UNIVERSITY of LOUISIANA LAFAYETTE

CSCE566-DATA MINING
WEEK 9

Convolutional Neural Networks

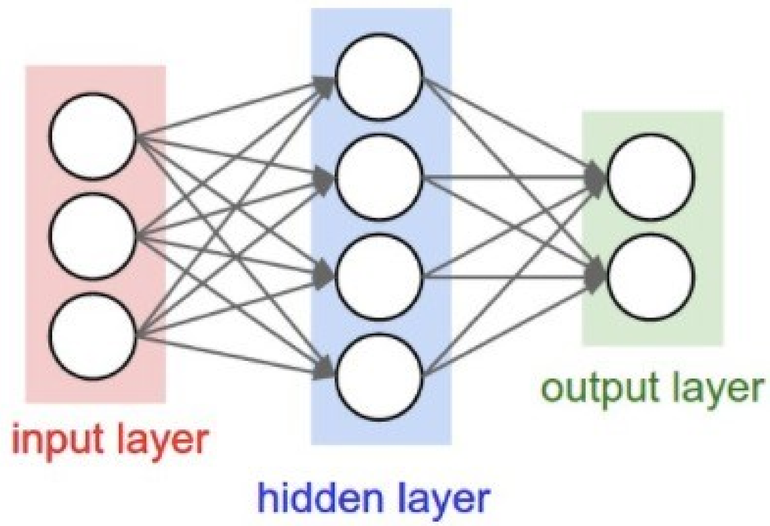
Min Shi
min.shi@louisiana.edu

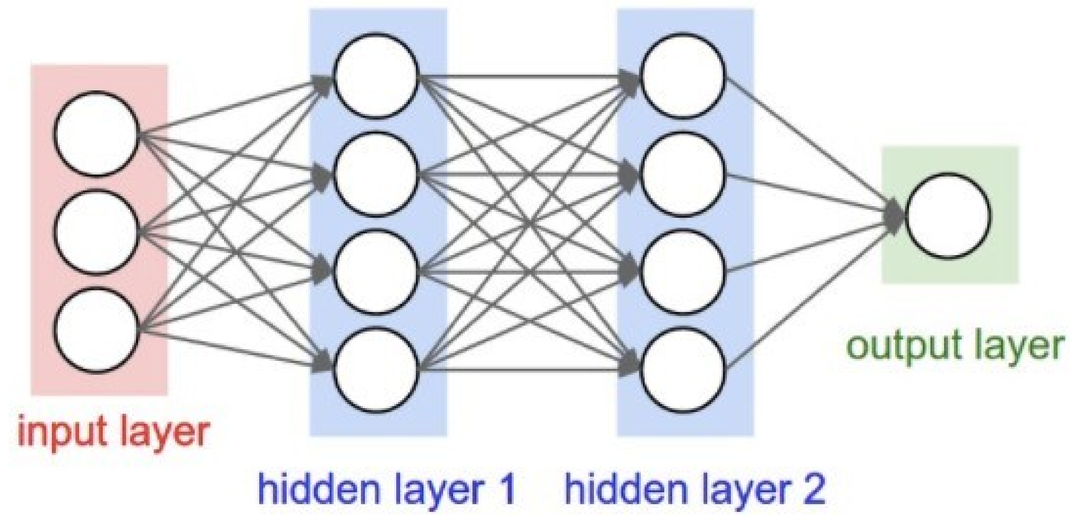
Oct 21, 2024

Outline: Image Data Mining

1. Introduction
2. Digital Image Processing
3. Convolutional Neural Network
4. Computer Vision Tasks

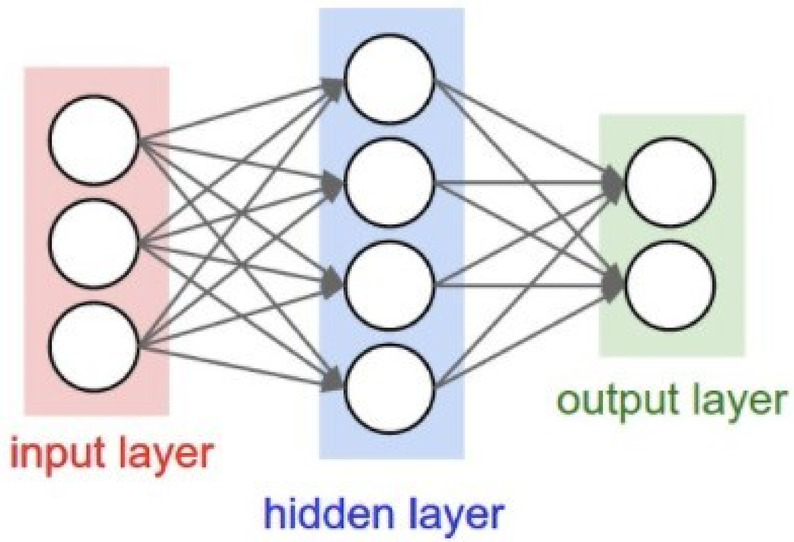
Recall: Feedforward neural networks

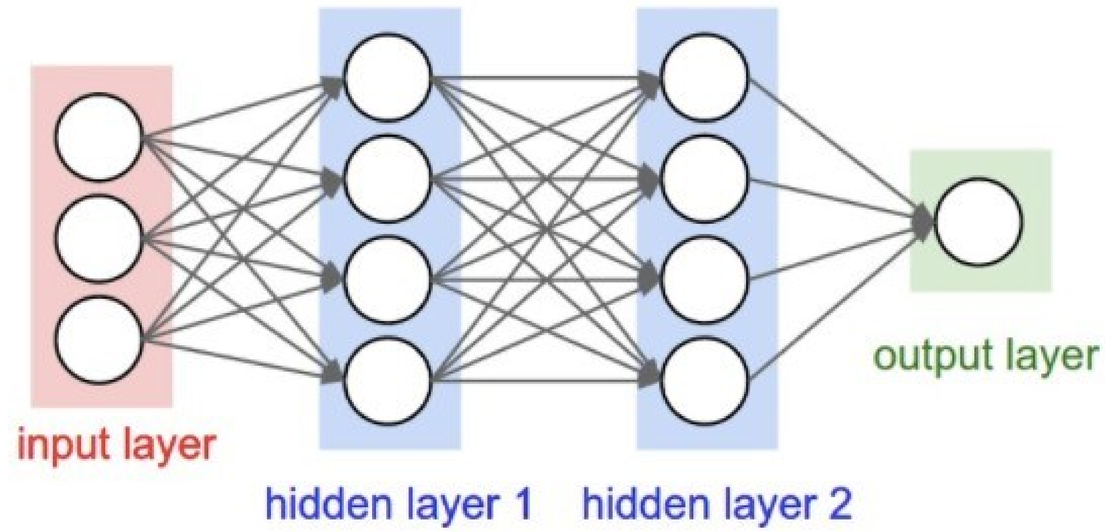




Each node provides input to each node in the next layer

Recall





- Assume 2-layer model with 100 nodes per layer
  - e.g., how many weights are in a 640x480 image?
    - 640x480x3x100 + 100x100 + 100x1 = 92,170,100
  - e.g., how many weights are in a 2048X1536 image (3.1 Megapixel image)?
    - 2048x1536x3x100 + 100x100 + 100x1 = 943,728,500

Recall

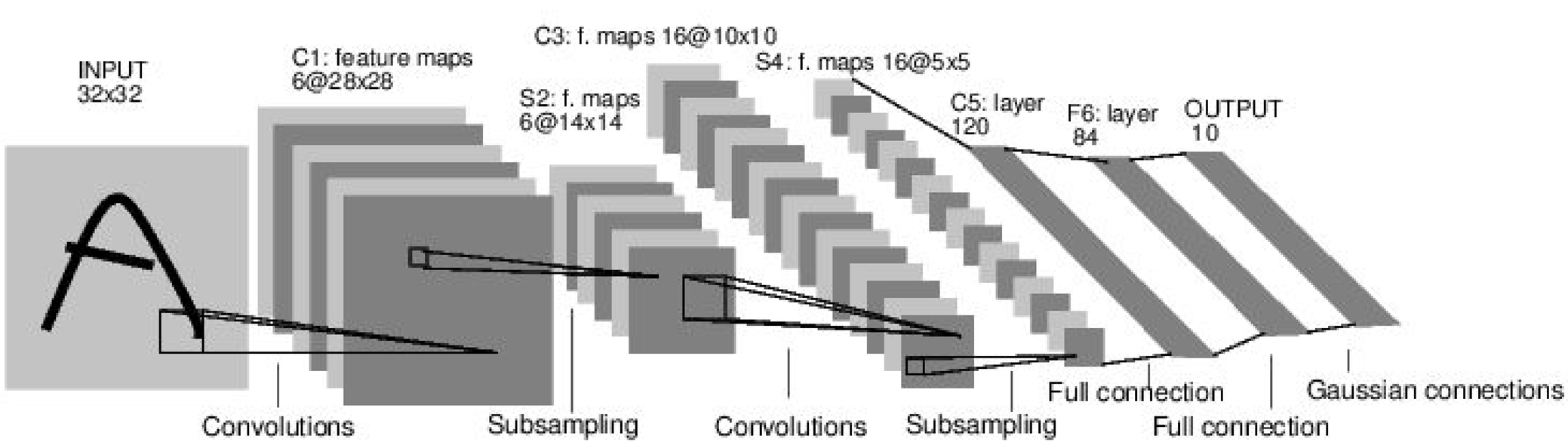
Many model parameters and so…

- greater chance to overfit
- increased training time
- needs more training data

- Assume 2 layer model with 100 nodes per layer
  - e.g. how many weights are in a 640x480 image?
    - 640x480x3x100 + 100x100 + 100x1 = 92,170,100
  - e.g. how many weights are in a 2048X1536 image (3.1 Megapixel image)?
    - 2048x1536x3x100 + 100x100 + 100x1 = 943,728,500

Convolutional layer

Idea: each node receives input only from a small neighborhood in previous layer



Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

A filter:

-1	1	-1

Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

A filter:

-1	1	-1

After convolution*:

Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

A filter:

-1	1	-1

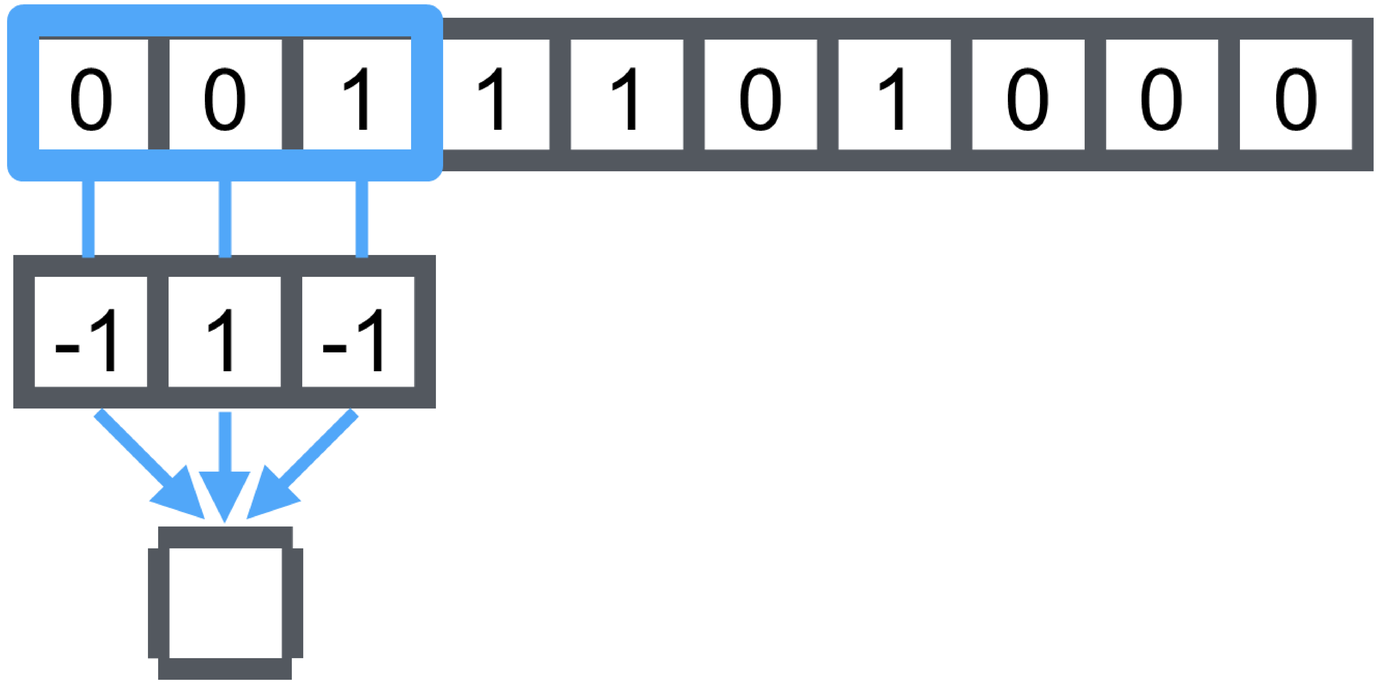
After convolution*:

Convolutional layer: 1D example

A 1D image:

A filter:

After convolution*:

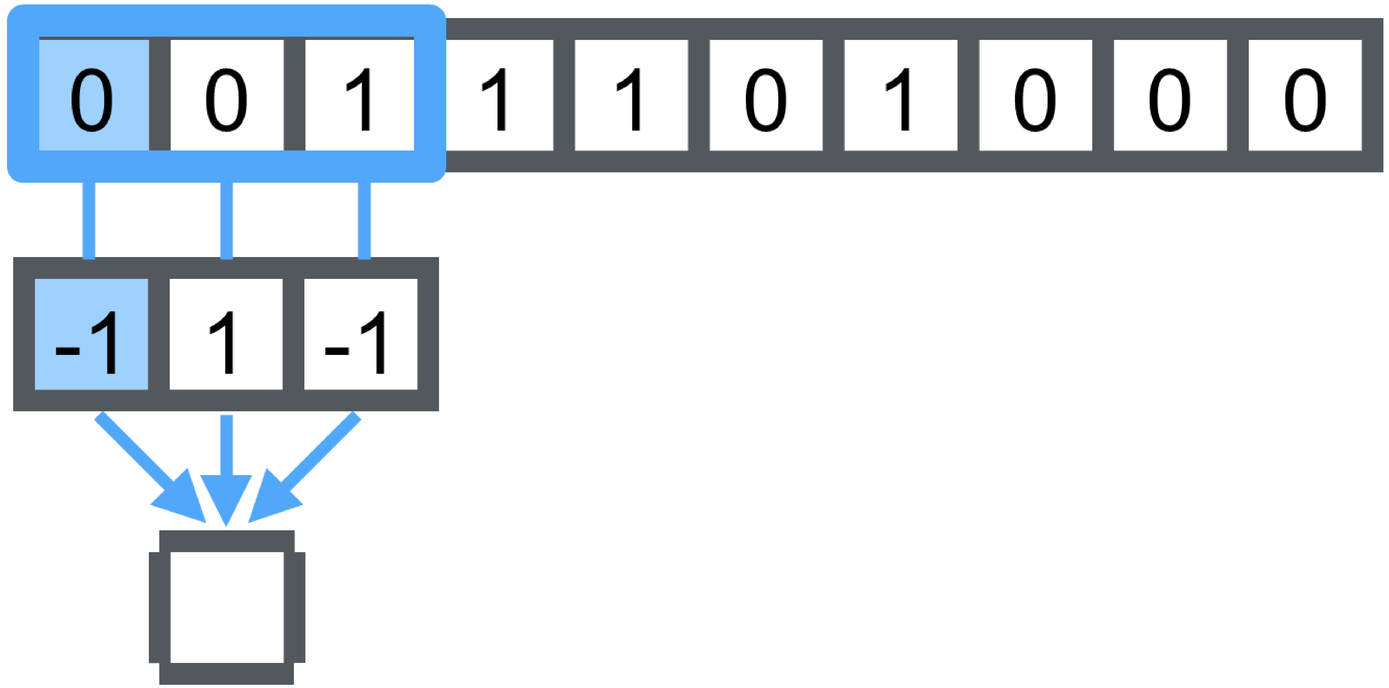


Convolutional layer: 1D example

A 1D image:

A filter:

After convolution*:

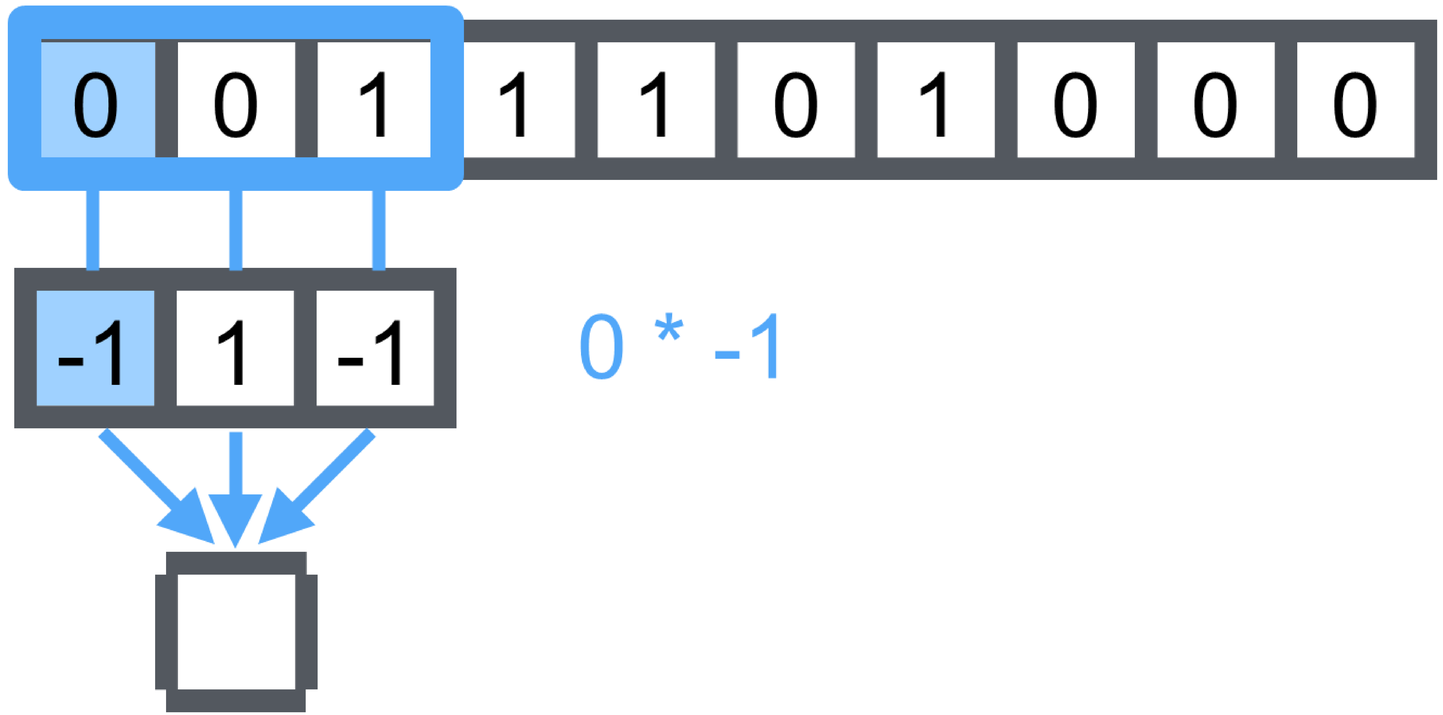


Convolutional layer: 1D example

A 1D image:

A filter:

After convolution*:

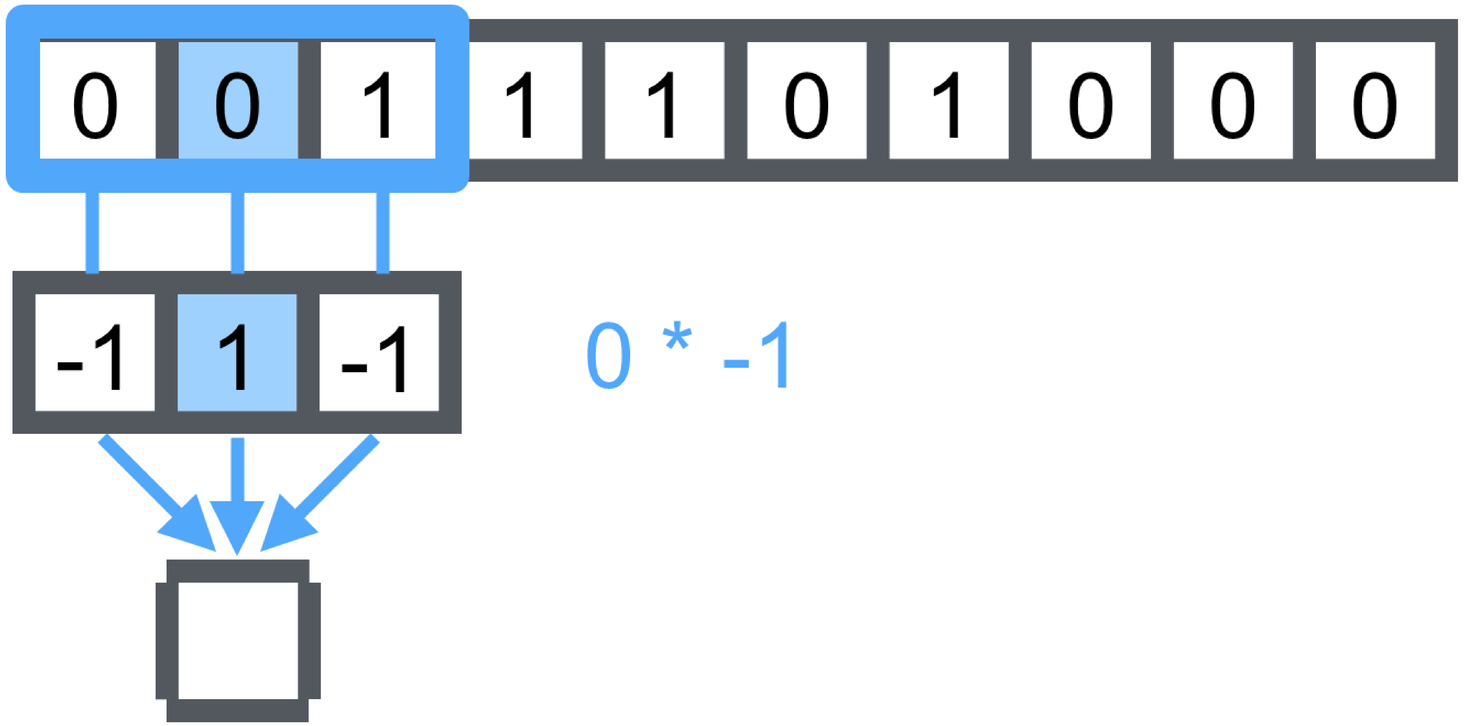


Convolutional layer: 1D example

A 1D image:

A filter:

After convolution*:

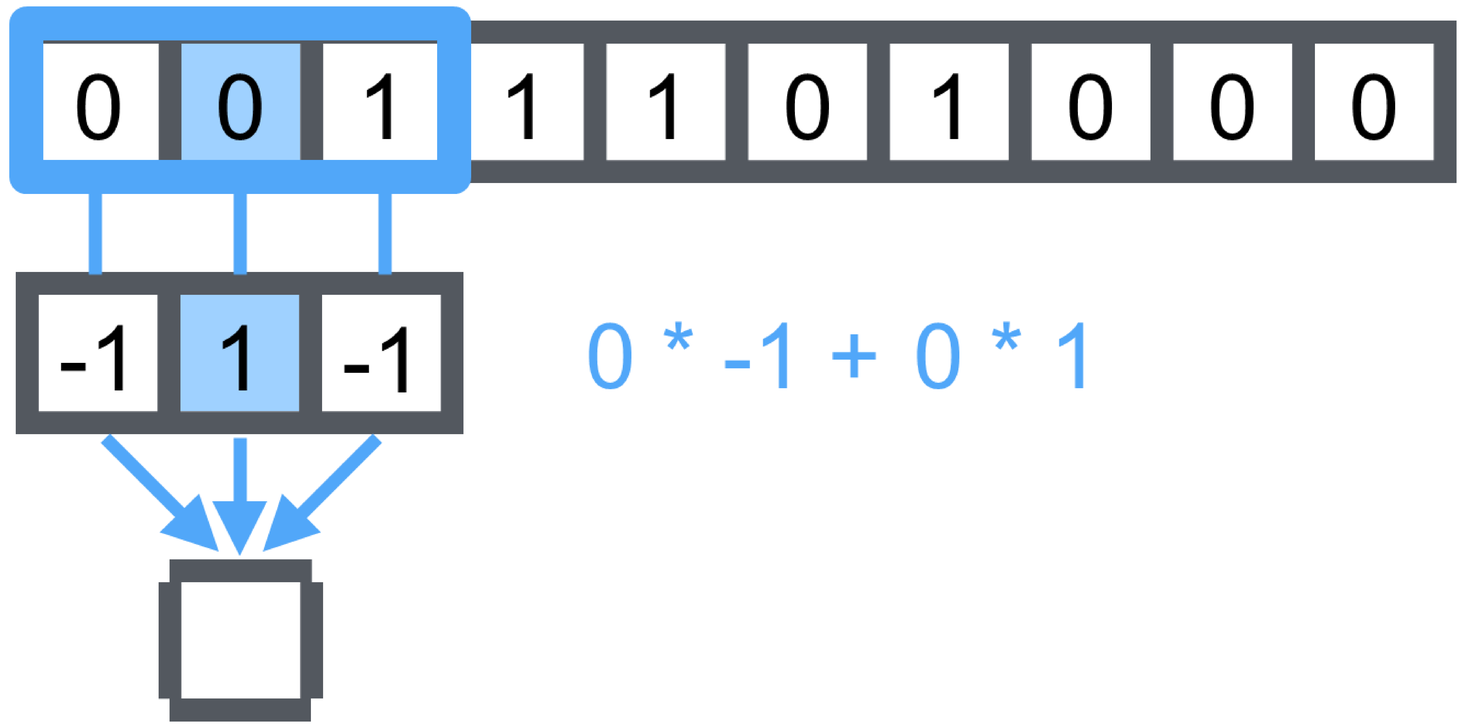


Convolutional layer: 1D example

A 1D image:

A filter:

After convolution*:



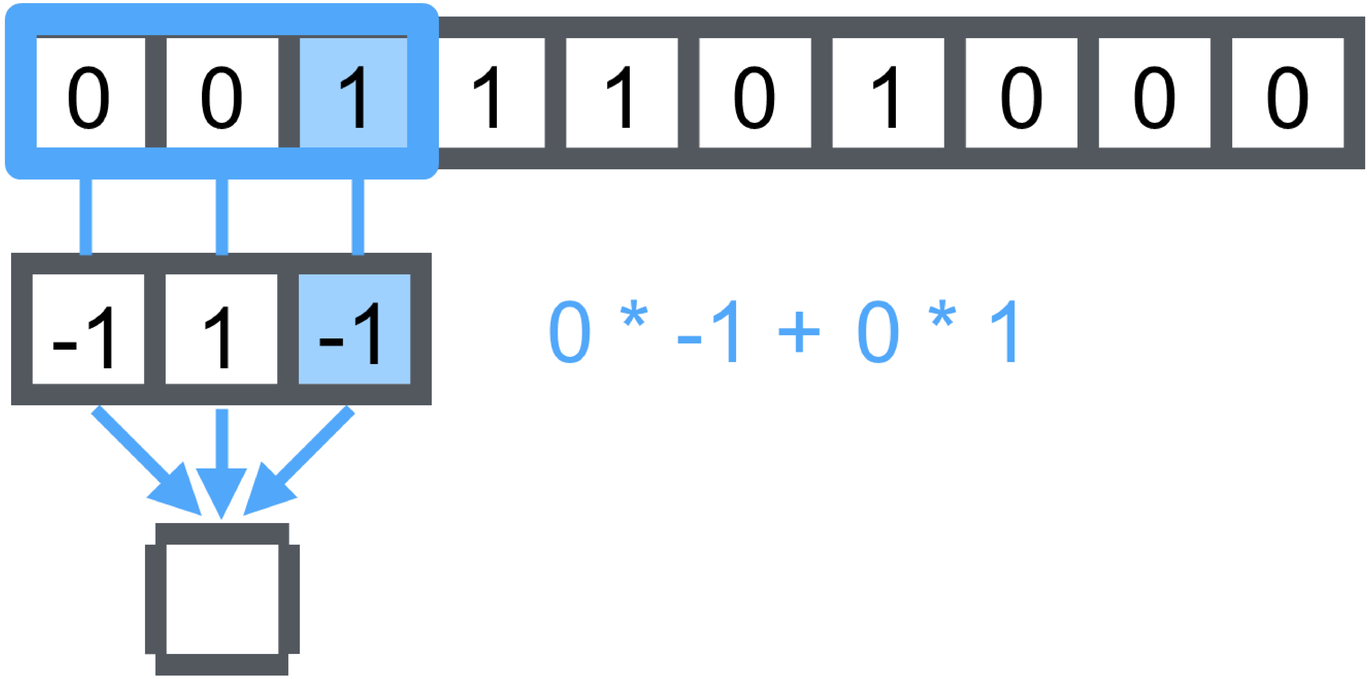
0 * -1 + 0 * 1

Convolutional layer: 1D example

A 1D image:

A filter:

After convolution*:



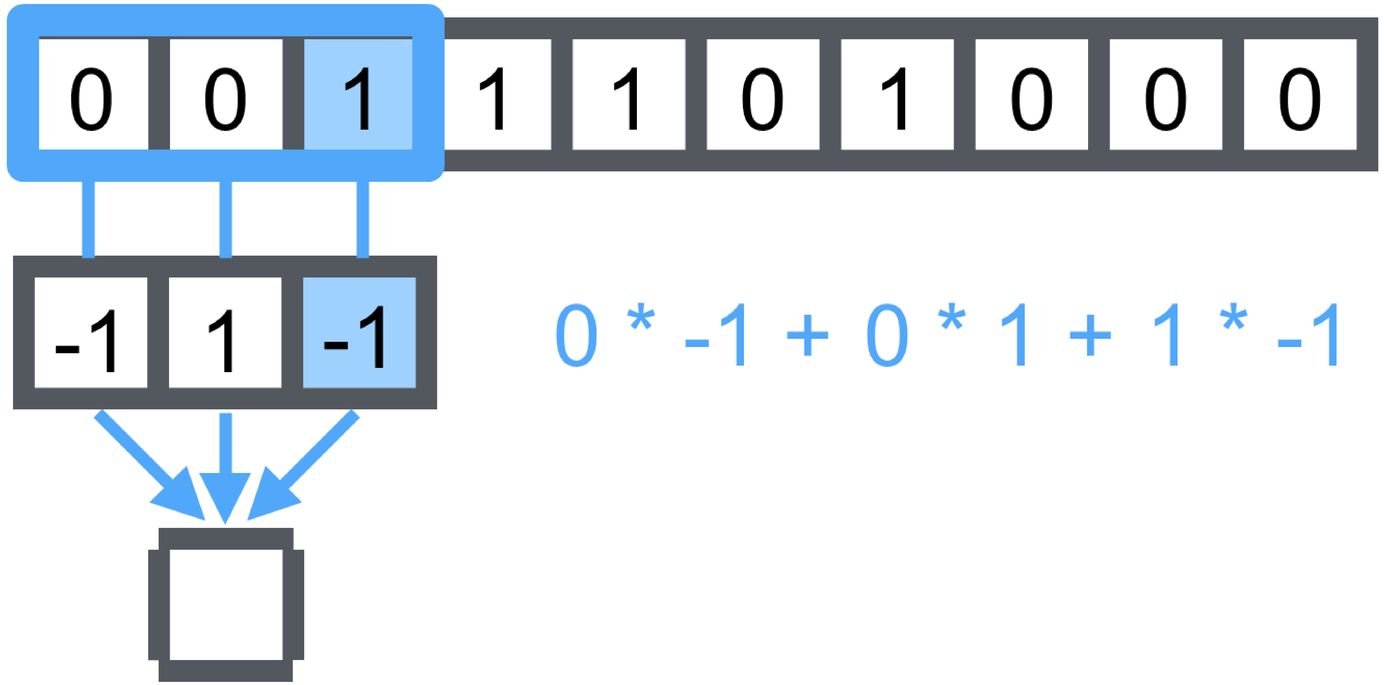
$0 * -1 + 0 * 1$

Convolutional layer: 1D example

A 1D image:

A filter:

After convolution*:



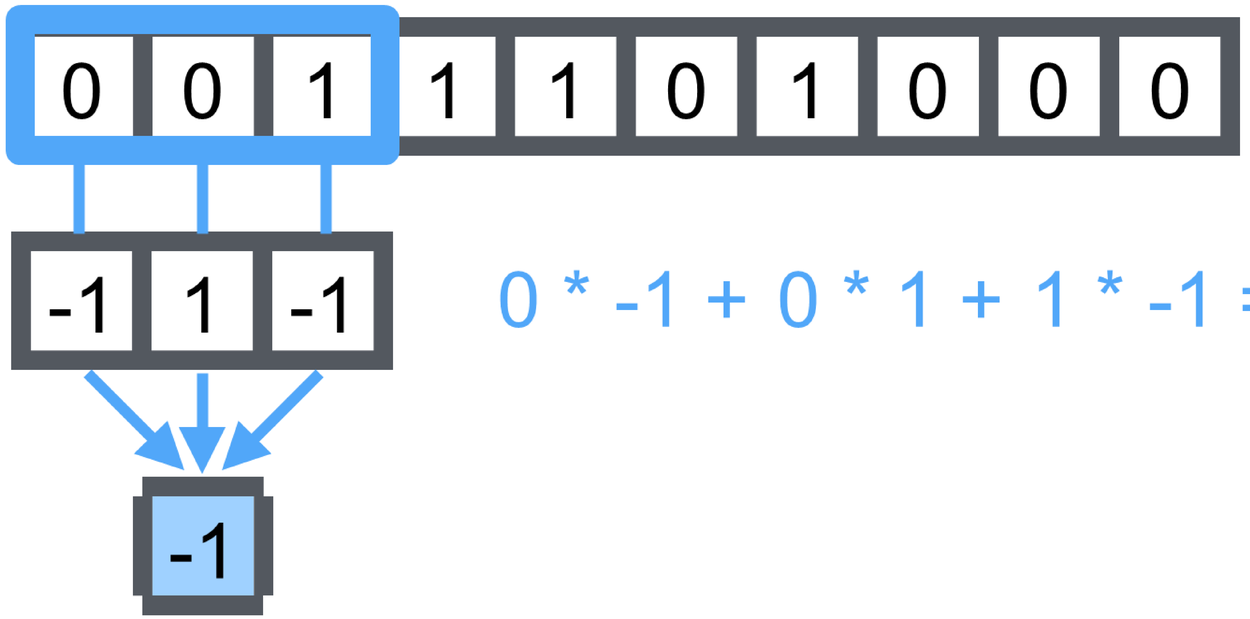
0 * -1 + 0 * 1 + 1 * -1

Convolutional layer: 1D example

A 1D image:

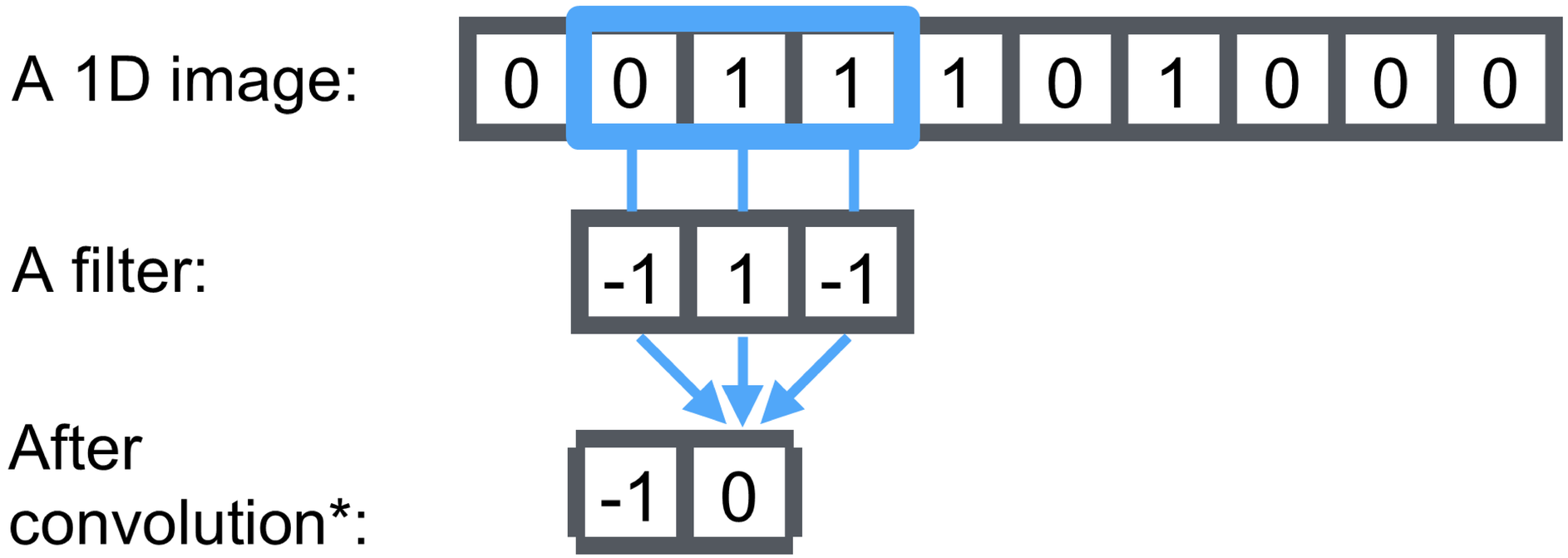
A filter:

After convolution*:

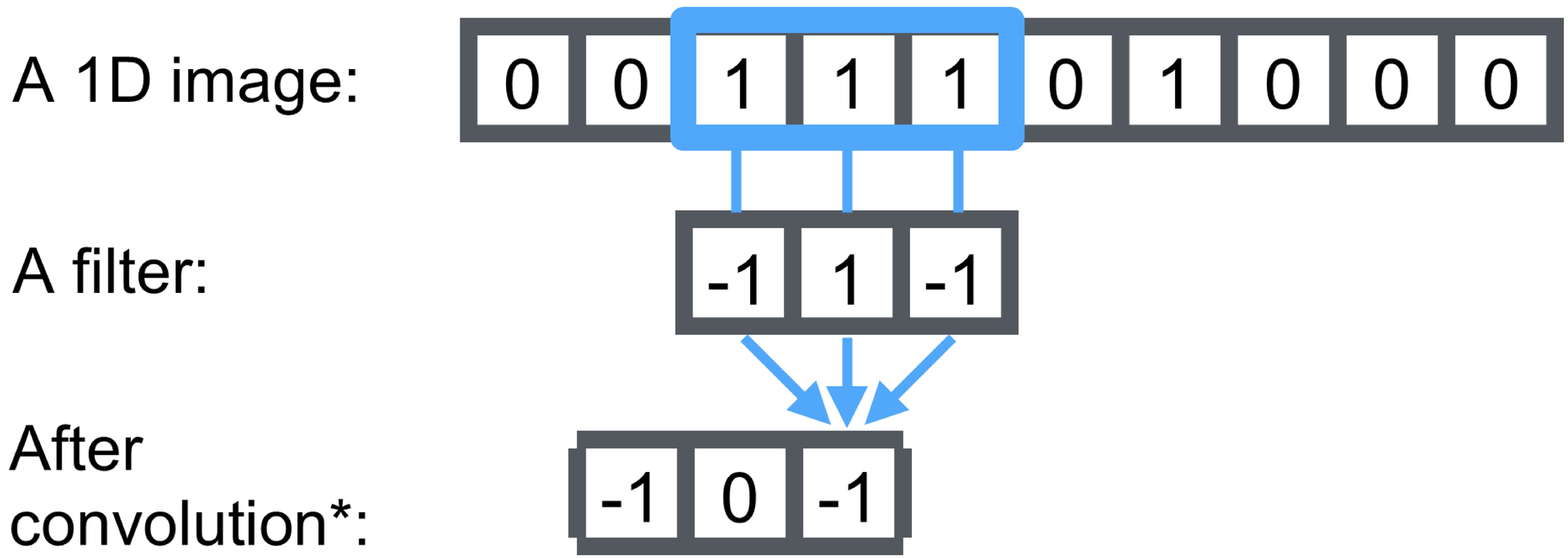


$0 * -1 + 0 * 1 + 1 * -1 = -1$

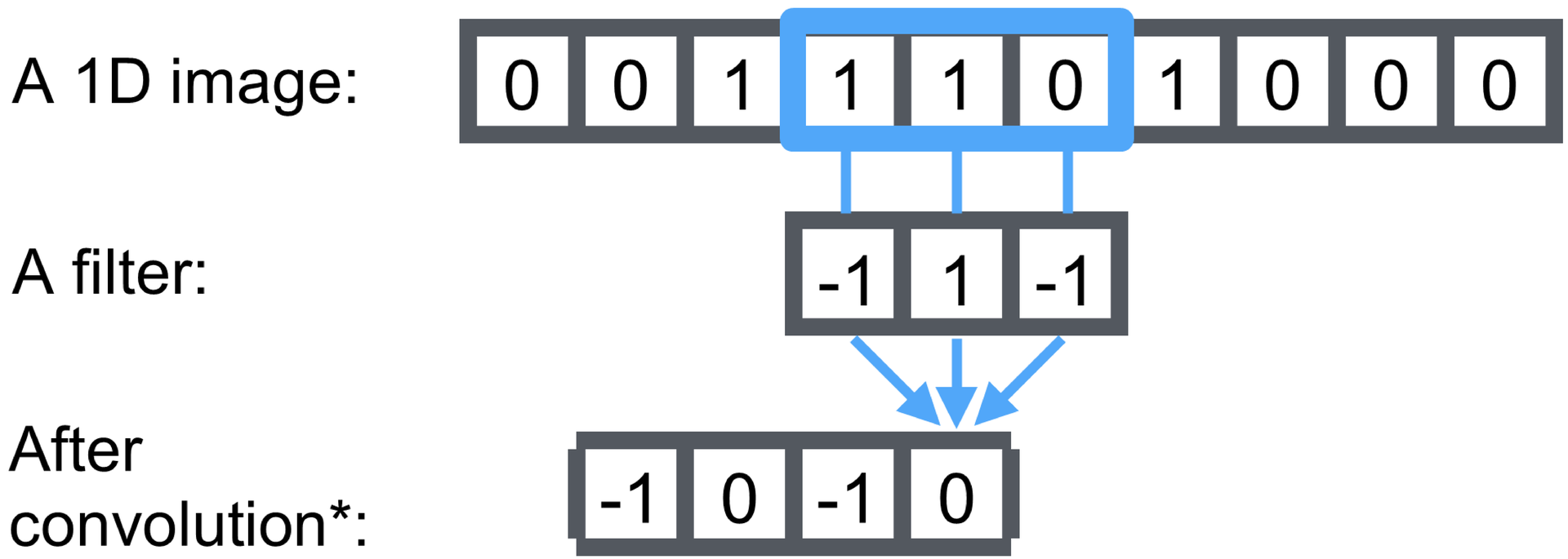
Convolutional layer: 1D example



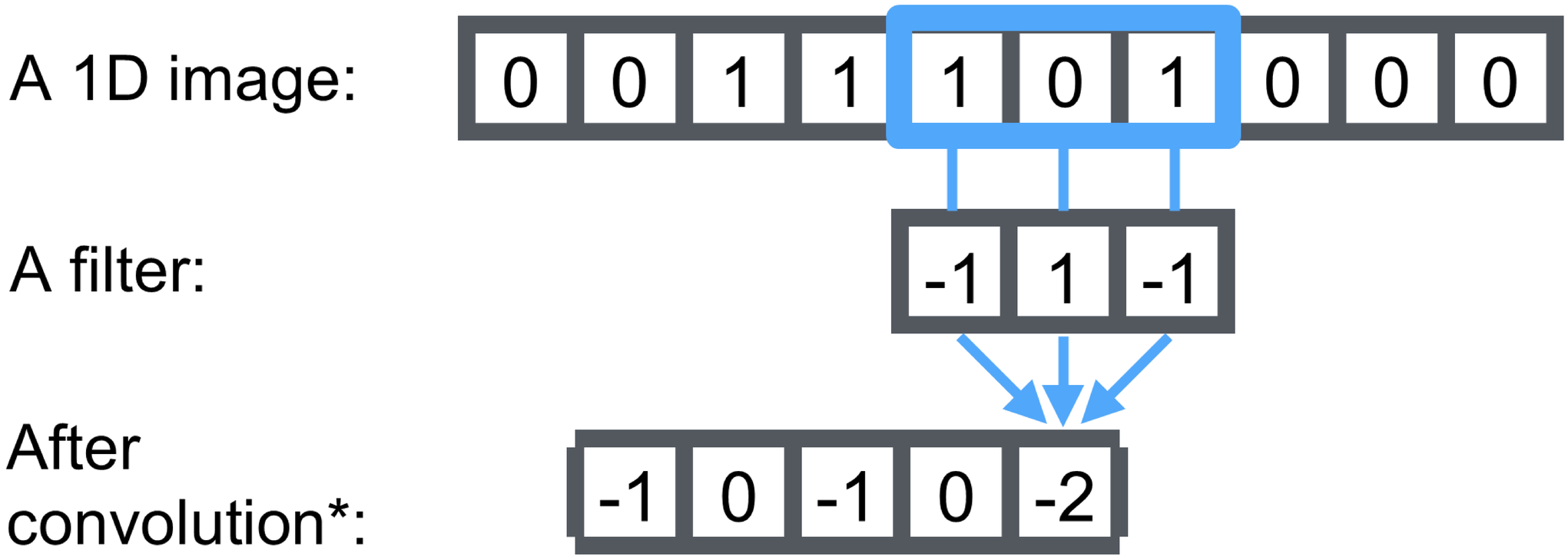
Convolutional layer: 1D example



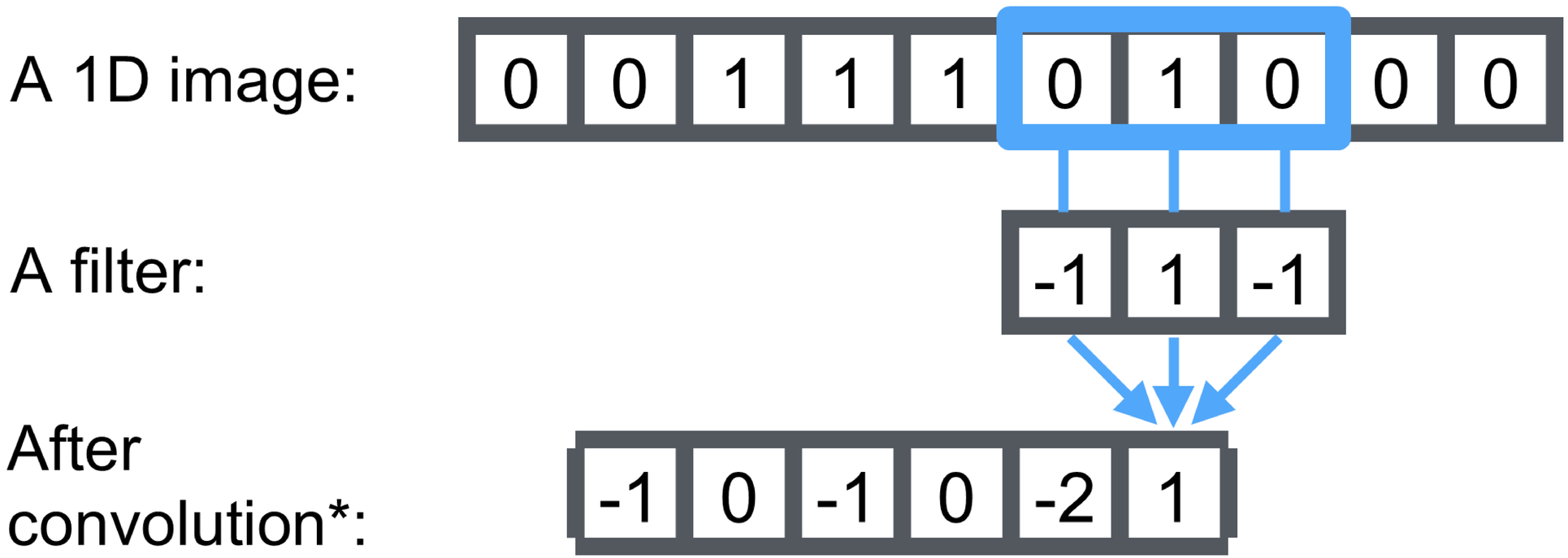
Convolutional layer: 1D example



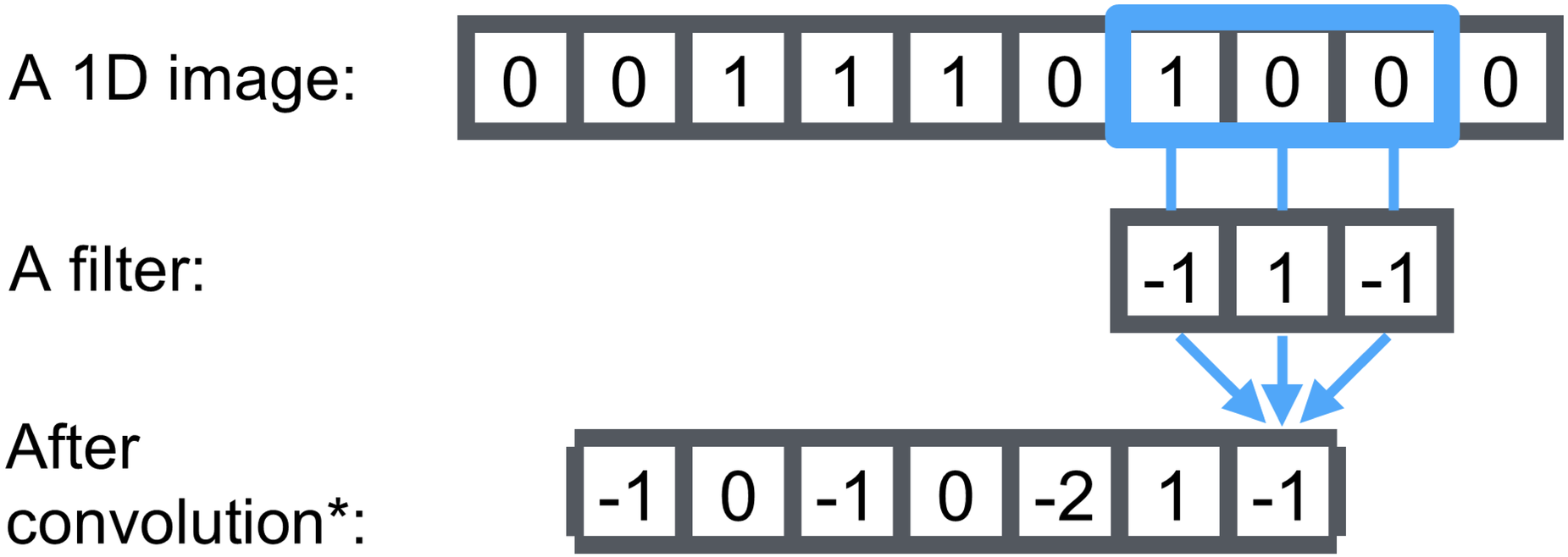
Convolutional layer: 1D example



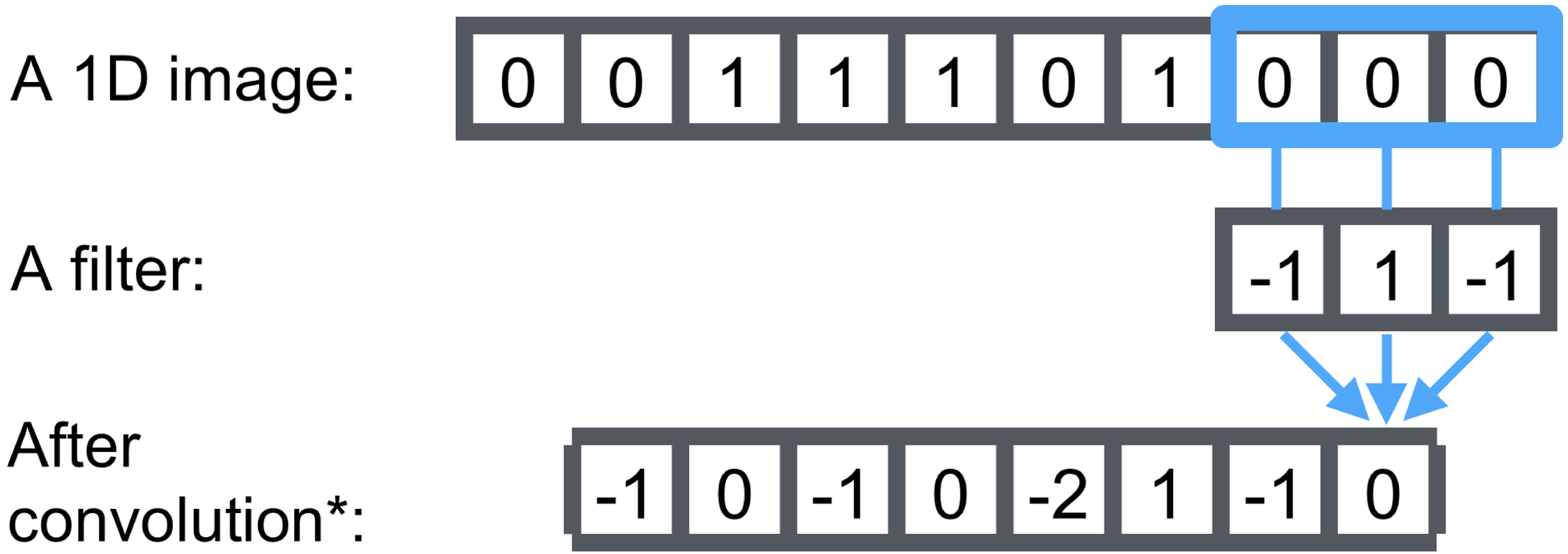
Convolutional layer: 1D example



Convolutional layer: 1D example



Convolutional layer: 1D example



Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

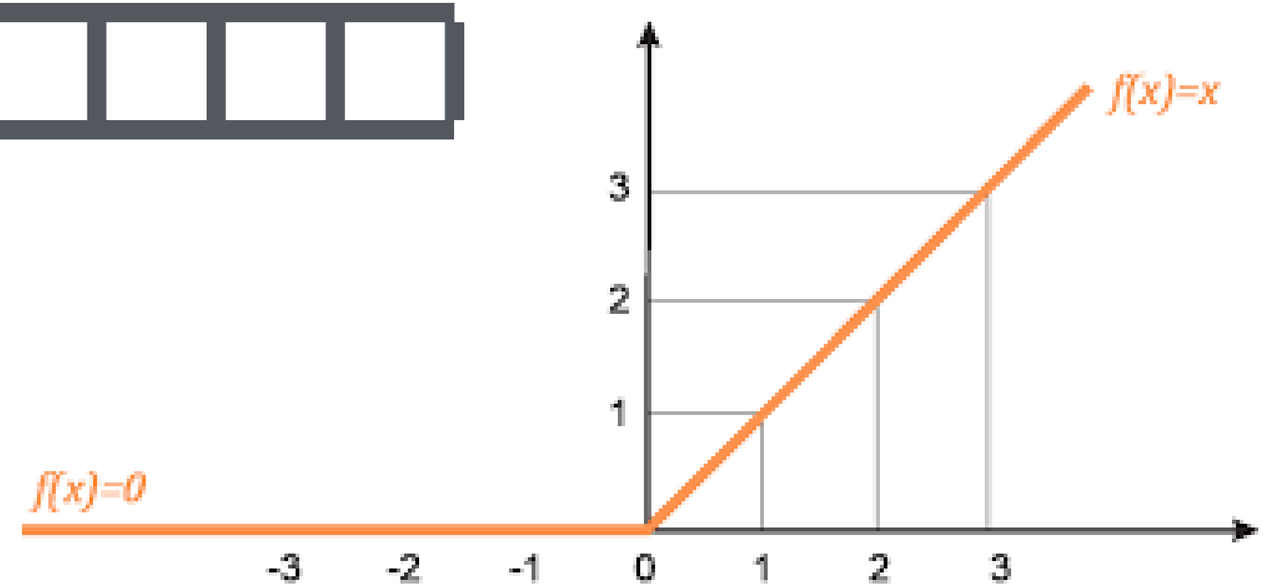
A filter:

-1	1	-1

After convolution*:

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

After ReLU:



Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

A filter:

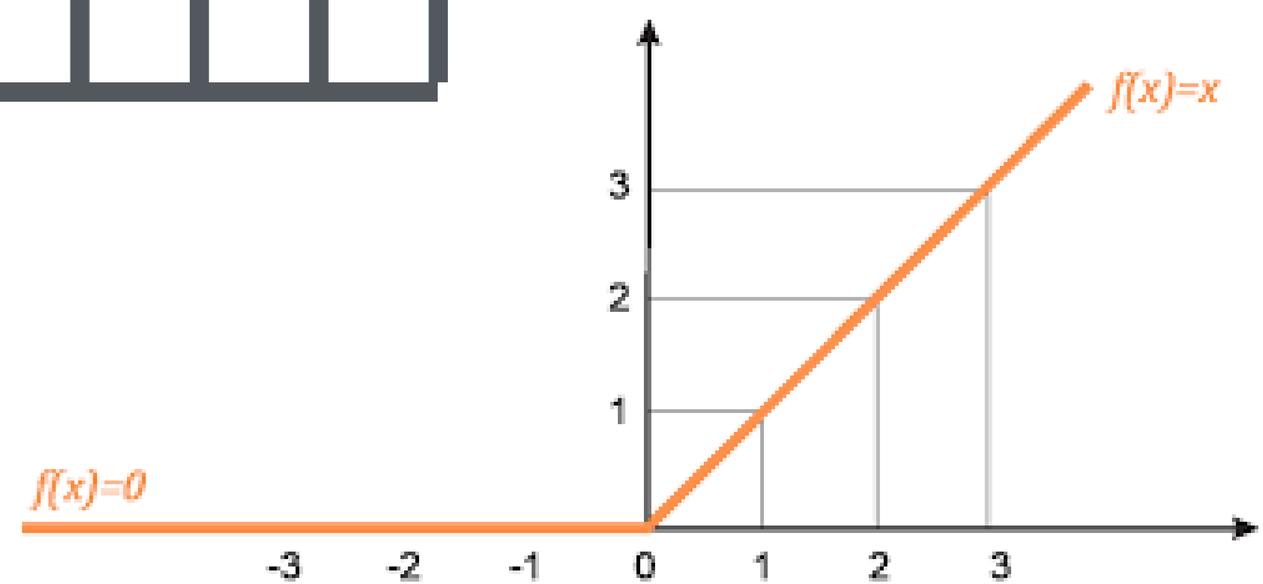
-1	1	-1

After convolution*:

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

After ReLU:

0							



Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

A filter:

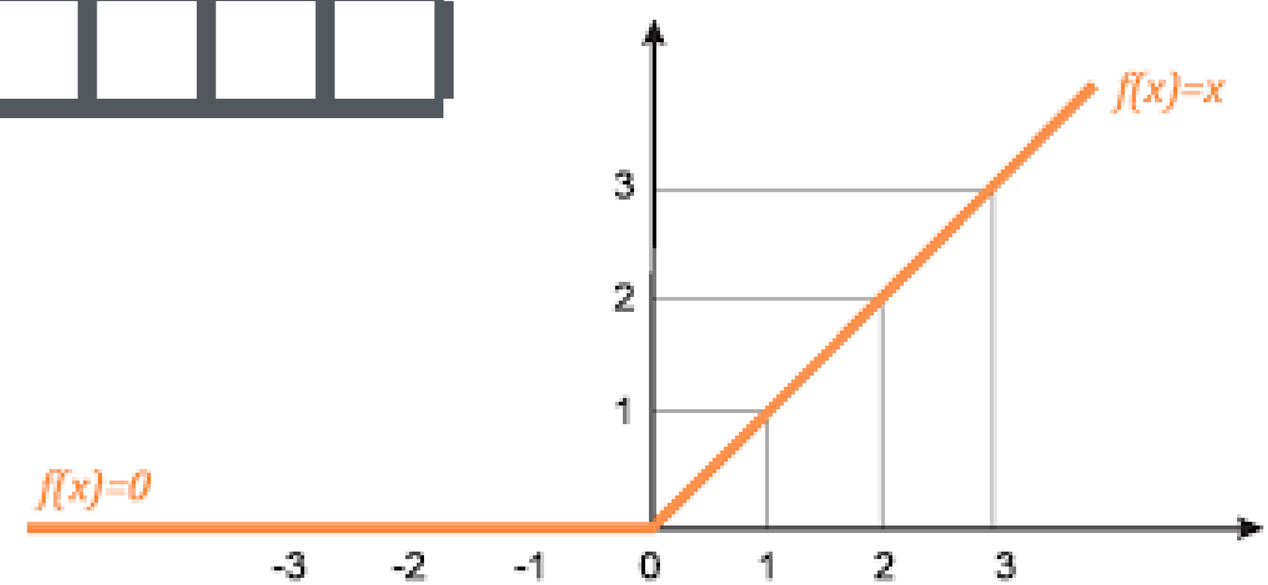
-1	1	-1

After convolution*:

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

After ReLU:

0	0						



Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

A filter:

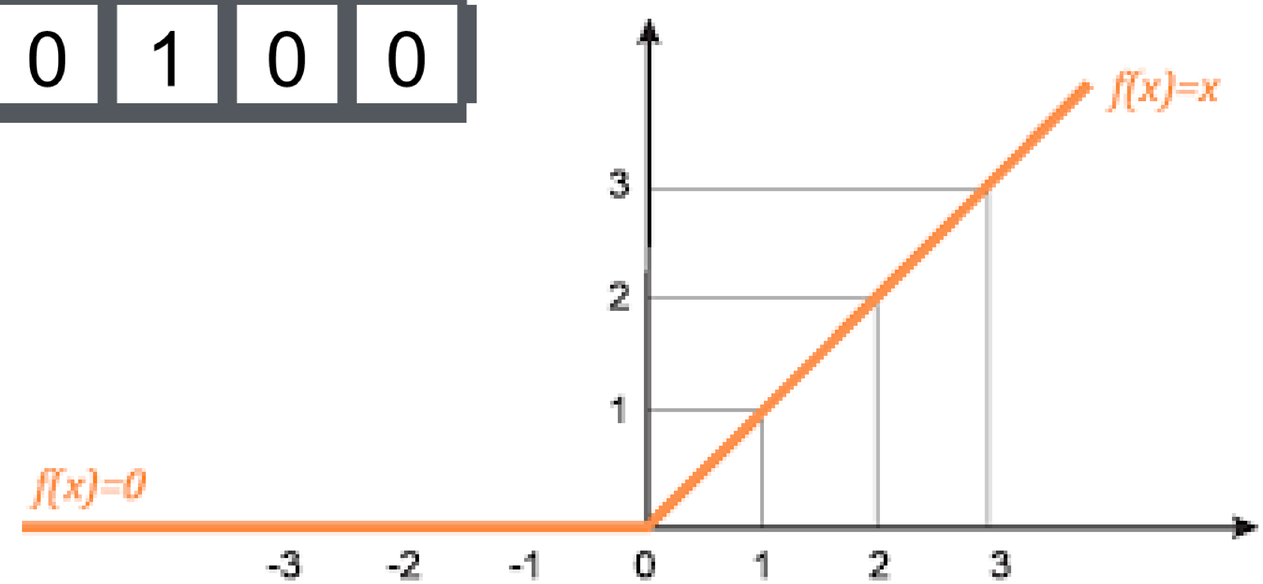
-1	1	-1

After convolution*:

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

After ReLU:

0	0	0	0	0	1	0	0



Convolutional layer: 1D example

A 1D image:

0	0	1	1	1	0	1	0	0	0

A filter:

-1	1	-1

After convolution*:

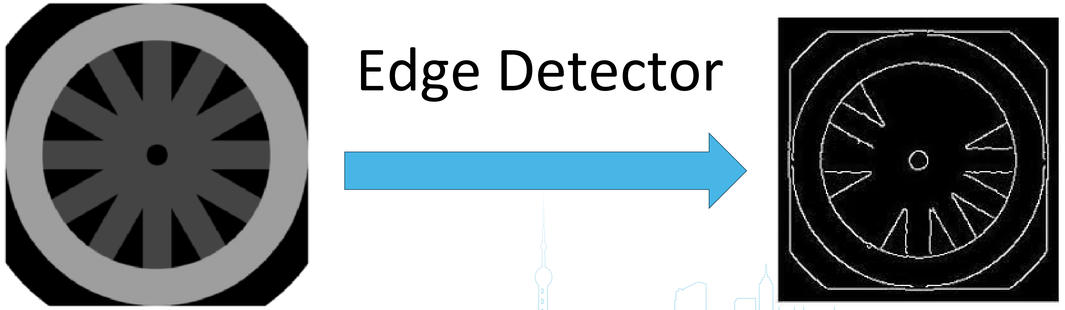
-1	0	-1	0	-2	1	-1	0

After ReLU:

0	0	0	0	0	1	0	0

What does the filter do?

The filter detects edges by computing the difference between neighboring values.



Convolutional layer: 1D example

A 1D image:

0	0	0	1	1	1	0	1	0	0	0	0

A filter:

-1	1	-1

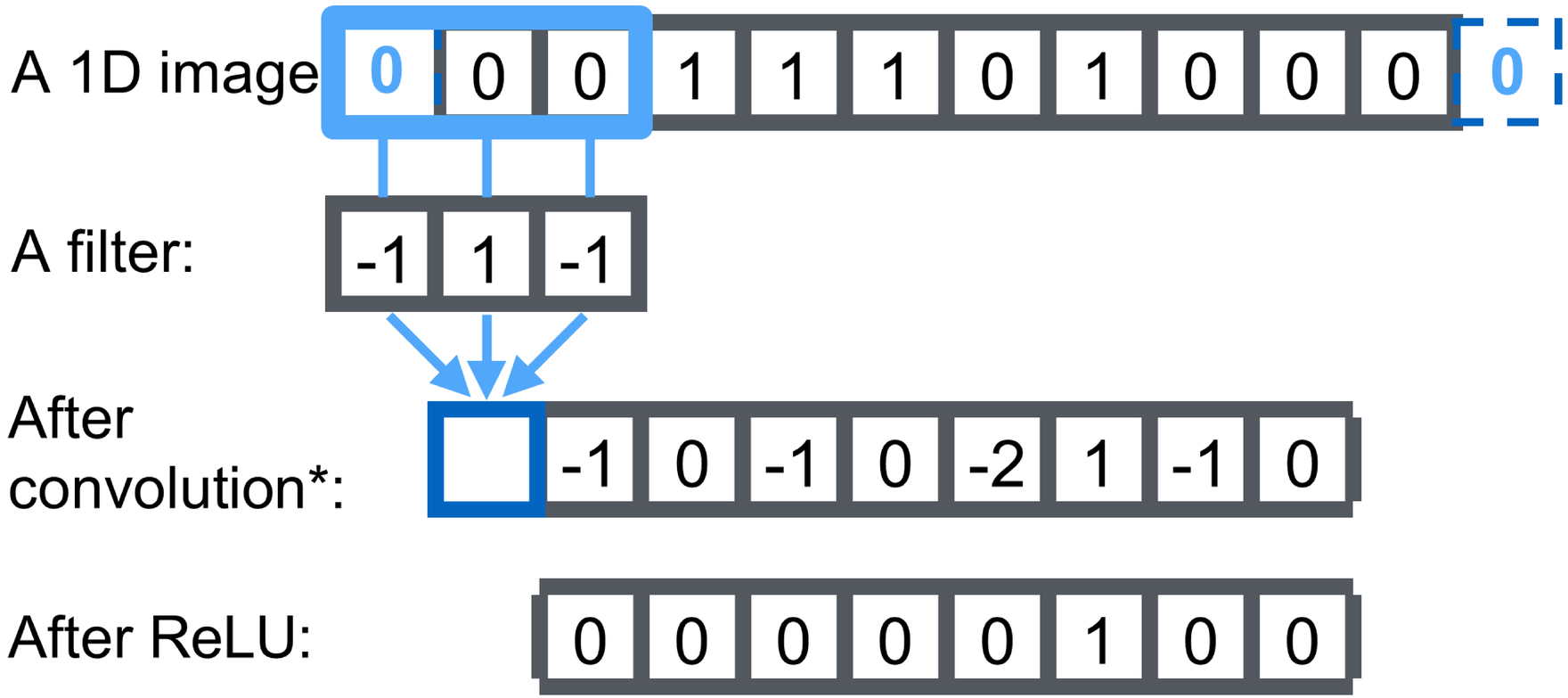
After convolution*:

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

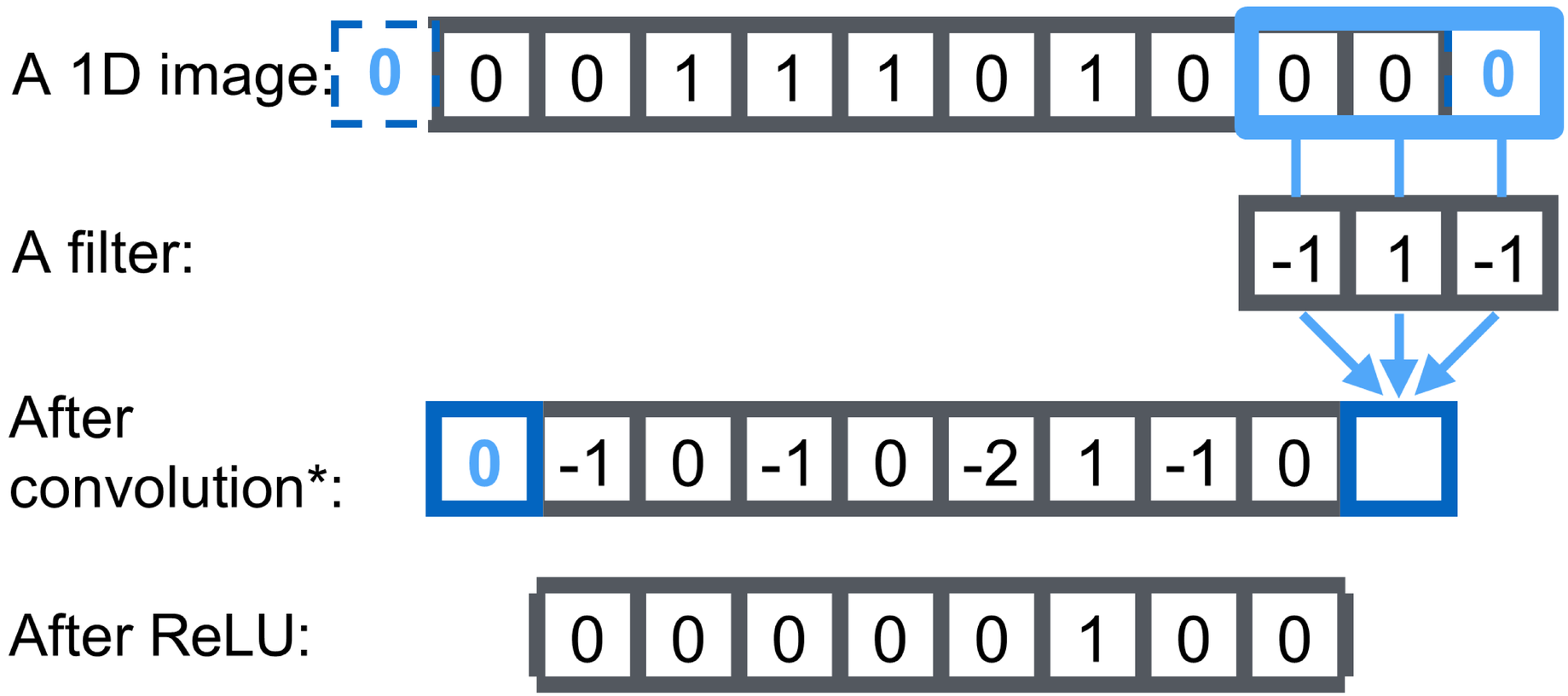
After ReLU:

0	0	0	0	0	1	0	0

Convolutional layer: 1D example



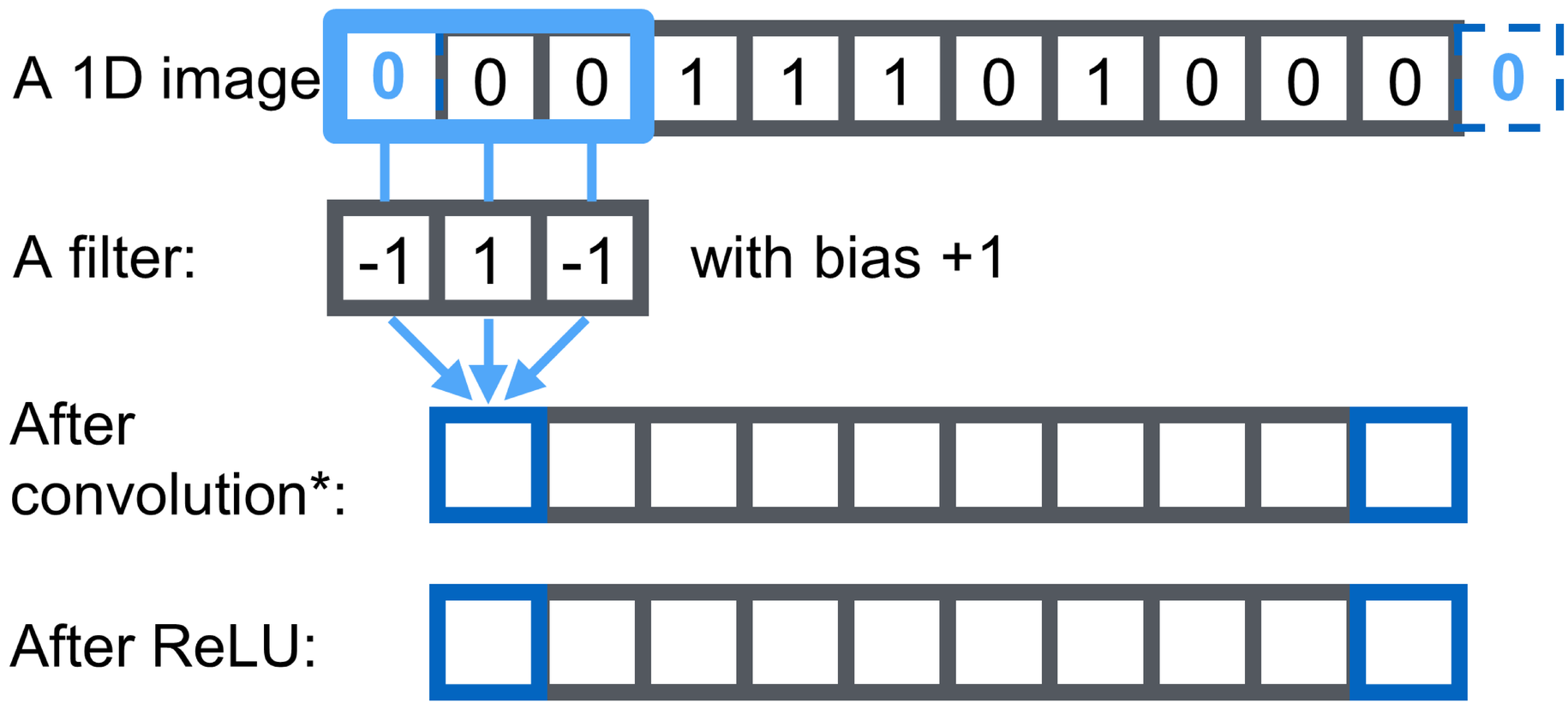
Convolutional layer: 1D example



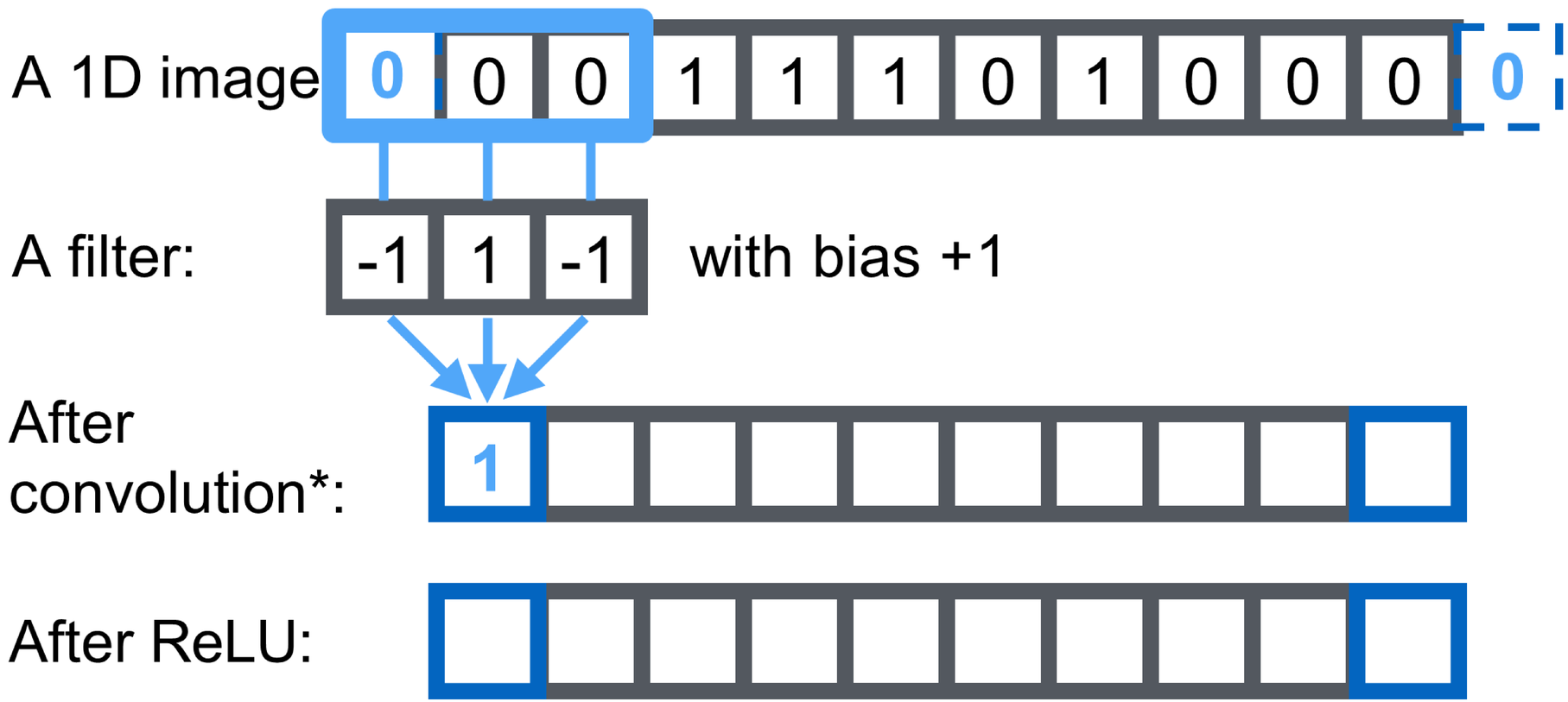
Convolutional layer: 1D example

A 1D image:	0	0	0	1	1	1	0	1	0	0	0	0
A filter:		-1	1	-1							
After convolution*:	0	-1	0	-1	0	-2	1	-1	0	0	
After ReLU:	0	0	0	0	0	0	1	0	0	0	

Convolutional layer: 1D example



Convolutional layer: 1D example



Convolutional layer: 1D example

A 1D image:

0	0	0	1	1	1	0	1	0	0	0

A filter:

-1	1	-1

with bias +1

After convolution*:

1	0	1	0	1	-1	2	0	1	1

After ReLU:

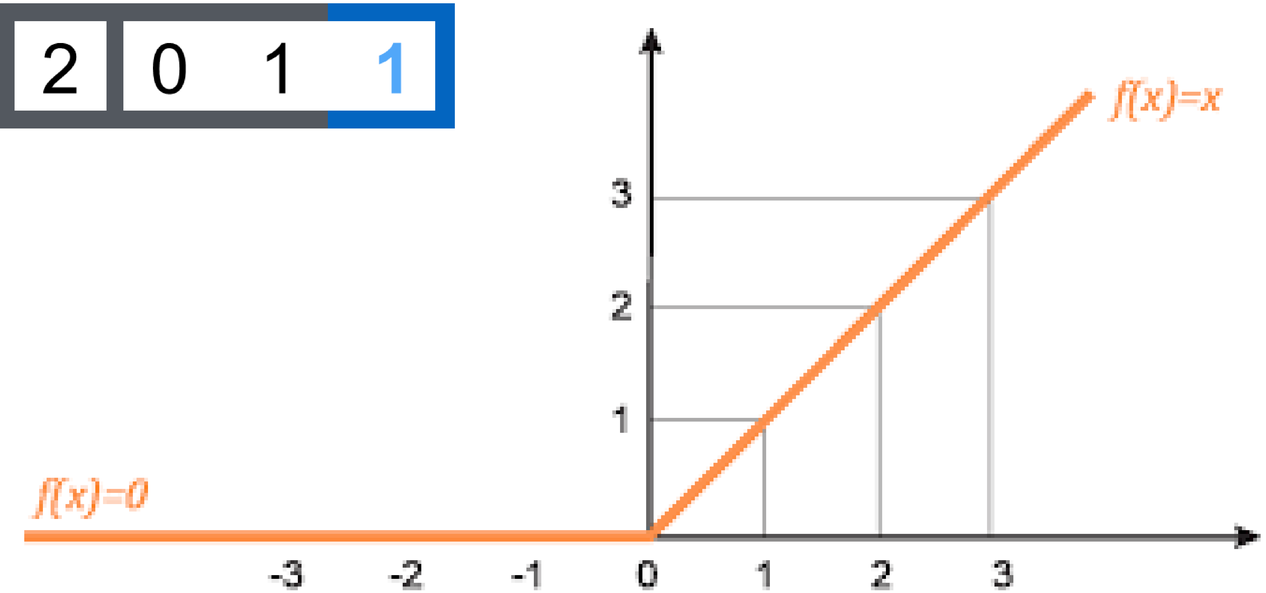
Convolutional layer: 1D example

A 1D image: 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0

A filter: -1 | 1 | -1 with bias +1

After convolution*: 1 | 0 | 1 | 0 | 1 | -1 | 2 | 0 | 1 | 1

After ReLU: 1 | 0 | 1 | 0 | 1 | 0 | 2 | 0 | 1 | 1



Convolutional layer: 1D example

A 1D image: 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0

A filter: w_1 | w_2 | w_3 with bias b

After convolution*:

After ReLU:

- How many weights (including bias)?

Convolutional layer: 1D example

A 1D image: 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0

A filter: w_1 | w_2 | w_3 with bias b

After convolution*:

After ReLU:

- How many weights (including bias)? 4

Convolutional layer: 1D example

A 1D image: 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0

A filter: w_1 | w_2 | w_3 with bias b

After convolution*:

After ReLU:

- How many weights (including bias)? 4
- How many weights (including biases) for fully connected layer with 10 inputs & 10 outputs?

Convolutional layer: 1D example

A 1D image: 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0

A filter: w_1 | w_2 | w_3 with bias b

After convolution*:

After ReLU:

- How many weights (including bias)? 4
- How many weights (including biases) for fully connected layer with 10 inputs & 10 outputs? 10 x 11 = 110

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution:

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

-1 + 0 + -1
+ -1 + 0 + -1
+ -1 + - 1 + -1
= -7

After convolution:

-7

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

-7	-2

After convolution:

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

-7	-2	

After convolution:

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution:

-7	-2	-4
-5		

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution:

-7	-2	-4
-5	-2	

Convolutional layer: 2D example

A 2D image:

1	0	1	0	0
1	0	1	0	1
1	1	1	0	0
1	0	1	0	1
1	0	1	0	1

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution:

-7	-2	-4
-5	-2	-5
-7	-2	-5

Convolutional layer: 2D example

A 2D image:

0	0	0	0	0	0	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	1	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution:

-7	-2	-4
-5	-2	-5
-7	-2	-5

Convolutional layer: 2D example

A 2D image:

0	0	0	0	0	0	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	1	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

0

After convolution:

-7	-2	-4
-5	-2	-5
-7	-2	-5

Convolutional layer: 2D example

A 2D image:

0			0	0	0	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	1	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution:

0	-4	
-7	-2	-4
-5	-2	-5
-7	-2	-5

Convolutional layer: 2D example

A 2D image:

0	0	0	0	0	0	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	1	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution:

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

Convolutional layer: 2D example

A 2D image:

0	0	0	0	0	0	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	1	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

A filter:

-1	-1	-1
-1	1	-1
-1	-1	-1

After convolution & ReLU:

0	0	0	0	0
0	0	0	0	1
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Convolutional layer: 2D example

A 2D image:

0	0	0	0	0	0	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	1	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

A filter:

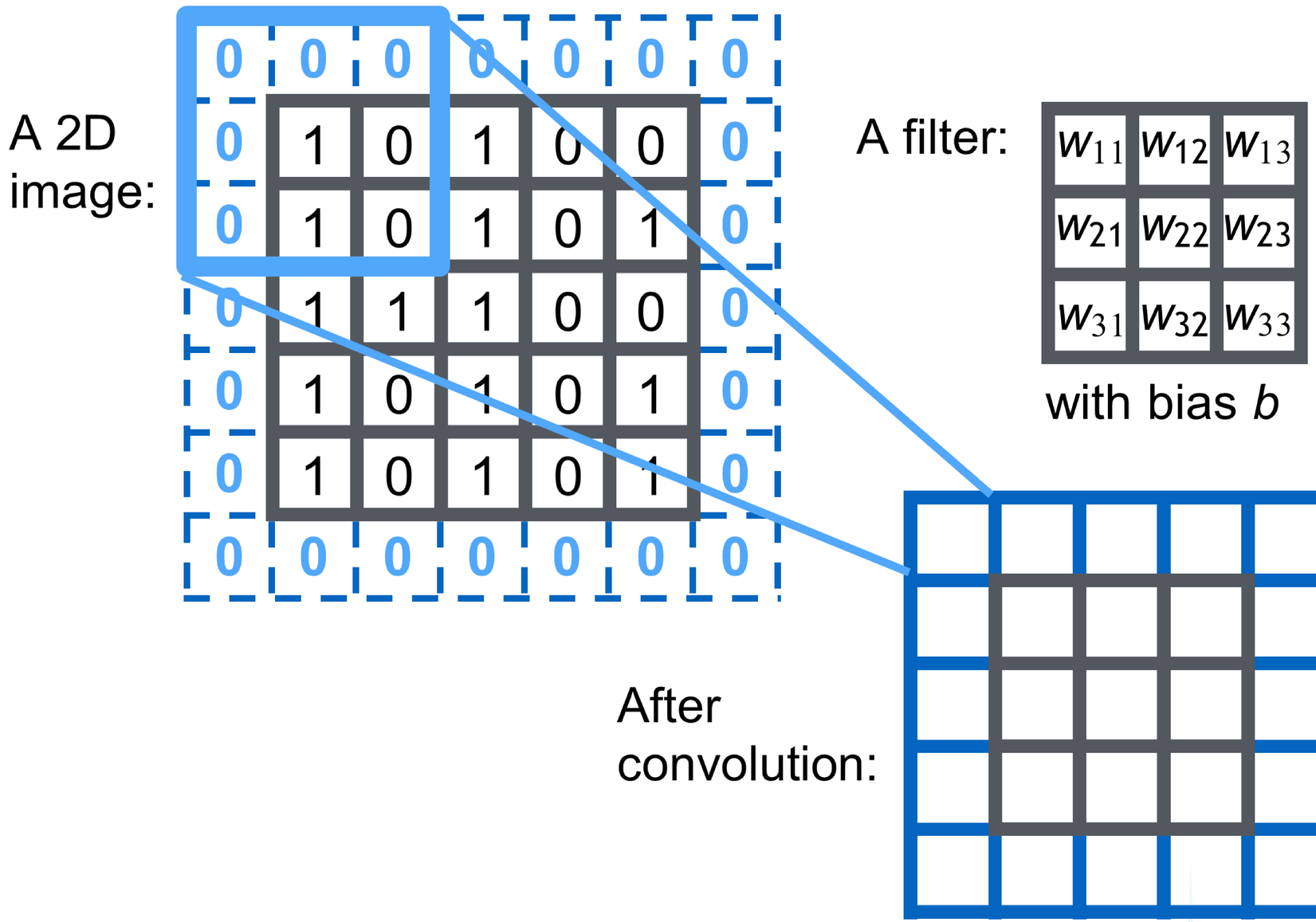
-1	-1	-1
-1	1	-1
-1	-1	-1

with bias 2

After convolution:

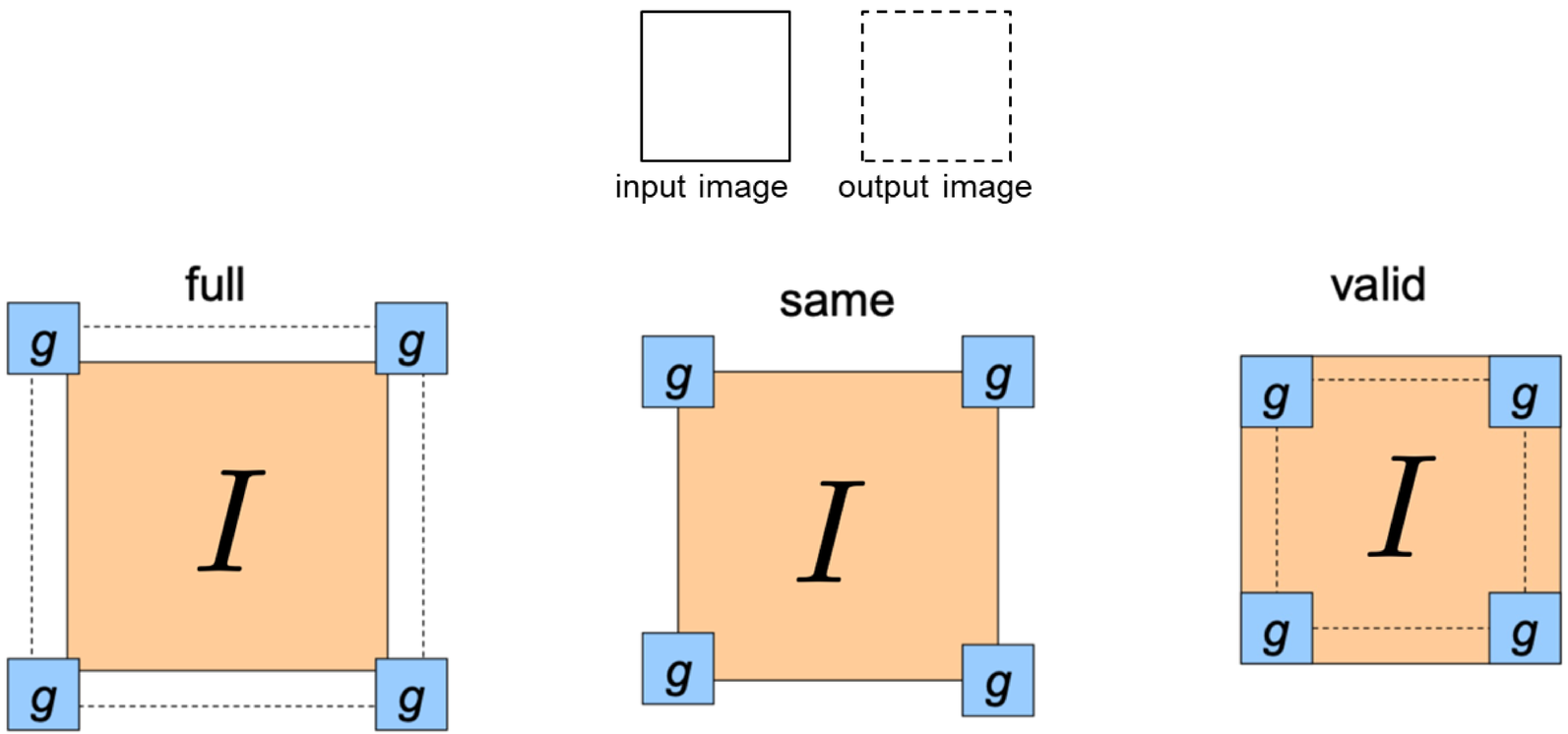
2	-2			

Convolutional layer: 2D example



Convolutional layer: padding

Design choice 1: How big to pad



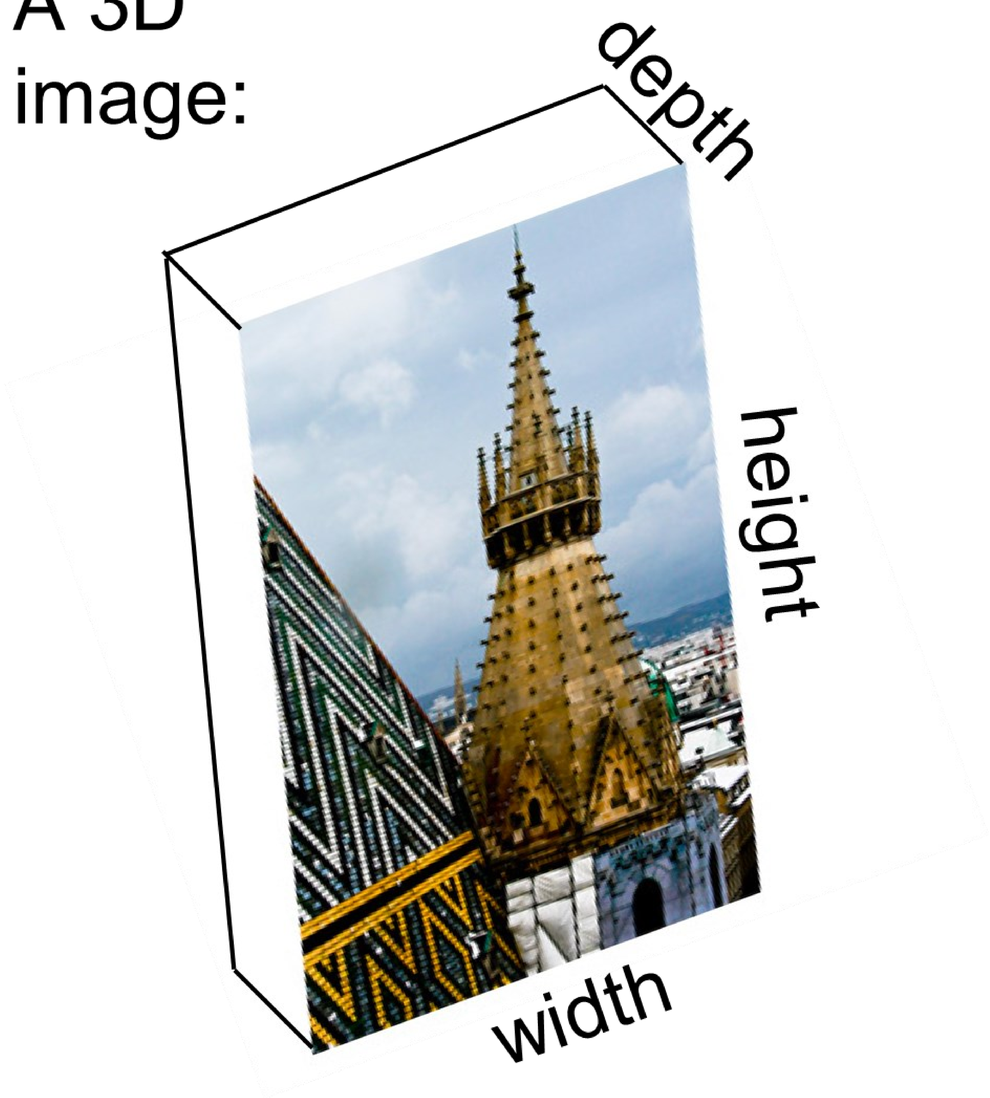
Pad: psz-1
Input: N
Output: N+psz-1

Pad: (psz-1)/2
Input: N
Output: N

Pad: 0
Input: N
Output: N-psz+1

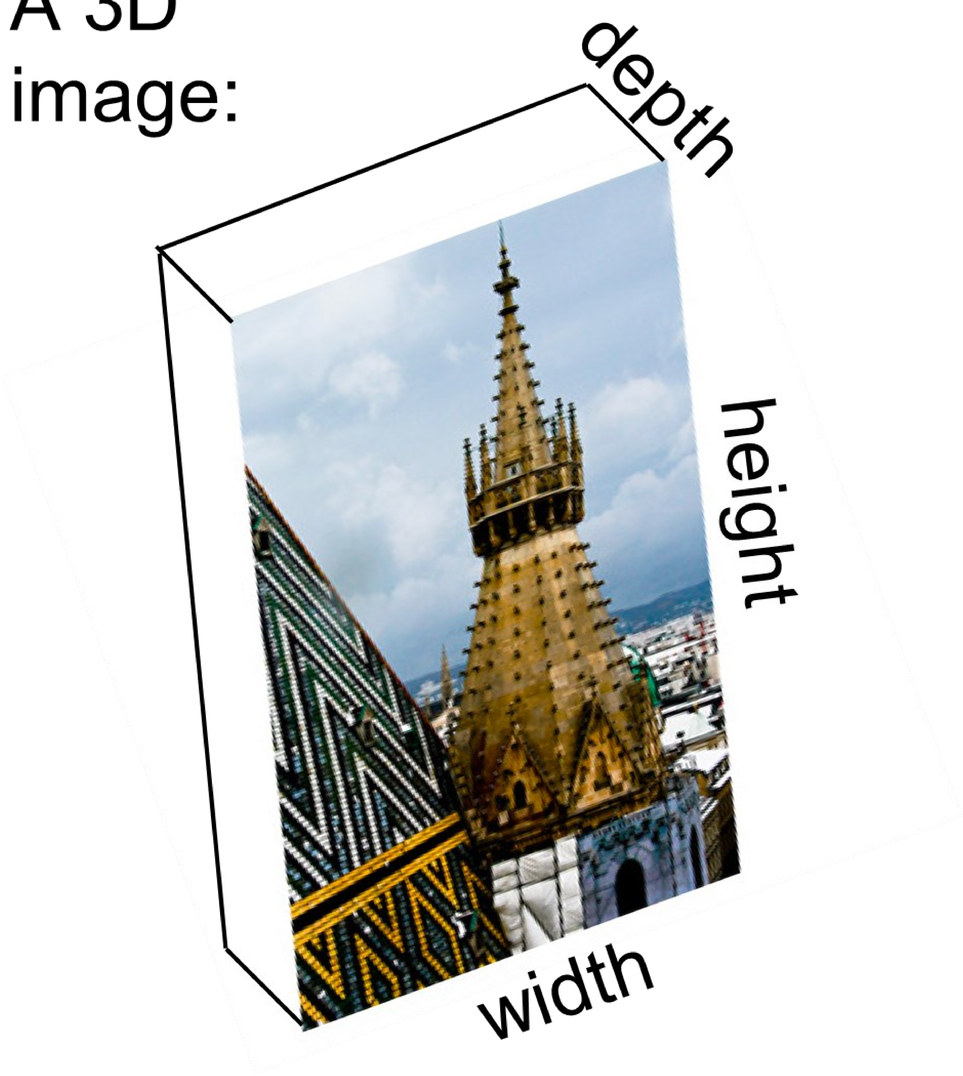
Convolutional layer: 3D example

A 3D image:

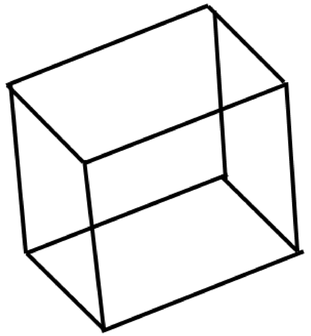


Convolutional layer: 3D example

A 3D image:

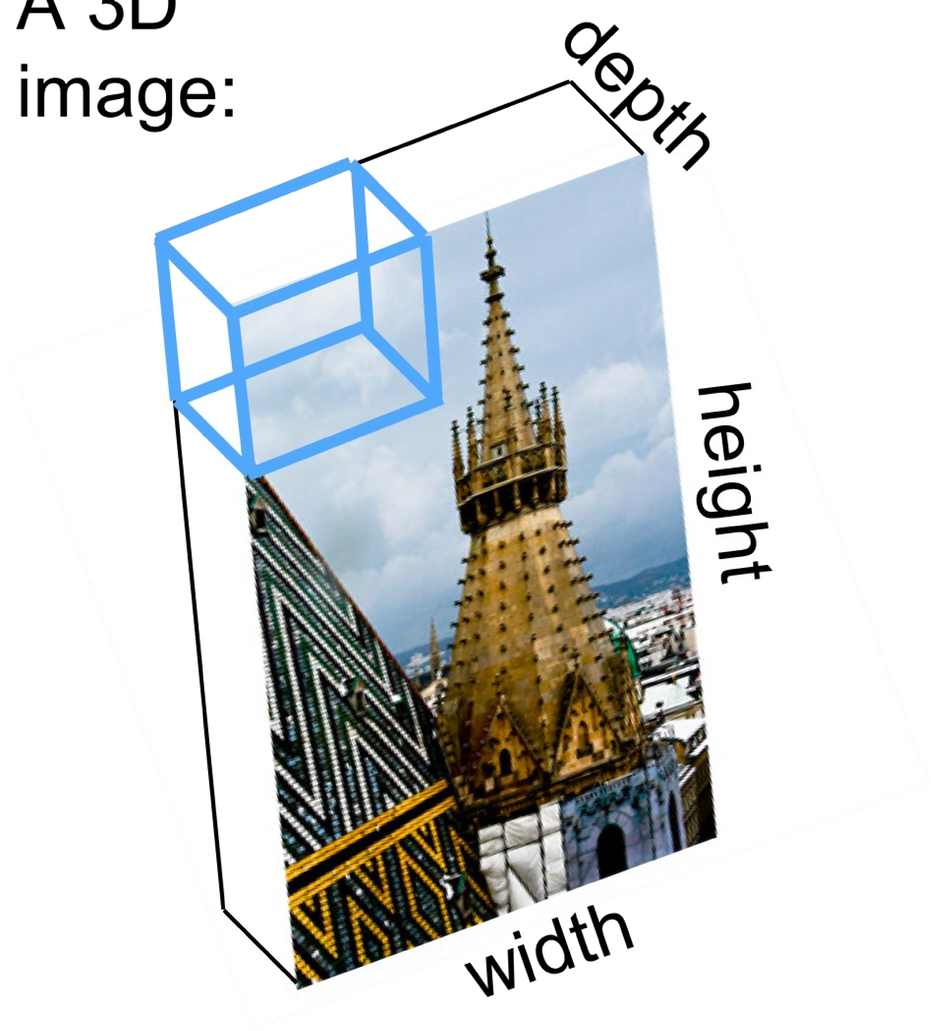


A filter:

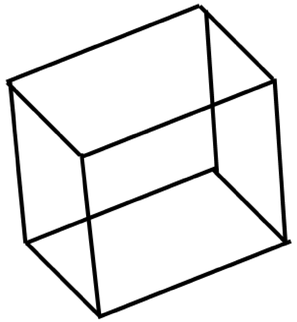


Convolutional layer: 3D example

A 3D image:

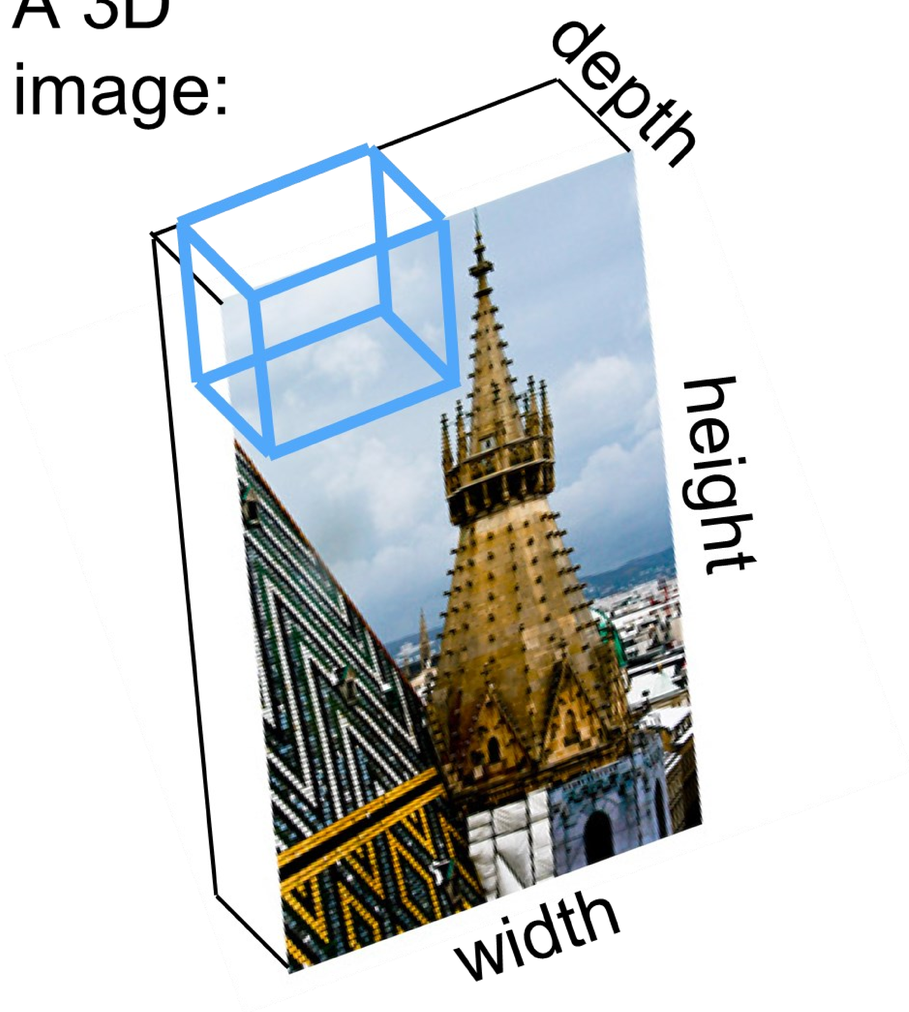


A filter:

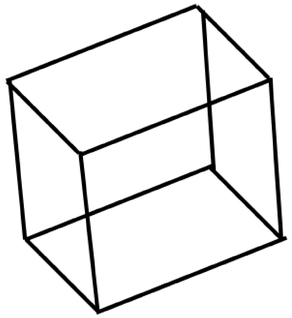


Convolutional layer: 3D example

A 3D image:

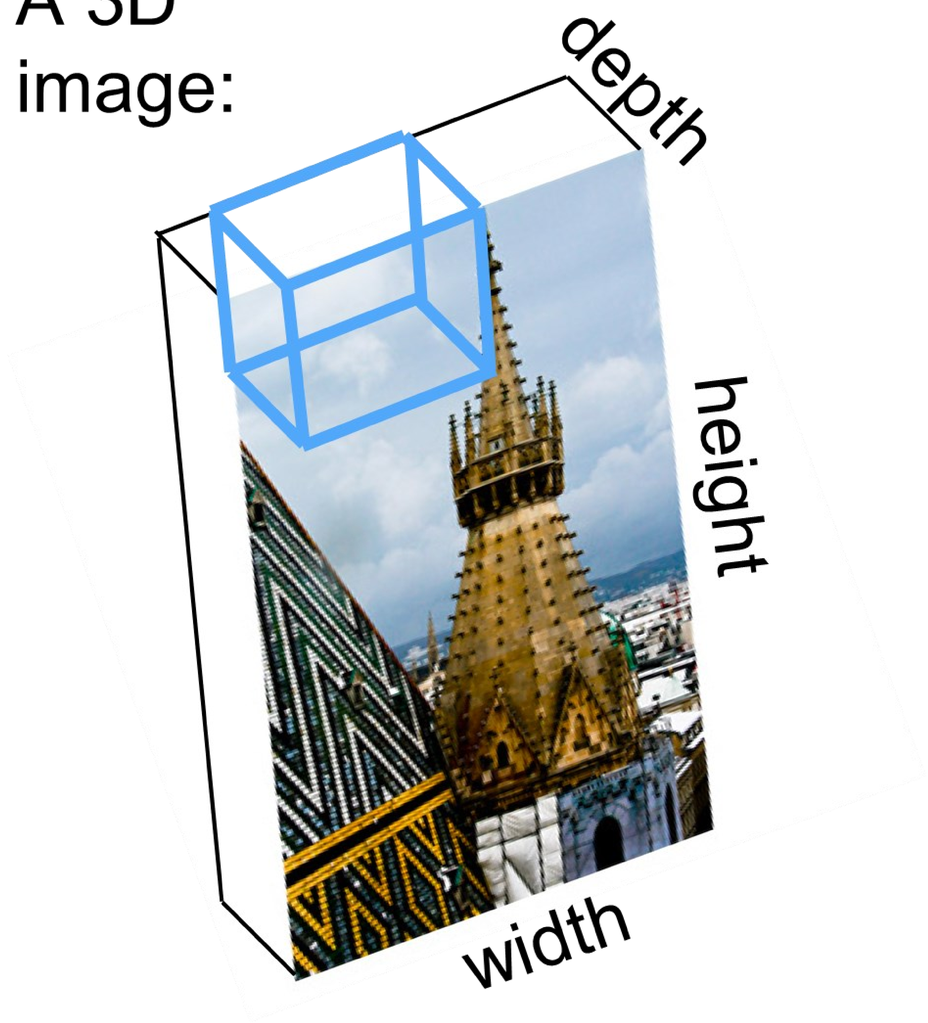


A filter:

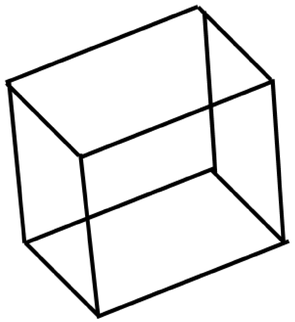


Convolutional layer: 3D example

A 3D image:

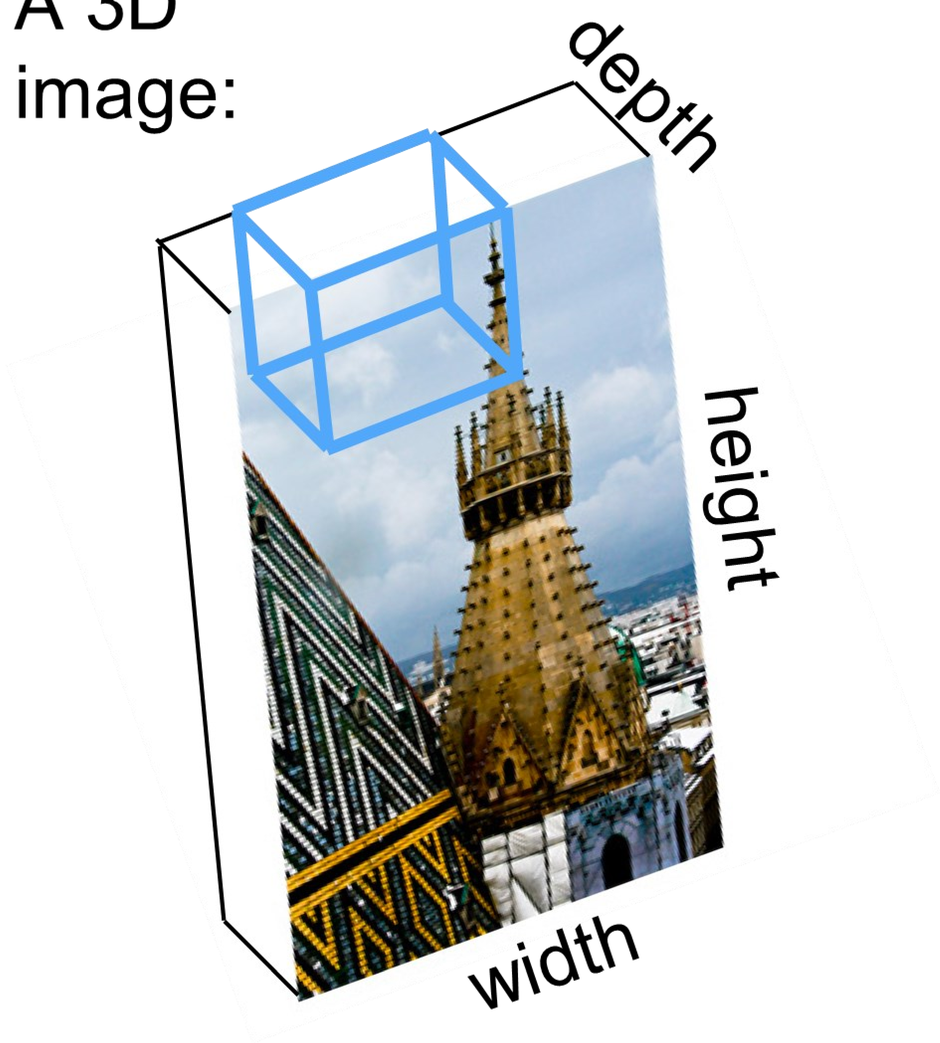


A filter:

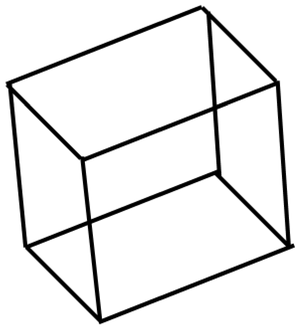


Convolutional layer: 3D example

A 3D image:

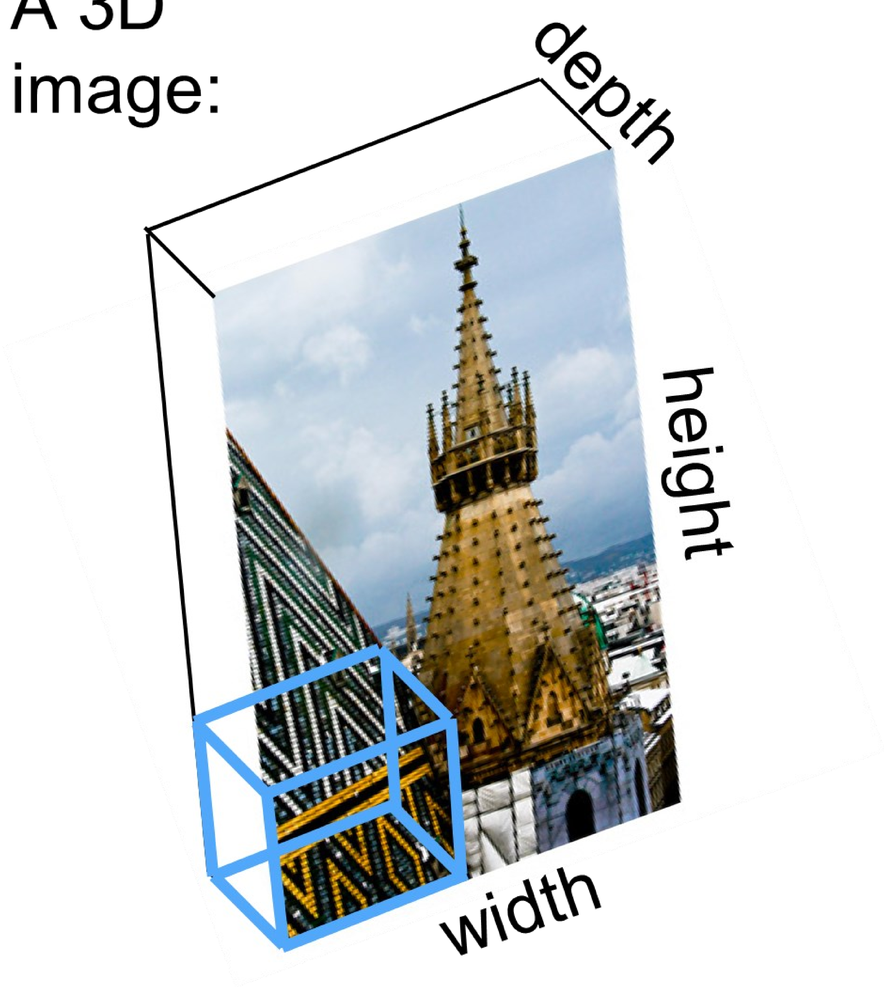


A filter:

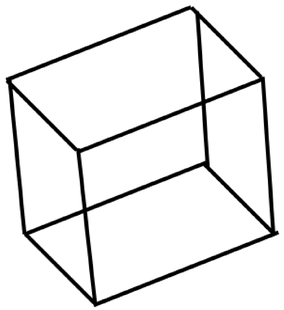


Convolutional layer: 3D example

A 3D image:

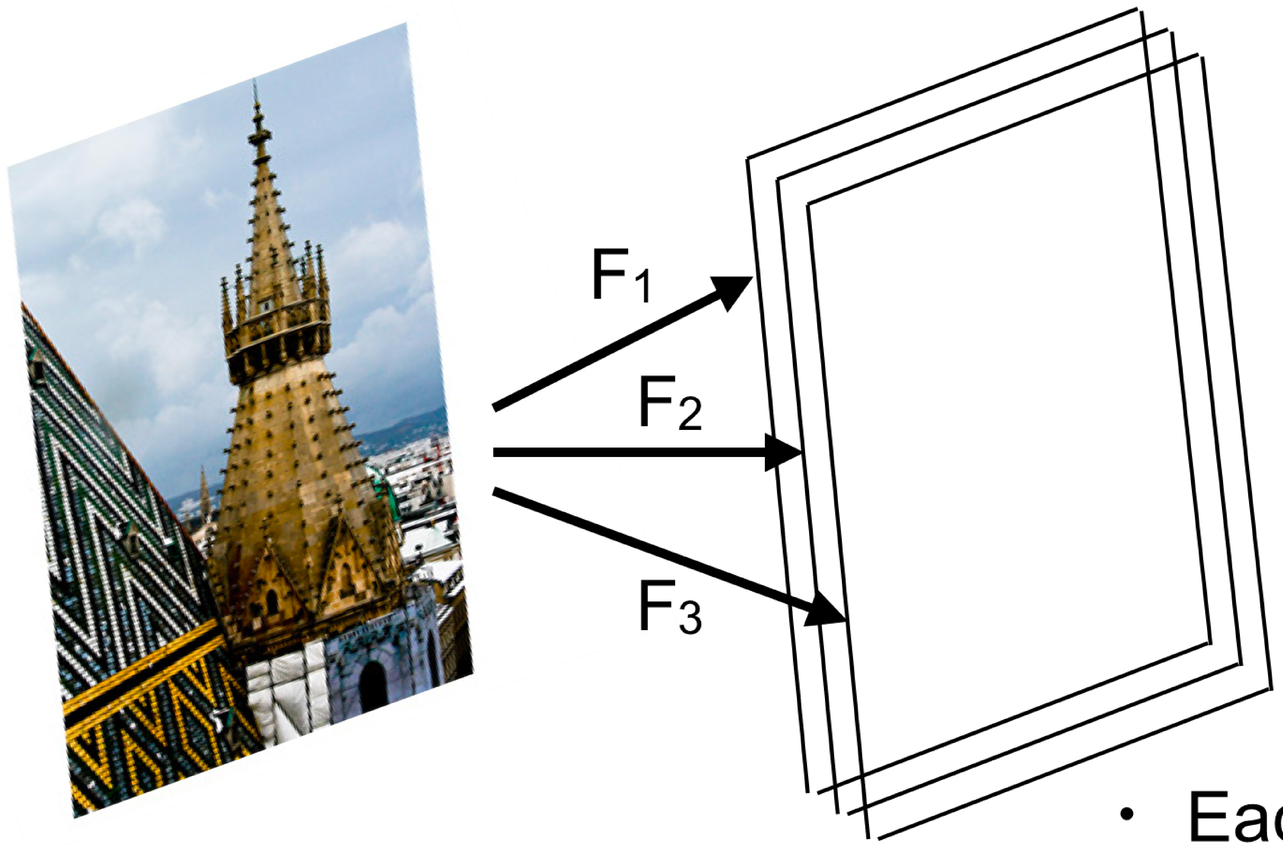


A filter:



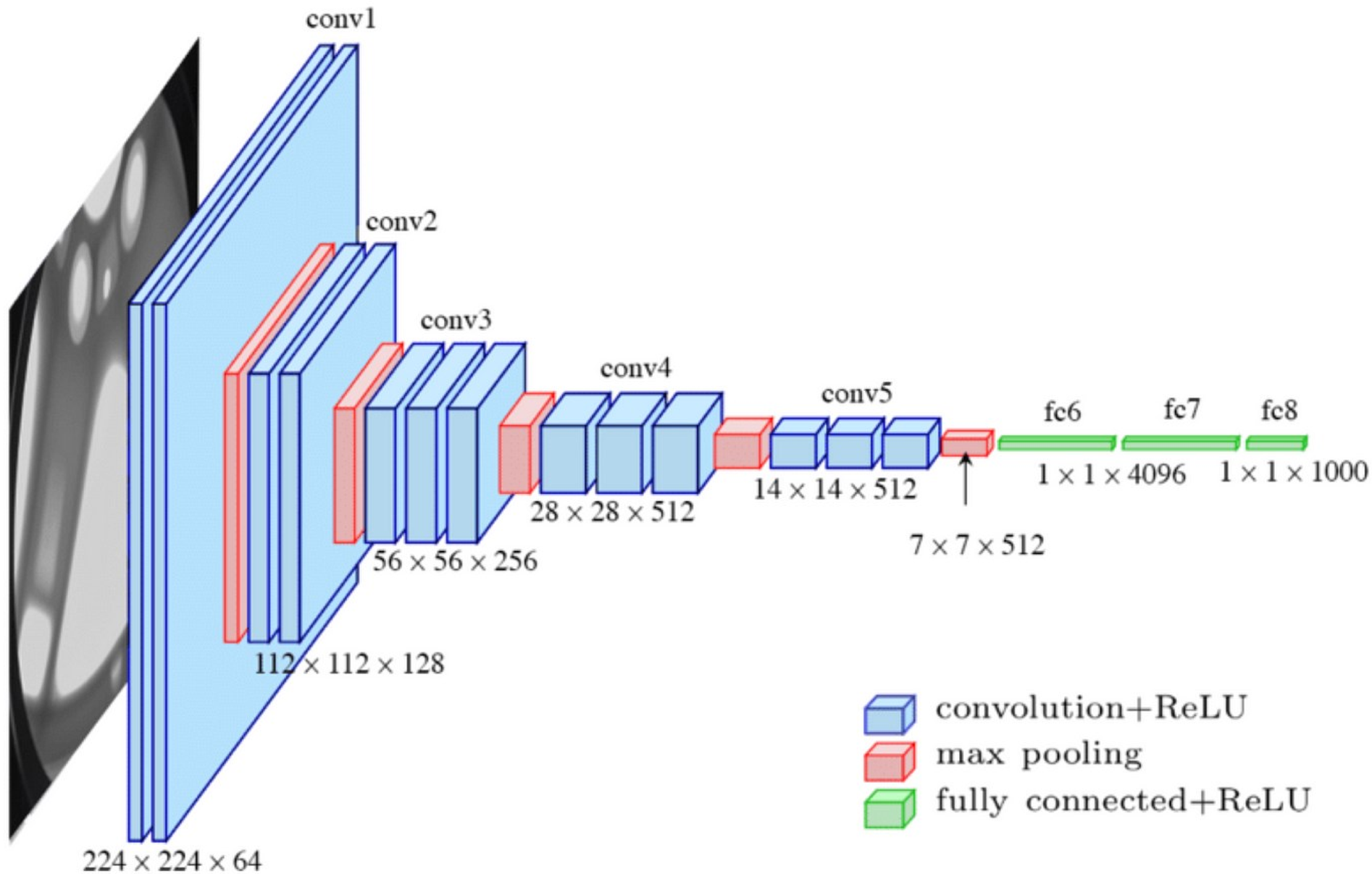
Convolutional layer: multiple filters

An image:



- Collection of filters in the layer: *filter bank*
- Each resulting image is a *channel*

Max pooling layer: 2D example



Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

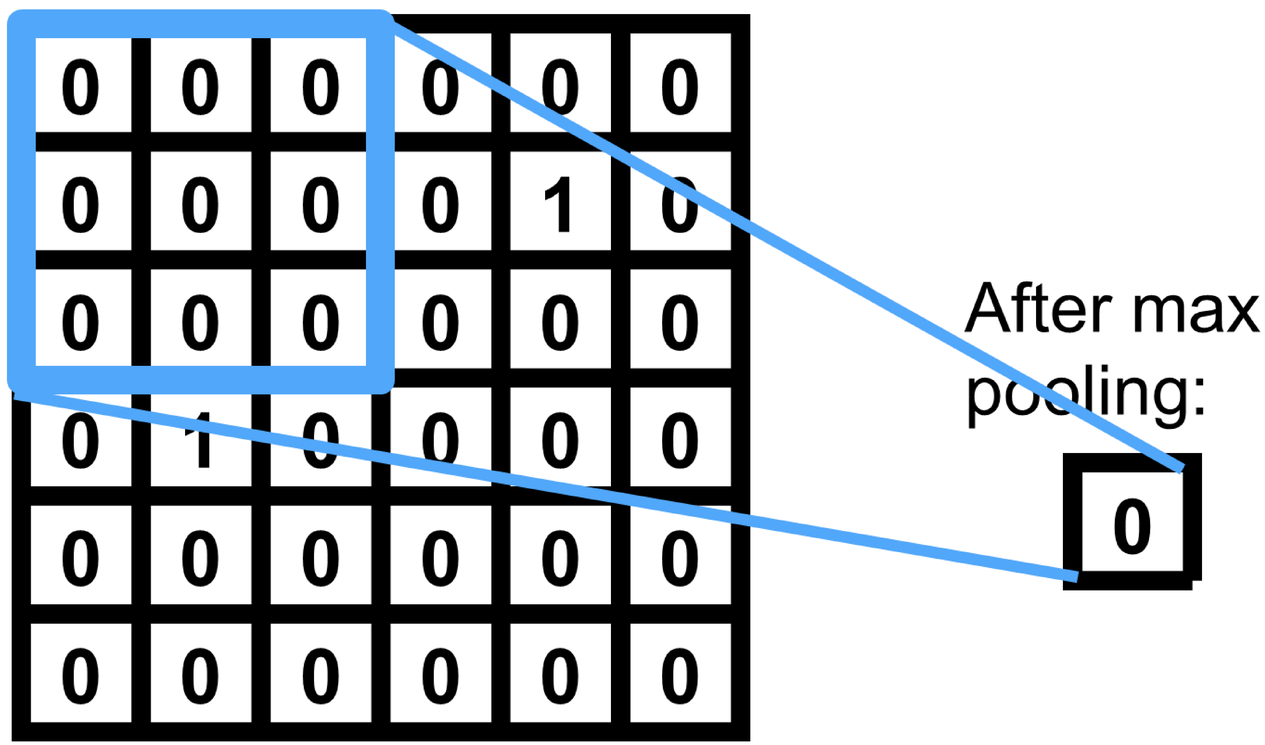
Max pooling: returns max of its arguments

- size 3x3 (“size 3”)

After max pooling:

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:



Max pooling: returns max of its arguments

- size 3x3 (“size 3”)

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

- size 3x3 (“size 3”)

After max pooling:

0	0	1	1

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

- size 3x3 (“size 3”)

After max pooling:

0	0	1	1
1	1	1	1
1	1	0	0
1	1	0	0

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

- size 3x3 (“size 3”)

After max pooling:

0	0	1	1
1	1	1	1
1	1	0	0
1	1	0	0

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

- size 3x3 (“size 3”)
- stride 3

After max pooling:

0

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

- size 3x3 (“size 3”)
- stride 3

After max pooling:

0	

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

- size 3x3 (“size 3”)
- stride 3

After max pooling:

0	1
1	

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

- size 3x3 (“size 3”)
- stride 3

After max pooling:

0	1
1	0

Max pooling layer: 2D example

Output from the convolutional layer & ReLU:

0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Max pooling: returns max of its arguments

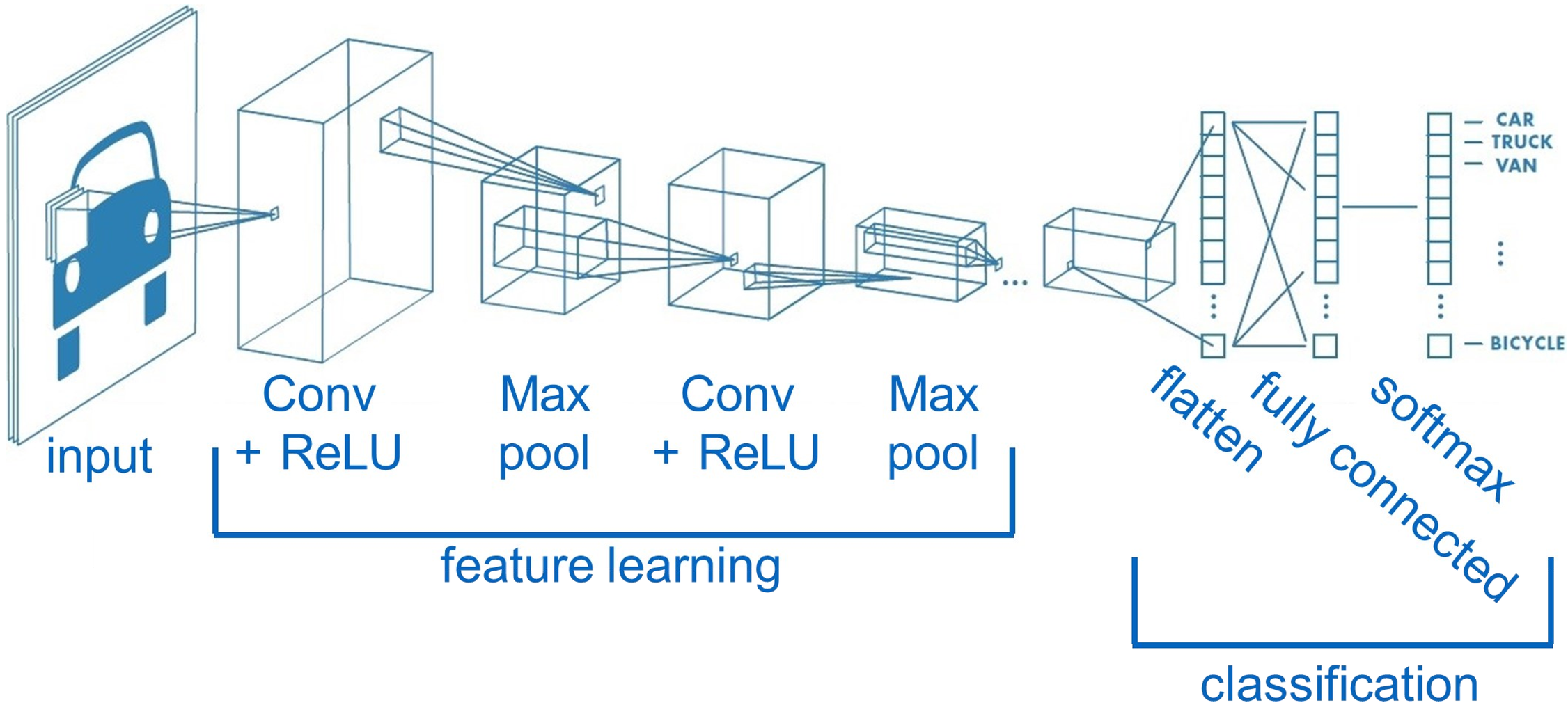
- size 3x3 (“size 3”)
- stride 3

After max pooling:

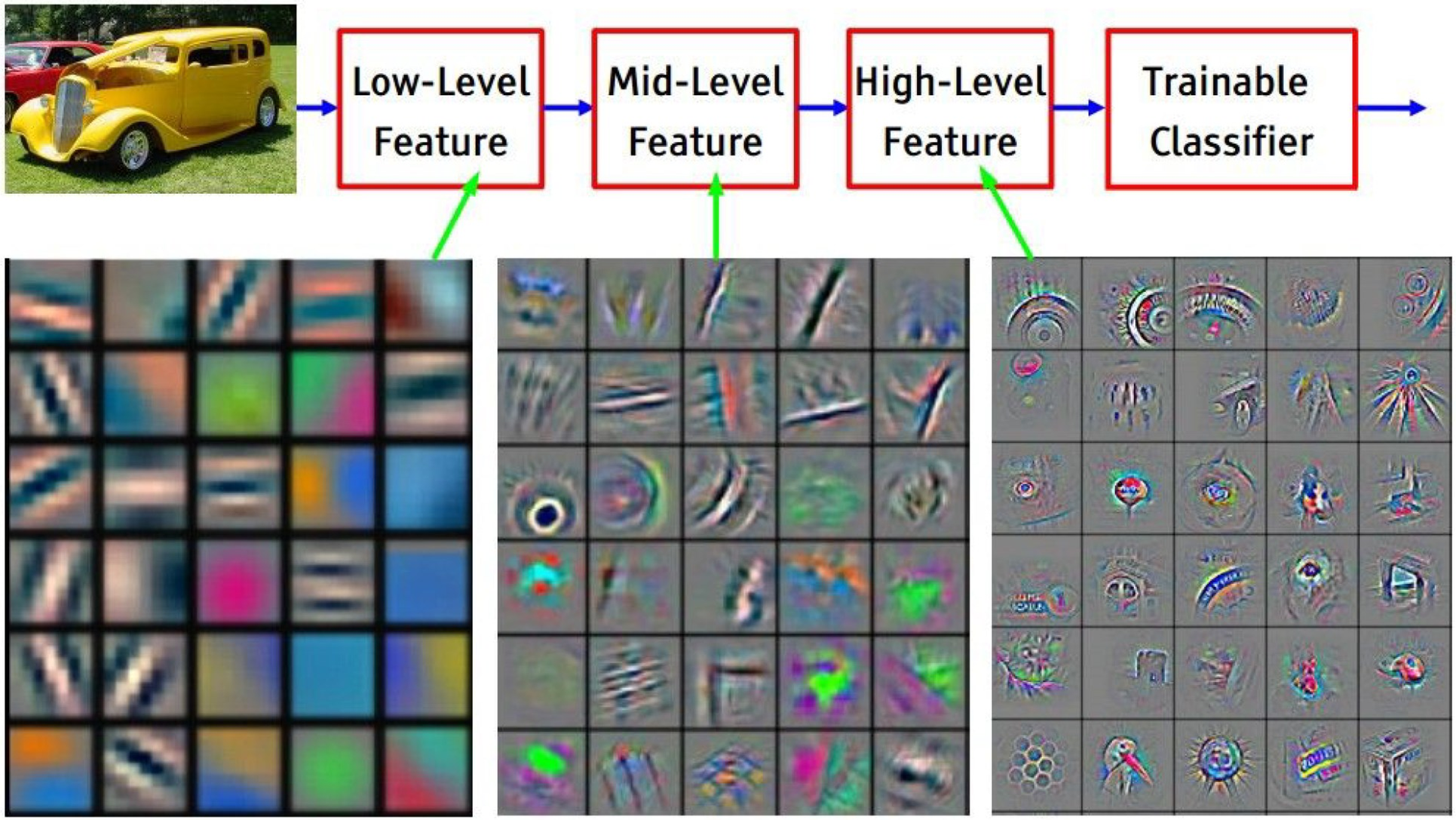
0	1
1	0

- Can use stride with filters too
- No weights in max pooling

CNNs: typical architecture

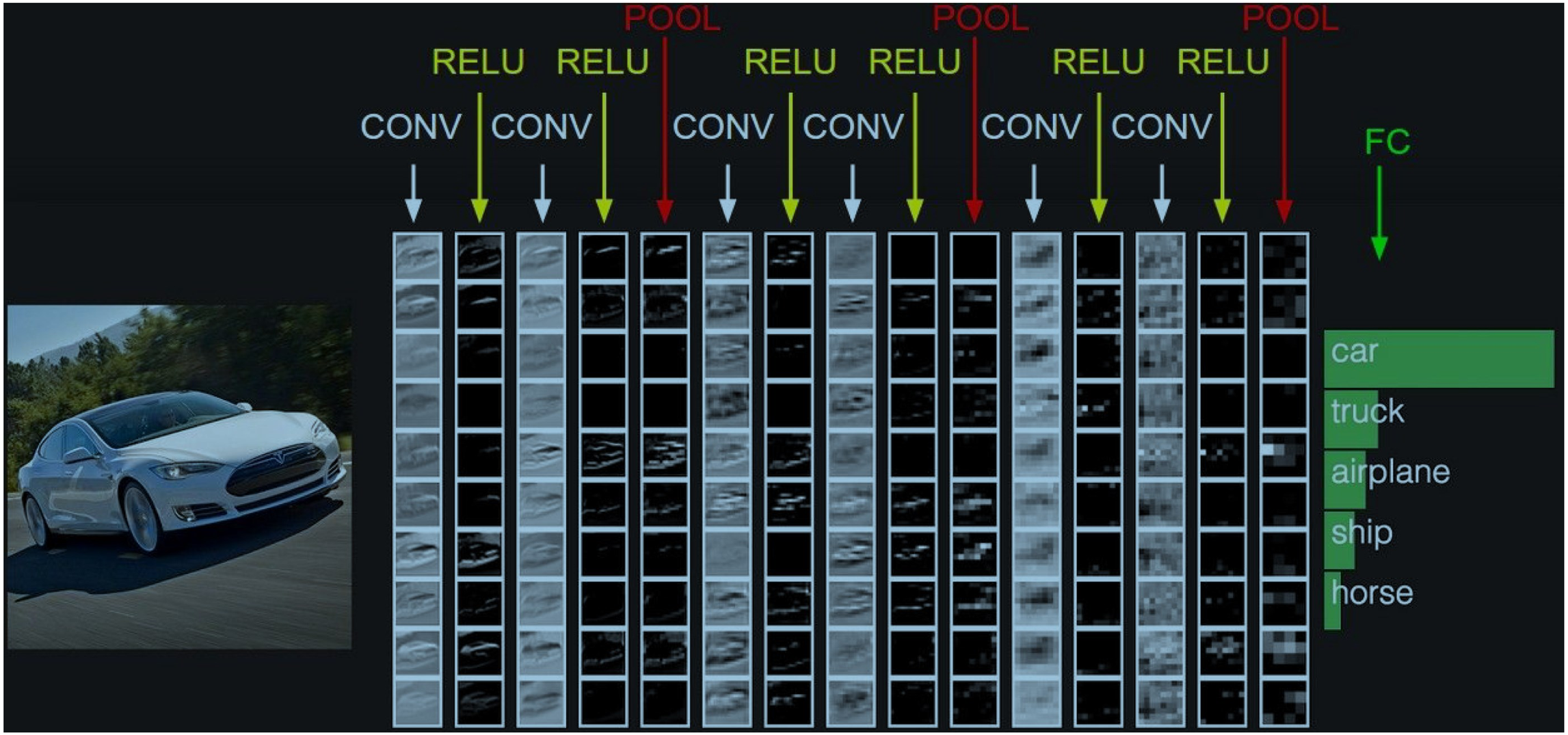


CNNs: feature learning



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

CNNs: feature learning



Feature Visualization: https://poloclub.github.io/cnn-explainer/

Outline: Image Data Mining

1. Introduction
2. Digital Image Processing
3. Convolutional Neural Network
4. Computer Vision Tasks

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for **So Many Tasks**!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for **So Many Tasks**!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…

e.g., take a picture of an object and find where to buy it

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for **So Many Tasks**!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…

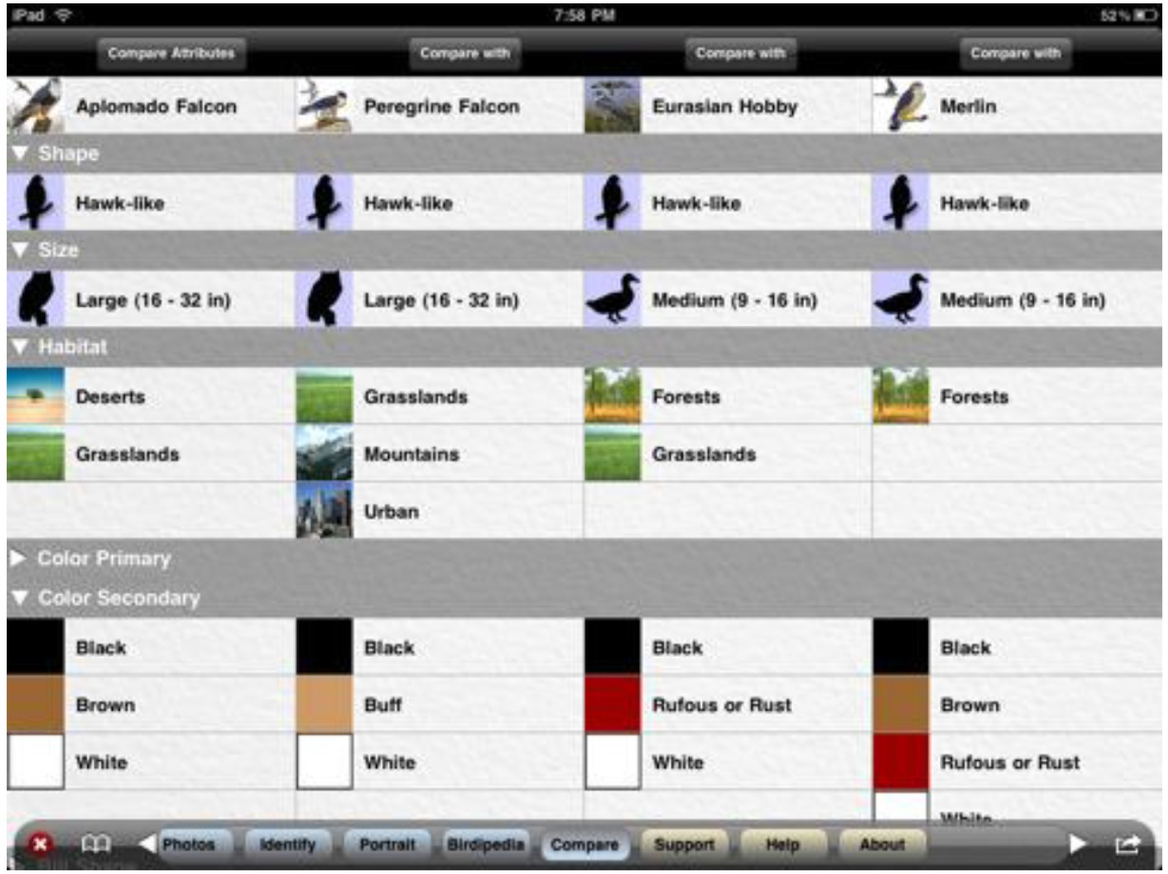
Kitchen

Store

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for **So Many Tasks**!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…



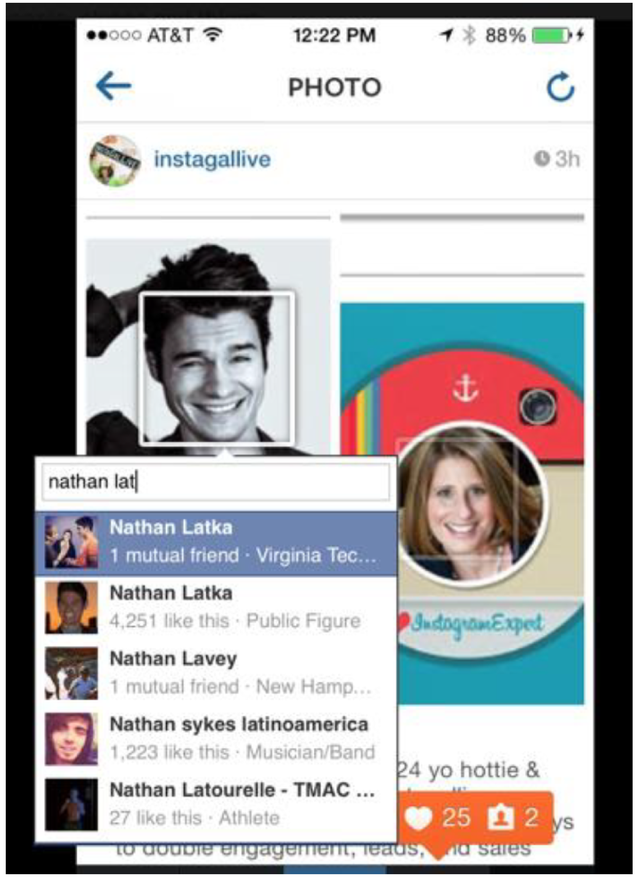
e.g., describe a bird to learn what type it is

Demo: https://www.youtube.com/watch?v=UPcz9Y17iCc

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for **So Many Tasks**!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…

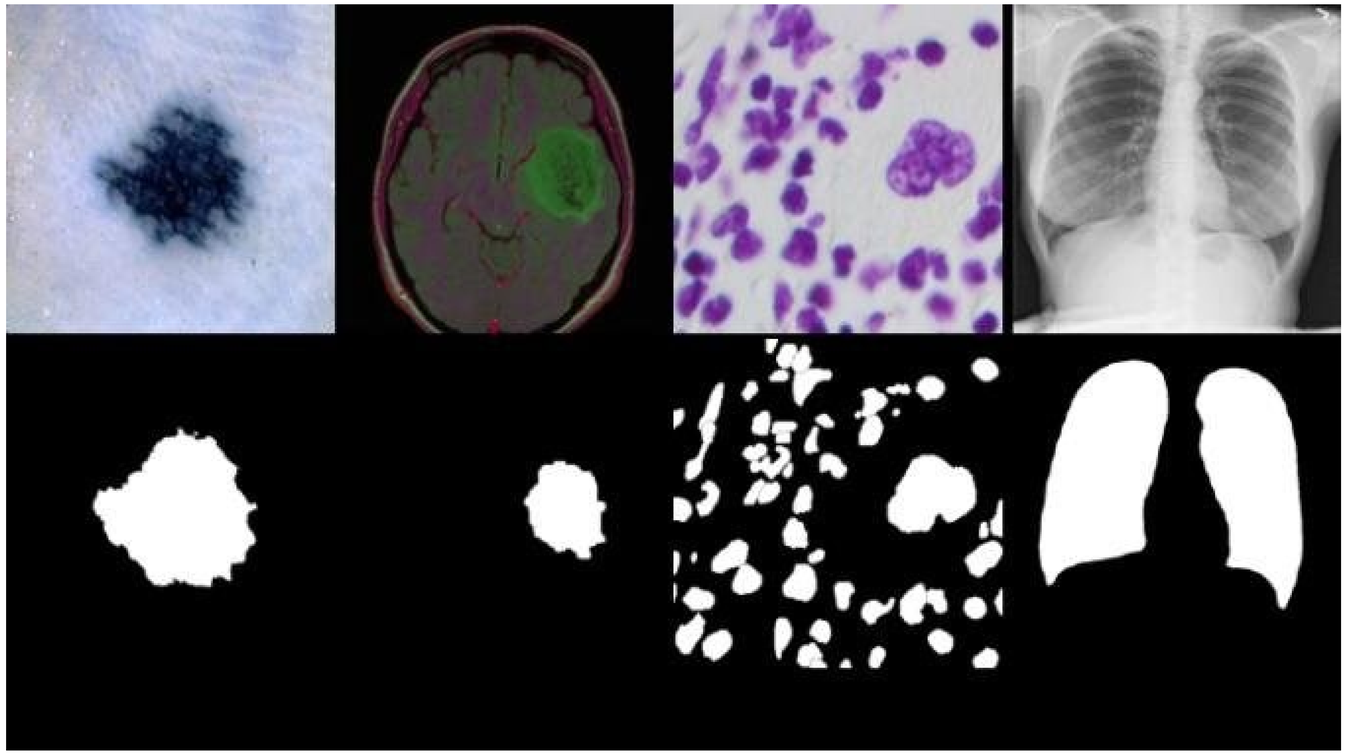


e.g., detect faces to tag

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for So Many Tasks!

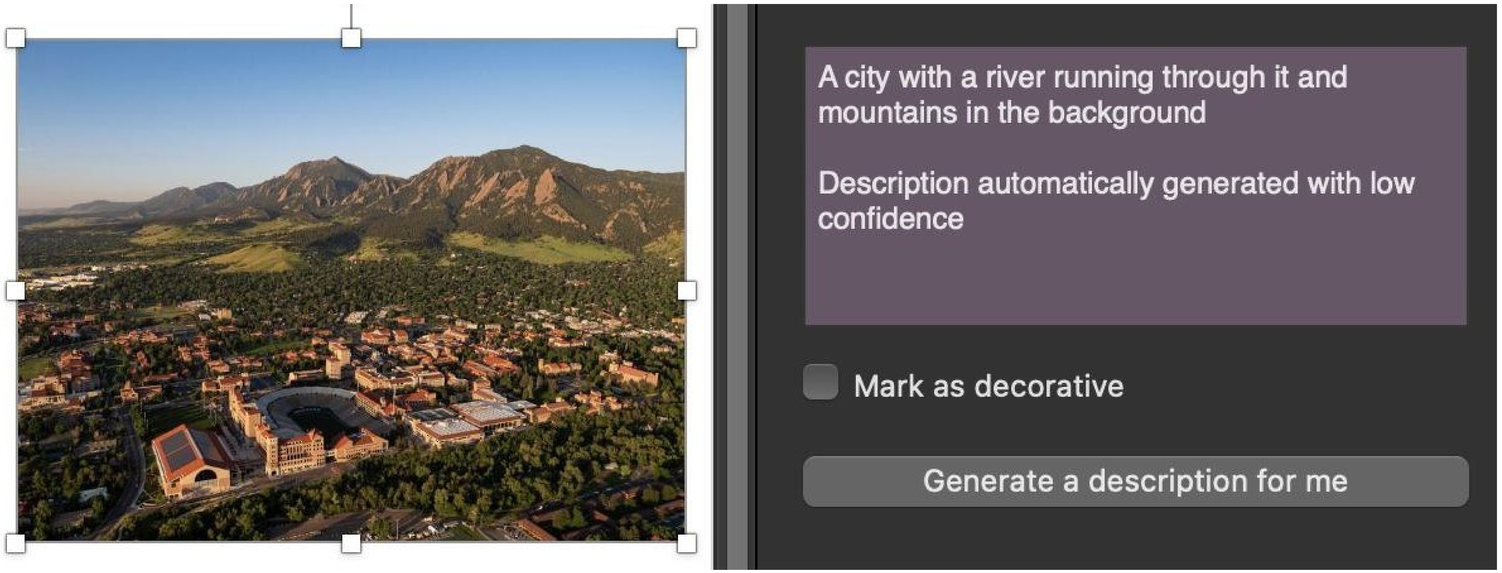
- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…



Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for So Many Tasks!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…

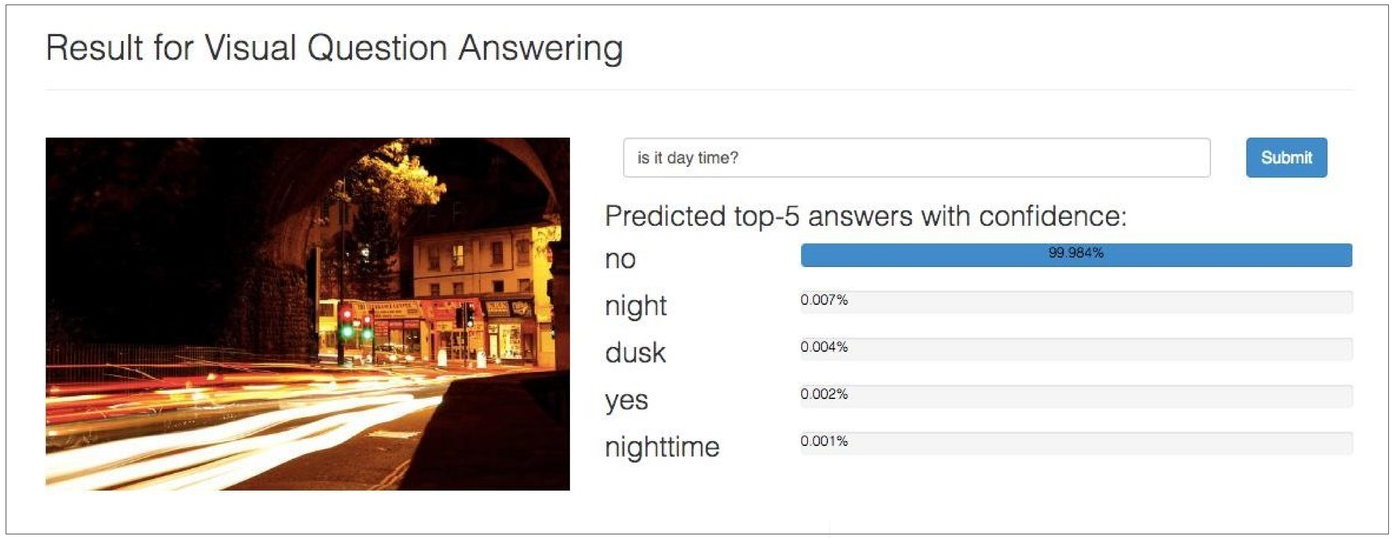


e.g., Microsoft Power Point

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for So Many Tasks!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…



Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for So Many Tasks!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…

e.g., shopping without a cashier

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for So Many Tasks!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…

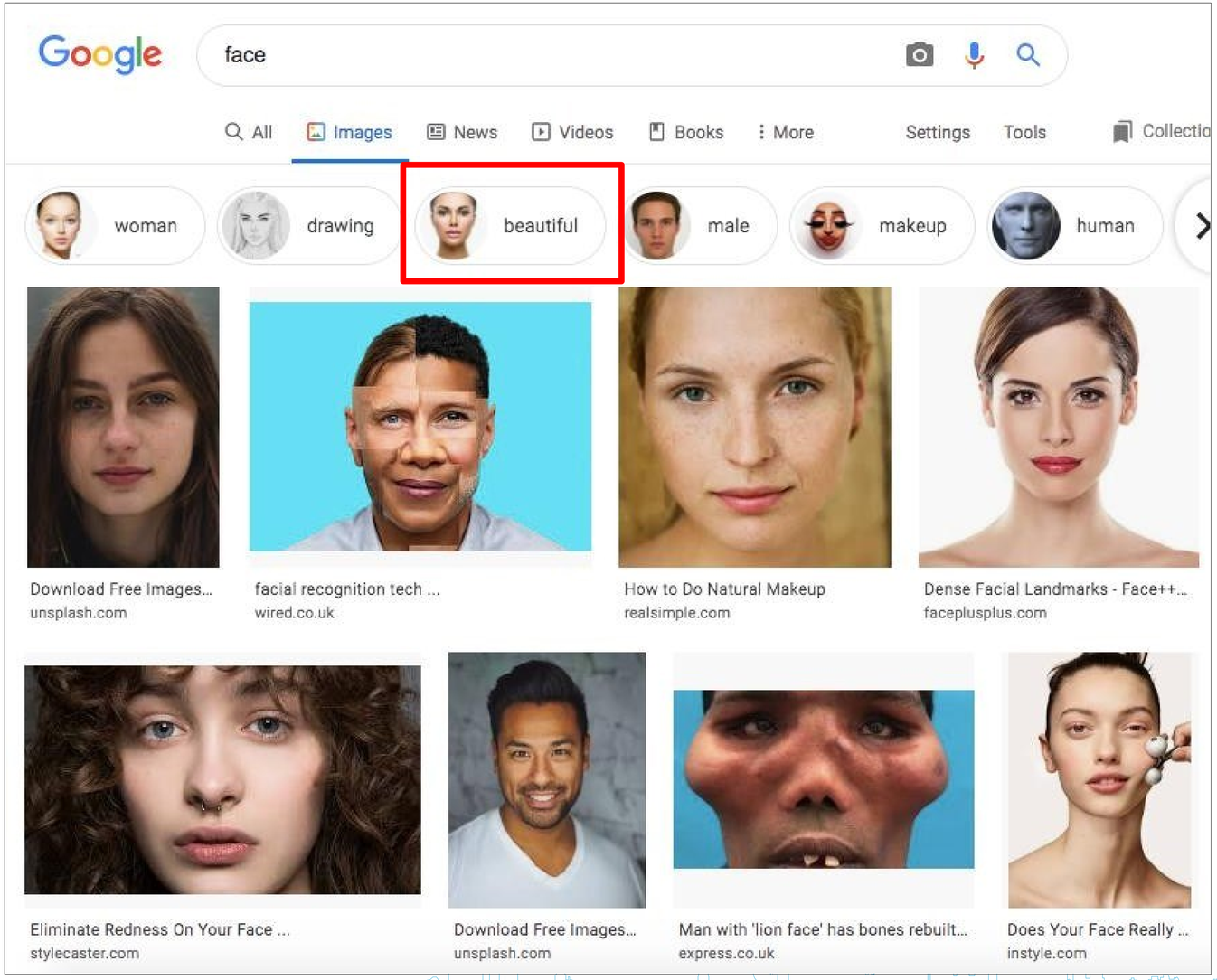
e.g., track bowling ball path

e.g., calculate bat speed

Computer vision tasks

Key Challenge: Replicate Human Vision for So Much Variation for So Many Tasks!

- Object recognition
- Scene classification
- Attribute labeling
- Object detection
- Segmentation
- Image Captioning
- Visual Question Answering
- Activity/Event Recognition
- Object Tracking
- Subjective Problems
- And more…



Email: min.shi@louisiana.edu