Convolution and Pooling

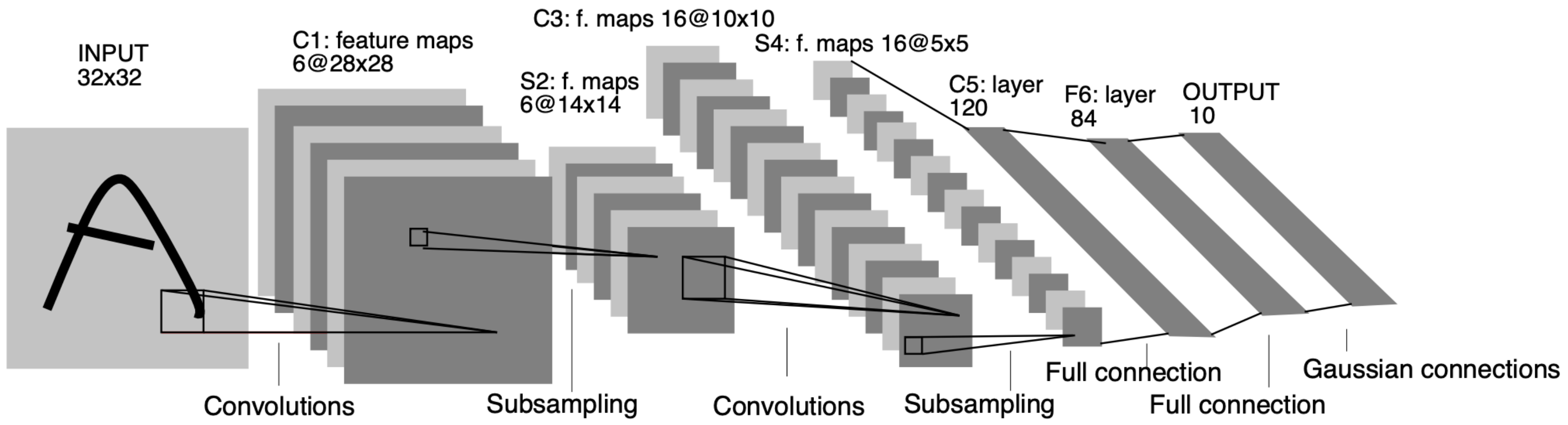
Sangheum Hwang

Computer Vision Tasks

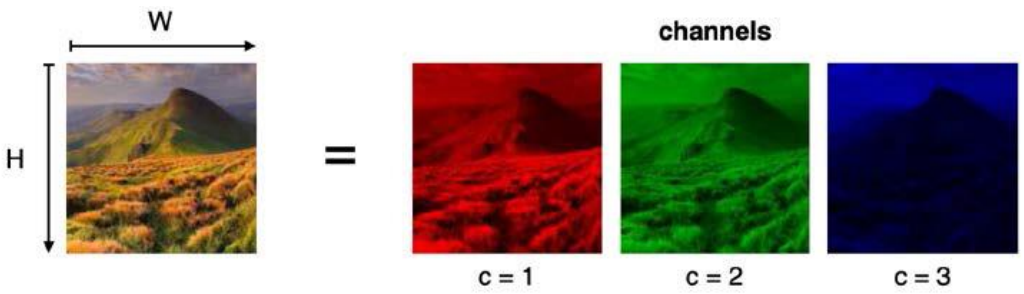
- Classification
- Object detection
- Semantic or instance segmentation
- Others
  - Tracking in videos, camera pose estimation, body pose estimation, 3D reconstruction, denoising, super-resolution, auto-captioning, synthesis, etc.

Convolutional Neural Networks

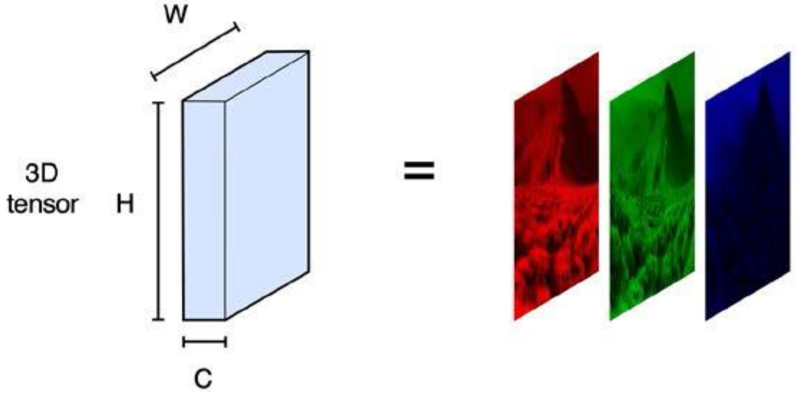
- A specialized kind of neural network for processing data that has a known grid-like topology such as time-series data, image data, video data, etc.
- Ex. LeNet-5 (LeCun et al., 1998)



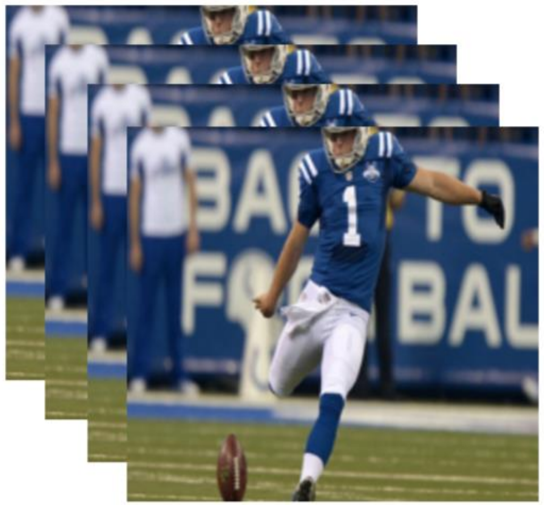
Recap. Tensors



- 3D tensor = 3-dimensional array
  - RGB color image
  - (height, width, channel)
  - Grayscale image = 2D tensor



- 4D tensor = 4-dimensional array
  - Color video
  - (time, height, width, channel)



Sangheum Hwang

Motivation

- A linear layer taking a 256*256 RGB image as input, and producing an image of same size would require $(256 \times 256 \times 3)^2 \cong 3.87 \times 10^{10}$.

- Some input signals have some "invariance in translation".
  - A function f of x is invariant to a transformation T if $f(T(x)) = f(x)$.
  - A transformation meaningful at a certain location can be used everywhere.

- A convolution layer embodies this idea.
  - It applies the same linear transformation locally, everywhere.

Convolution

- Convolution is a spatial filtering.

$* \; 1/8$

0	1	0
1	4	1
0	1	0

$*$

0	-1	0
-1	4	-1
0	-1	0

$*$

1	0	-1
2	0	-2
1	0	-1

Convolution with 1D Array Input

- **kernel** = [a,b] ← learnable parameters
  (*a.k.a.* filter)
- input size m=6, kernel size k=2, stride(kernel step size) s=1, output size n= (m-k)/s+1=5

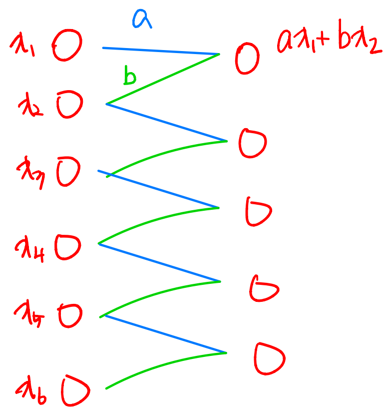
$$\frac{6-2}{1}+1=5$$

output

ax_1+bx_2	ax_2+bx_3	ax_3+bx_4	ax_4+bx_5	ax_5+bx_6

$+bias$

locally LT



kernel

a	b

⇒ shared all connections

input
6 node

x_1	x_2	x_3	x_4	x_5	x_6

Figure credit: SKKU Prof. Kang

Convolution with 1D Array Input

- **kernel** = [a,b] ← parameters
- input size m=6, kernel size k=2, stride(kernel step size) s=1, output size n= (m-k)/s+1=5

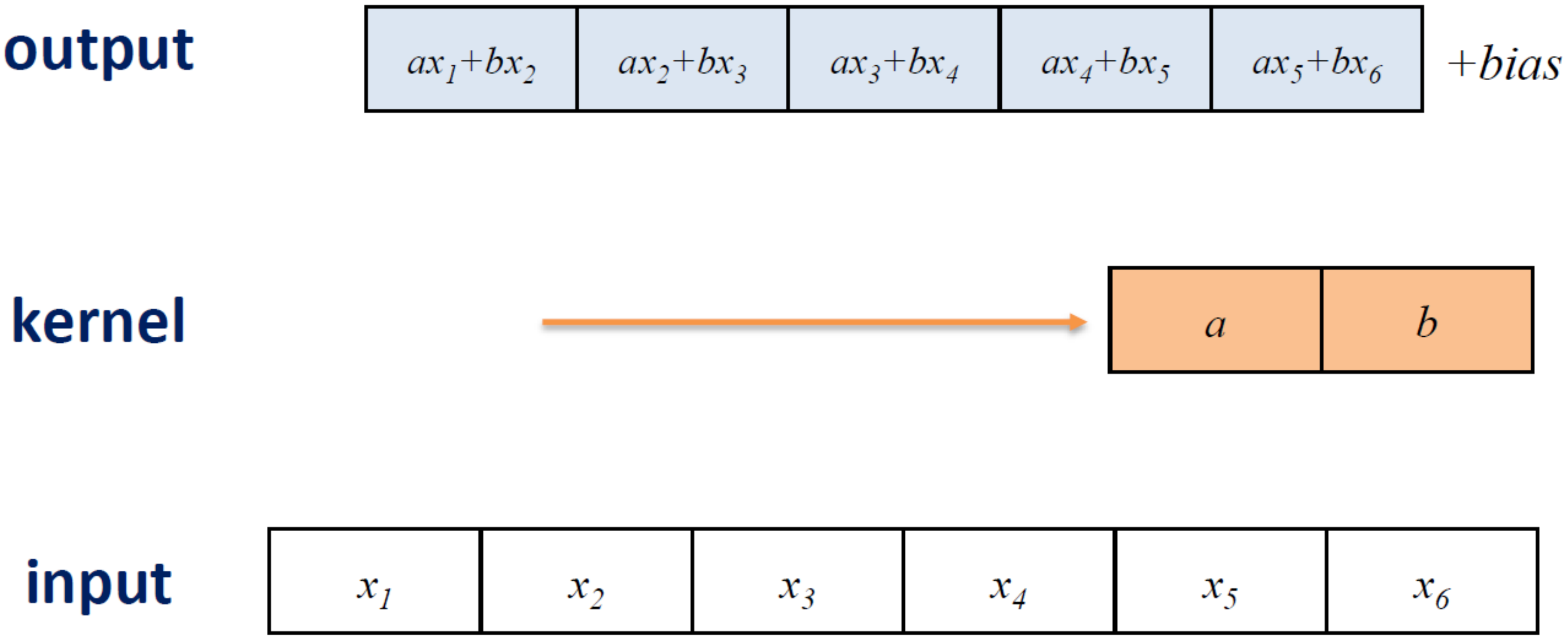


Figure credit: SKKU Prof. Kang

Sangheum Hwang

Convolution with 1D Array Input

- **kernel** = [a,b] ← parameters
- input size m=6, kernel size k=2, stride(kernel step size) s=2, output size n= (m-k)/s+1=3

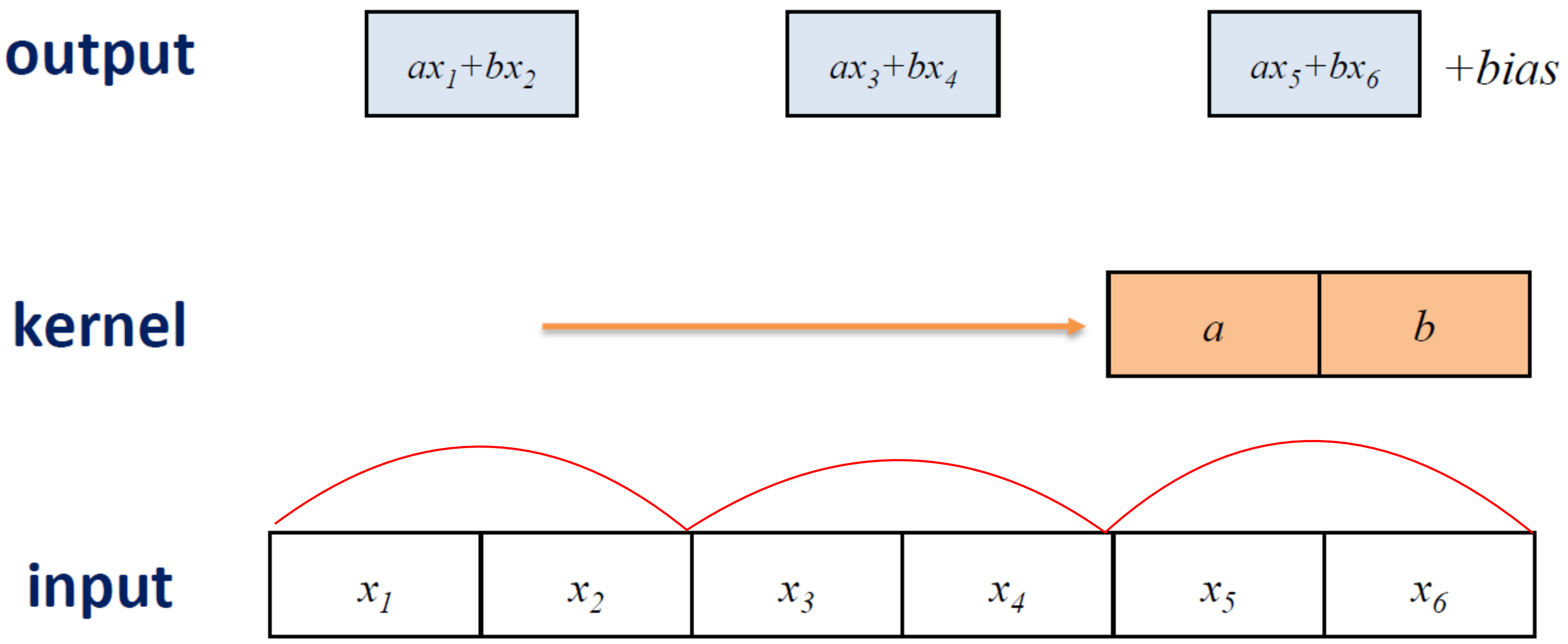


Figure credit: SKKU Prof. Kang

Convolution with 1D Array Input

- **kernel** = [a,b,c] ← parameters
- input size m=6, kernel size k=3, stride(kernel step size) s=1, output size n= (m-k)/s+1=4

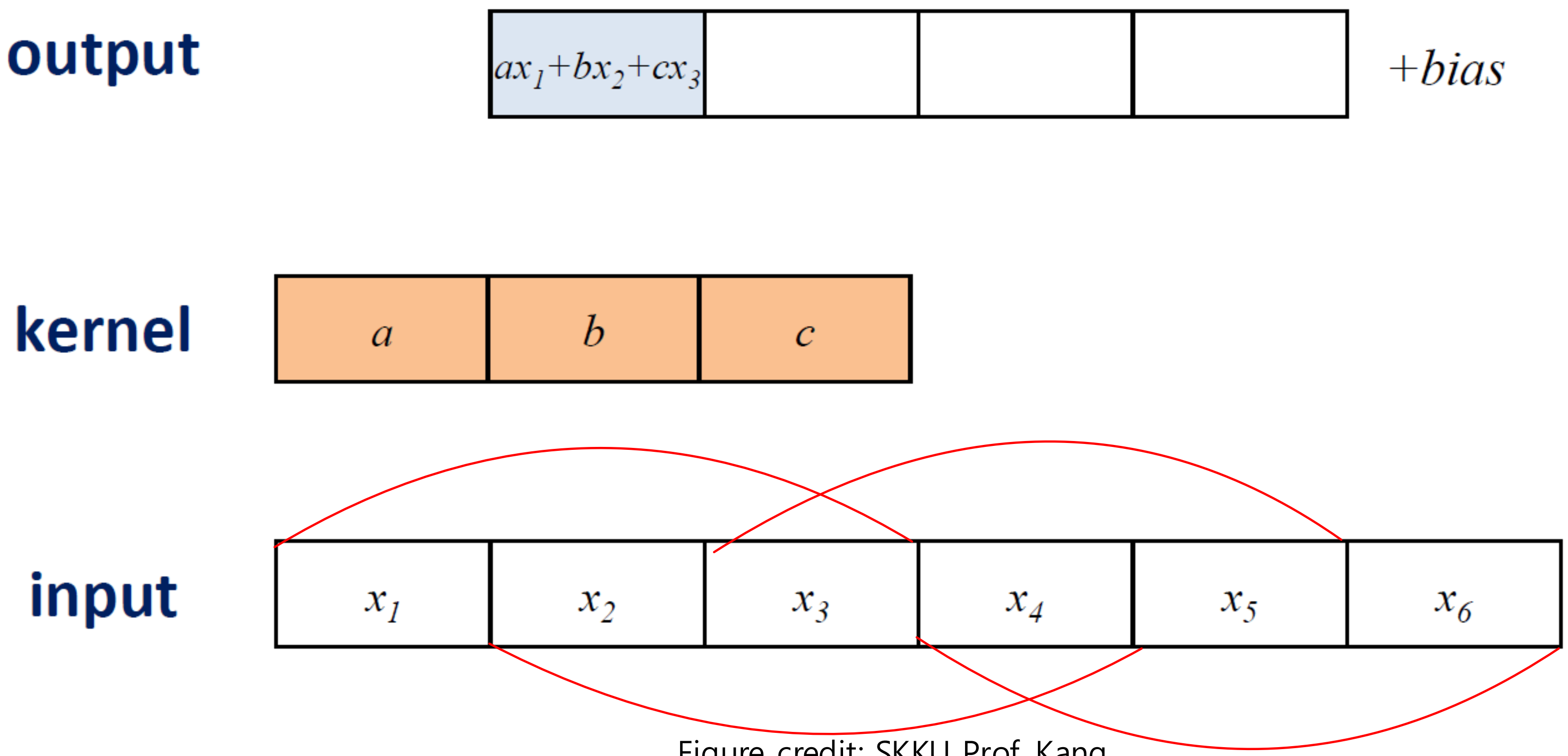


Figure credit: SKKU Prof. Kang

Convolution with 1D Array Input

- **kernel** = [a,b,c] ← parameters
- input size m=6, kernel size k=3, stride(kernel step size) s=1, output size n= (m-k)/s+1=4

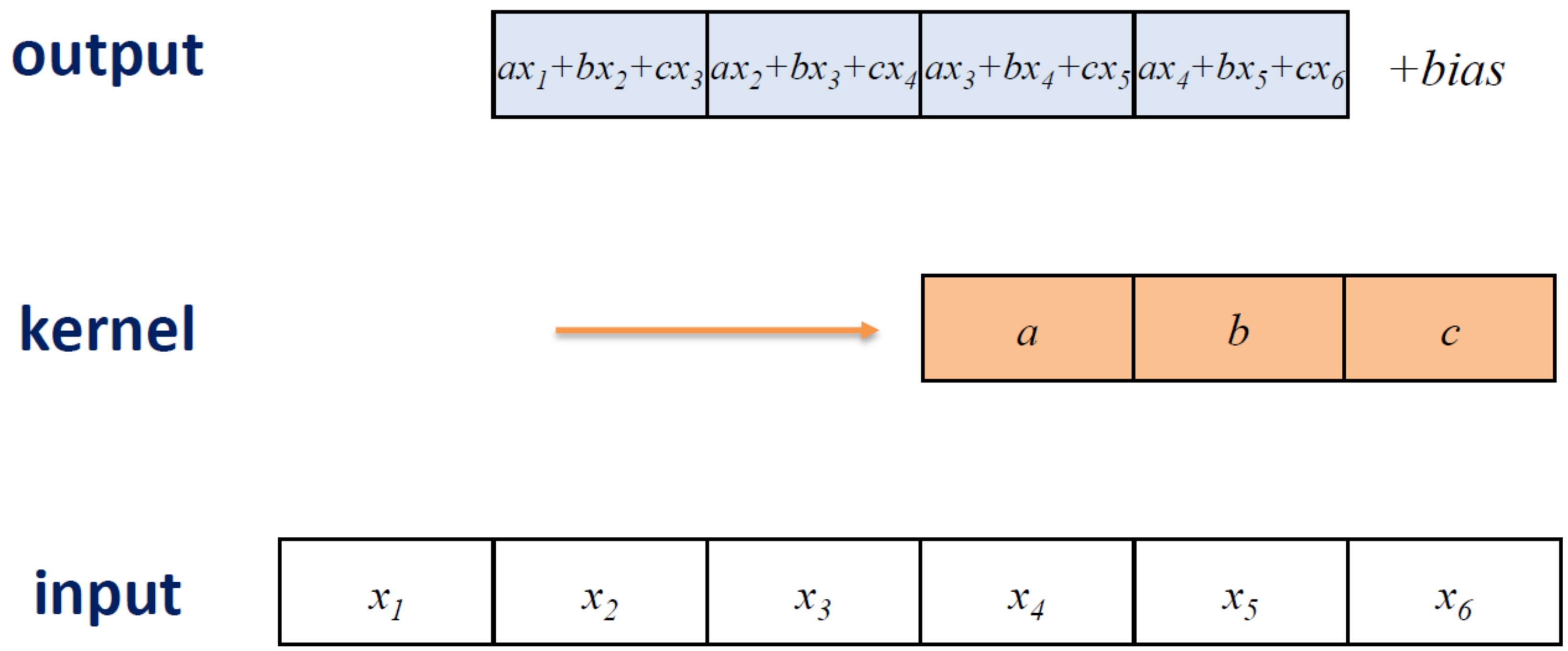


Figure credit: SKKU Prof. Kang

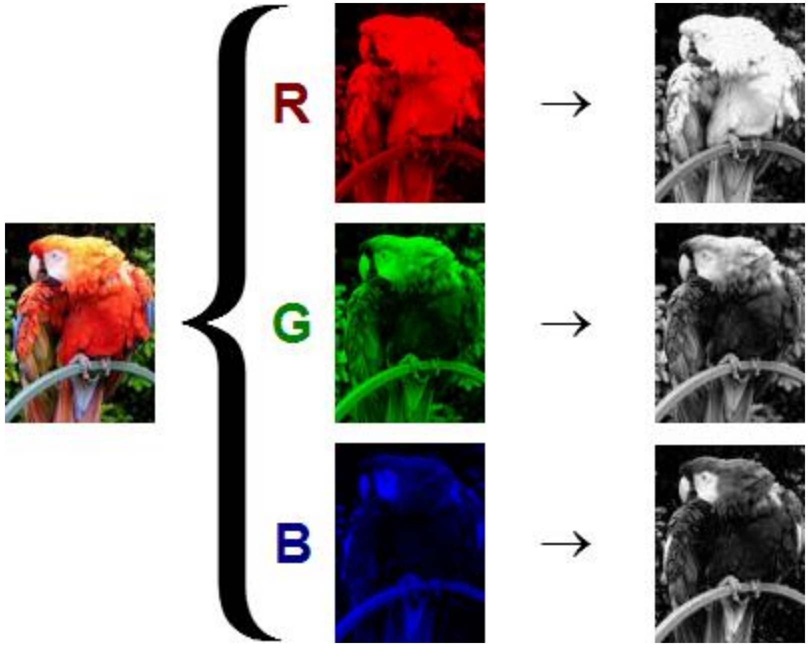
How Computers See Images

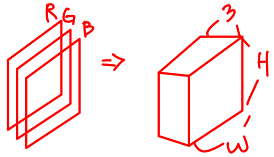
- Image is just an array of numbers.

2D tensor

```
[[  9   1  29  70 114  76   0   8   4   5   5   0 111 162   9   8  62  62]
 [  3   0  33  61 102 106  34   0   0   0   0  49 182 150   1  12  65  62]
 [  1   0  40  54 123  90  72  77  52  51  49 121 205  98   0  15  67  59]
 [  3   1  41  57  74  54  96 181 220 170  90 149 208  56   0  16  69  59]
 [  6   1  32  36  47  81  85  90 176 206 140 171 186  22   3  15  72  63]
 [  4   1  31  39  66  71  71  97 147 214 203 190 198  22   6  17  73  65]
 [  2   3  15  30  52  57  68 123 161 197 207 200 179   8   8  18  73  66]
 [  2   2  17  37  34  40  78 103 148 187 205 225 165   1   8  19  76  68]
 [  2   3  20  44  37  34  35  26  78 156 214 145 200  38   2  21  78  69]
 [  2   2  20  34  21  43  70  21  43 139 205  93 211  70   0  23  78  72]
 [  3   4  16  24  14  21 102 175 120 130 226 212 236  75   0  25  78  72]
 [  6   5  13  21  28  28  97 216 184  90 196 255 255  84   4  24  79  74]
 [  6   5  15  25  30  39  63 105 140  66 113 252 251  74   4  28  79  75]
 [  5   5  16  32  38  57  69  85  93 120 128 251 255 154  19  26  80  76]
 [  6   5  20  42  55  62  66  76  86 104 148 242 254 241  83  26  80  77]
 [  2   3  20  38  55  64  69  80  78 109 195 247 252 255 172  40  78  77]
 [ 10   8  23  34  44  64  88 104 119 173 234 247 253 254 227  66  74  74]
 [ 32   6  24  37  45  63  85 114 154 196 226 245 251 252 250 112  66  71]]
```

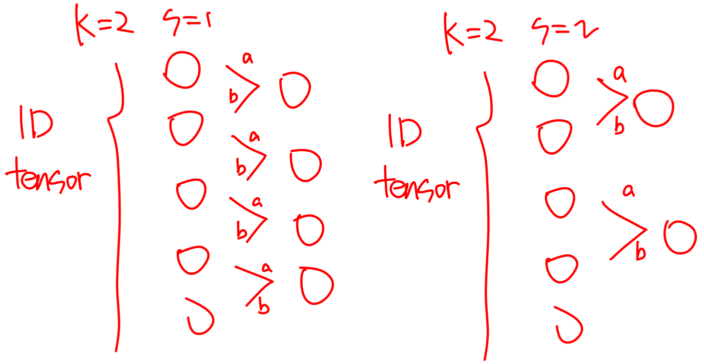
3 channels

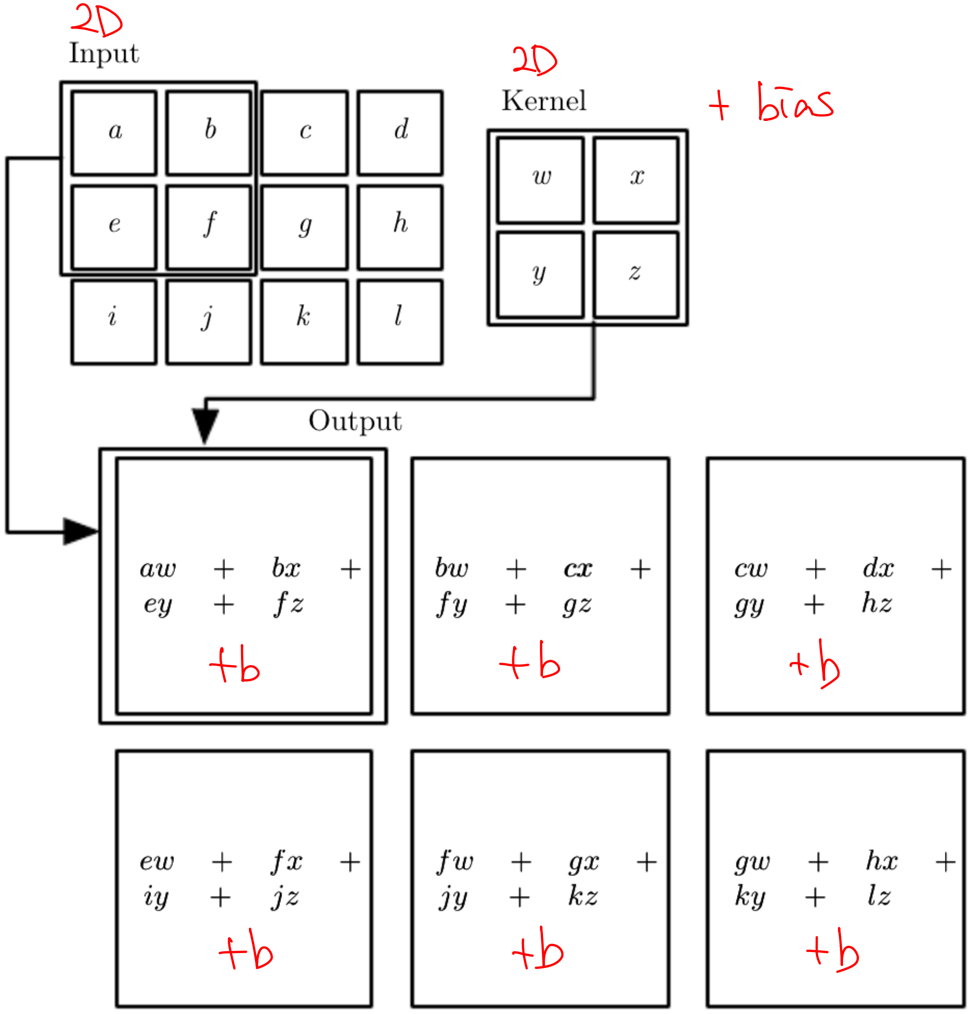




https://savan77.github.io/blog/how-computers-see-image.html

Convolution with 2D Tensor





kernel & bias

What are parameters? (w x y z b)

What are hyperparameters? kernel size, stride

\# output node

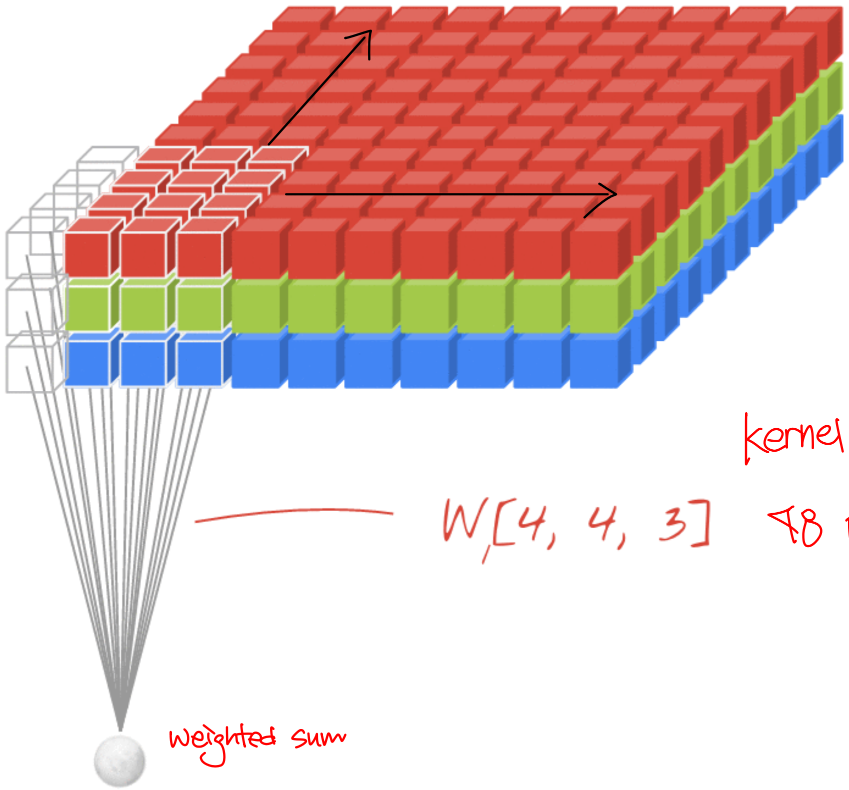
indirectly specifying!

https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

Sangheum Hwang

Convolution with 3D Tensor

(H,W) ⇒ df=2.



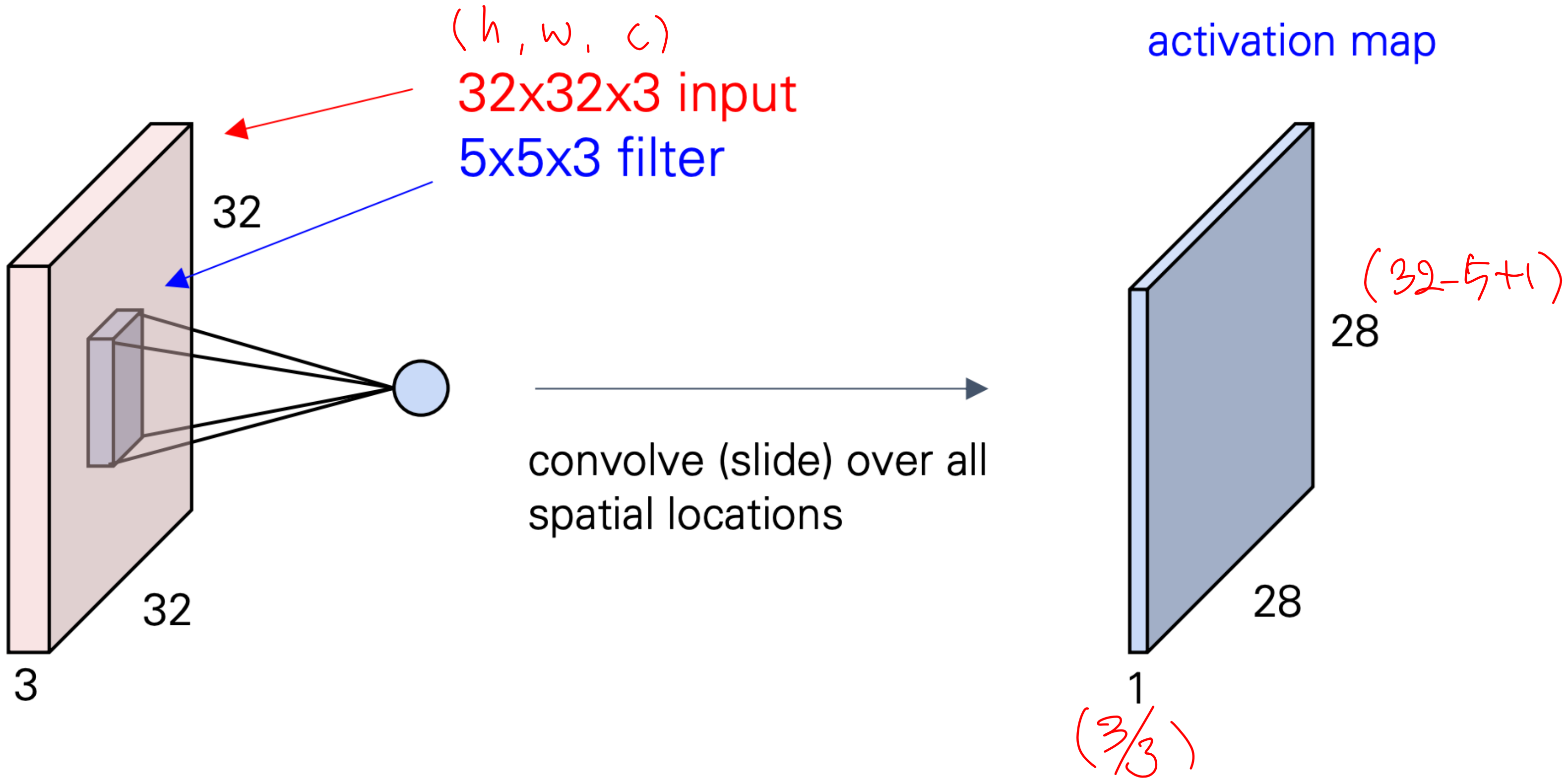
kernel ⇒ 3D tensor.

W[4, 4, 3] 48 param.

weighted sum

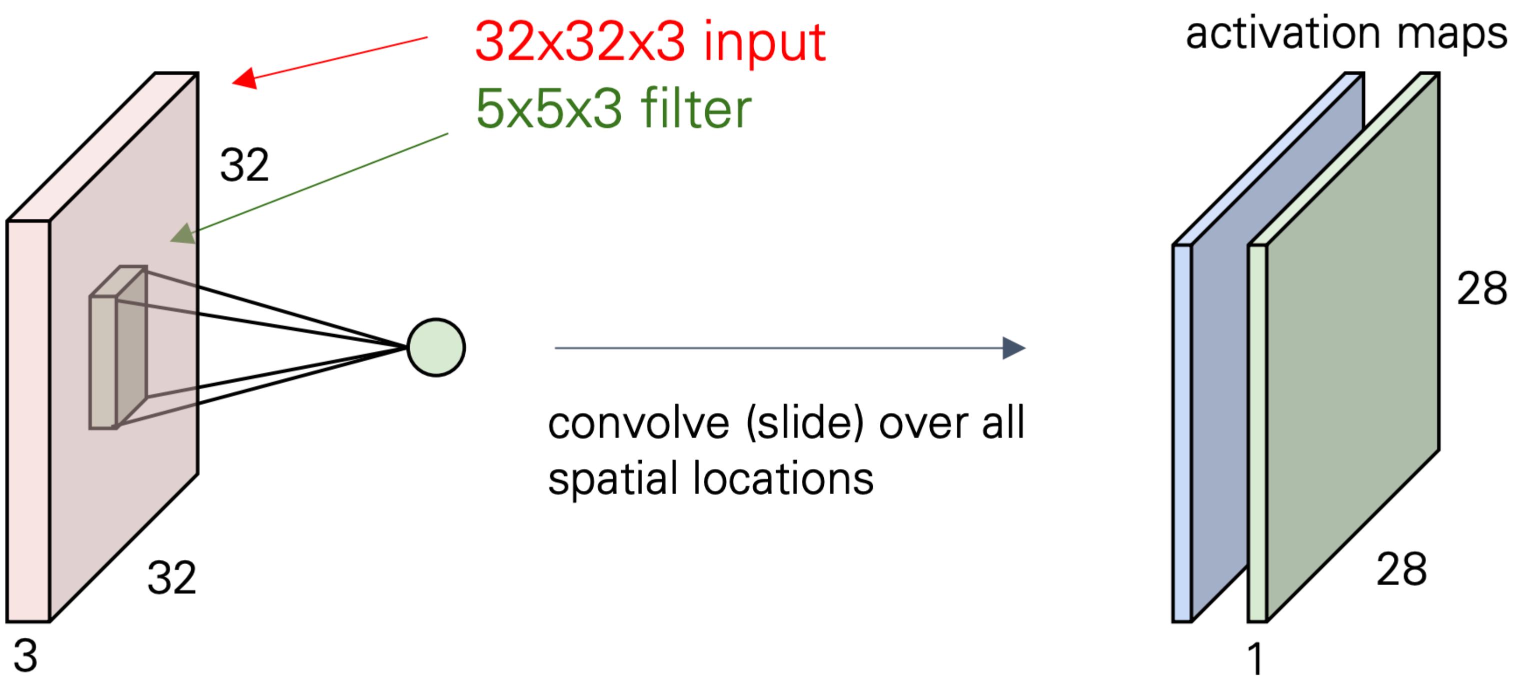
Convolution with 3D Tensor

- Generally, we use multiple kernels for single convolutional layer.



Convolution with 3D Tensor

- Generally, we use multiple kernels for single convolutional layer.

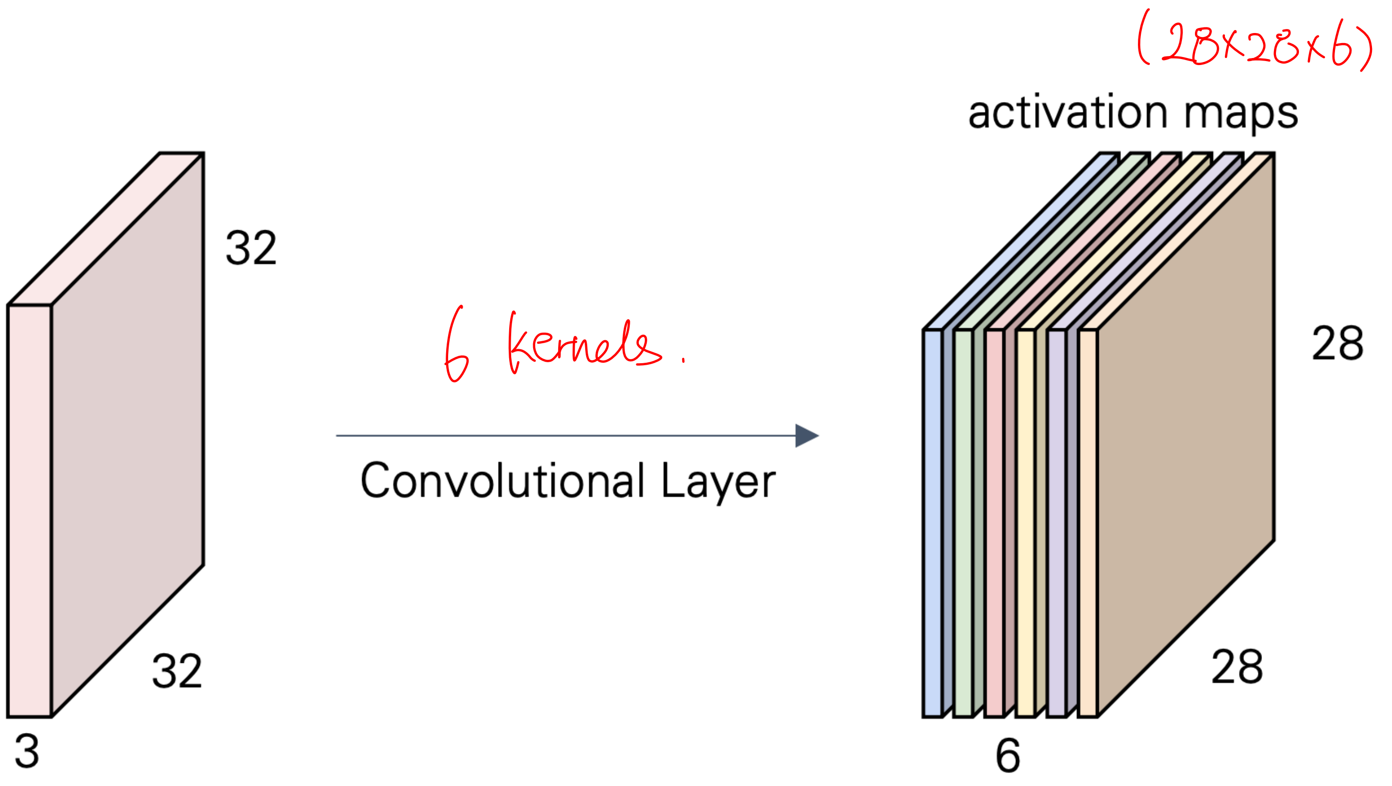


Sangheum Hwang

Convolution with 3D Tensor

kernel size + stride + # kernels

- Generally, we use multiple kernels for single convolutional layer.



Parameters in Convolution

- The **padding** specifies the size of a zeroed frame added around the input.

- The **stride** specifies a step size when moving the kernel across the signal.

- The **dilation** modulates the expansion of the filter without adding weights.

optional … skip

Parameters in Convolution: Padding

- There is border effects in convolution.
- Valid convolution

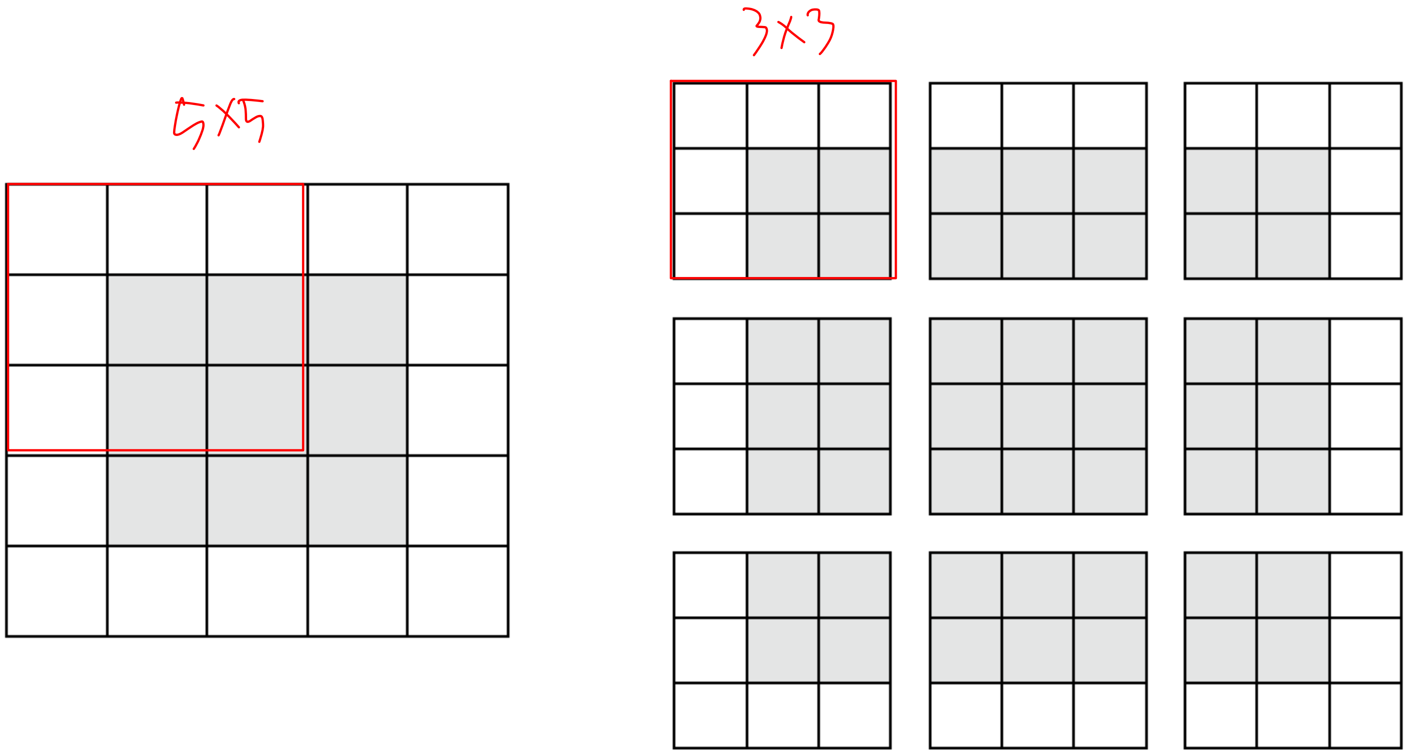
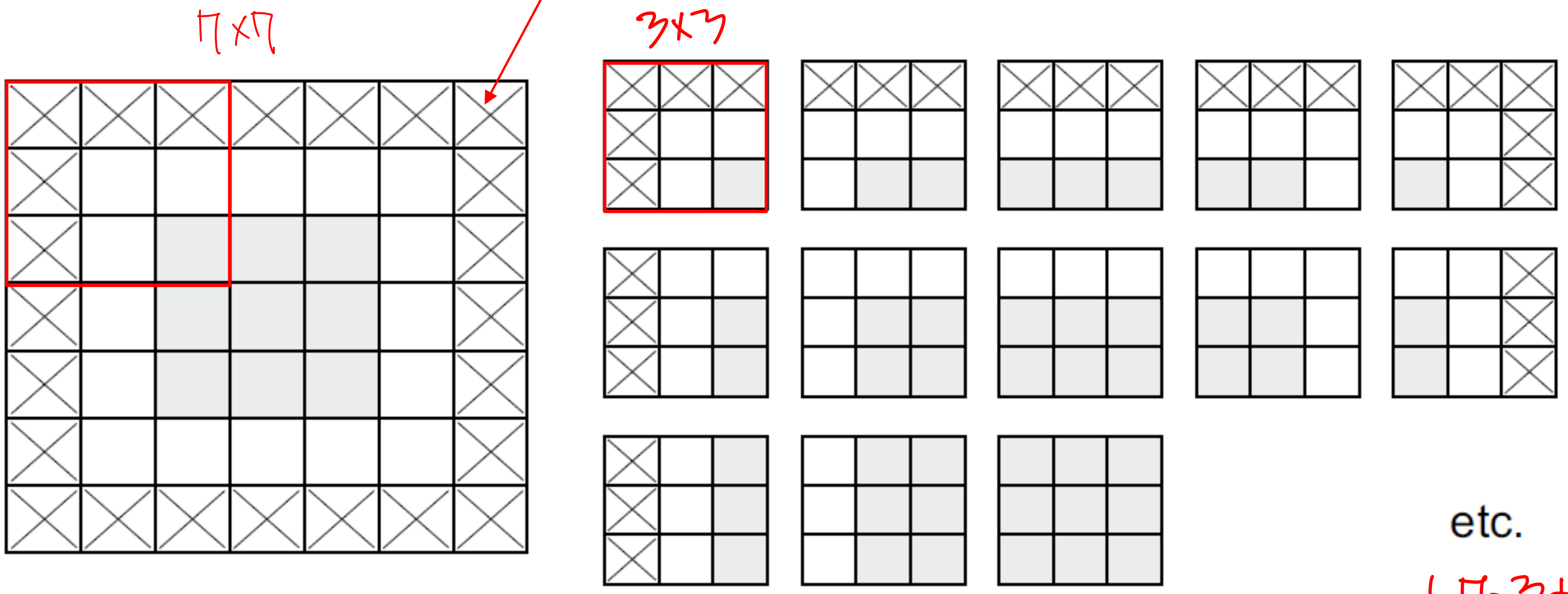


Figure 8.5 Valid locations of 3 × 3 patches in a 5 × 5 input feature map

Parameters in Convolution: Padding

- Padding to an input

Typically set to zero → Zero padding



$(\underline{7-3}+1)\times(\underline{7-3}+1)$
$= 25$ patches

Figure 8.6 Padding a 5 × 5 input in order to be able to extract 25 3 × 3 patches

Parameters in Convolution: Stride

- Strided convolution: convolutions with a stride higher than 1

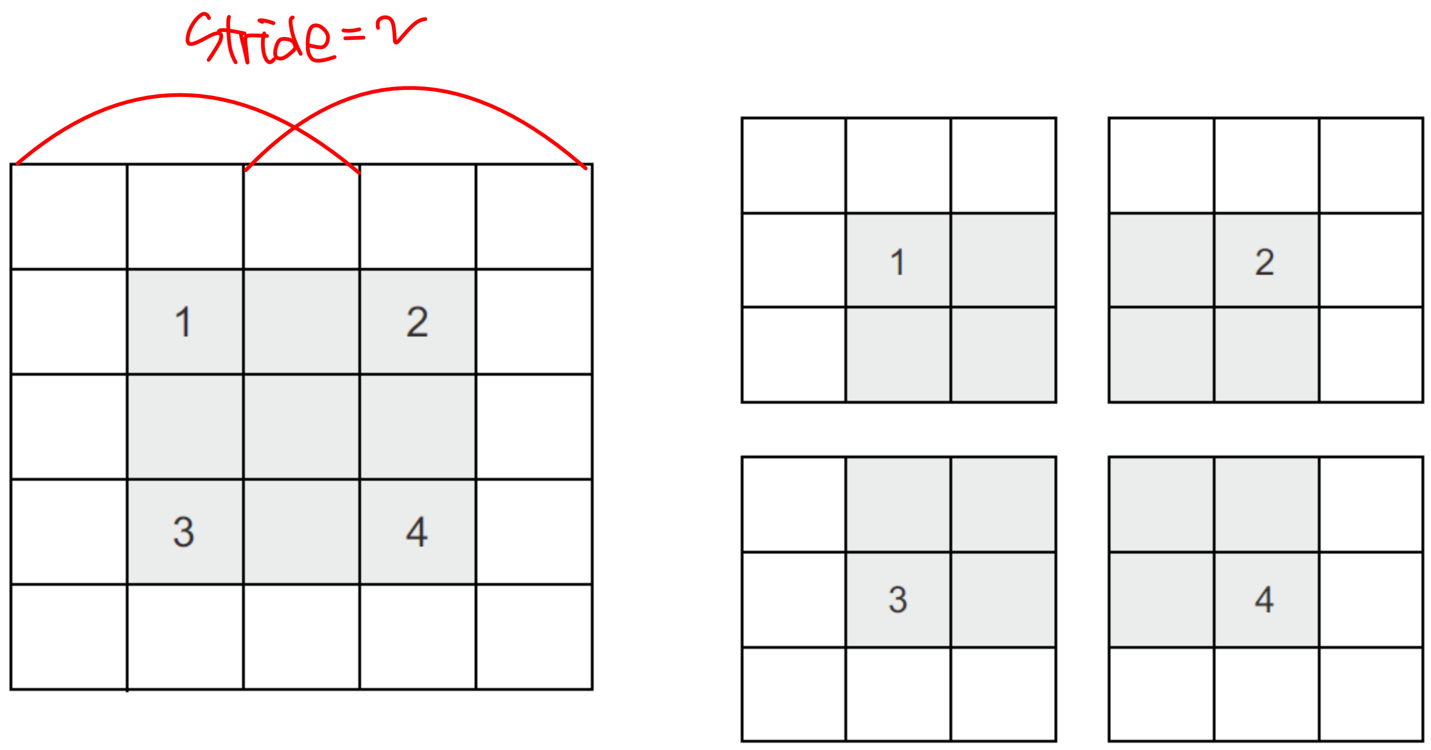
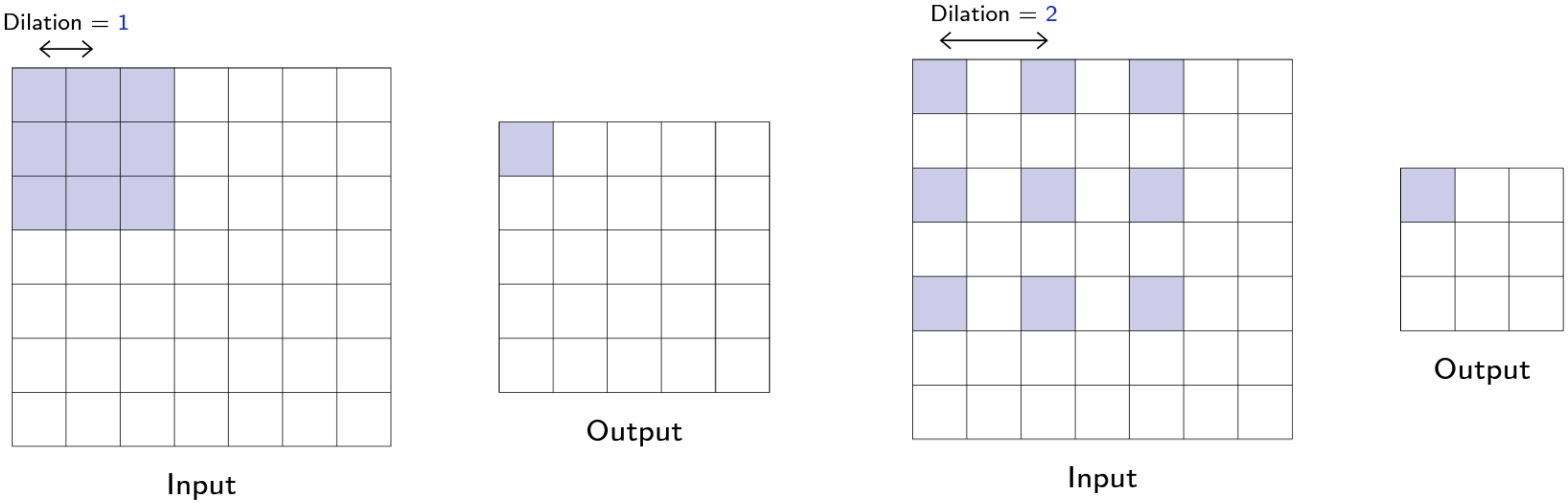


Figure 8.7 3 × 3 convolution patches with 2 × 2 strides

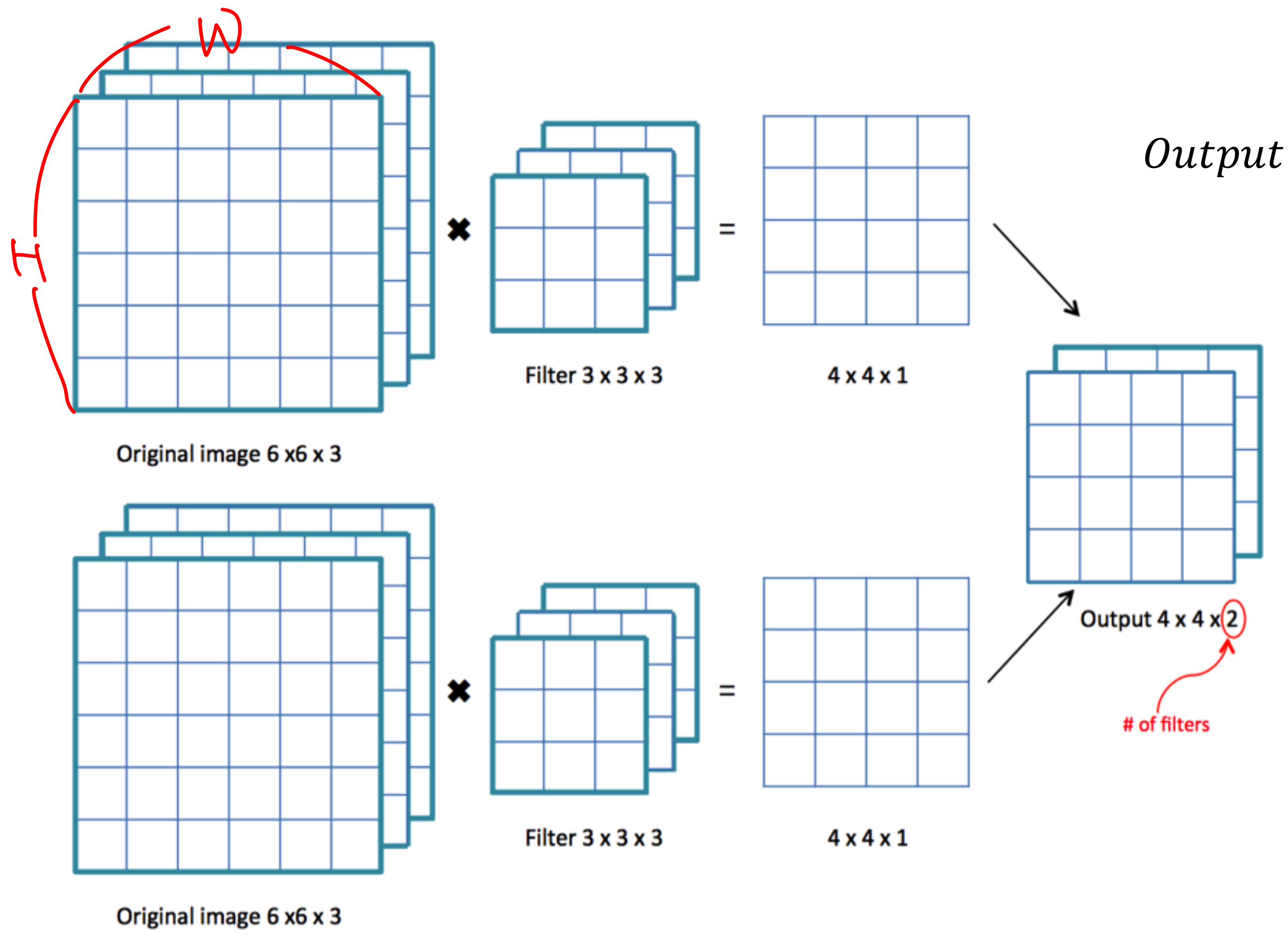
Parameters in Convolution: Dilation

skip

- The expansion of a filter by adding rows and columns of zeros between coefficient.



How to Compute Output Shape



padding filter size

$$Output\ Size = \left(\frac{H + 2P - F}{S} + 1\right) \times \left(\frac{W + 2P - F}{S} + 1\right)$$

stride

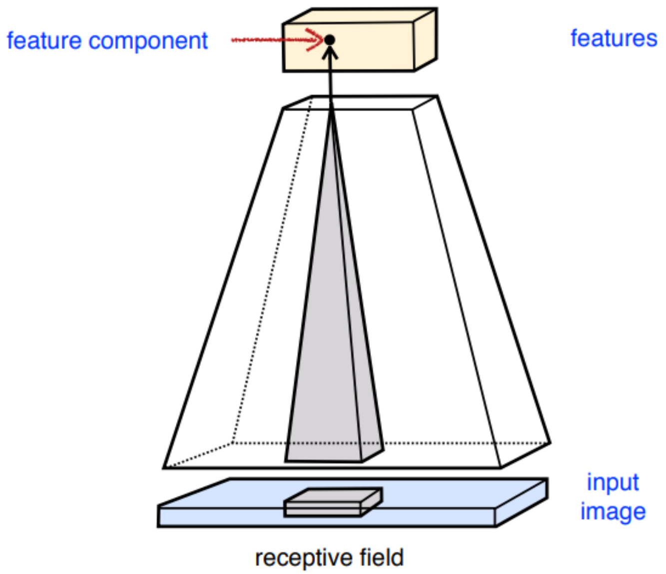
$$\left(\frac{6+0-3}{1}+1\right)\left(\frac{6+0-3}{1}+1\right)=16$$

Properties of Convolution

- A convolution preserves the signal support structure. ?
- Sparse interactions
  - Inputs and outputs are not fully connected but have local connectivity
- Parameter sharing
  - The same kernel is used repeatedly.
- Equivariance to transition
  - convolution(shift(input)) = shift(convolution(input))

?

- **Receptive field: Spatial locality**
  - Each element of the feature map process only for its receptive field (a local region of the input)
  - higher kernel size k → larger receptive field
  - Higher-level layers → larger receptive field

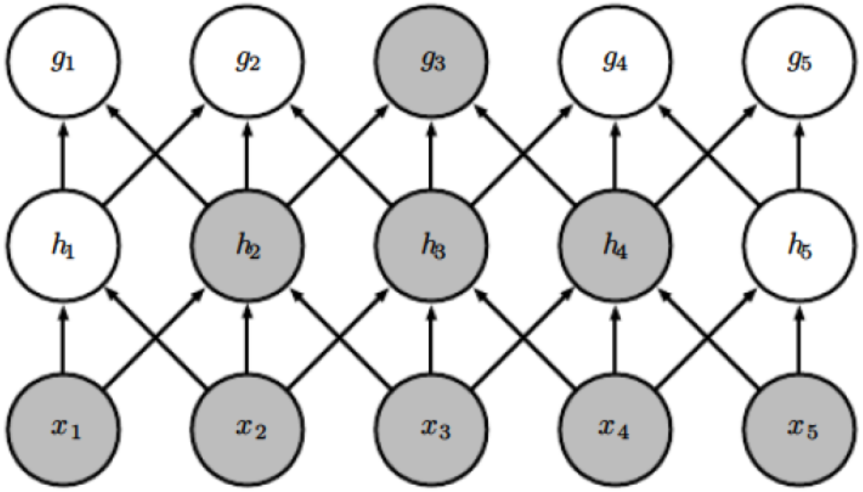


Example: a simple CNN with k=3

Receptive field of h_3 in the input layer = {x2,x3,x4}

Receptive field of g_3 in the input layer = {x1,x2,x3,x4,x5}

input



Example: a simple FNN (fully connected)

Receptive field of a hidden unit? $\{x_1, x_2, x_3, x_4, x_5\}$

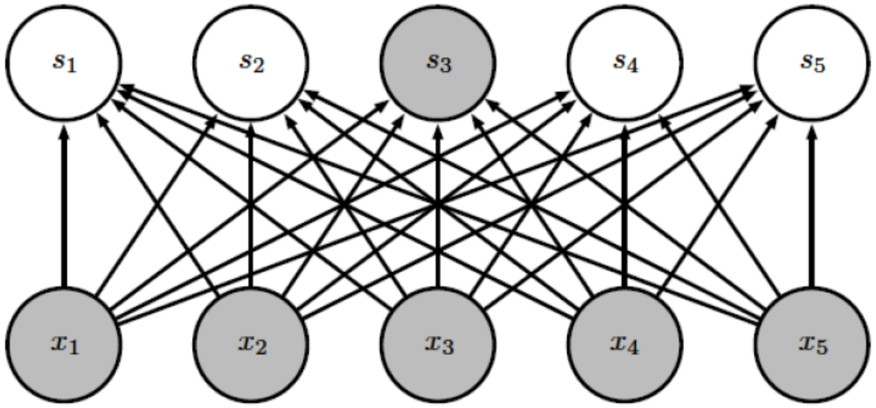
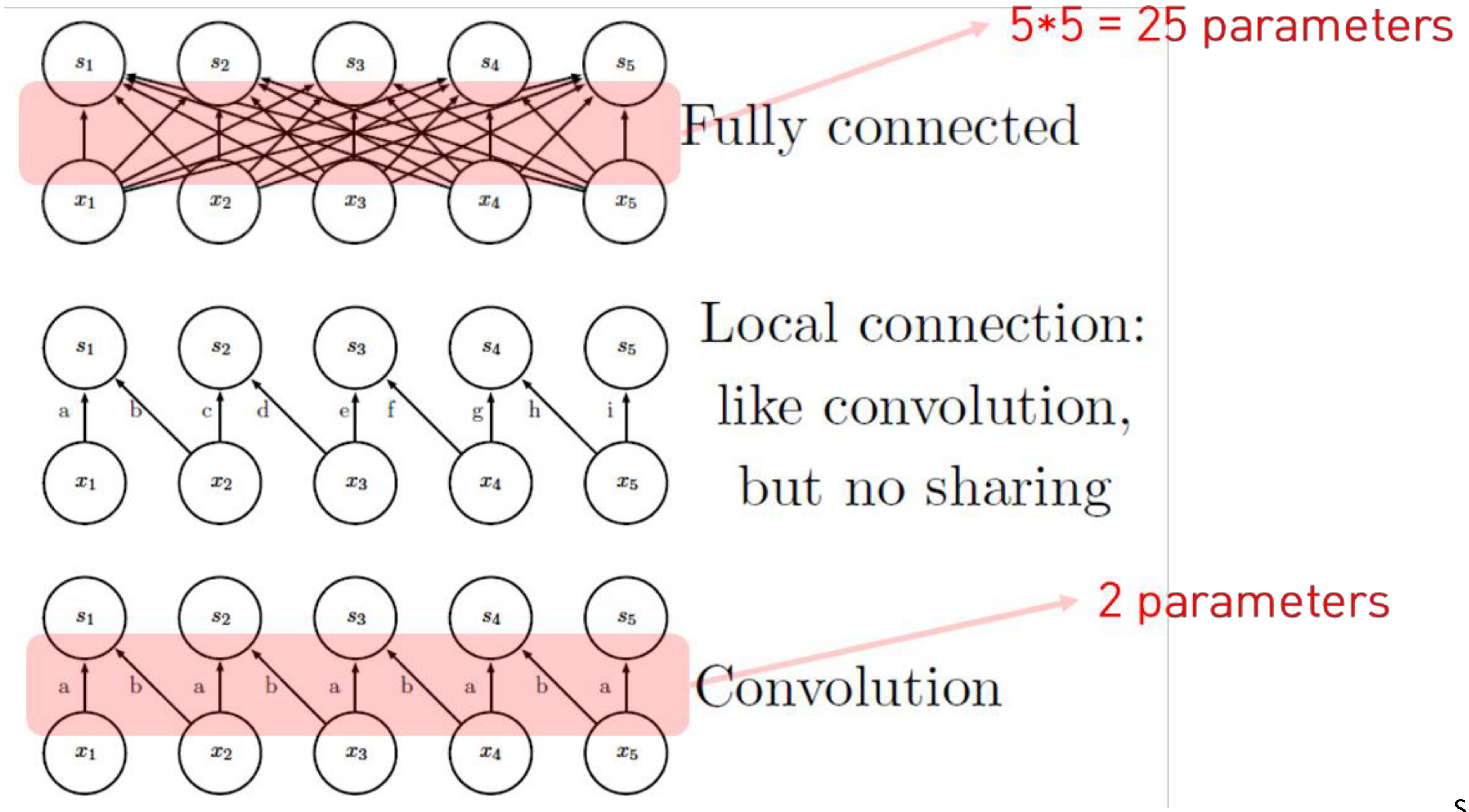


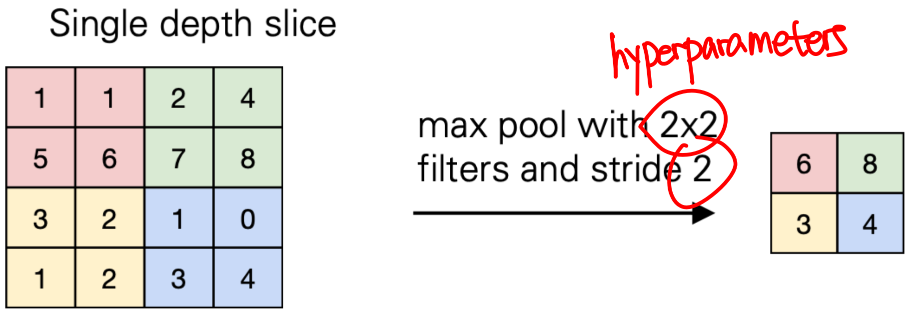
Figure credit: SKKU Prof. Kang

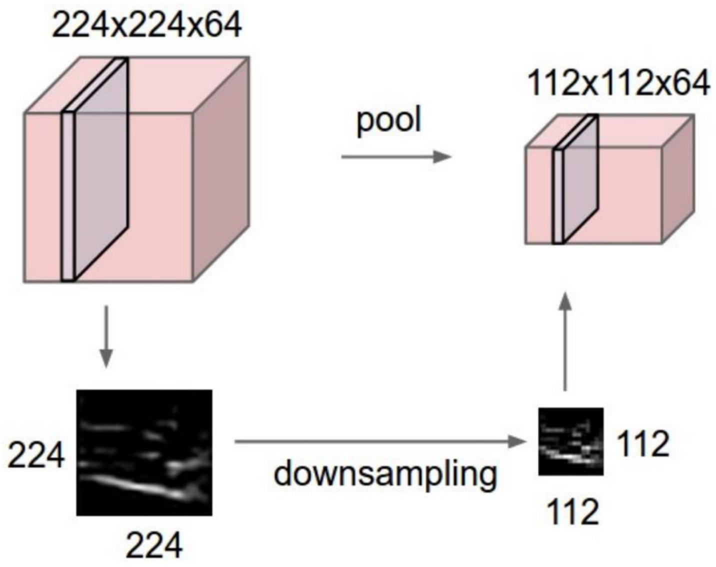
Properties of Convolution



Pooling

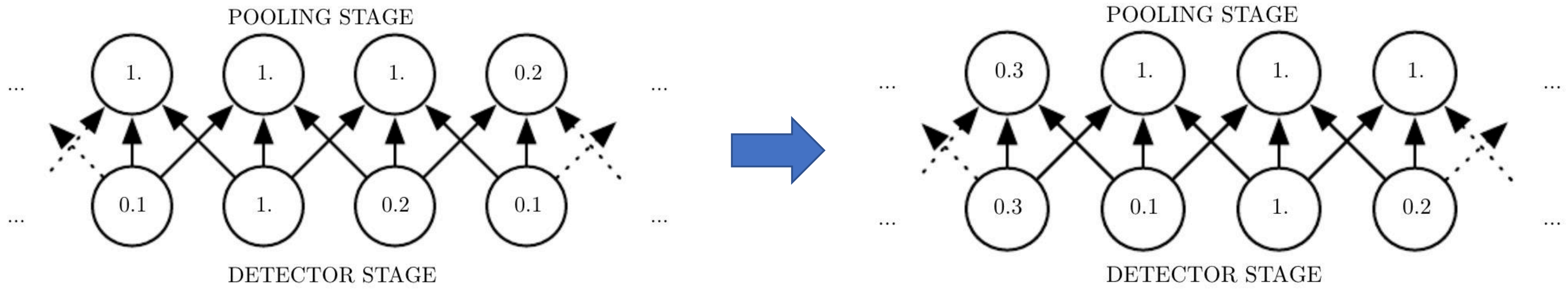
- Pooling function
  - replaces the output of the layer at a certain location with a summary statistic of the nearby outputs.
  - makes the representation smaller and more manageable.
  - operates over each activation map independently. ?
  - Max pooling, average pooling, etc.





Pooling

- Invariant to small translations of the input
  - It means that if we translate the input by a small amount, the values of most of the pooled outputs do not change.



Reading assignments

- “Dive into deep learning”
  - Chapter 7
- “Understanding deep learning”
  - Chapter 10

perceptron.

backprop

grad computation

overfit / regularization

techniques dropout

MLP ⇒ compute gradient

"chain rule" terms

vanishing gradient

← sigmoid activation

↳ drawback

swap

relu

why $z_j \to 1$? $\frac{\partial L}{\partial w_{ji}} \to 0$?

Reading assignments

compute
trainable params

modern CNN (VGGNet ResNet)
architecture

- "Dive into deep learning"
  - Chapter 7
- "Understanding deep learning"
  - Chapter 10