# 11210IPT553000 Deep Learning in Biomedical Optical Imaging

## Week 7

## Convolutional Neural Network

### Foundations of Convolutional Neural Networks

Instructor: Hung-Wen Chen
2023/10/23 @NTHU, Fall 2023

coursera deeplearning.ai

SPONSORED BY NVIDIA DEEP LEARNING INSTITUTE

NVIDIA DEEP LEARNING INSTITUTE CERTIFIED INSTRUCTOR

- Computer Vision
- Foundations of Convolutional Neural Networks (Course 4 Week 1)
- Lab Practice: Transfer Learning

# Computer Vision

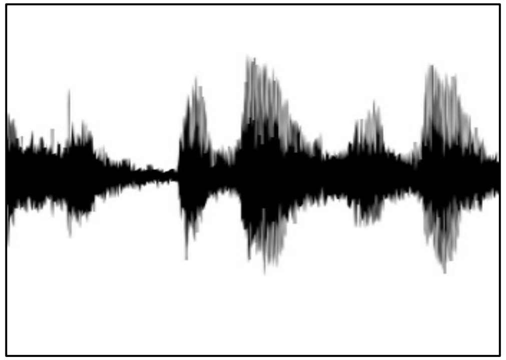
# Supervised Learning

## Structured Data

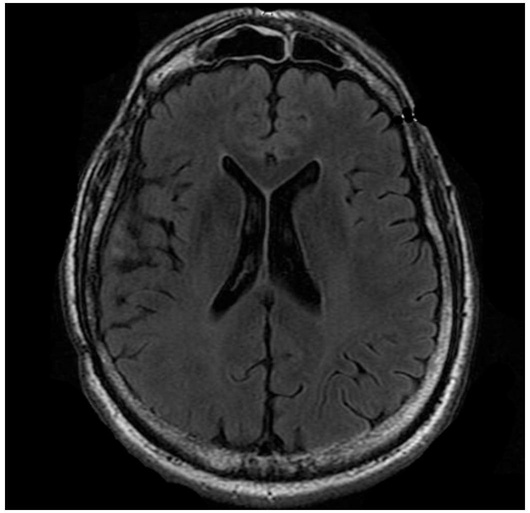
| Age | Weight | ⋯ | Gender |
|---|---|---|---|
| 50 | 72 | | Male |
| 33 | 82 | | Female |
| 18 | 66 | | Female |
| ⋮ | ⋮ | | ⋮ |
| 80 | 55 | | Male |

| User Age | Ad Id | ⋯ | Click |
|---|---|---|---|
| 41 | 93242 | | 1 |
| 80 | 93287 | | 0 |
| 18 | 87312 | | 1 |
| ⋮ | ⋮ | | ⋮ |
| 27 | 71244 | | 1 |

## Unstructured Data



Audio



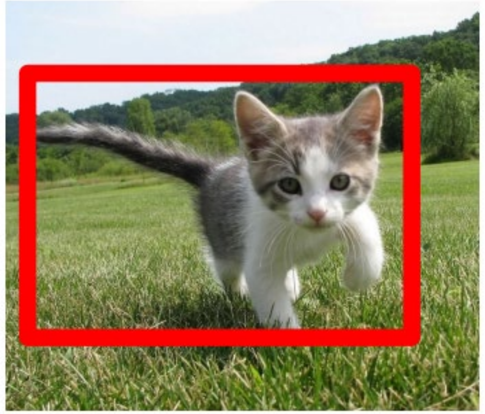
Four scores and seven years ago…

Text

## Tasks

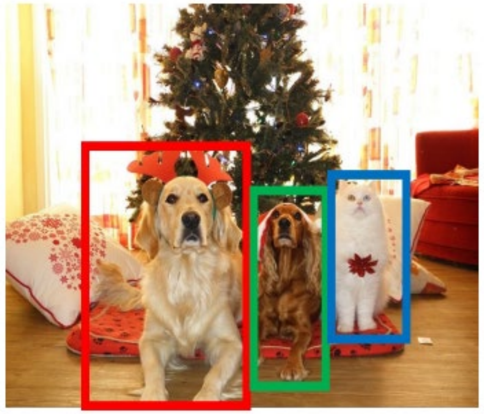
**Classification**

**Classification + Localization**



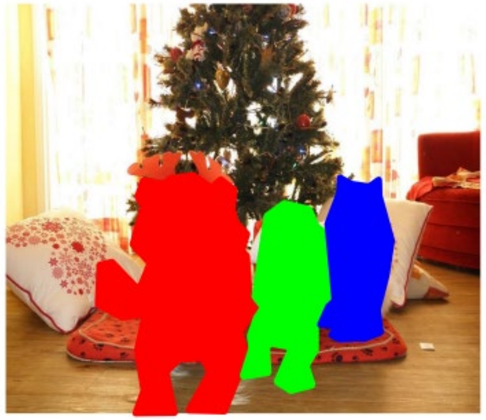
CAT

**Object Detection**



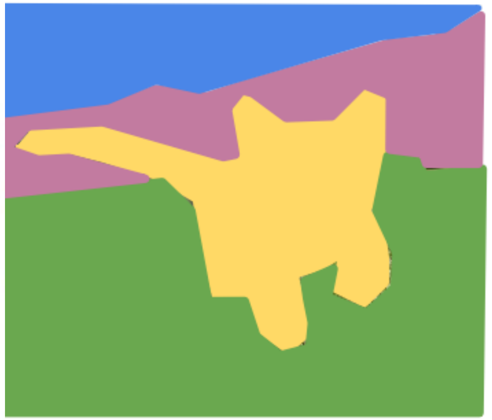
DOG, DOG, CAT

**Instance Segmentation**



DOG, DOG, CAT

**Semantic Segmentation**



GRASS, CAT, TREE, SKY

Single Object

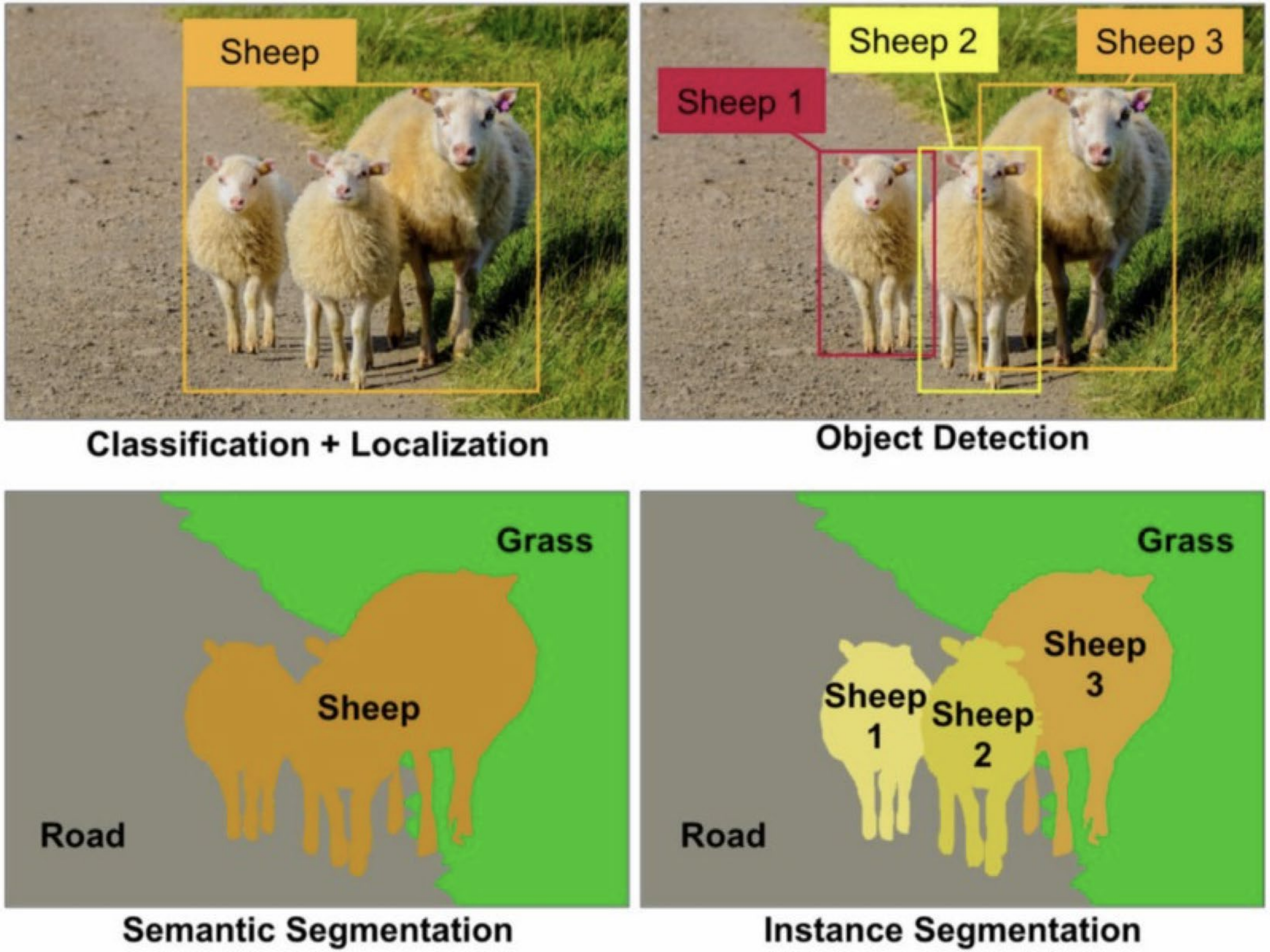
Multiple Object



No objects, just pixels

Modified from Stanford CS231n

## Instance segmentation vs Semantic segmentation



# Computer Vision

## Deep Learning on large images

→ Cat? (0/1)

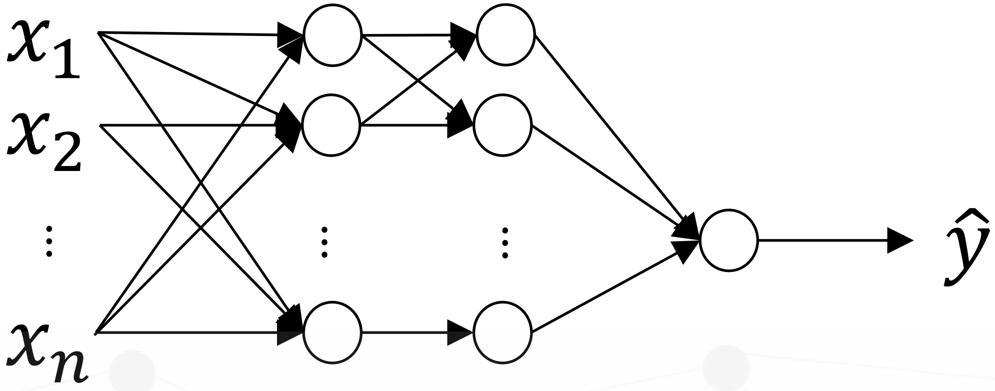
64x64 x3

12288

3 **billion**!!!

$w^{[1]}$ (1000 , 3M)

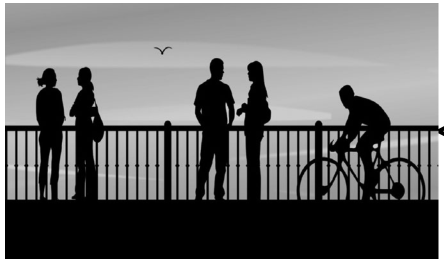
1000 x 1000 x 3
= 3 million



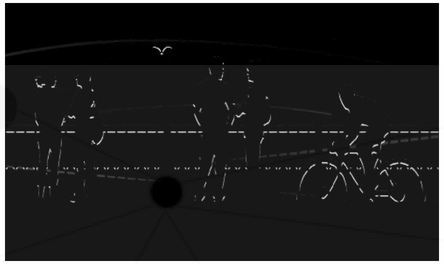

3M 1000

# Computer Vision

## Computer vision problem



vertical edges

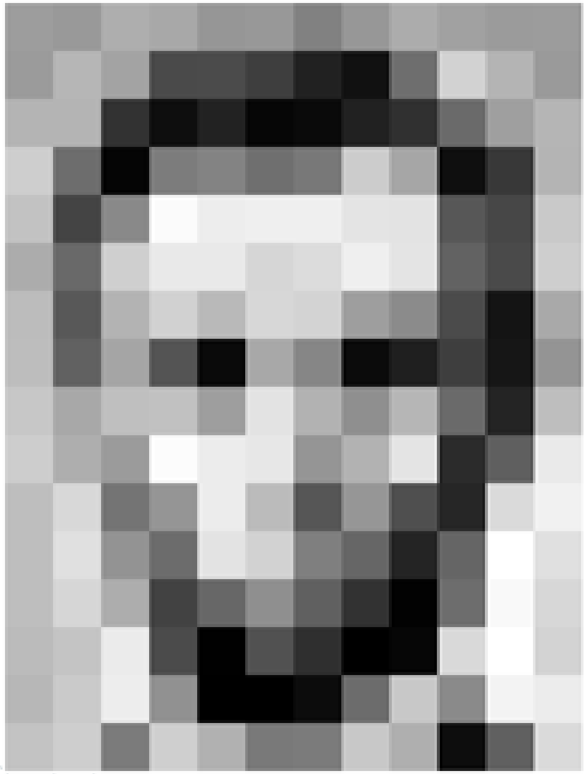


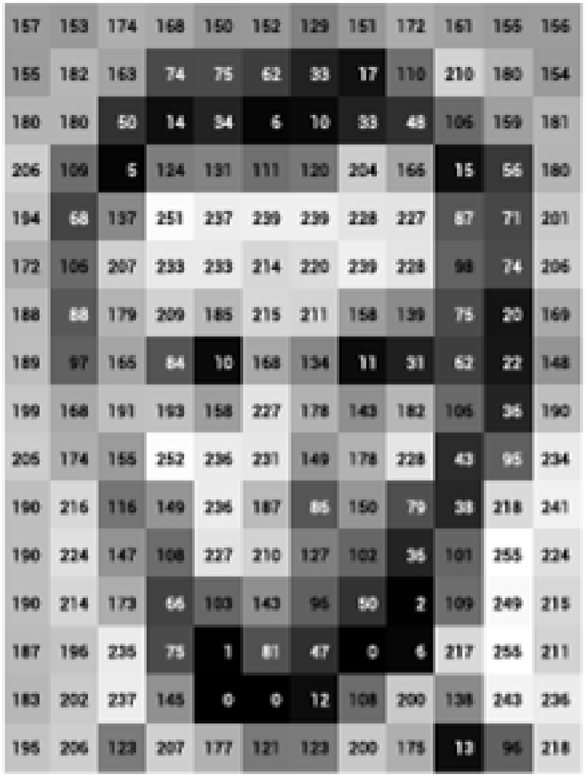
horizontal edges

# Image Kernels (Filters)

The representation of an image

# 1 channel: Black -> 0, White -> 255 // 2 dimensions





| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 157 | 153 | 174 | 168 | 150 | 152 | 129 | 151 | 172 | 161 | 155 | 156 |
| 155 | 182 | 163 | 74 | 75 | 62 | 33 | 17 | 110 | 210 | 180 | 154 |
| 180 | 180 | 50 | 14 | 34 | 6 | 10 | 33 | 48 | 106 | 159 | 181 |
| 206 | 109 | 5 | 124 | 131 | 111 | 120 | 204 | 166 | 15 | 56 | 180 |
| 194 | 68 | 137 | 251 | 237 | 239 | 239 | 228 | 227 | 87 | 71 | 201 |
| 172 | 105 | 207 | 233 | 233 | 214 | 220 | 239 | 228 | 98 | 74 | 206 |
| 188 | 88 | 179 | 209 | 185 | 215 | 211 | 158 | 139 | 75 | 20 | 169 |
| 189 | 97 | 165 | 84 | 10 | 168 | 134 | 11 | 31 | 62 | 22 | 148 |
| 199 | 168 | 191 | 193 | 158 | 227 | 178 | 143 | 182 | 106 | 36 | 190 |
| 205 | 174 | 155 | 252 | 236 | 231 | 149 | 178 | 228 | 43 | 95 | 234 |
| 190 | 216 | 116 | 149 | 236 | 187 | 86 | 150 | 79 | 38 | 218 | 241 |
| 190 | 224 | 147 | 108 | 227 | 210 | 127 | 102 | 36 | 101 | 255 | 224 |
| 190 | 214 | 173 | 66 | 103 | 143 | 96 | 50 | 2 | 109 | 249 | 215 |
| 187 | 196 | 235 | 75 | 1 | 81 | 47 | 0 | 6 | 217 | 255 | 211 |
| 183 | 202 | 237 | 145 | 0 | 0 | 12 | 108 | 200 | 138 | 243 | 236 |
| 195 | 206 | 123 | 207 | 177 | 121 | 123 | 200 | 175 | 13 | 96 | 218 |

12X16

## Vertical edge detection

3X1+1X1+1X2+0X0+0X5+0X7+1X-1+8X-1+2X-1

| | | | | | |
|---|---|---|---|---|---|
| 3 | 0 | 1 | 2 | 7 | 4 |
| 1 | 5 | 8 | 9 | 3 | 1 |
| 2 | 7 | 2 | 5 | 1 | 3 |
| 0 | 1 | 3 | 1 | 7 | 8 |
| 4 | 2 | 1 | 6 | 2 | 8 |
| 2 | 4 | 5 | 2 | 3 | 9 |

6 X 6

*

Filter or Kernel

| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

3 X 3

=

| | | | |
|---|---|---|---|
| -5 | -4 | 0 | 8 |
| -10 | -2 | 2 | 3 |
| 0 | -2 | -4 | -7 |
| -3 | -2 | -3 | -16 |

4 X 4

Convolution

## Kernels and Convolution

**Blur Kernel**

| | | |
|---|---|---|
| .06 | .13 | .06 |
| .13 | .25 | .13 |
| .06 | .13 | .06 |

*

**Original Image**

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |

=

**Convolved Image**

| | | | |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

# Kernels and Convolution

## Blur Kernel

| | | |
|---|---|---|
| .06 | .13 | .06 |
| .13 | .25 | .13 |
| .06 | .13 | .06 |

*

## Original Image

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |

=

## Convolved Image

Kernels and Convolution

## Kernels and Convolution

## Kernels and Convolution

**Blur Kernel**

| | | |
|---|---|---|
| .06 | .13 | .06 |
| .13 | .25 | .13 |
| .06 | .13 | .06 |

*

**Original Image**

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | .13 | .06 | 0 | 1 |
| 0 | .13 | 0 | 0 | 1 | 0 |
| 0 | .06 | .13 | .06 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |

=

**Convolved Image**

| | | | |
|---|---|---|---|
| .56 | $\boxed{.57}$ | | |
| | | | |
| | | | |
| | | | |

# Kernels and Convolution

## Blur Kernel

| | | |
|---|---|---|
| .06 | .13 | .06 |
| .13 | .25 | .13 |
| .06 | .13 | .06 |

*

## Original Image

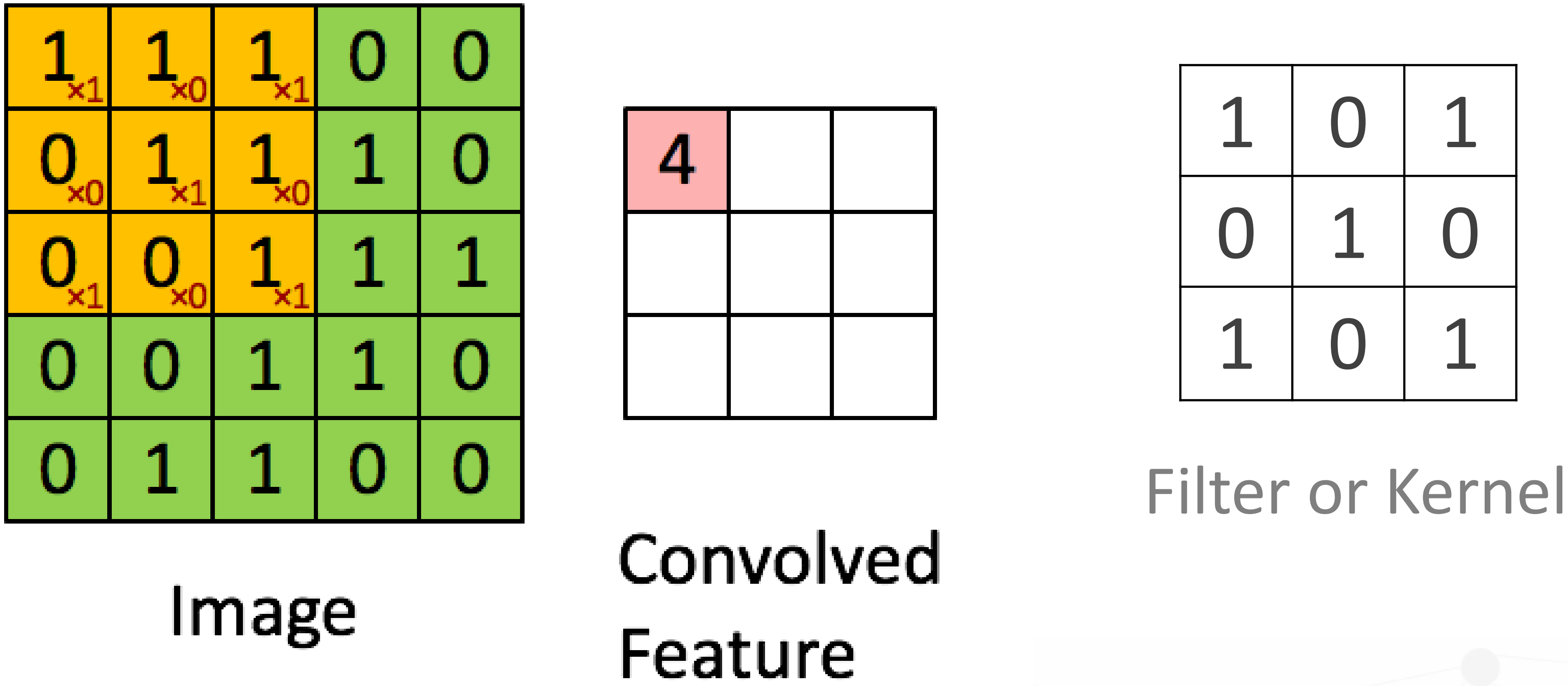
| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |

=

## Convolved Image

| | | | |
|---|---|---|---|
| .56 | .57 | .57 | .56 |
| .7 | .82 | .82 | .7 |
| .69 | .95 | .95 | .69 |
| .64 | .69 | .69 | .64 |

# Animation



## Vertical edge detection

| | | | | | |
|---|---|---|---|---|---|
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |

6 X 6

*

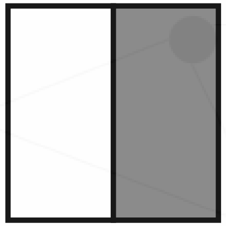
| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

3 X 3

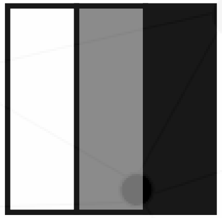
=

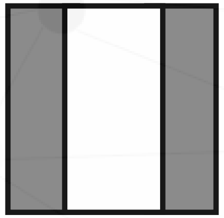
| | | | |
|---|---|---|---|
| 0 | 30 | 30 | 0 |
| 0 | 30 | 30 | 0 |
| 0 | 30 | 30 | 0 |
| 0 | 30 | 30 | 0 |

4 X 4



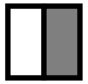
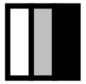
*



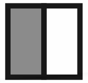
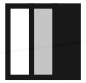


## Vertical edge detection examples

$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix}$$


$$\begin{bmatrix} 0 & 0 & 0 & 10 & 10 & 10 \\ 0 & 0 & 0 & 10 & 10 & 10 \\ 0 & 0 & 0 & 10 & 10 & 10 \\ 0 & 0 & 0 & 10 & 10 & 10 \\ 0 & 0 & 0 & 10 & 10 & 10 \\ 0 & 0 & 0 & 10 & 10 & 10 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & -30 & -30 & 0 \\ 0 & -30 & -30 & 0 \\ 0 & -30 & -30 & 0 \\ 0 & -30 & -30 & 0 \end{bmatrix}$$


## Vertical and horizontal edge detection

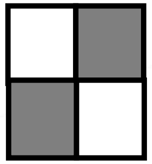
| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

Vertical

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| -1 | -1 | -1 |

Horizontal

| | | | | | |
|---|---|---|---|---|---|
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 10 | 10 | 10 | 0 | 0 | 0 |
| 0 | 0 | 0 | 10 | 10 | 10 |
| 0 | 0 | 0 | 10 | 10 | 10 |
| 0 | 0 | 0 | 10 | 10 | 10 |



*

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| -1 | -1 | -1 |

=

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 30 | 10 | -10 | -30 |
| 30 | 10 | -10 | -30 |
| 0 | 0 | 0 | 0 |

## Learning to detect edges

| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

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

Sobel Filter

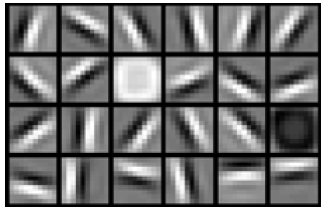
| | | |
|---|---|---|
| 3 | 0 | -3 |
| 10 | 0 | -10 |
| 3 | 0 | -3 |

Scharr Filter

| | | | | | |
|---|---|---|---|---|---|
| 3 | 0 | 1 | 2 | 7 | 4 |
| 1 | 5 | 8 | 9 | 3 | 1 |
| 2 | 7 | 2 | 5 | 1 | 3 |
| 0 | 1 | 3 | 1 | 7 | 8 |
| 4 | 2 | 1 | 6 | 2 | 8 |
| 2 | 4 | 5 | 2 | 3 | 9 |

*

| | | |
|---|---|---|
| $w_1$ | $w_2$ | $w_3$ |
| $w_4$ | $w_5$ | $w_6$ |
| $w_7$ | $w_8$ | $w_9$ |



=

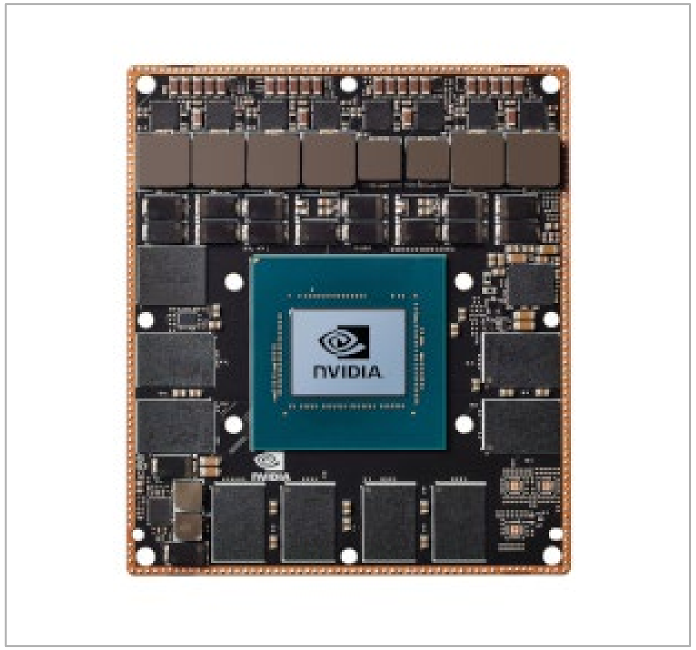
45 degree
60 degree
90 degree
……

| | | | |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

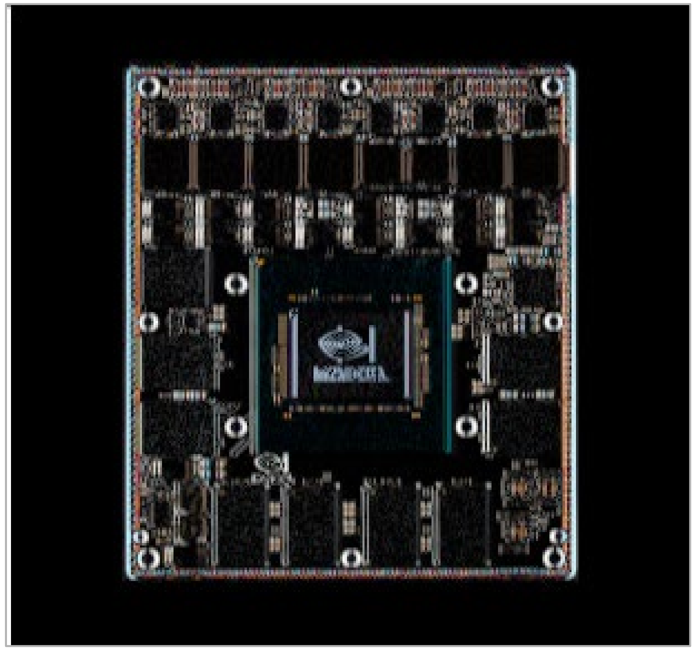
# Computer Vision

## Edge detection example

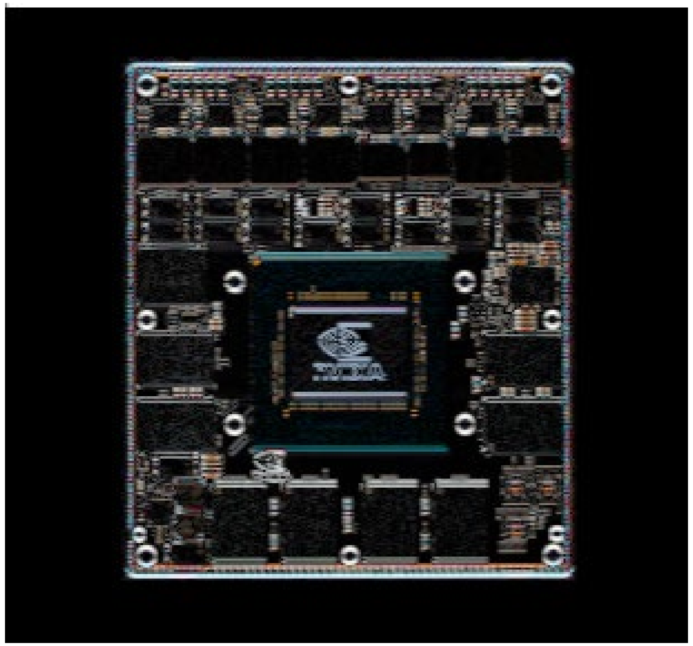
| Original Image | Vertical Edges | Horizontal Edges |
|---|---|---|




| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |



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



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

# Computer Vision

## Kernels and Convolution

Original Image

Blur

Sharpen

Brighten

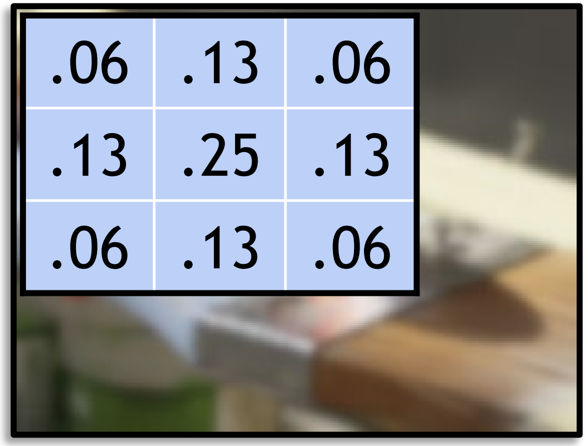
Darken

# Computer Vision

## Kernels and Convolution

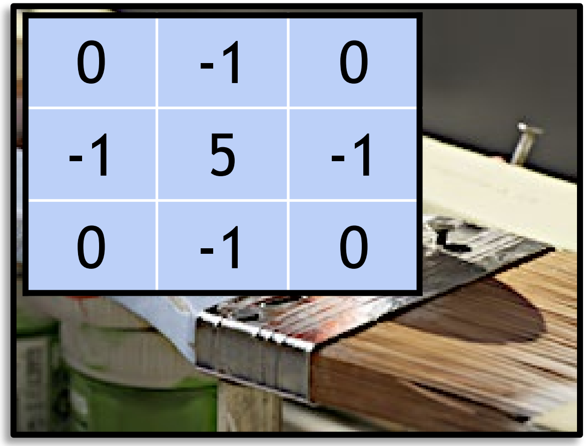
Original Image

Blur



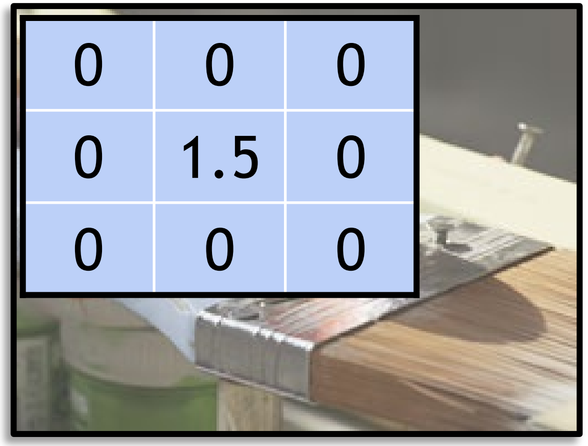
| | | |
|---|---|---|
| .06 | .13 | .06 |
| .13 | .25 | .13 |
| .06 | .13 | .06 |

Sharpen



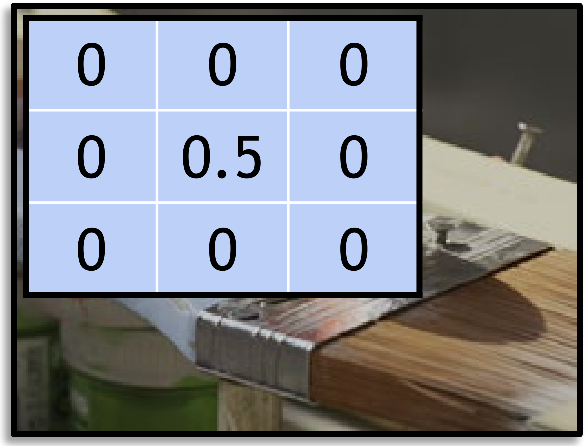
| | | |
|---|---|---|
| 0 | -1 | 0 |
| -1 | 5 | -1 |
| 0 | -1 | 0 |

Brighten



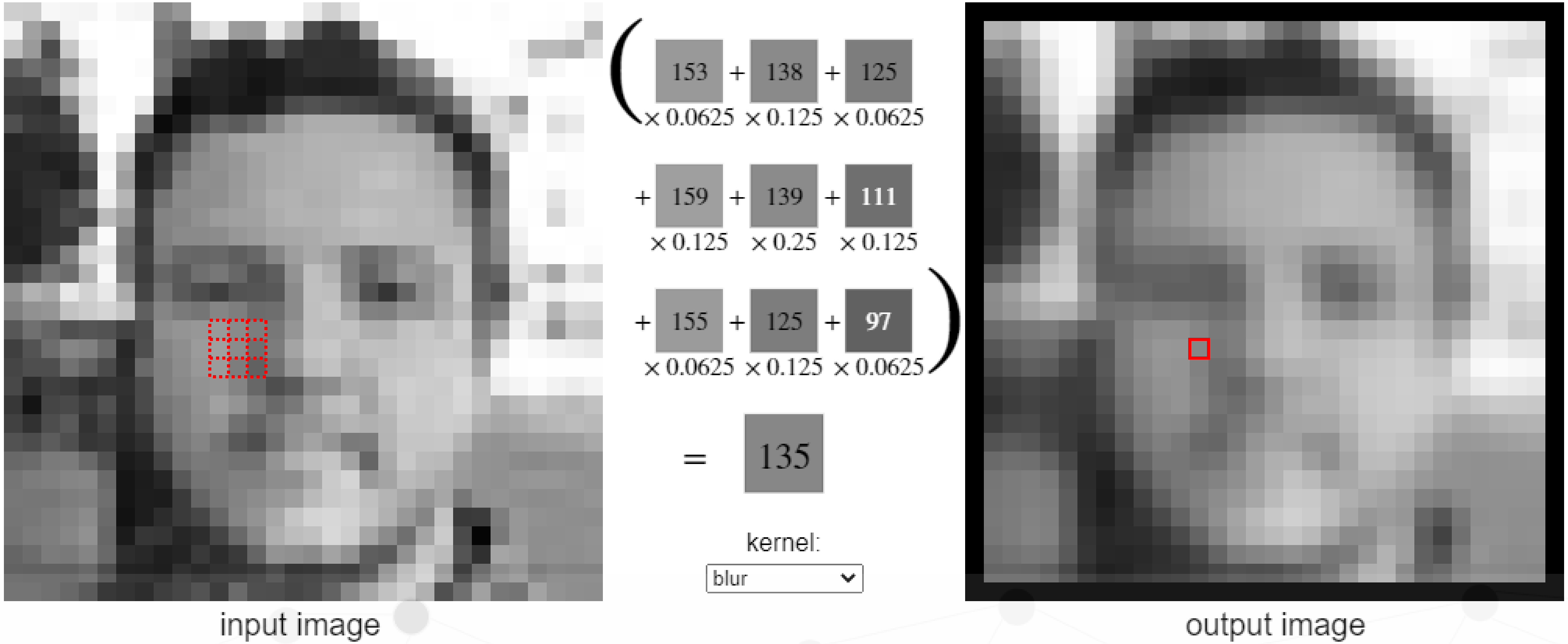
| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1.5 | 0 |
| 0 | 0 | 0 |

Darken



| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0.5 | 0 |
| 0 | 0 | 0 |

# Image kernels (filters) explained visually




https://setosa.io/ev/image-kernels/

# Computer Vision

## Image kernels (filters) explained visually

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

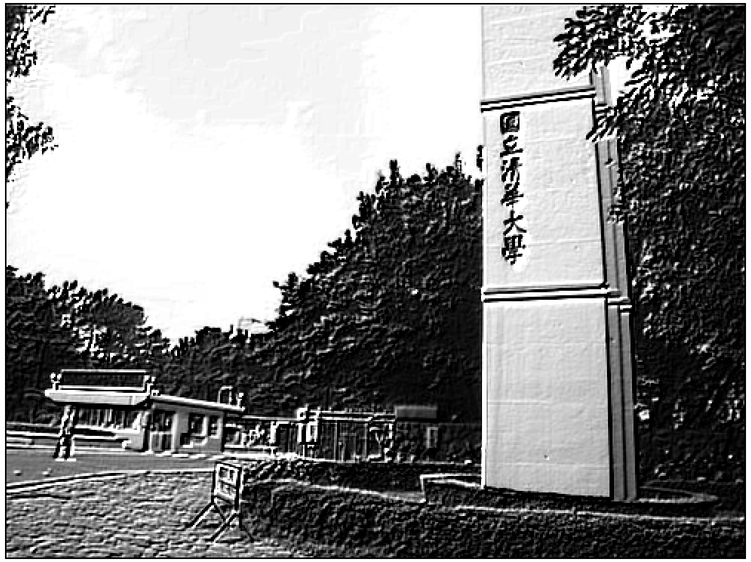
identity


| | | |
|---|---|---|
| -1 | -1 | -1 |
| -1 | 8 | -1 |
| -1 | -1 | -1 |

outline

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

emboss

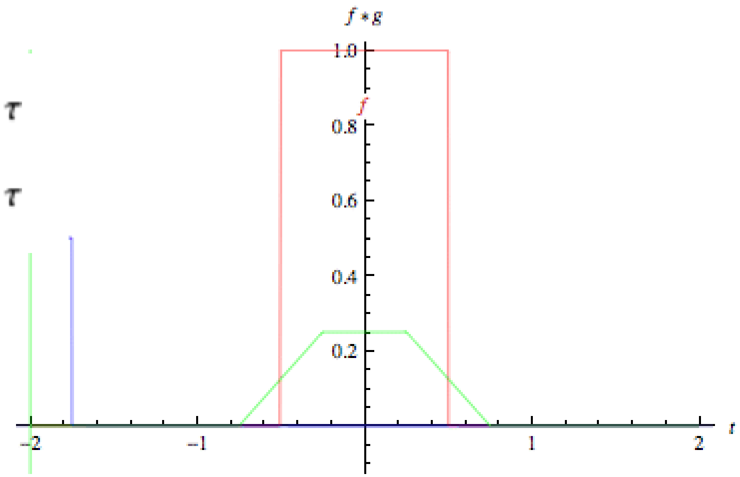


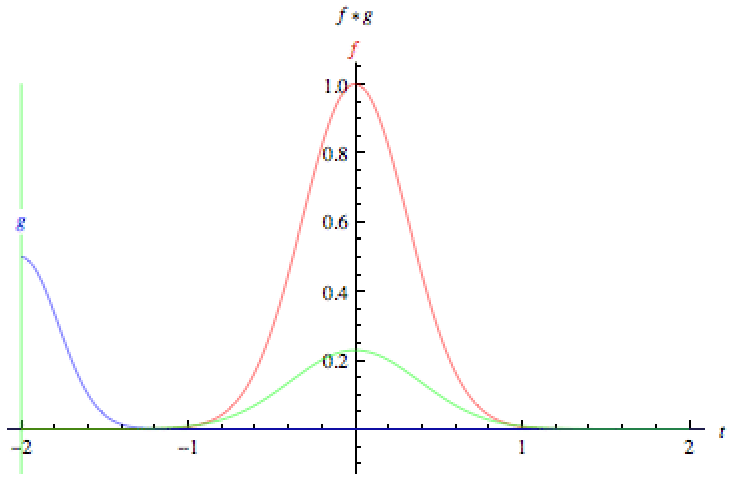
https://setosa.io/ev/image-kernels/

# Convolution in 1D

Convolution is more often taken over an infinite range,

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




The animations above graphically illustrate the convolution of two boxcar functions (left) and two Gaussians (right). In the plots, the green curve shows the convolution of the blue and red curves as a function of *t*, the position indicated by the vertical green line. The gray region indicates the product $g(\tau) f(t-\tau)$ as a function of *t*, so its area as a function of *t* is precisely the convolution. One feature to emphasize and which is not conveyed by these illustrations (since they both exclusively involve symmetric functions) is that the function *g* must be mirrored before lagging it across *f* and integrating. (https://mathworld.wolfram.com/Convolution.html)

The name Convolutional Neural Networks (CNN) suggests that they use the convolution operation, but in the usual way to describe CNN, **it is correlation that it's using**. However, convolution and correlation can be interchanged through a simple rotation operation. Therefore, the name Convolutional Neural Networks is also justified.

Convolution in math textbook

| | | | | | |
|---|---|---|---|---|---|
| 2 | 3 | 7 | 4 | 6 | 2 |
| 6 | 6 | 9 | 8 | 7 | 4 |
| 3 | 4 | 8 | 3 | 8 | 9 |
| 7 | 8 | 3 | 6 | 6 | 3 |
| 4 | 2 | 1 | 8 | 3 | 4 |
| 3 | 2 | 4 | 1 | 9 | 8 |

*

| | | |
|---|---|---|
| 3 | 4 | 5 |
| 1 | 0 | 2 |
| -1 | 9 | 7 |

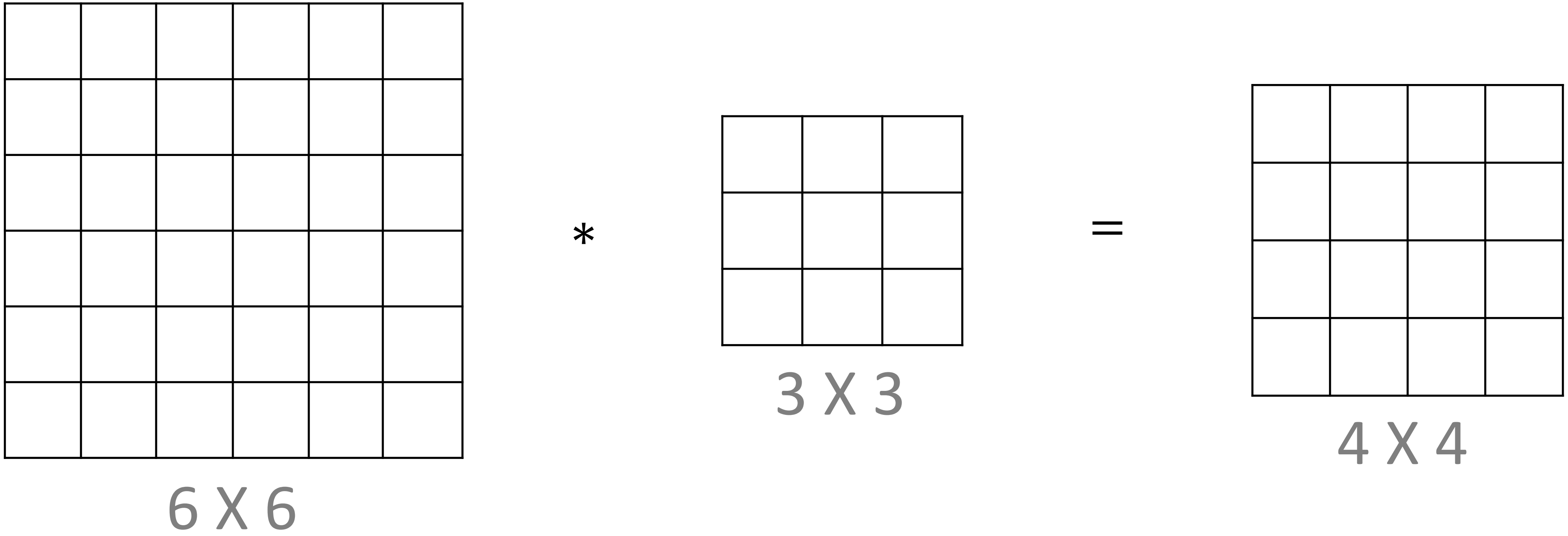
| | | |
|---|---|---|
| **7** | **9** | **-1** |
| **2** | **0** | **1** |
| **5** | **4** | **3** |

Mirroring it both on the vertical and horizontal axes

# Padding

## Padding

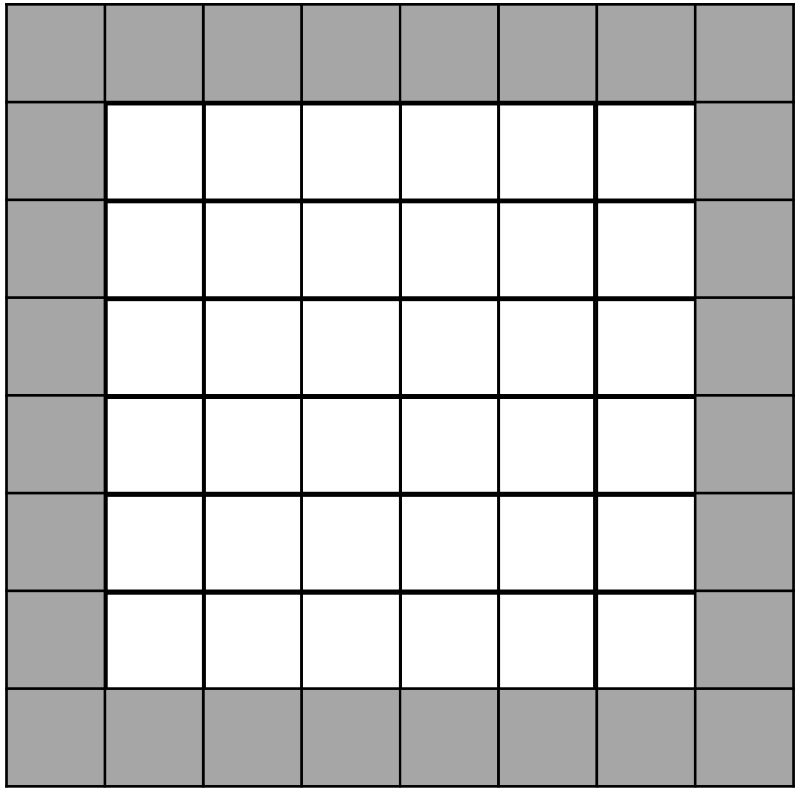
Output dimension without padding



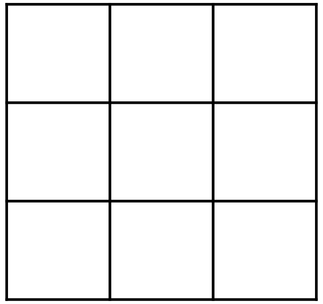

Generalize: *n X n* *f X f* *(n-f+1)X (n-f+1)*

# Padding

## Output dimension with padding

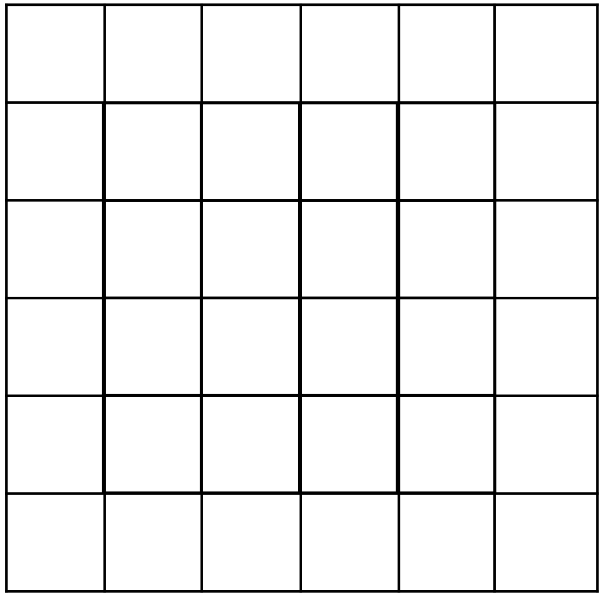


*



3 X 3

=



4 X 4

6 X 6

6 X 6

p = 1 8 X 8

Generalize: *(n+**2p**) X (n+**2p**)* *f X f* *(n+**2p**-f+1) X (n+**2p**-f+1)*

## Valid convolution and Same convolution

No padding

"Valid": $(nXn) \quad * \quad (fXf) \quad = \quad (n\text{-}f+1)\ X\ (n\text{-}f+1)$

$(6X6) \quad * \quad (3X3) \quad = \quad 4X4$

"Same": Pad so that output size is the same as the input size.

$(n+2p\text{-}f+1)\ X\ (n+2p\text{-}f+1)$

$n+2p-f+1=n \rightarrow p=(f-1)/2$

$(3X3) \rightarrow p=(3-1)/2=1$

$(5X5) \rightarrow p=(5-1)/2=2$

The value of f is usually "odd"

# Zero Padding

## Original Image

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |

## Zero Padding

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Mirror Padding

## Original Image

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |

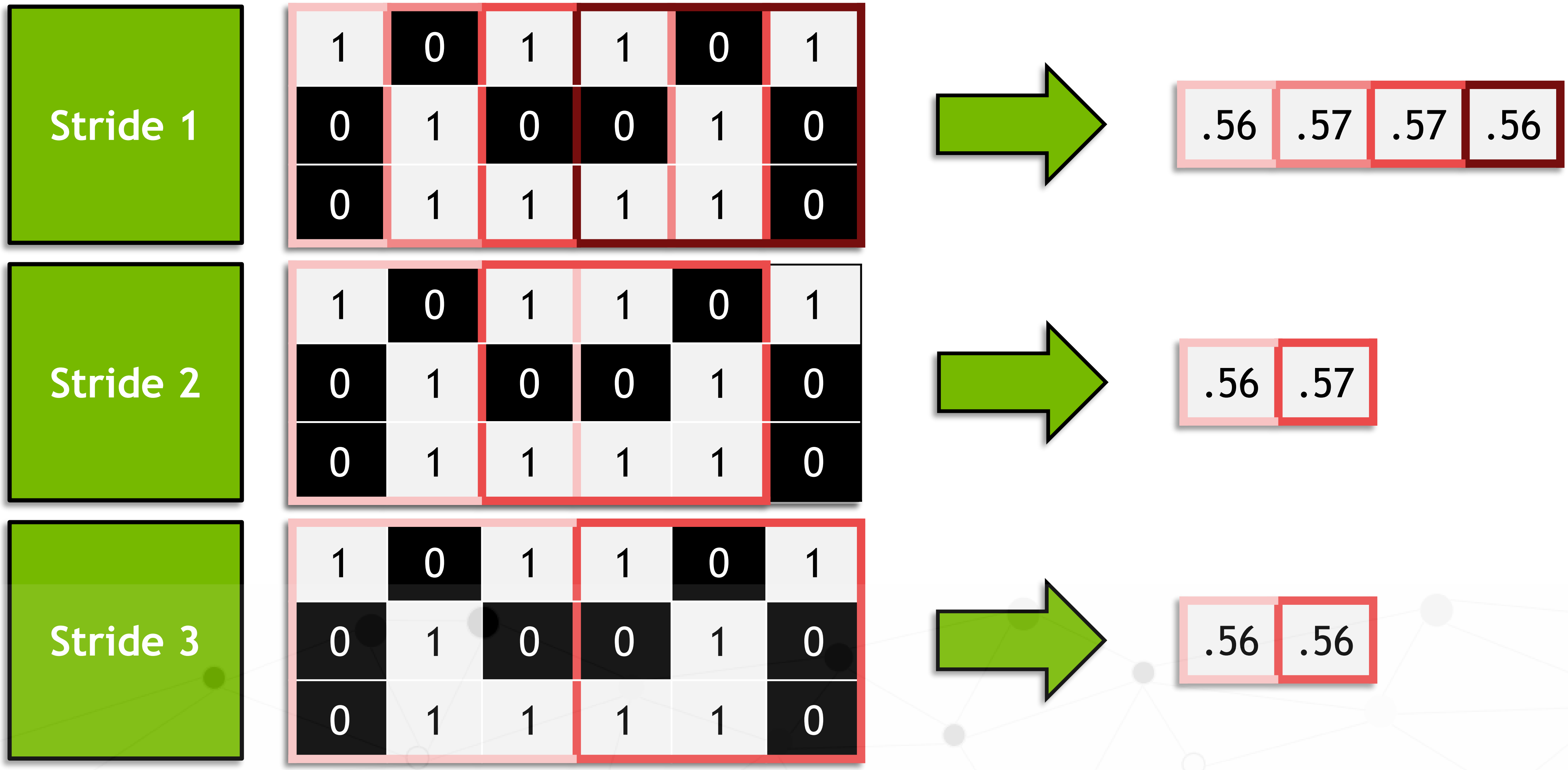
## Mirror Padding

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |

# Strided Convolutions

# Strided convolutions

## Different strides



# Strided Convolutions

## Example

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 3 | 7 | 4 | 6 | 2 | 9 |
| 6 | 6 | 9 | 8 | 7 | 4 | 3 |
| 3 | 4 | 8 | 3 | 8 | 9 | 7 |
| 7 | 8 | 3 | 6 | 6 | 3 | 4 |
| 4 | 2 | 1 | 8 | 3 | 4 | 6 |
| 3 | 2 | 4 | 1 | 9 | 8 | 3 |
| 0 | 1 | 3 | 9 | 2 | 1 | 4 |

7 X 7

$*$

| | | |
|---|---|---|
| 3 | 4 | 4 |
| 1 | 0 | 2 |
| -1 | 0 | 3 |

3 X 3

**Stride s =2**

=

| | | |
|---|---|---|
| | | |
| | | |
| | | |

3 X 3

$\lfloor x \rfloor = floor\ (x)$

$n \times n$ image $*$ $f \times f$ filter

padding $p$ stride $s$

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

$$\left\lfloor \frac{7+2*0-3}{2} + 1 \right\rfloor = 3$$

Output dimension

$n \times n$ image $\quad f \times f$ filter $\quad$ padding $p \quad$ stride $s$

Output size

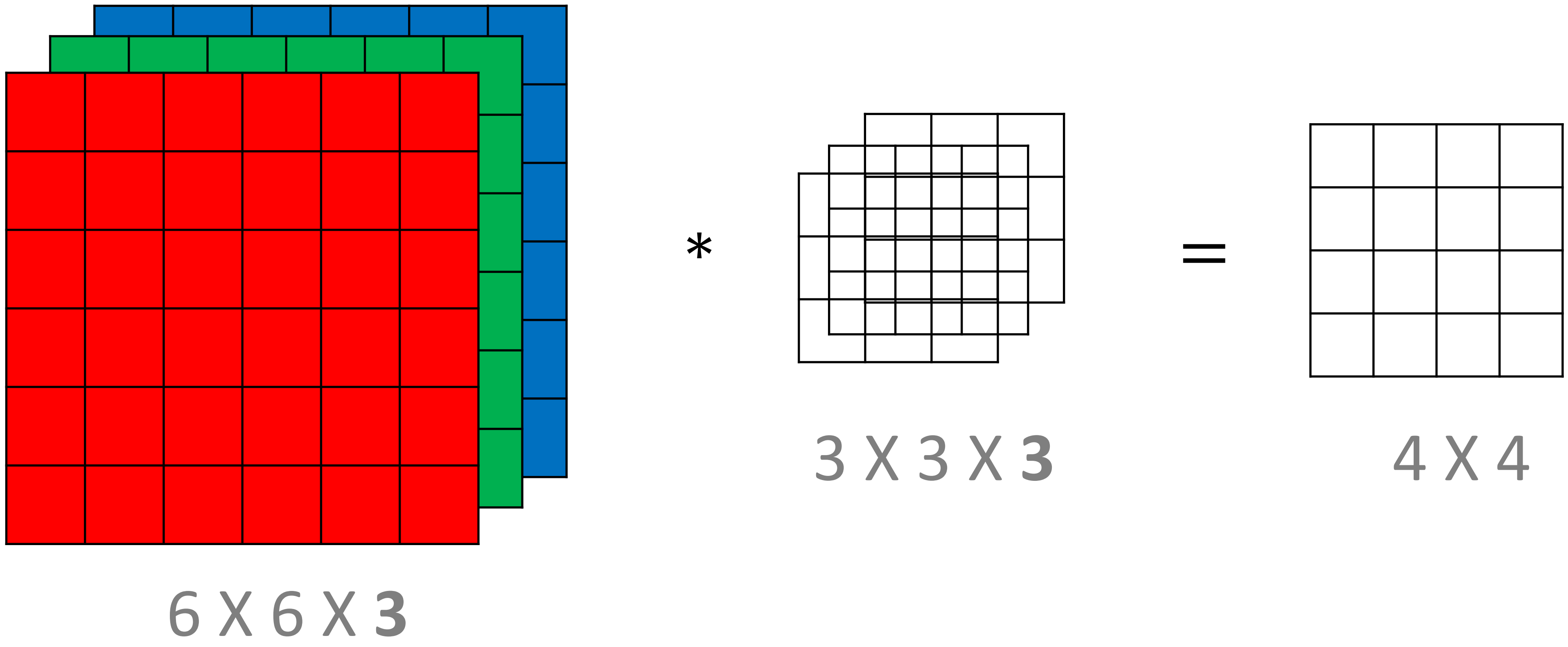
$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

$\lfloor x \rfloor = floor\ (x)$

# Convolutions Over Volumes

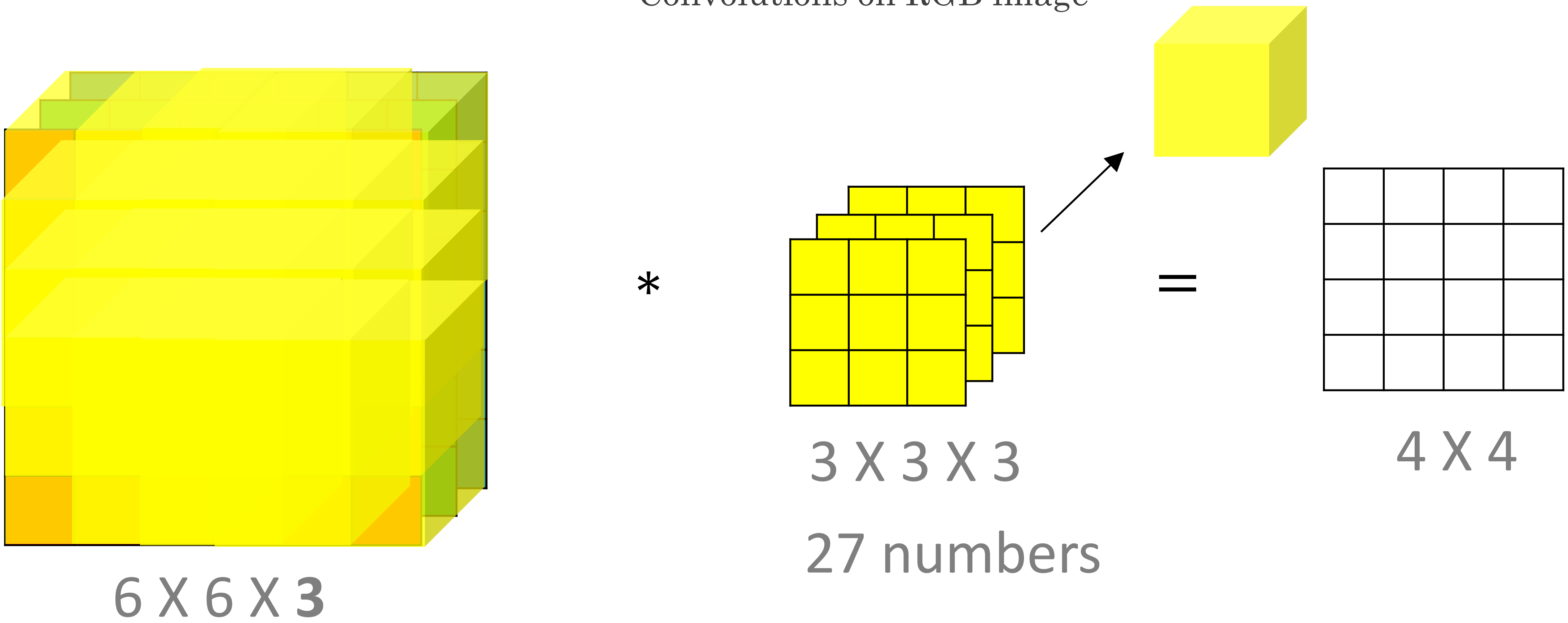
# Convolution Over Volumes

Convolutions on RGB images



# Convolution Over Volumes

Convolutions on RGB image




| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

3 X 3 X 3

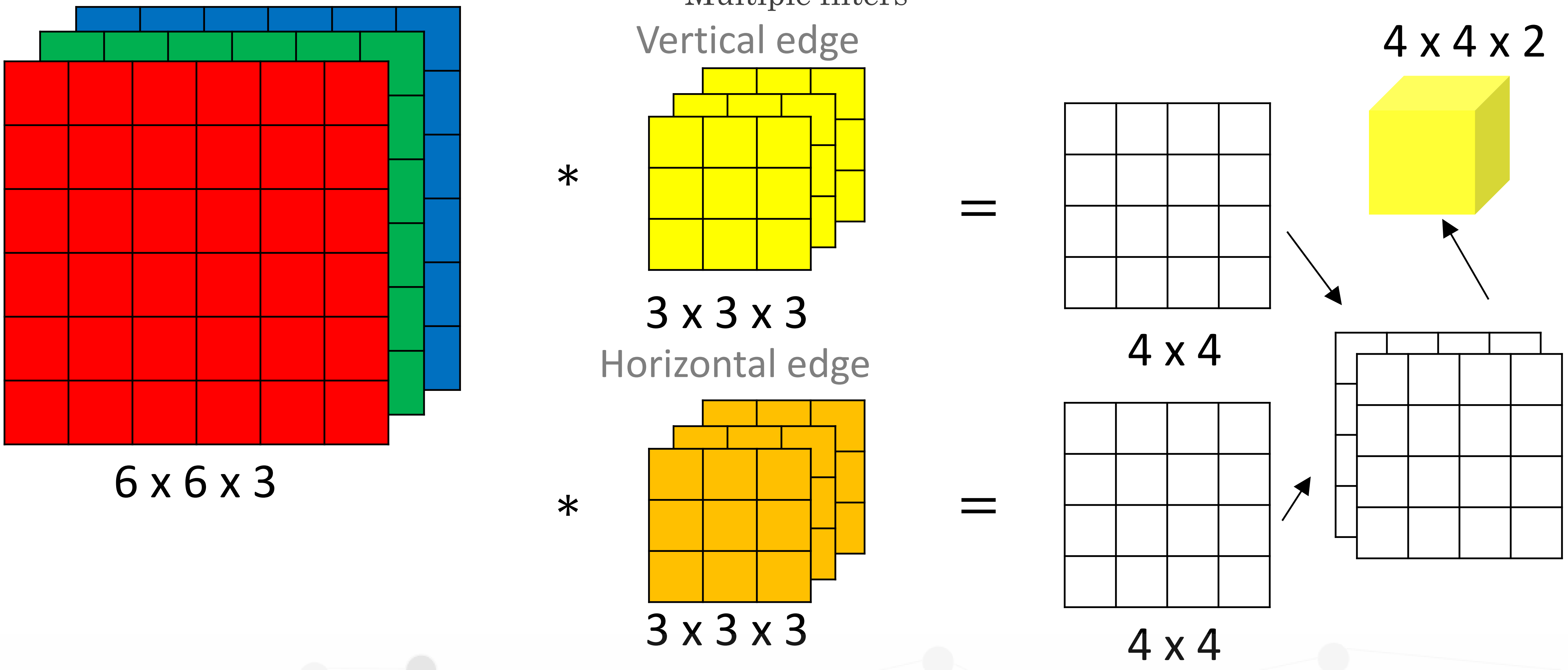
| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

| | | |
|---|---|---|
| 1 | 0 | -1 |
| 1 | 0 | -1 |
| 1 | 0 | -1 |

# Convolution Over Volumes

Multiple filters




\# of filters

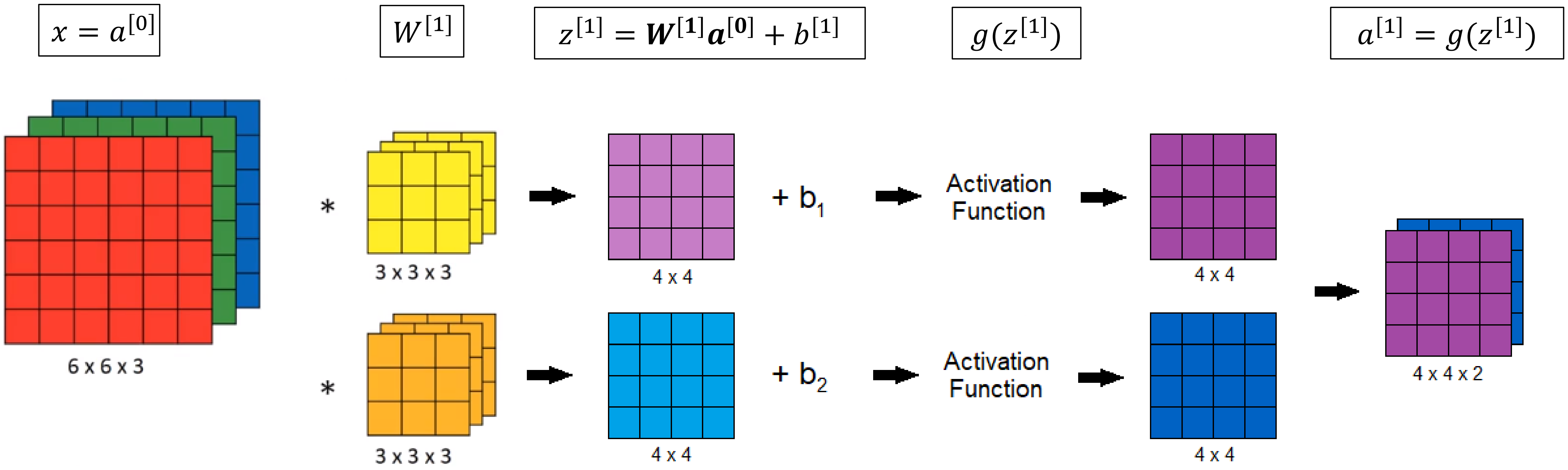
Summary: $n \times n \times n_c$ * $f \times f \times n_c$ → $(n-f+1) \times (n-f+1) \times n'_c$

# One Layer of a Convolutional Network


Yann LeCun

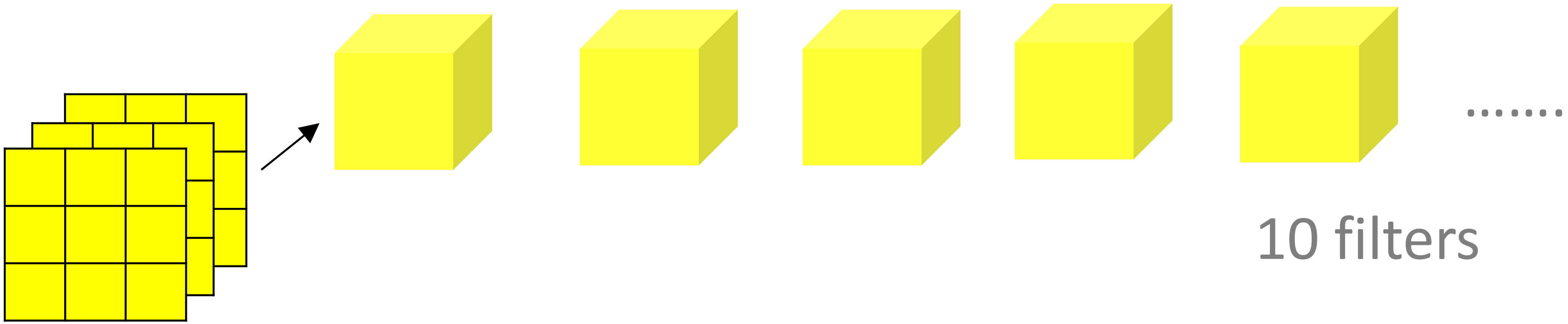
# One Layer of a Convolutional Network

## Example of a layer



Number of parameters in one layer

# If you have 10 filters that 3 x 3 x 3 in one layer of a neural network, how many parameters does this layer have?




3 X 3 X 3

27 parameters + 1 bias
= 28 parameters

28 X 10 = 280 parameters

Summary of notation

# If layer $l$ is a convolutional layer:

$f^{[l]}$: filter size in layer $l$
$p^{[l]}$: padding in layer $l$
$s^{[l]}$: stride in layer $l$
$n_c^{[l]}$ : # of filters

Input: $n_H^{[l-1]}$ x $n_w^{[l-1]}$ x $n_c^{[l-1]}$
Output: $n_H^{[l]}$ x $n_w^{[l]}$ x $n_c^{[l]}$

$$n_H^{[l]} = \left\lfloor \frac{n_H^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + \mathbf{1} \right\rfloor$$

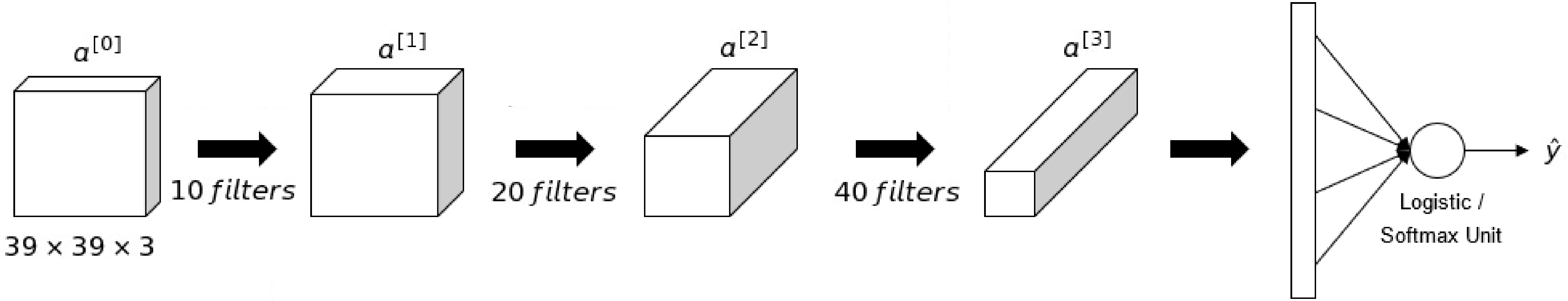
Each filters is: $f^{[l]}$ x $f^{[l]}$ x $n_c^{[l-1]}$
Activation $a^{[l]}$ : $n_H^{[l]}$ x $n_w^{[l]}$ x $n_c^{[l]}$
Weights: $f^{[l]}$ x $f^{[l]}$ x $n_c^{[l-1]}$ x $n_c^{[l]}$
Bias: $n_c^{[l]}$ or (1,1,1, $n_c^{[l]}$)

$A^{[l]}$ : m x $n_H^{[l]}$ x $n_w^{[l]}$ x $n_c^{[l]}$

# A Simple Convolution Network Example

# A Simple Convolution Network Example

Example ConvNet




$$n_H^{[l]} = \left\lfloor \frac{n_H^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + \mathbf{1} \right\rfloor$$

## Types of layer in a convolutional network

- Convolution (Conv net) Layer
- Pooling (Pool) Layer
- Fully connected (FC) Layer

# Next:

Lab Practice

Transfer Learning