

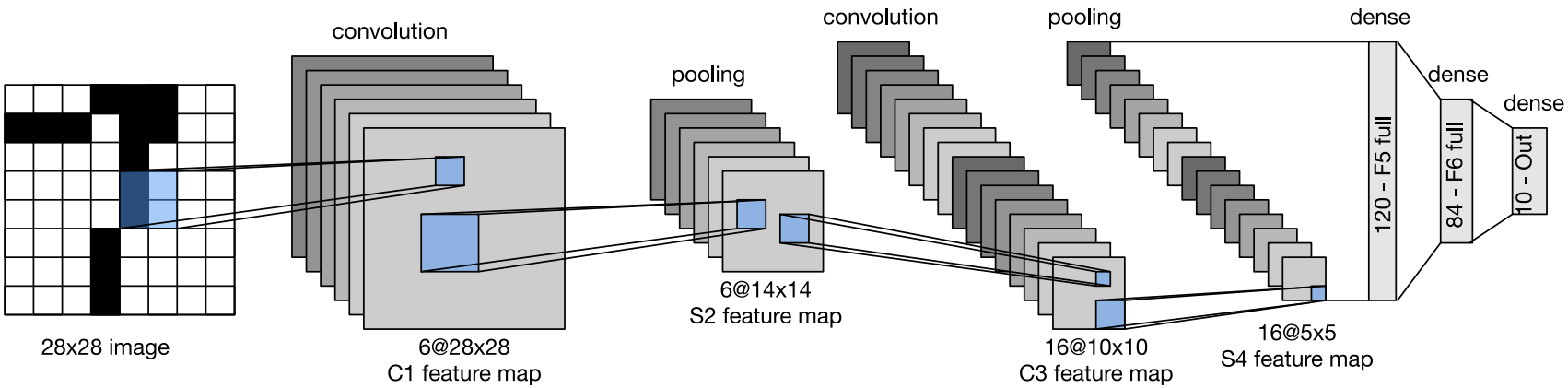


Figure: LeNet5 for MNIST classification (Test accuracy: 98.8% after 1 epoch)

LeNet5

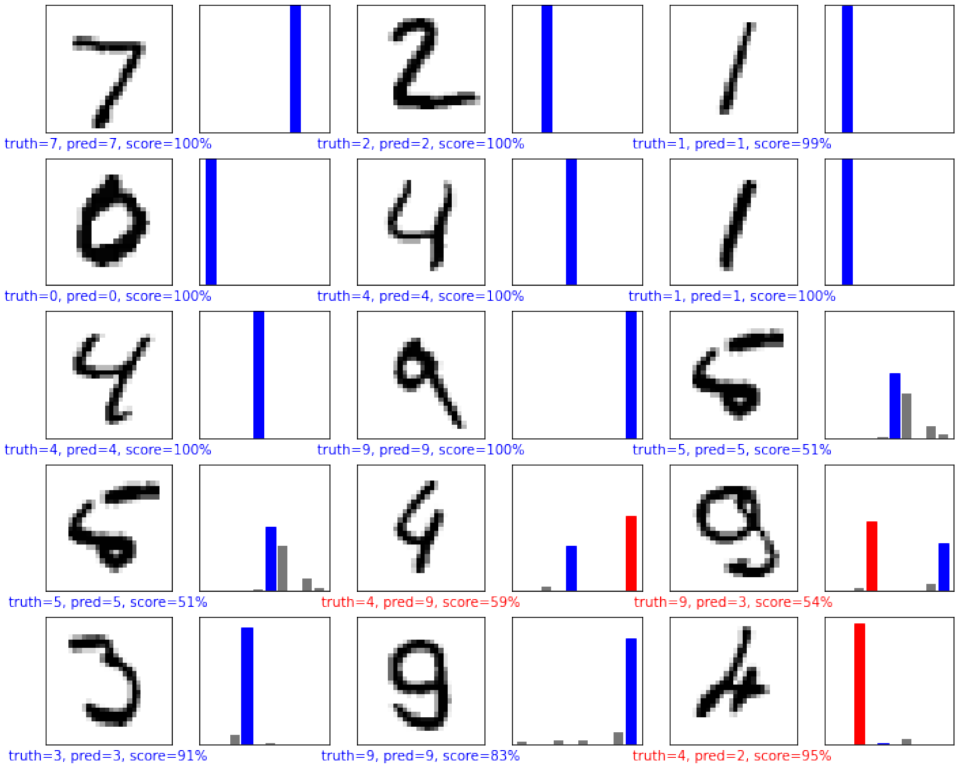


Figure: Result of LeNet5 for MNIST classification