Deep Learning Basics Convolution

HKUST MSBD 6000B

Instructor: Yu Zhang

Convolutional neural networks

- Strong empirical application performance

- Convolutional networks: neural networks that use convolution in place of general matrix multiplication in at least one of their layers

$$h = \sigma(W^T x + b)$$

for a specific kind of weight matrix W

Convolution

Convolution: math formula

- Given functions $u(t)$ and $w(t)$, their convolution is a function $s(t)$

$$s(t) = \int u(a)w(t-a)da$$

- Written as

$$s = (u * w) \quad \text{or} \quad s(t) = (u * w)(t)$$

Convolution: discrete version

- Given array u_t and w_t, their convolution is a function s_t

$$s_t = \sum_{a=-\infty}^{+\infty} u_a w_{t-a}$$

- Written as

$$s = (u * w) \quad \text{or} \quad s_t = (u * w)_t$$

- When u_t or w_t is not defined, assumed to be 0

Illustration 1

w = [z, y, x]
u = [a, b, c, d, e, f]

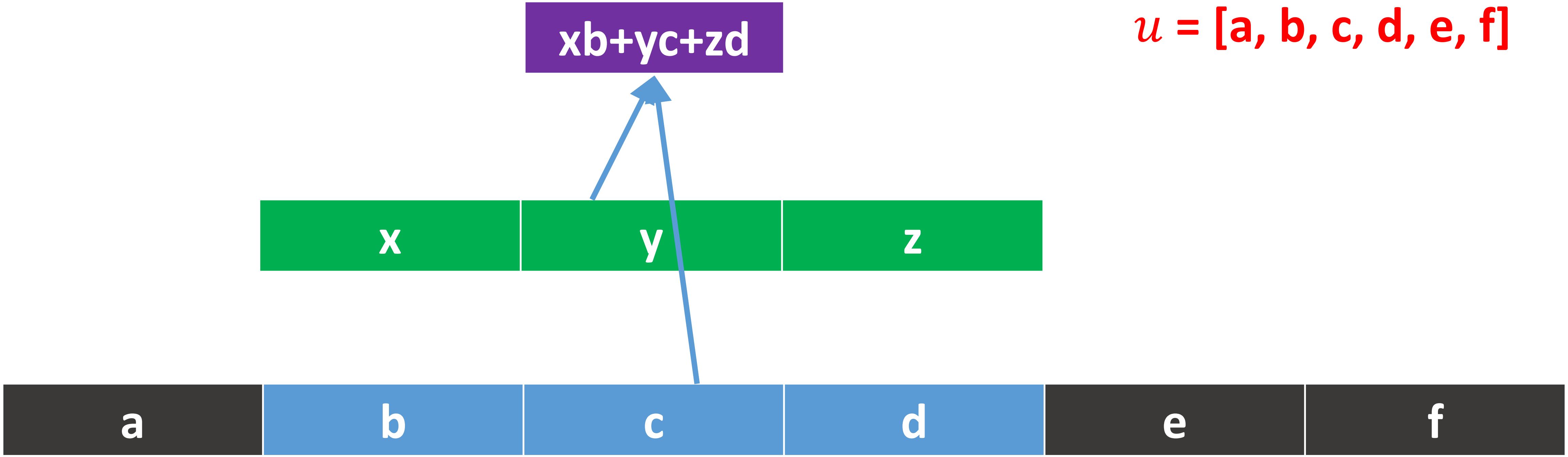


Illustration 1

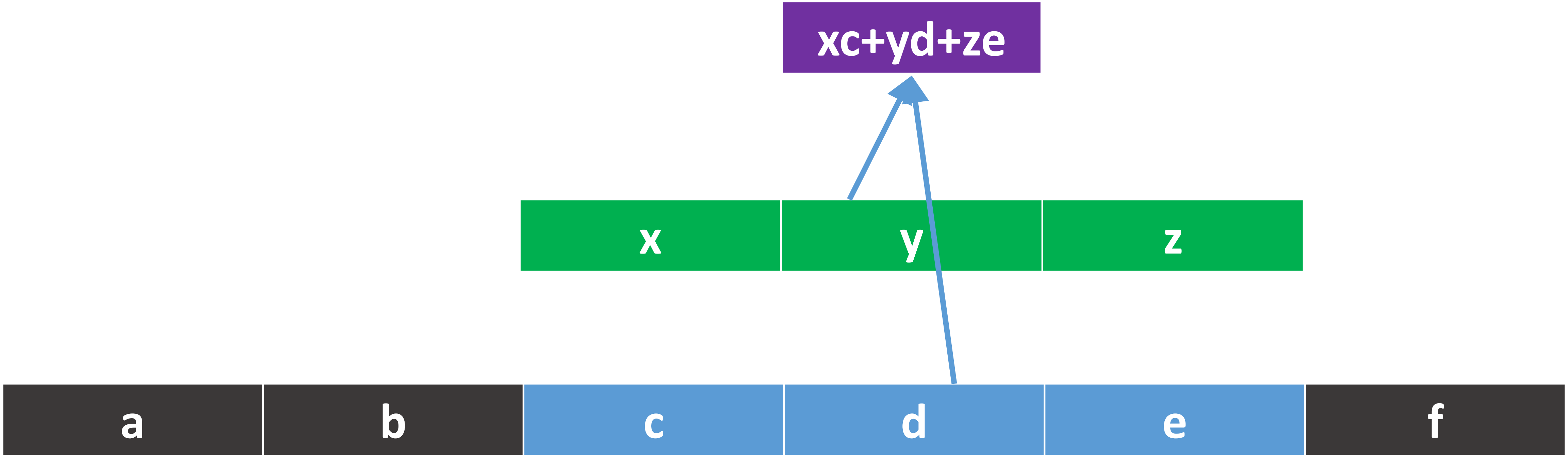


Illustration 1

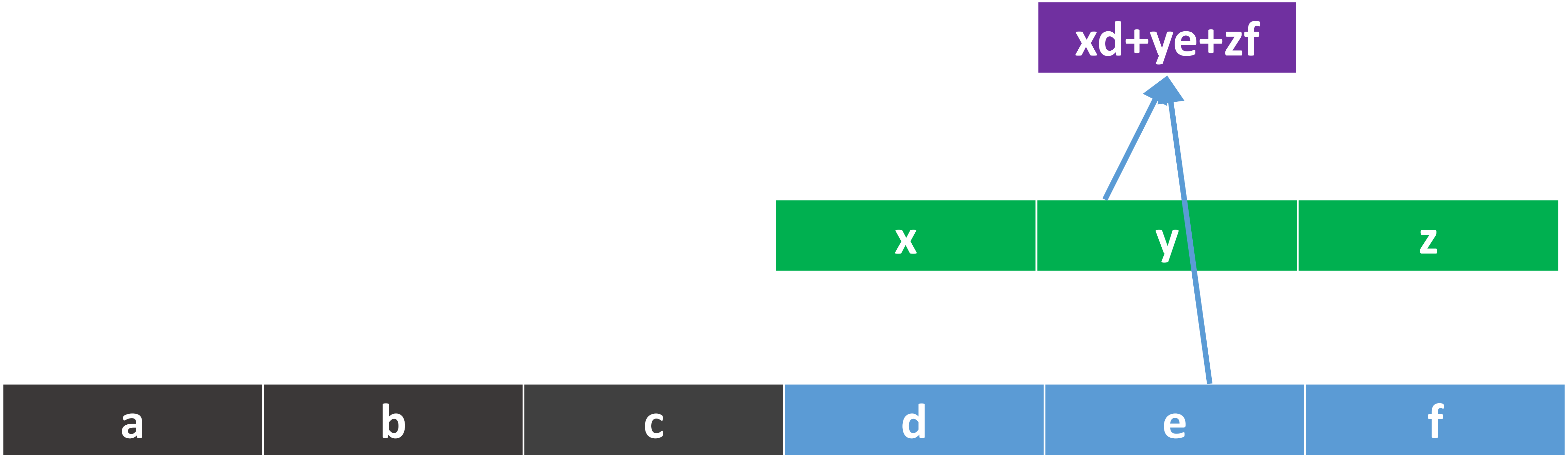


Illustration 1: boundary case

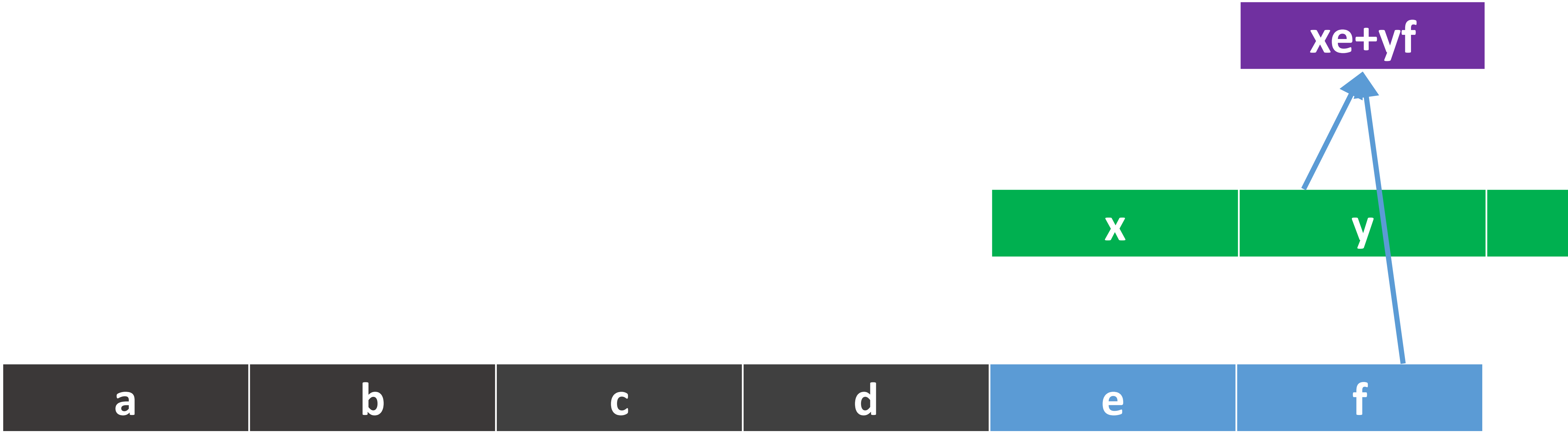


Illustration 1 as matrix multiplication

$$
\begin{bmatrix}
y & z & & & & \\
x & y & z & & & \\
 & x & y & z & & \\
 & & x & y & z & \\
 & & & x & y & z \\
 & & & & x & y
\end{bmatrix}
\begin{bmatrix}
a \\ b \\ c \\ d \\ e \\ f
\end{bmatrix}
$$

Illustration 2: two dimensional case

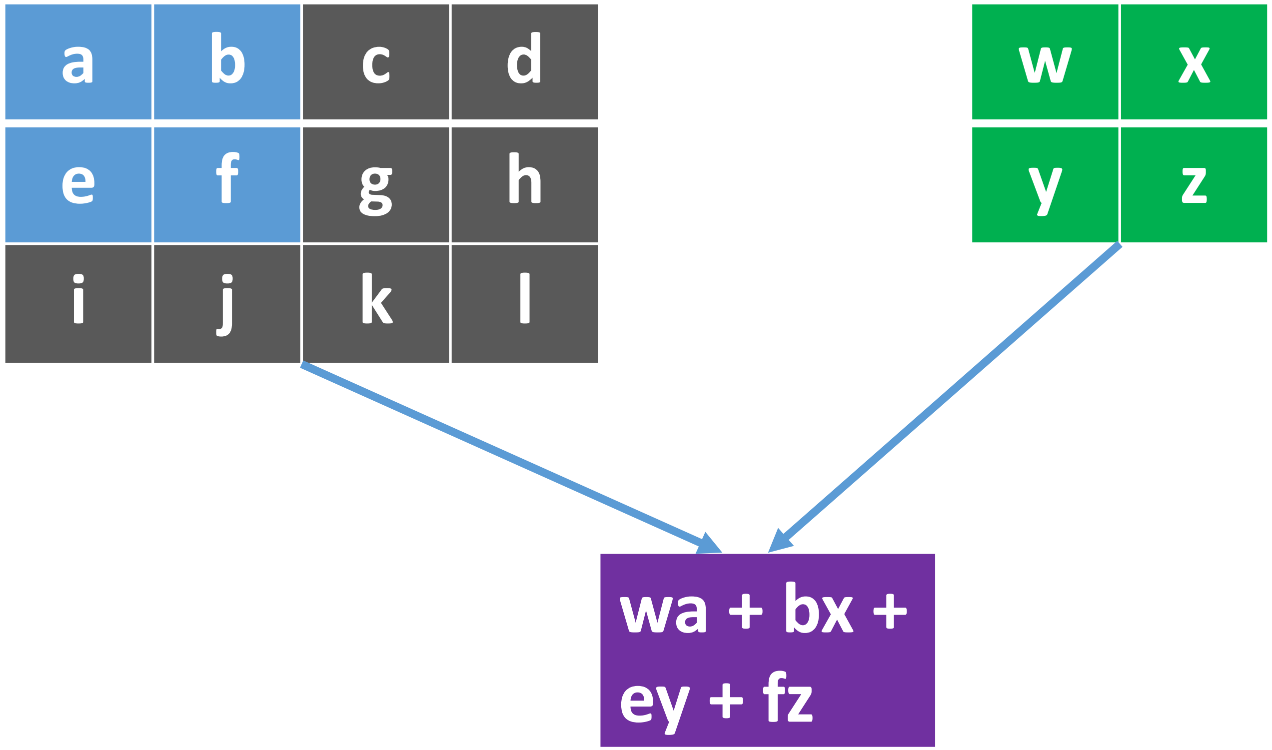


Illustration 2

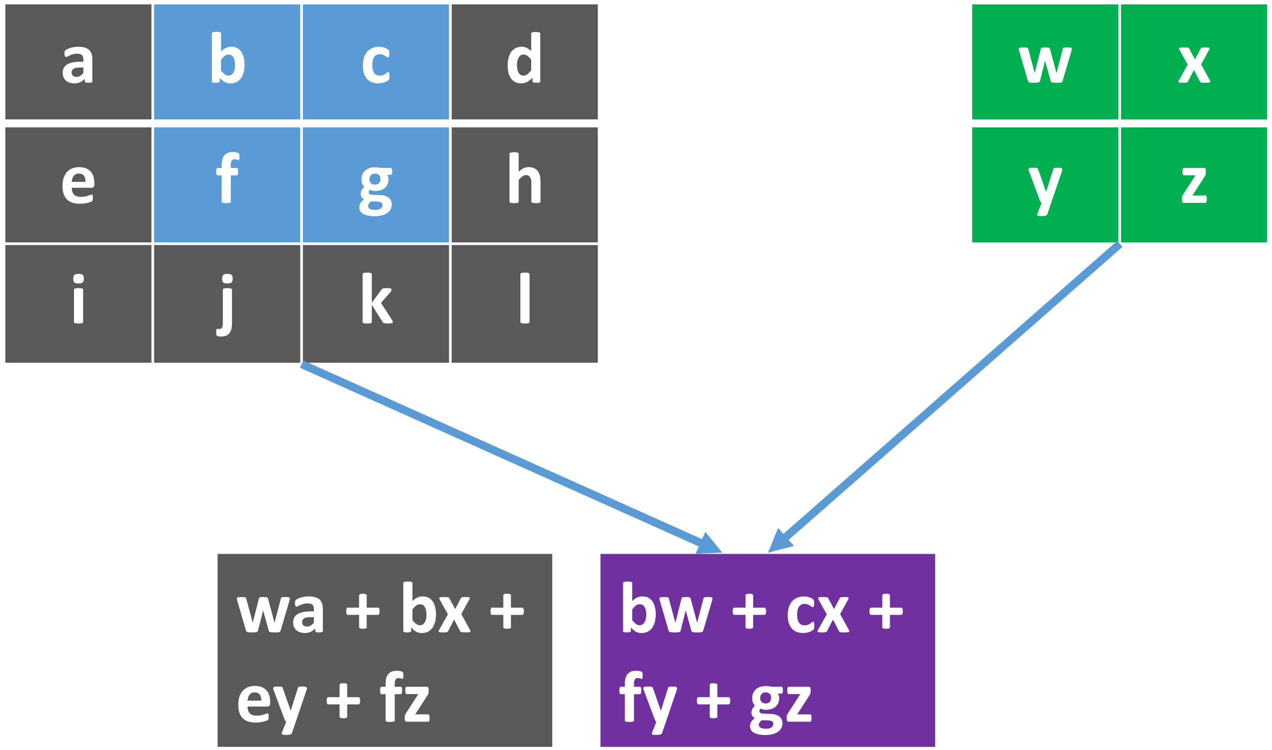
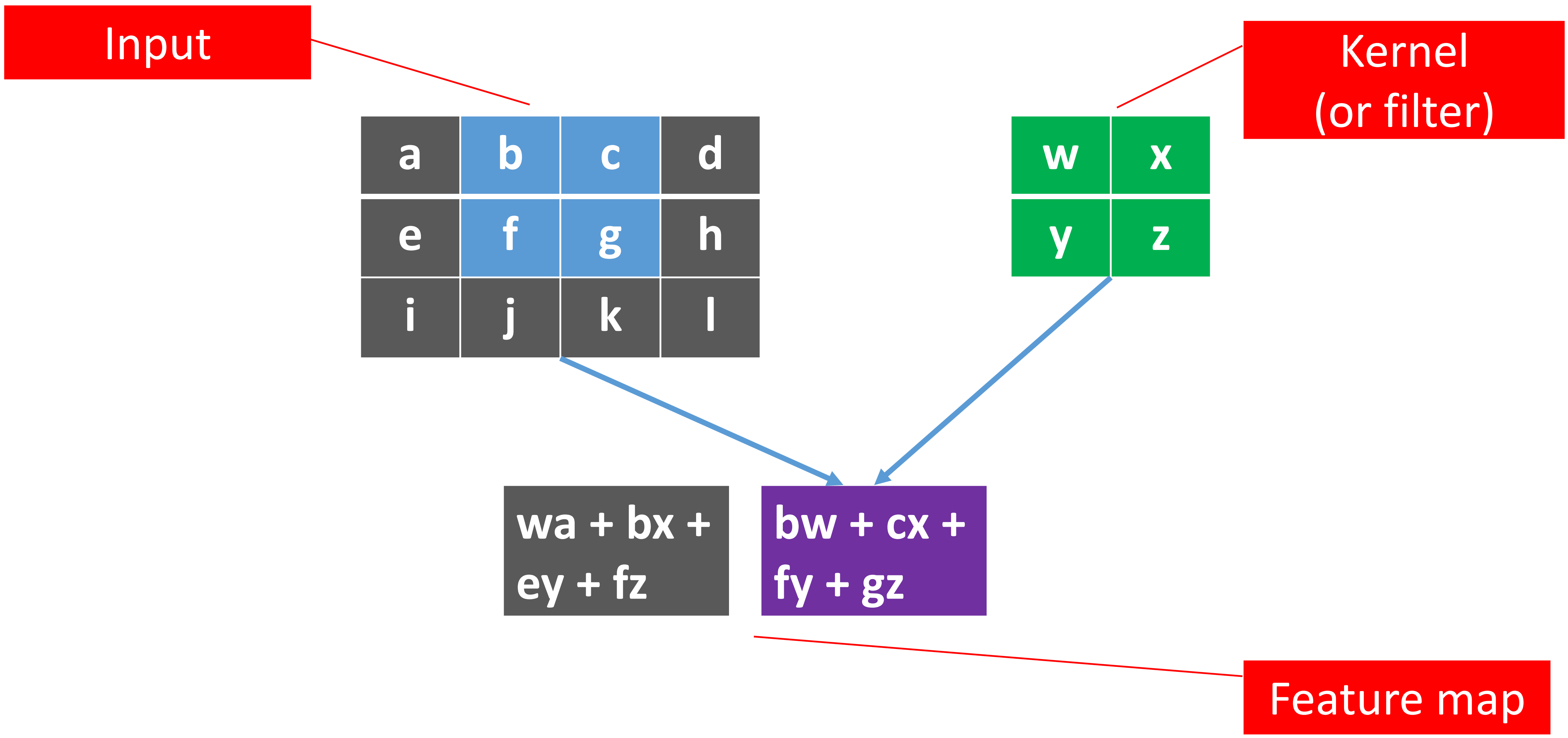
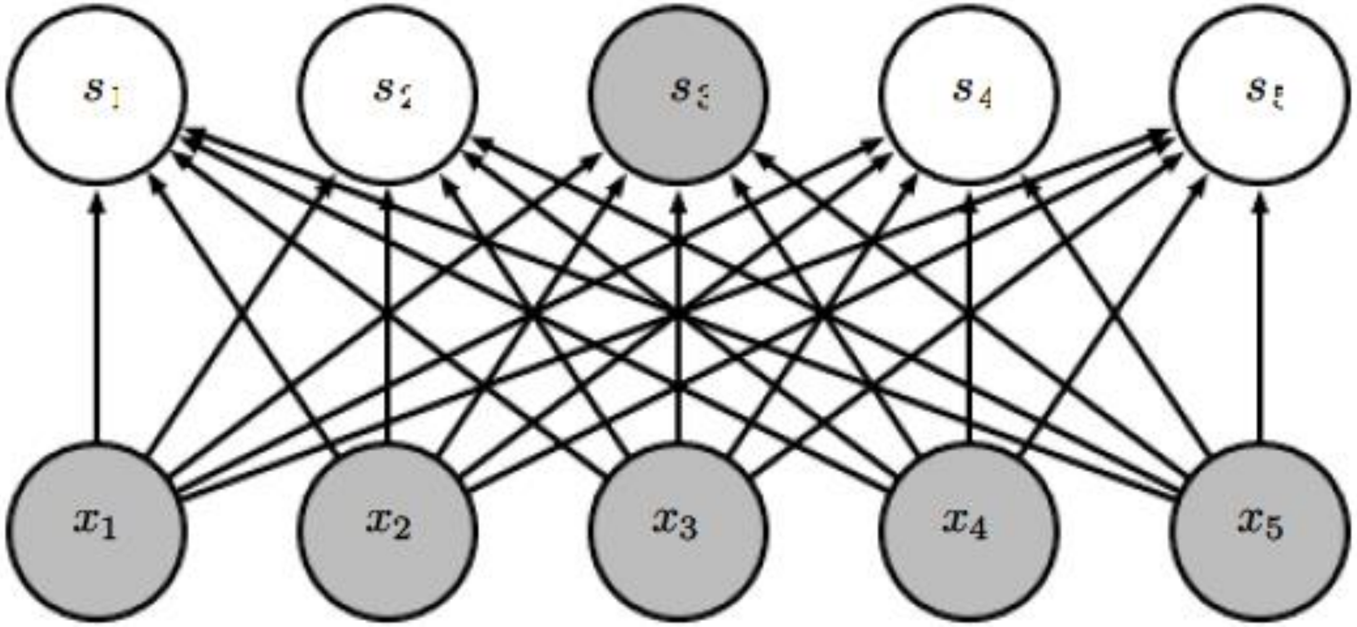


Illustration 2



Advantage: sparse interaction

Fully connected layer, $m \times n$ edges



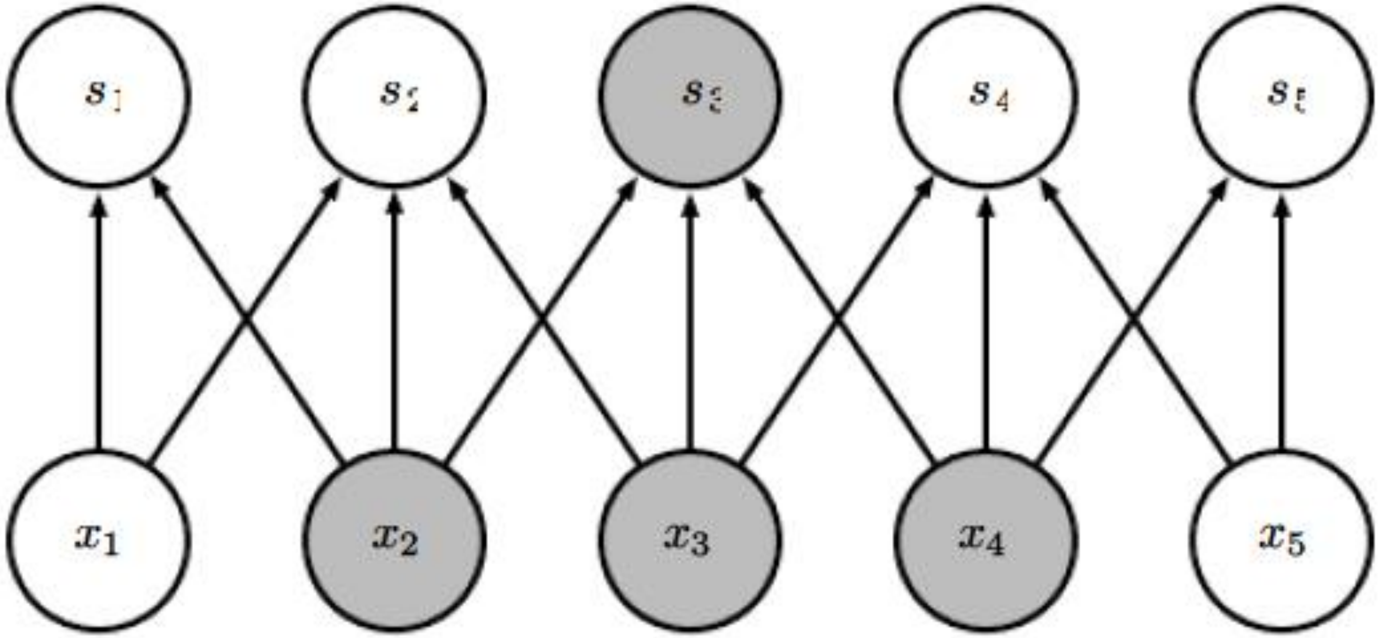
m output nodes

n input nodes

Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Advantage: sparse interaction

Convolutional layer, $\leq m \times k$ edges



m output nodes

k kernel size

n input nodes

Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Advantage: sparse interaction

Multiple convolutional layers: larger receptive field

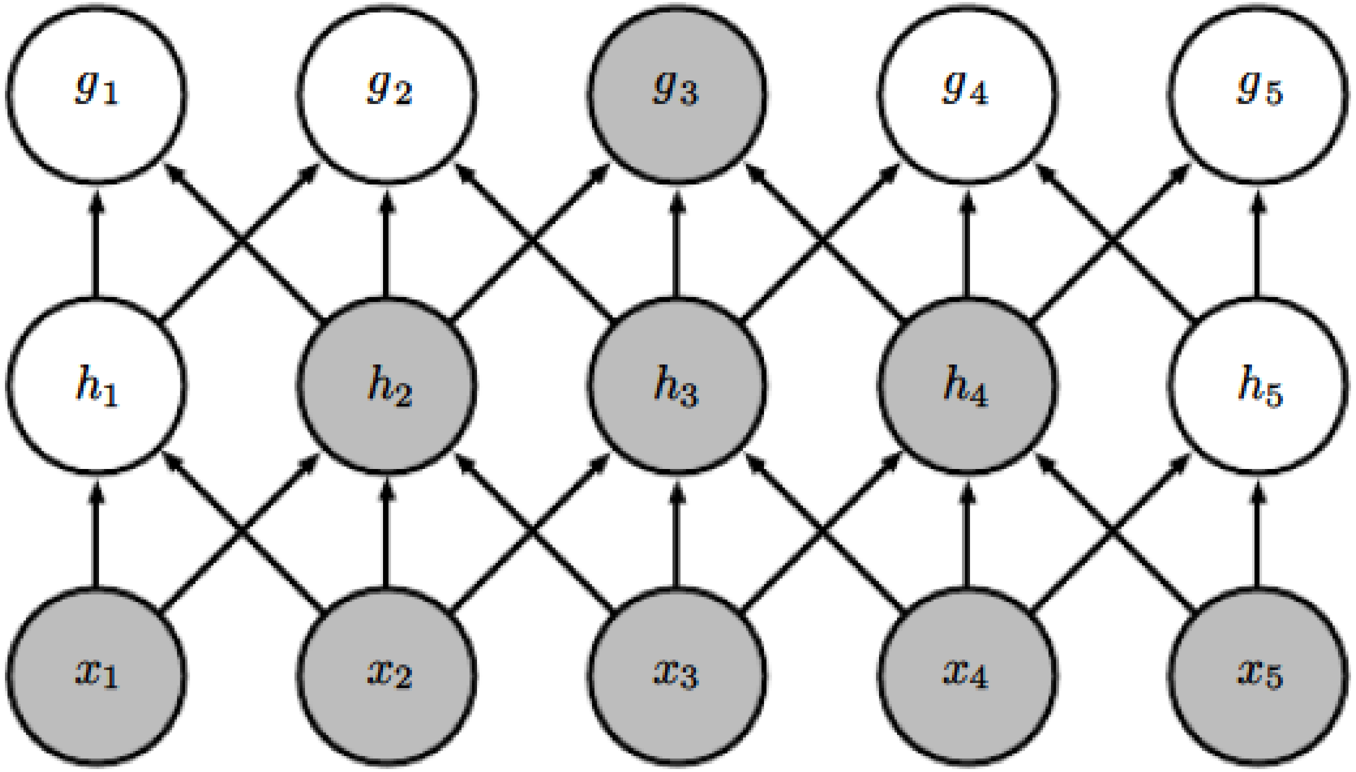
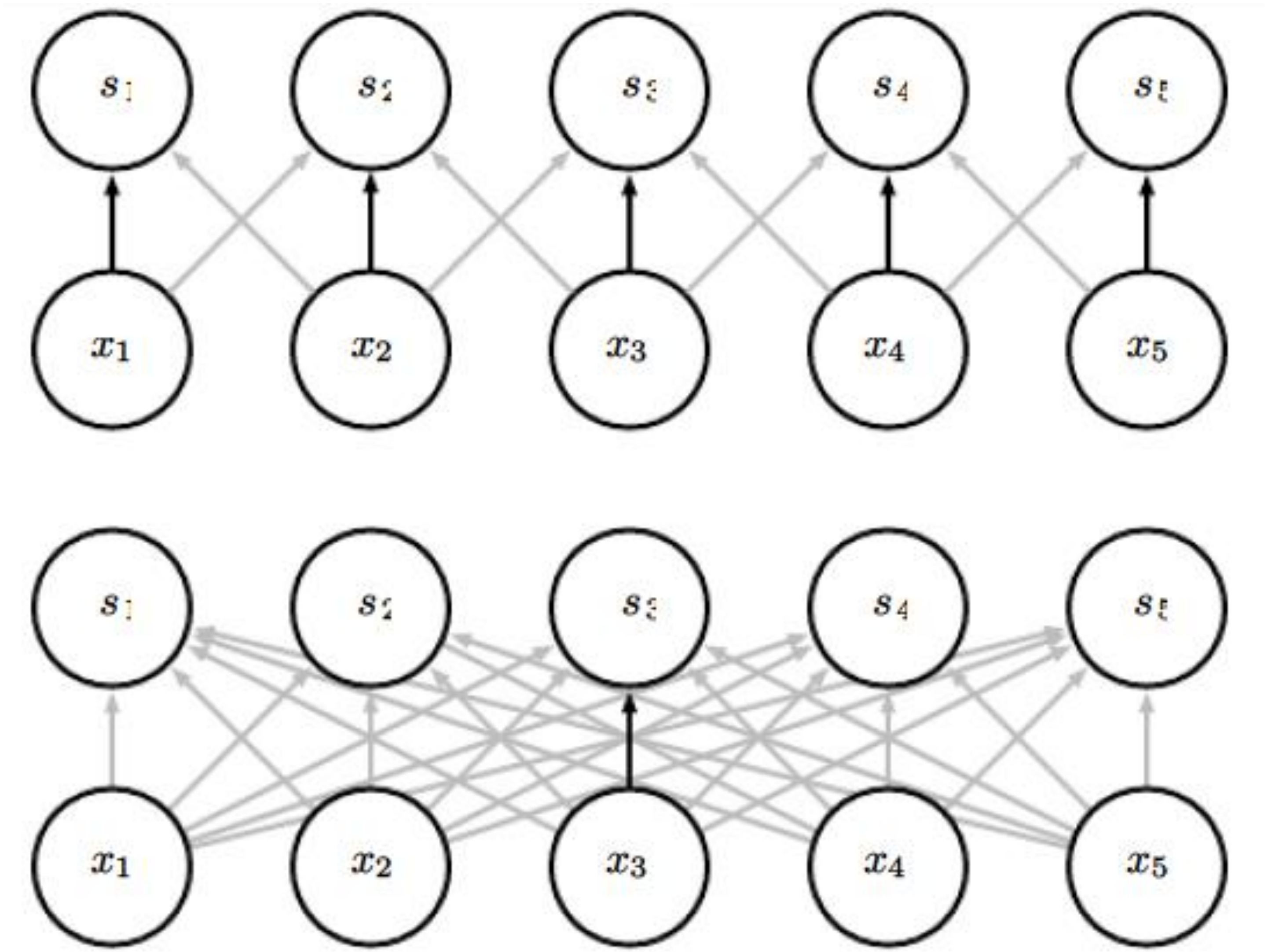


Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Advantage: parameter sharing



The same kernel are used repeatedly. E.g., the black edge is the same weight in the kernel.

Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Advantage: equivariant representations

- Equivariant: transforming the input = transforming the output

- Example: input is an image, transformation is shifting
- Convolution(shift(input)) = shift(Convolution(input))

- Useful when care only about the existence of a pattern, rather than the location

Pooling

Terminology

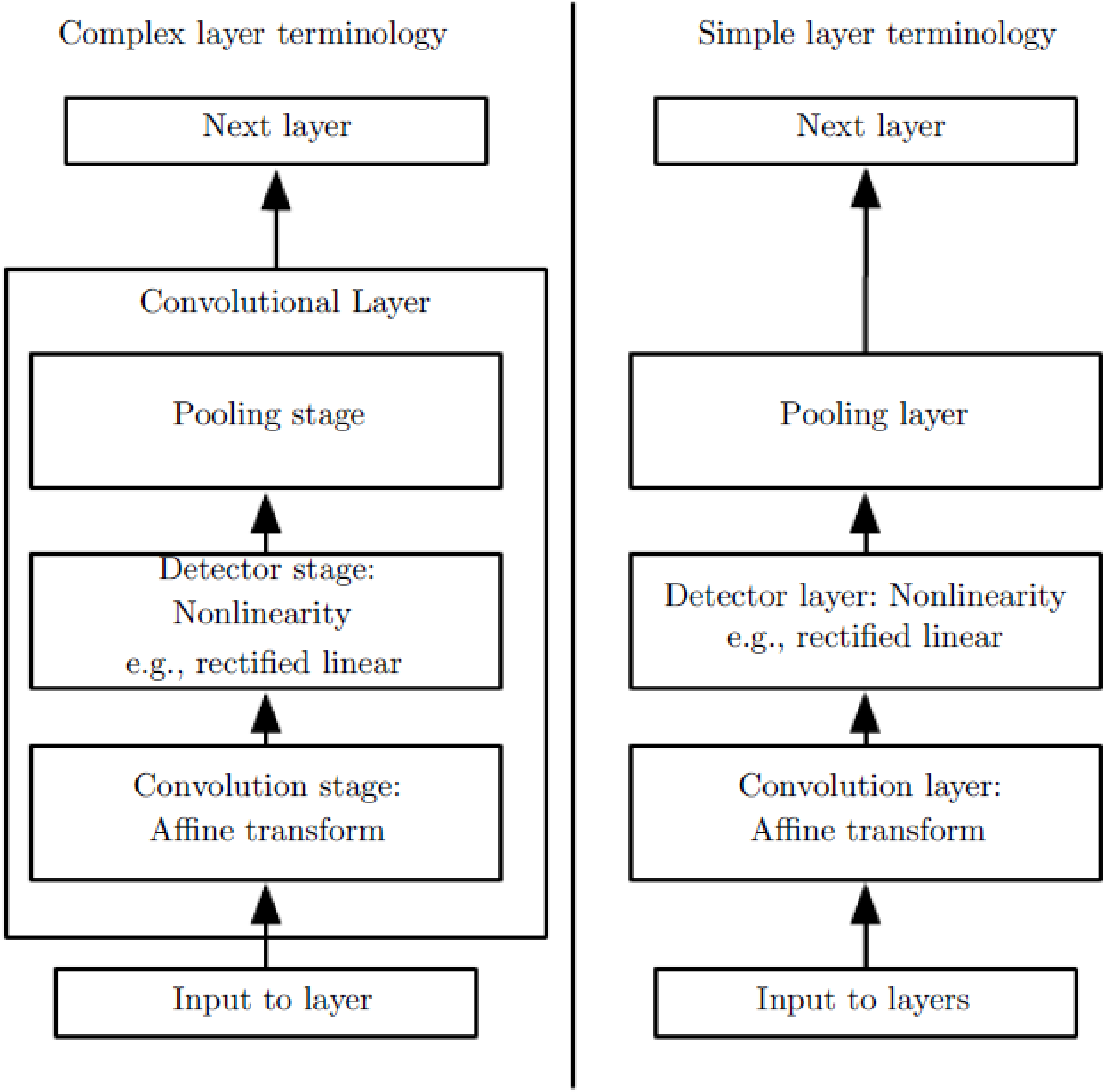


Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Pooling

- Summarizing the input (i.e., output the max of the input)

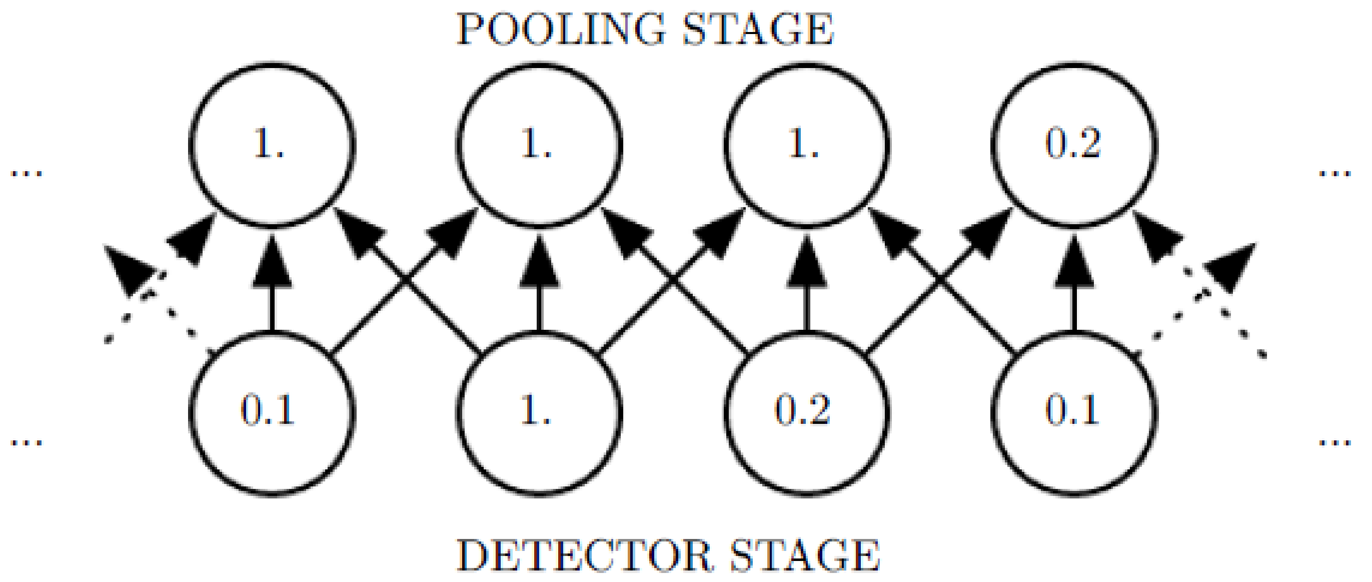


Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Motivation from neuroscience

- David Hubel and Torsten Wiesel studied early visual system in human brain (V1 or primary visual cortex), and won Nobel prize for this

- V1 properties
  - 2D spatial arrangement
  - Simple cells: inspire convolution layers
  - Complex cells: inspire pooling layers

Variants of convolution and pooling

Variants of convolutional layers

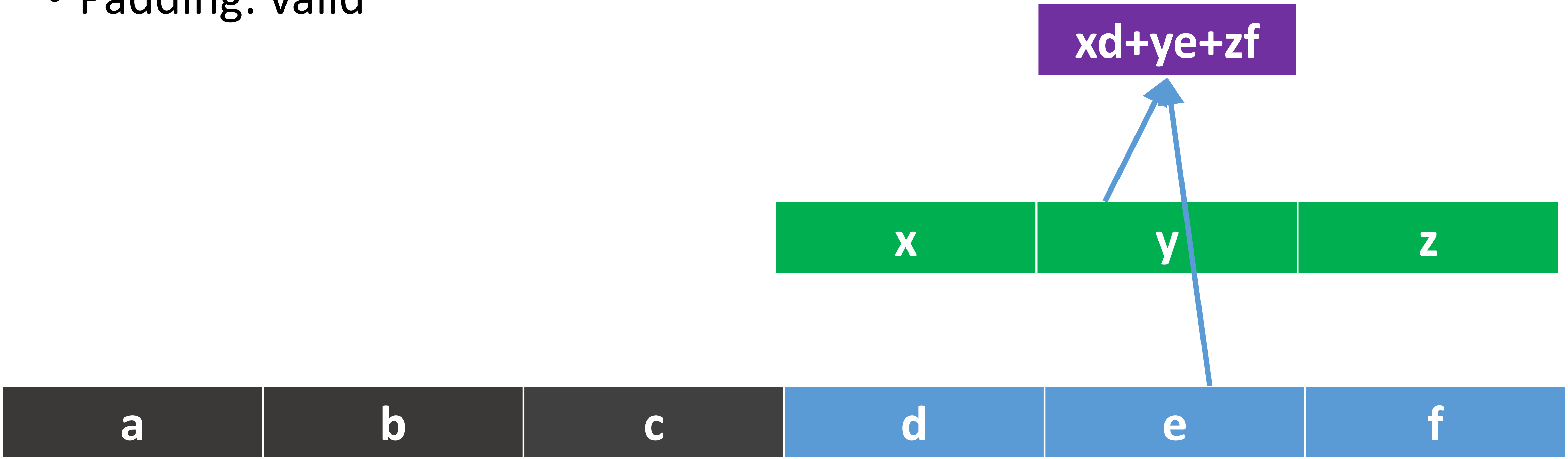
- Multiple dimensional convolution

- Input and kernel can be 3D
  - E.g., images have (width, height, RBG channels)
- Multiple kernels lead to multiple feature maps (also called channels)

- Mini-batch of images have 4D: (image_id, width, height, RBG channels)

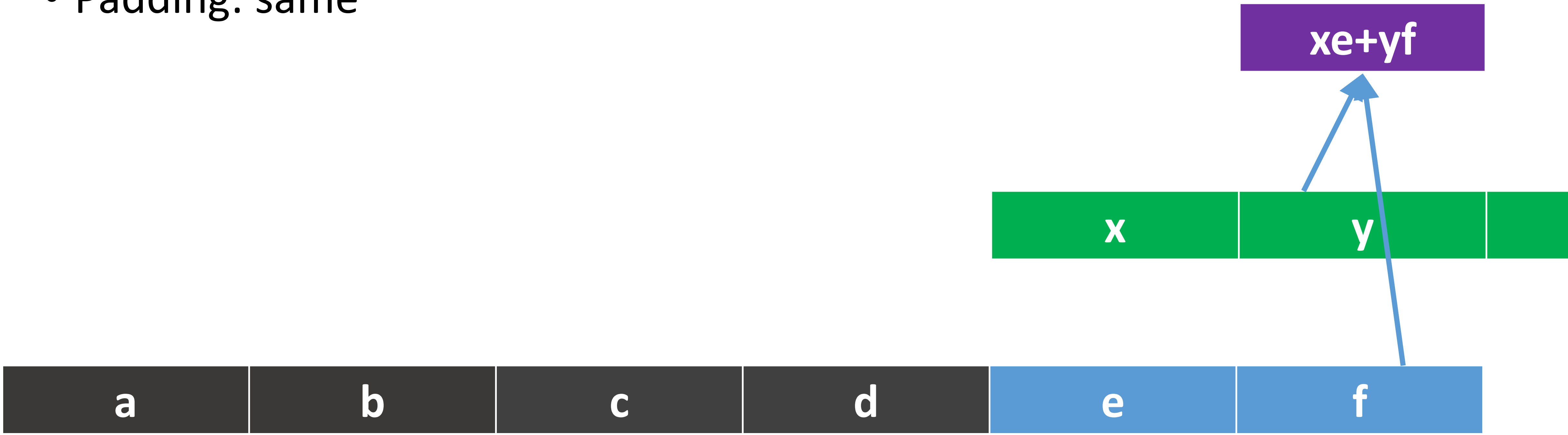
Variants of convolutional layers

- Padding: valid



Variants of convolutional layers

- Padding: same



Variants of convolutional layers

- Stride

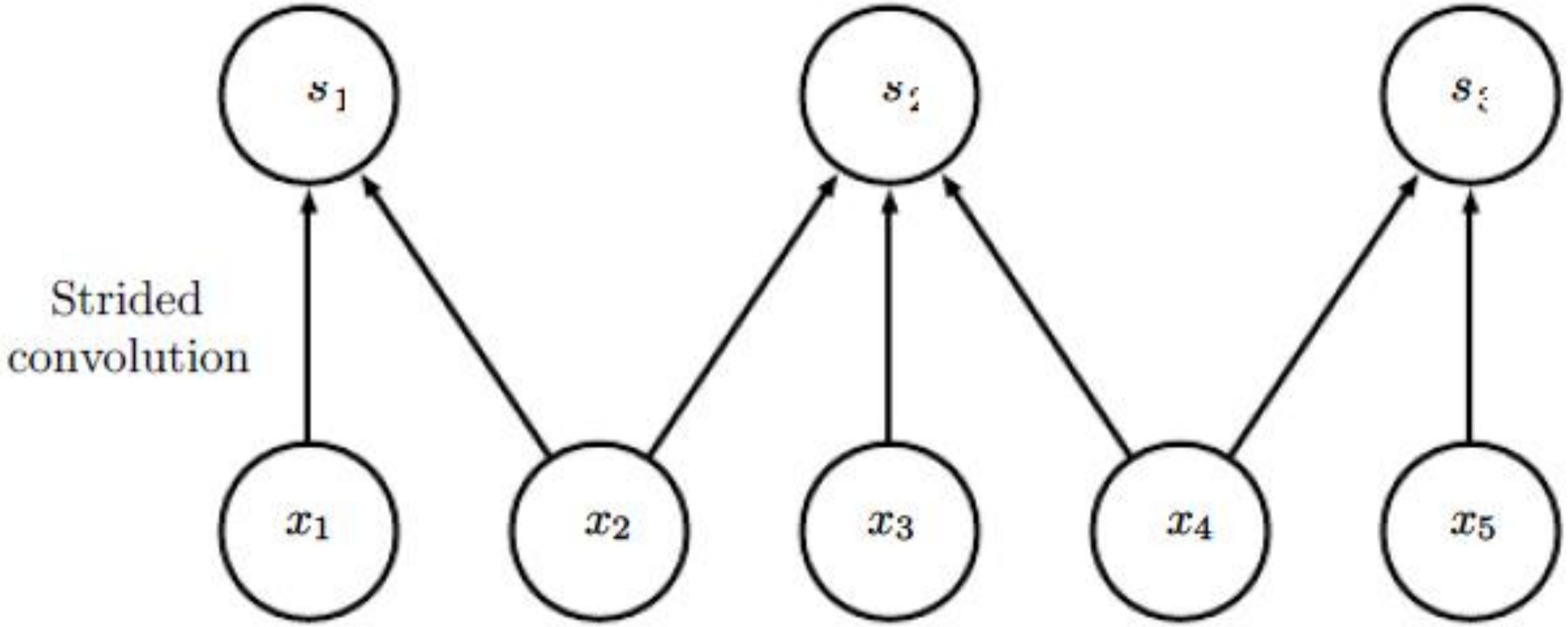


Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Variants of pooling

- Stride and padding

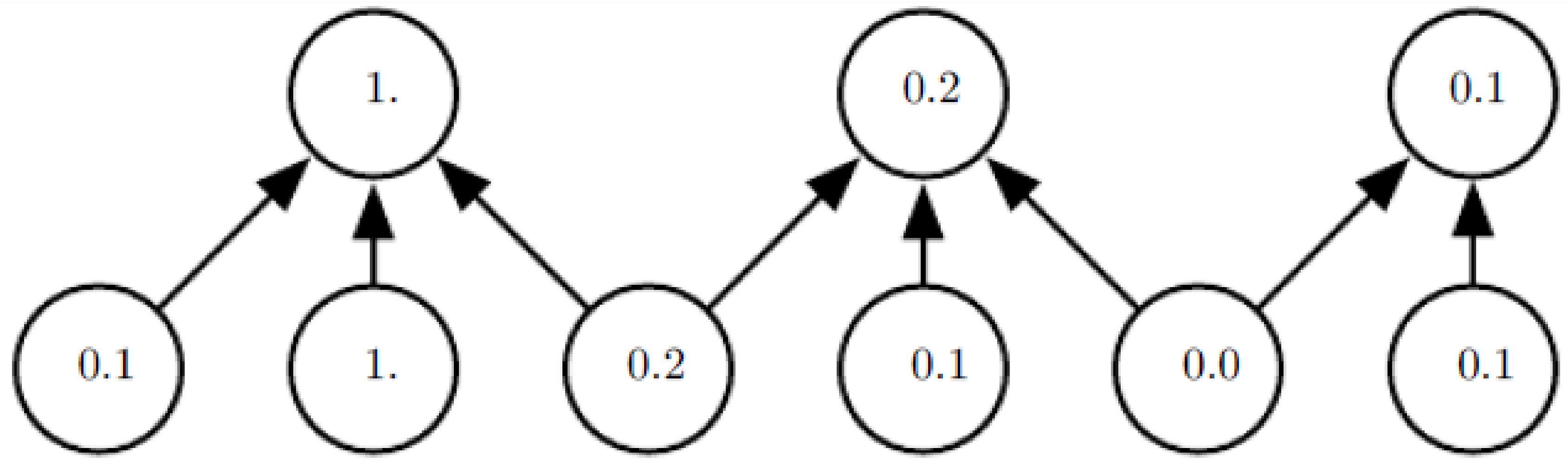


Figure from *Deep Learning,* by Goodfellow, Bengio, and Courville

Variants of pooling

- Max pooling $y = \max\{x_1, x_2, \ldots, x_k\}$
- Average pooling $y = \text{mean}\{x_1, x_2, \ldots, x_k\}$

- Others like max-out