MachineLearnAthon –Microlecture Computer Vision

MachineLearnAthon
A project Co-funded by the Erasmus+ programme of the European Union

13.03.2024

Learning outcomes of today

After successfully completing this lecture, you will be able to…

- Explain how image data is “seen” by machines
- Identify challenges in applying computer vision to real world applications
- Explain the concept of feature in computer vision
- Explain standard Convolutional Neural Network (CNN) architectures

Agenda

- What is computer vision?
- What do machines see?
- Important concepts in computer vision
  - Filter
  - Padding
  - Stride
  - Pooling
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- Outlook on today’s assignment

Agenda

- **What is computer vision?**
- What do machines see?
- Important concepts in computer vision
  - Filter
  - Padding
  - Stride
  - Pooling
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- Outlook on today's assignment

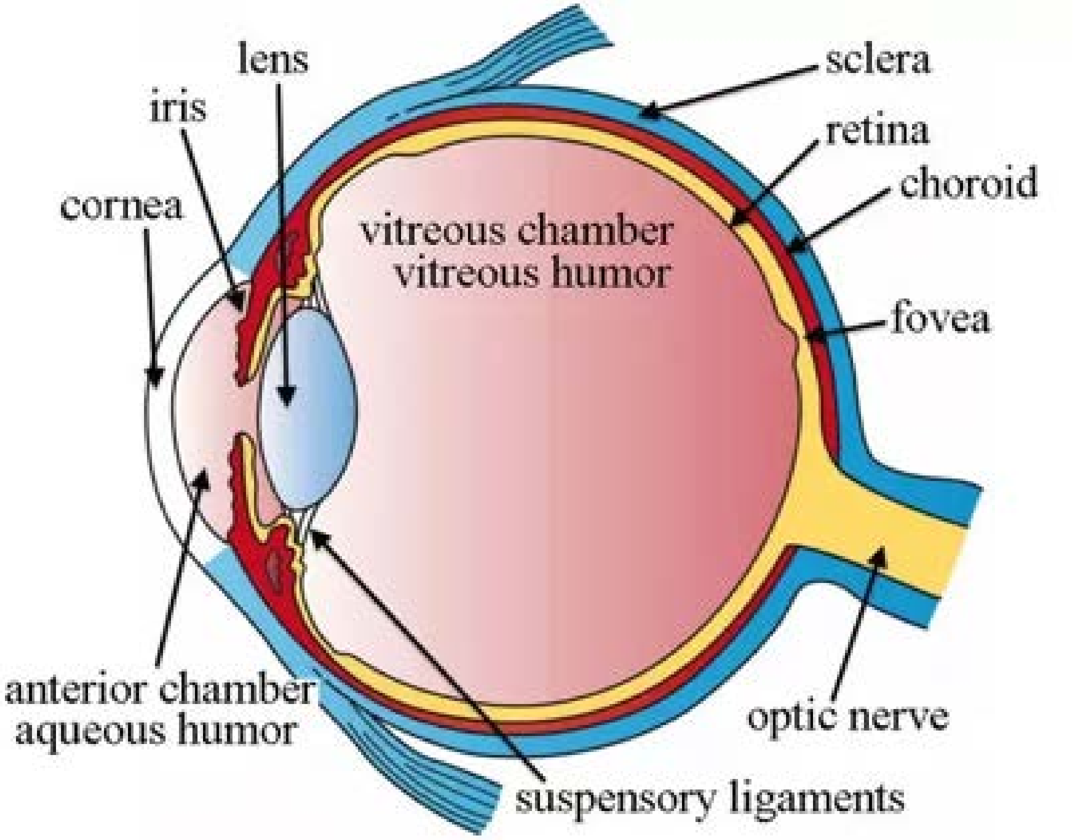
Goal of computer vision

“Computer vision is the process of using computers to **extract from images useful information** about the physical world, including meaningful descriptions of physical objects.”[1]

[1] Encyclopedia of Computer Science, https://dl.acm.org/doi/10.5555/1074100.1074274
[2] Figure: https://unsplash.com/photos/fRVPzBYcd5A

How do humans see?

- Vision:
  - **Light enters** through the cornea and focused by the lens onto the retina
  - In the retina the light is **converted into electrical signals**
  - These signals travel to the **brain for interpretation**
- Information from both eyes enables a 3D perception of the surroundings
- The resolution of the human eye is ~ 576 megapixels



https://www.vedantu.com/question-answer/draw-a-diagram-of-the-human-eye-as-seen-in-a-class-10-biology-cbse-6080f647dfee7e00e205f722

What is the difference?

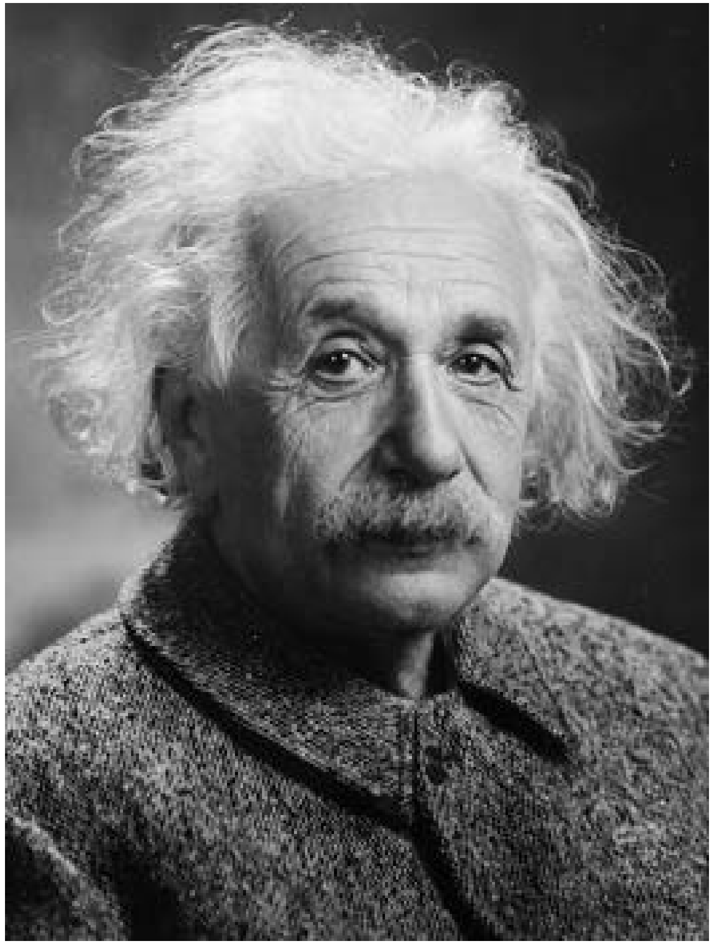


Figure of Ape: https://commons.wikimedia.org/wiki/File:Vespa_truck.jpg
Figure of Albert Einstein: https://commons.wikimedia.org/wiki/File:Albert_Einstein_Head.jpg

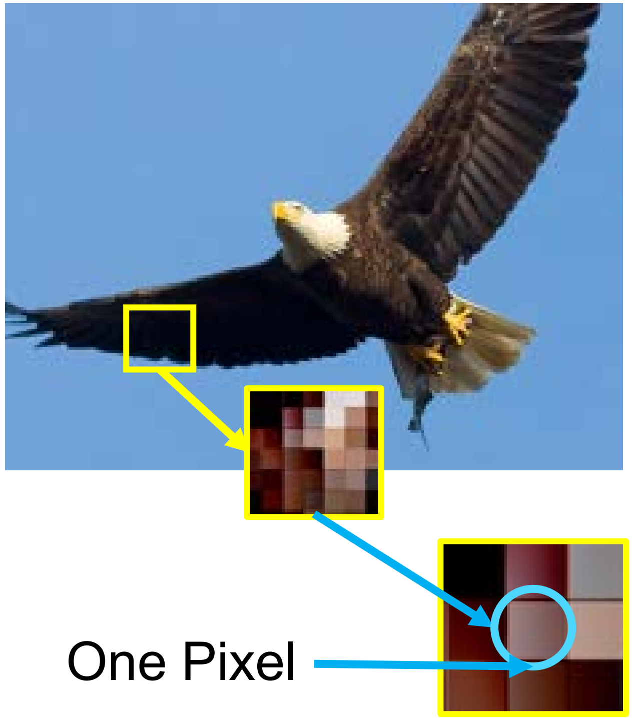
How can a computer differentiate an image of “car” from a “human”?

Agenda

- What is computer vision?
- **What do machines see?**
- Important concepts in computer vision
  - Filter
  - Padding
  - Stride
  - Pooling
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- Outlook on today’s assignment

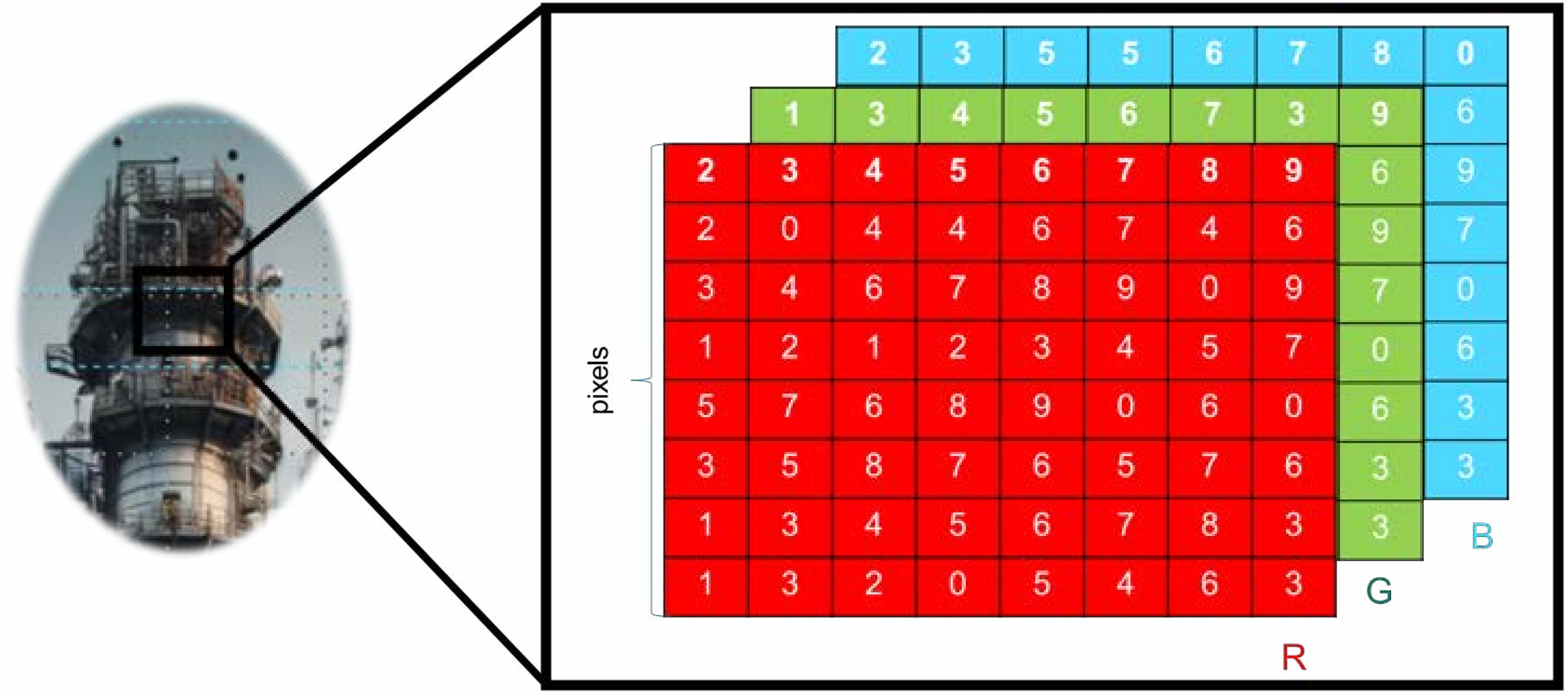
Digital images are composed of pixels

- Pixel basics
  - Each pixel is the smallest unit of a digital image
  - Pixels are organized in a grid to compose the image
- Image resolution
  - Resolution refers to the pixel count in an image
  - Higher resolution means more pixels and more detail
- Image formats
  - Common formats include JPEG and PNG
  - These use compression to reduce file size



Picture courtesy: "Bald eagle with fish" by U.S. Fish and Wildlife Service - Northeast Region is marked with Public Domain Mark 1.0.

Color spaces to represent images



- RGB images consist of three matrices laid over each other, with values between 0-255
- Alternative, Gray scale images only have a single matrix with values between 0-1

Are the pixel values of these two images similar?

Figure of Ape: https://commons.wikimedia.org/wiki/File:Vespa_truck.jpg
Figure of BMW Isetta: https://commons.wikimedia.org/wiki/File:BMW_Isetta_(2015-08-29_3124_b_Sp).JPG

Challenges for representing images as matrices

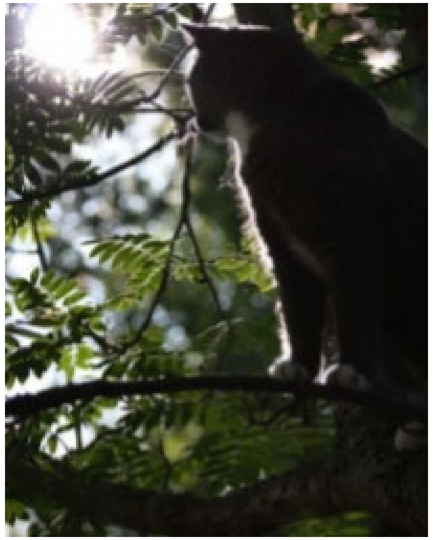
- Processing images as matrices is challenging using traditional computing
- Some of the challenges include…
  - Viewpoint
  - Illumination
  - Intraclass variability
  - Deformation
  - Background clutter
  - and many more…

Challenges: Viewpoint

Depending on the viewpoint, the same object has a completely different matrix

Challenges: Illumination

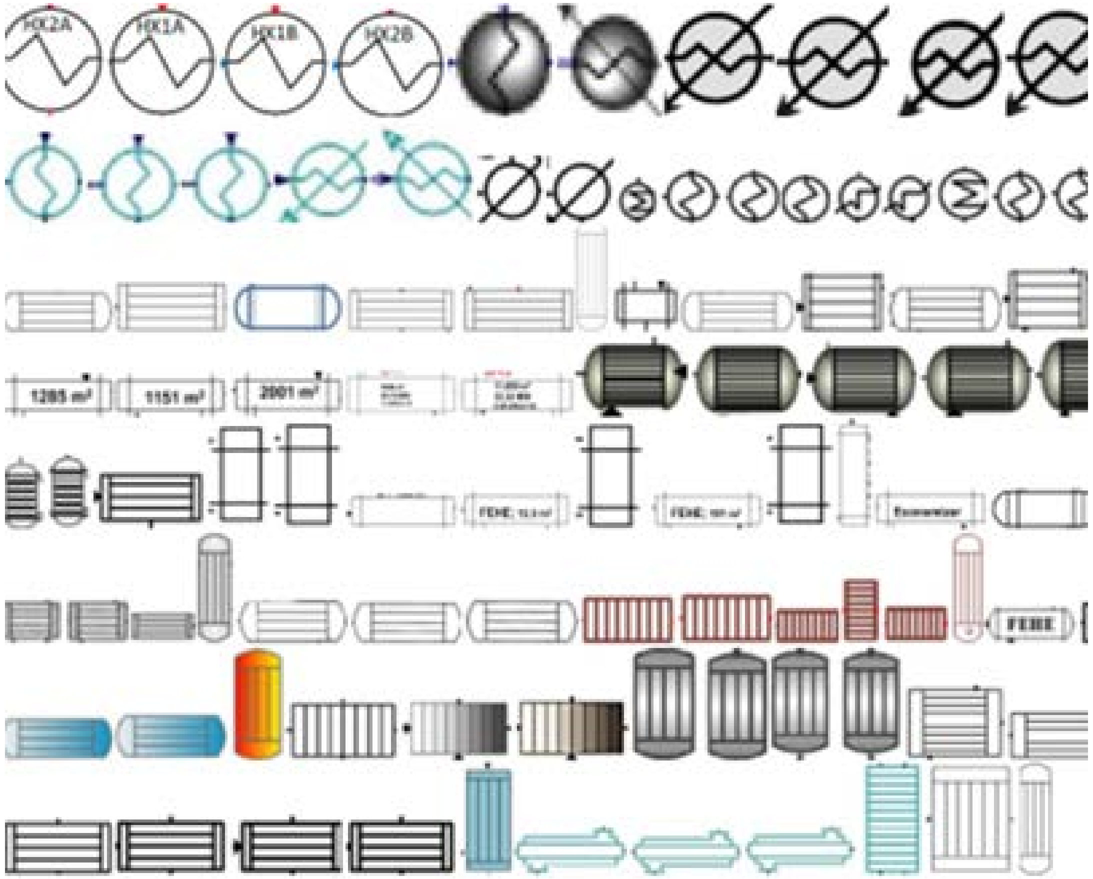


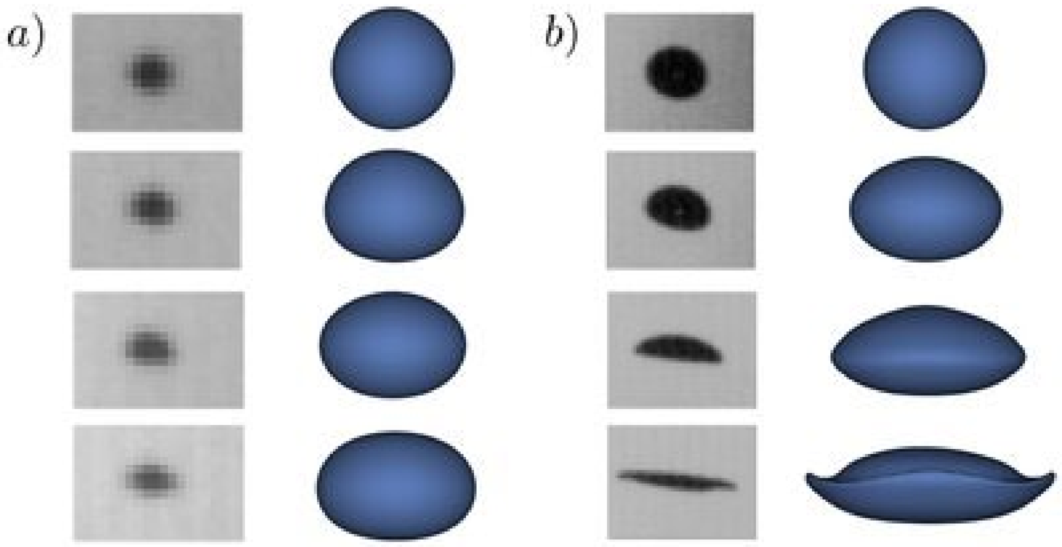
https://www.freepik.com/free-photo/cute-cat-darkness_9932116.htm#fromView=search&page=1&position=6&uuid=e5b09c2b-079d-4815-9e91-6a449816921c

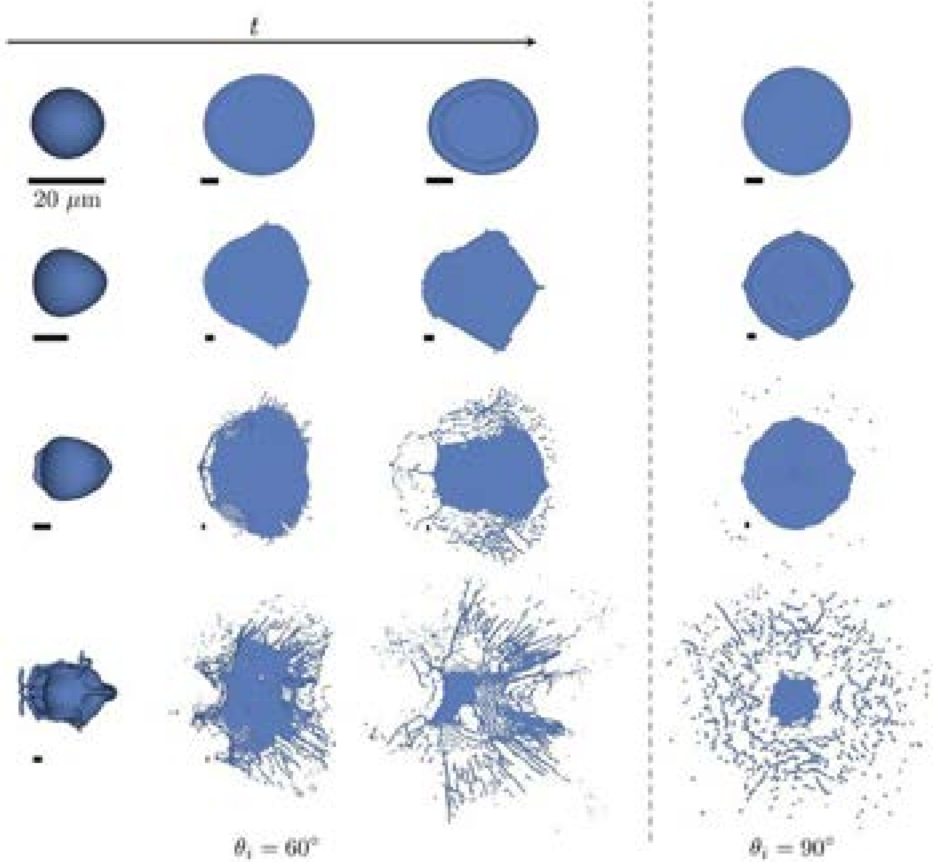
Challenges: Intraclass variability

- Oftentimes, we group things together that not always look completely alike
- We call this intraclass variability
- Computers need to know that certain depictions belong to the same class



Challenges: Deformation





Cimpeanu, R., & Papageorgiou, D. T. (2018). Three-dimensional high speed drop impact onto solid surfaces at arbitrary angles. International Journal of Multiphase Flow, 107, 192-207.

Challenges: Background clutter

https://www.freepik.com/free-photo/closeup-shot-cat-green-leaves_17419966.htm#fromView=search&page=1&position=0&uuid=e231445c-0165-4d85-91a7-b86ed34e44af

Agenda

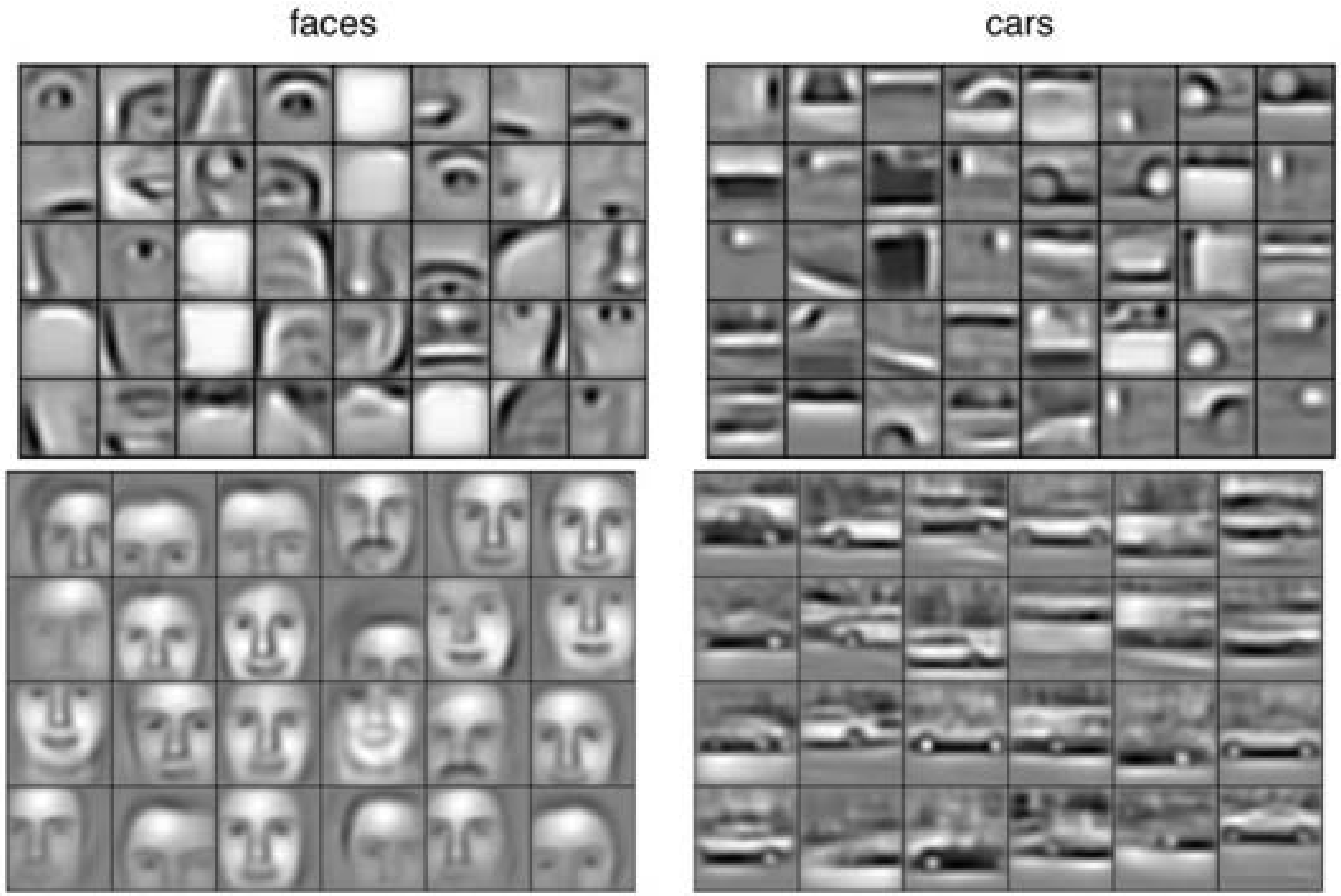
- What is computer vision?
- What do machines see?
- **Important concepts in computer vision**
  - **Filter**
  - Padding
  - Stride
  - Pooling
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- Outlook on today’s assignment

How can a computer differentiate an image of “car” from a “human”?

Features of humans and cars

Low-level features

High-level features



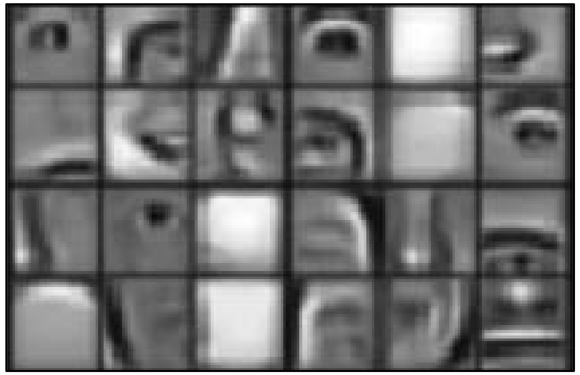
https://towardsdatascience.com/building-a-similar-images-finder-without-any-training-f69c0db900b5

We need to abstract relevant features from images

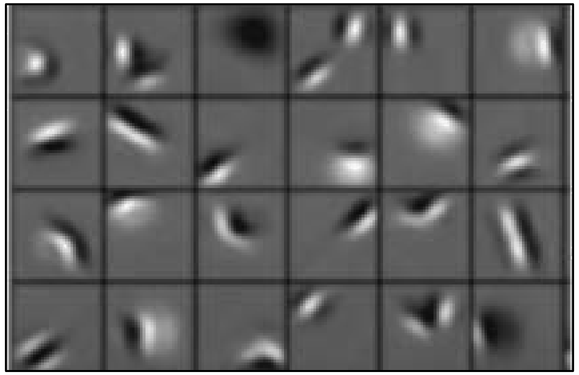
- To overcome these challenges, we need to represent the images robustly
- We need to find **features** that characterize objects in images
- But how can we find these features?

high-level features:
facial structure

mid-level features:
eyes, ears, noses



low-level features:
edges, dark spots

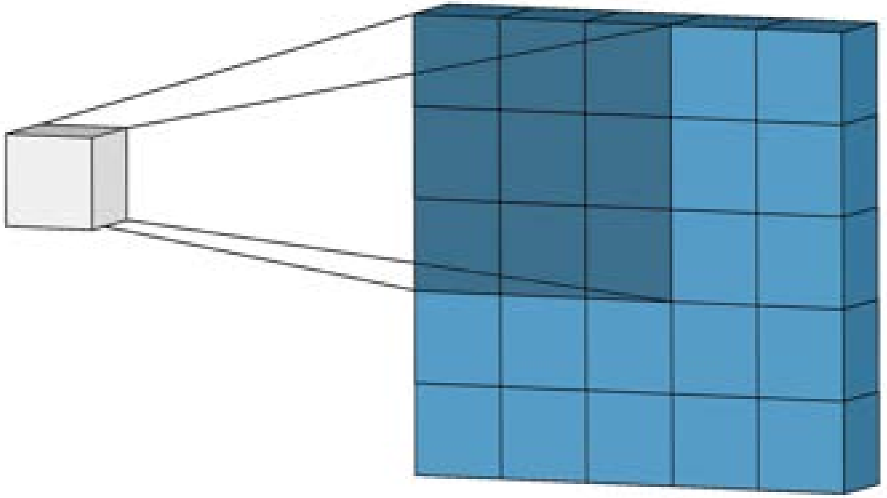


Increasing level of abstraction, focusing on lower-level features

[1] Bertasius, G., et al (2015) “High-for-Low and Low-for-High: Efficient Boundary Detection from Deep Object Features and its Applications to High-Level Vision”

Introduction to filters

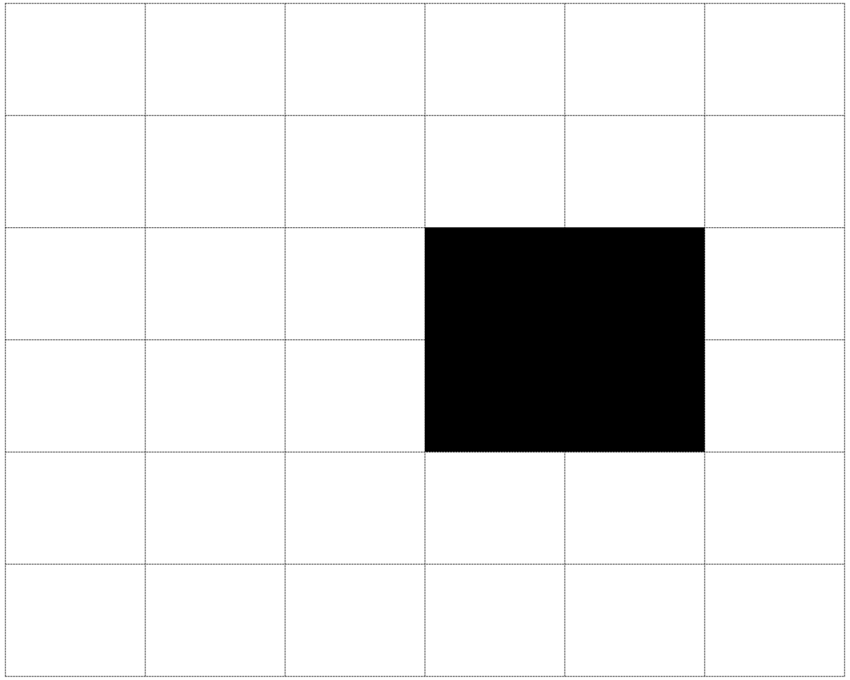
- **Filters** are mathematical operators applied to images to extract information and/or change appearance
- The result of a filter is referred to as a **feature map**
- In a **convolution operation**, a filter is slided over an image to obtain feature maps
- Convolution, kernel and filters are often used as synonyms



Filters transform images to new images (aka feature maps)

[1] Intuitively Understanding Convolutions for Deep Learning (Intuitively Understanding Convolutions for Deep Learning | by Irhum Shafkat | Towards Data Science)

A simple example “Image”



Actual image: 6x6

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

Matrix representation: 6x6

Conceptualize filters: What will happen if we apply the average filter?

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

Some image: 6x6

*

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

moving average 3x3

=

Conceptualize filters: What will happen if we apply the average filter?

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

Some image: 6x6

$*$

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

moving average 3x3

$=$

	1				

Conceptualize filters: What will happen if we apply the average filter?

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

Some image: 6x6

*

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

moving average 3x3

=

	1	0.88			

Conceptualize filters: What will happen if we apply the average filter?

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

Some image: 6x6

*

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

moving average 3x3

=

	1	0.88	0.77		

Co-funded by the Erasmus+ programme of the European Union

MachineLearnAthon

Conceptualize filters: What will happen if we apply the average filter?

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

Some image: 6x6

*

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

moving average 3x3

=

	1	0.88	0.77	0.77	
	1	0.77	0.55	0.55	
	1	0.77	0.55	0.55	
	1	0.88	0.77	0.77	

What happened to the image? – *It shrank*

Code example

https://github.com/process-intelligence-research/AI-in-Bio-Chemical-Engineering-Lecture-Coding

```python
import numpy as np

# 6x6 image matrix
image = np.array([[1, 1, 1, 1, 1, 1],
                  [1, 1, 1, 1, 1, 1],
                  [1, 1, 1, 0, 0, 1],
                  [1, 1, 1, 0, 0, 1],
                  [1, 1, 1, 1, 1, 1],
                  [1, 1, 1, 1, 1, 1]])

# 3x3 moving average filter
filter = np.array([[1/9, 1/9, 1/9],
                   [1/9, 1/9, 1/9],
                   [1/9, 1/9, 1/9]])

# Perform the convolution operation
output = np.zeros((4, 4))

for i in range(4):
    for j in range(4):
        # Extract the 3x3 patch from the image
        patch = image[i:i+3, j:j+3]
        # Compute the element-wise multiplication and sum
        output[i, j] = np.sum(patch * filter)

# Print the resulting output matrix
print(output)
```

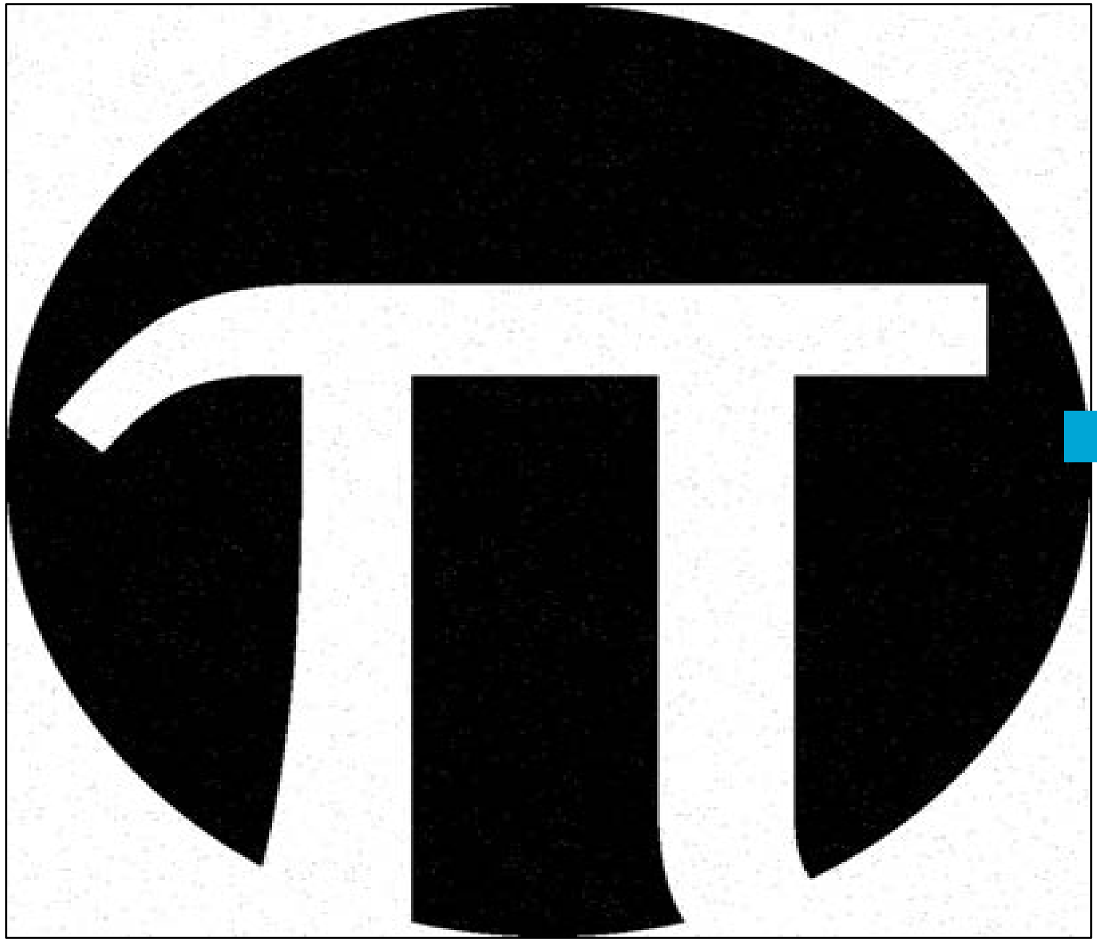
How do express a filter mathematically?

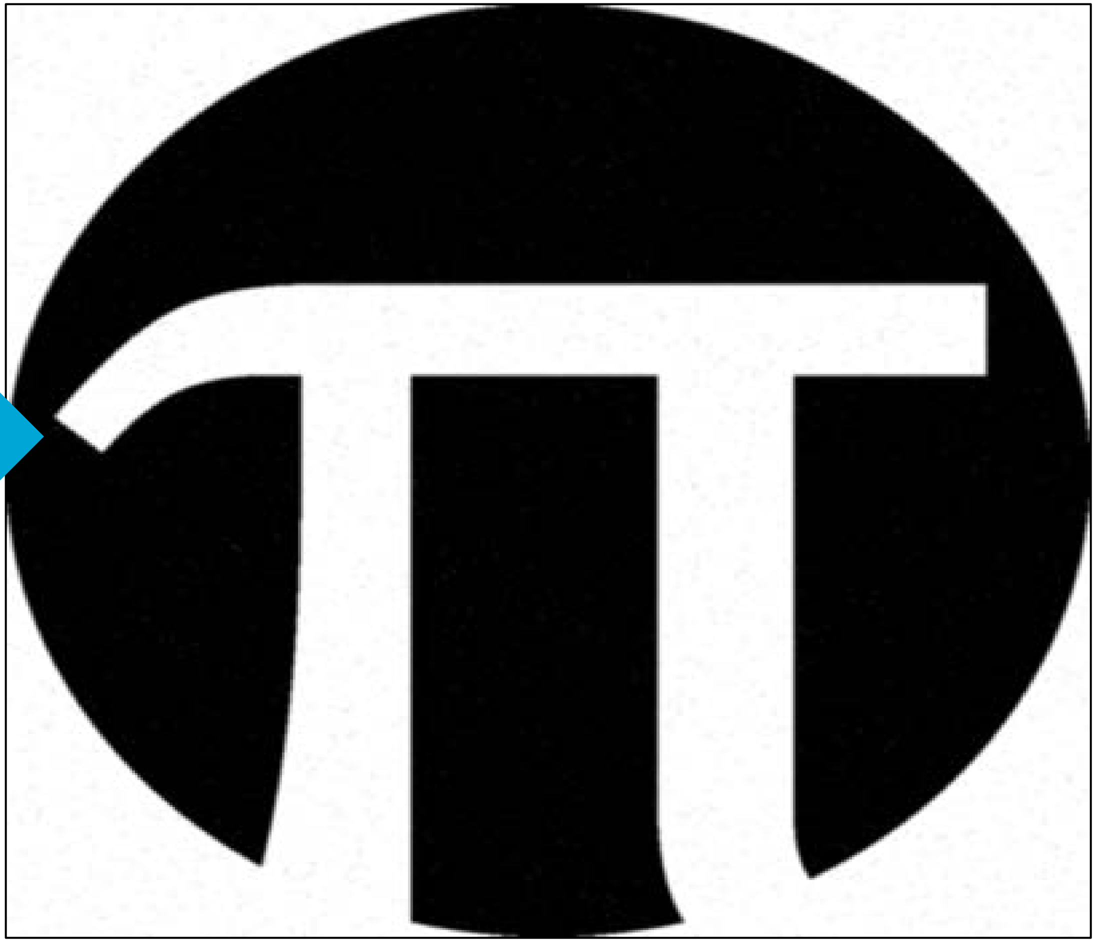
- The filtering can be expressed as a sparse matrix multiplication:

  - $C(x, y) = \sum_{i=-\frac{h-1}{2}}^{\frac{h-1}{2}} \sum_{j=-\frac{w-1}{2}}^{\frac{w-1}{2}} I(x+i, y+j) * K(i + \frac{h-1}{2}, j + \frac{w-1}{2})$

  - Where $I(x+i, y+j)$ is the pixel value at position $(x+i, y+j)$,
  - $K(i + \frac{h-1}{2}, j + \frac{w-1}{2})$ is the value of the kernel at $\left(i + \frac{h-1}{2}, j + \frac{w-1}{2}\right)$,
  - And h, w are the kernel height and width respectively

Image example: Effect of smoothing on noisy image





Join the Vevox session

Go to **vevox.app**

Enter the session ID: **199-929-003**

Or scan the QR code

Select the resultant once the following filter is applied

A

0%

B

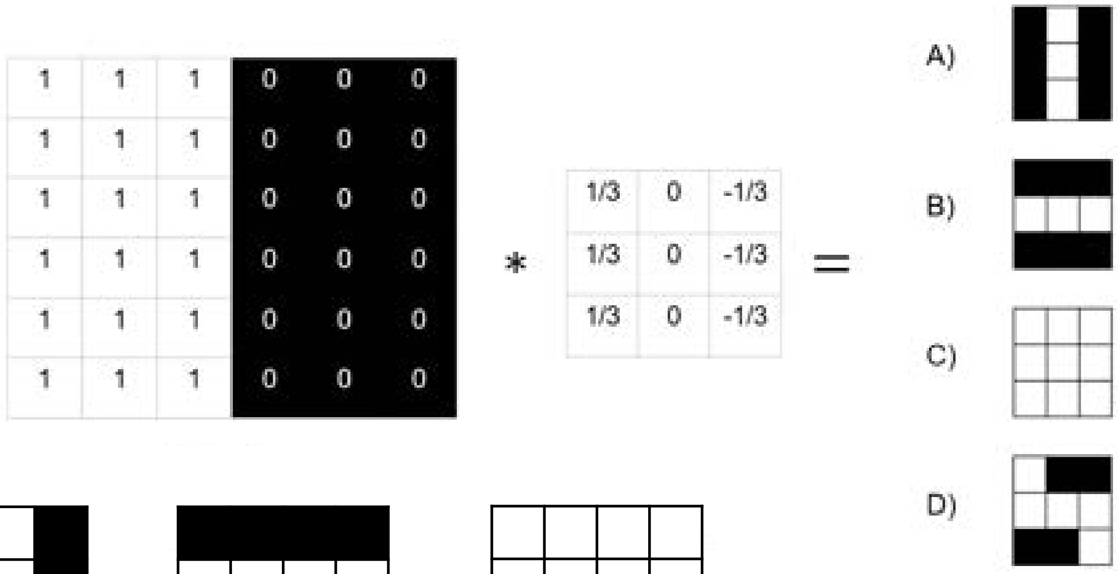
0%

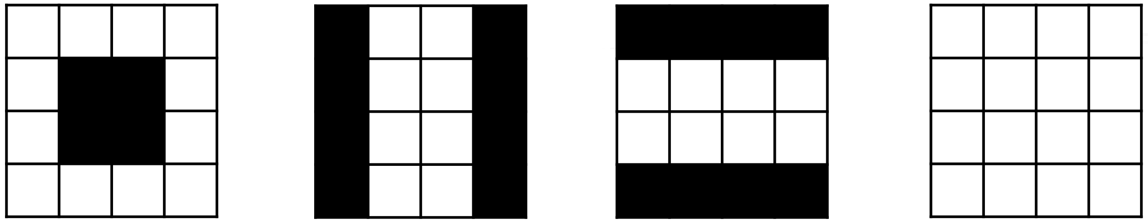
C

0%

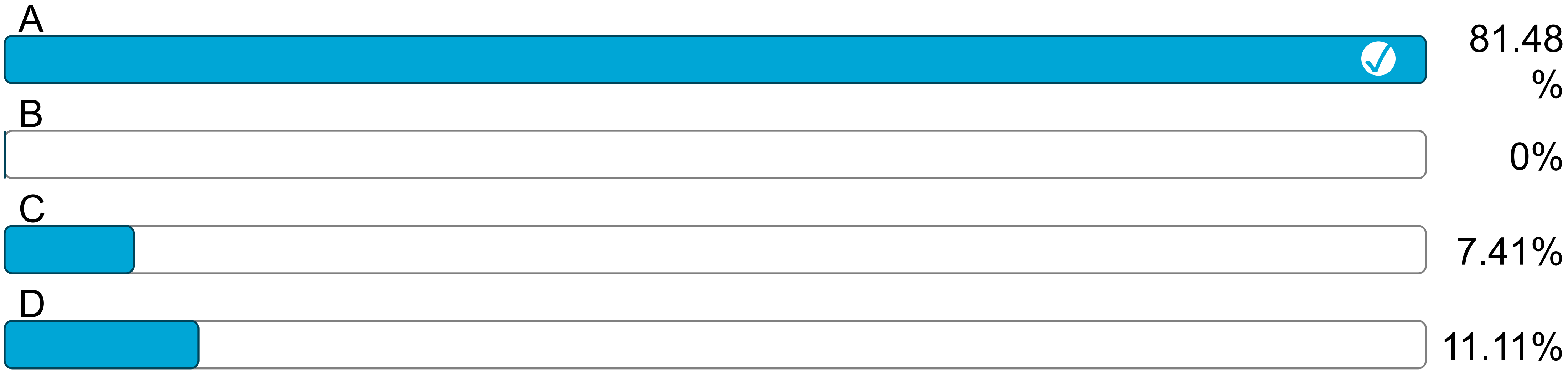
D

0%





Select the resultant once the following filter is applied



We can also apply other filters

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

*

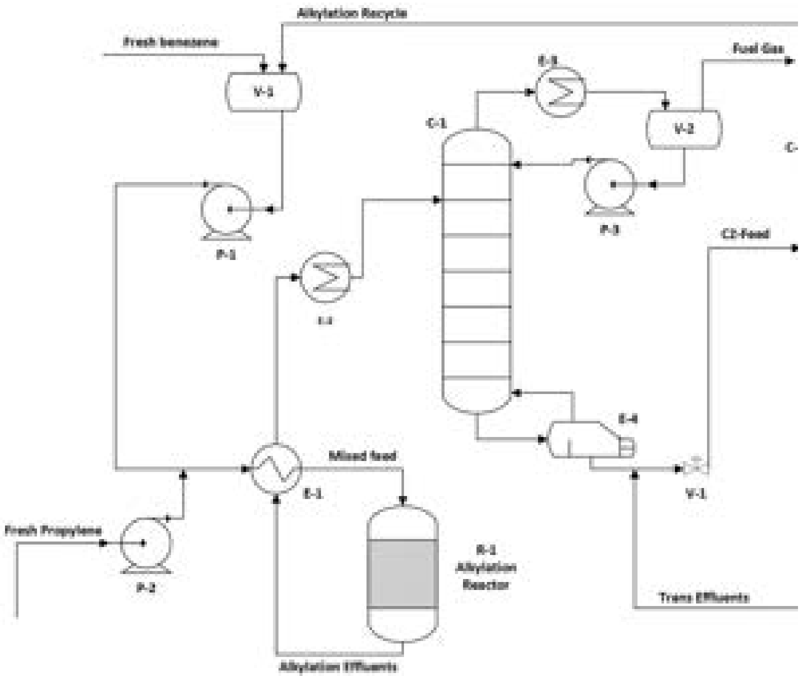
1/3	0	-1/3
1/3	0	-1/3
1/3	0	-1/3

=

What happens if we apply it?

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

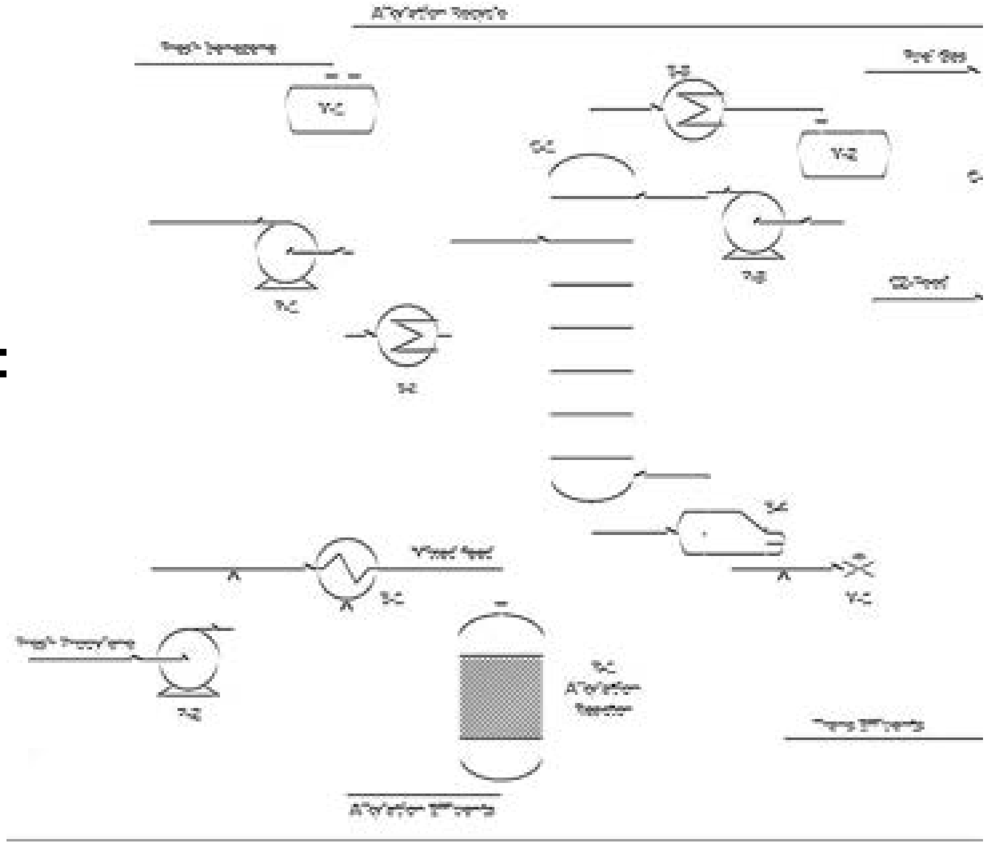
We can also apply multiple kernels to get different feature maps!



*

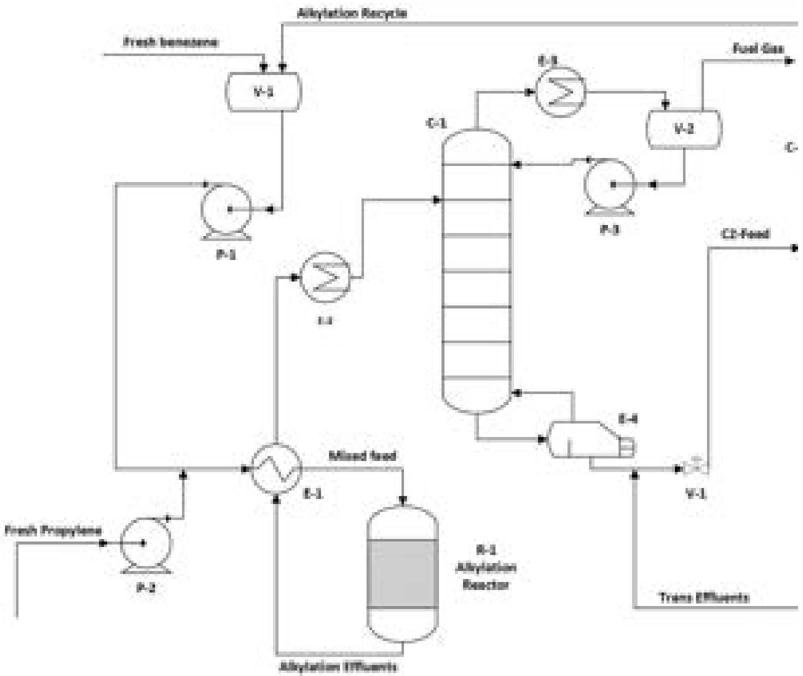
1	0	-1
1	0	-1
1	0	-1

=



Resulting images are inverted for visibility!

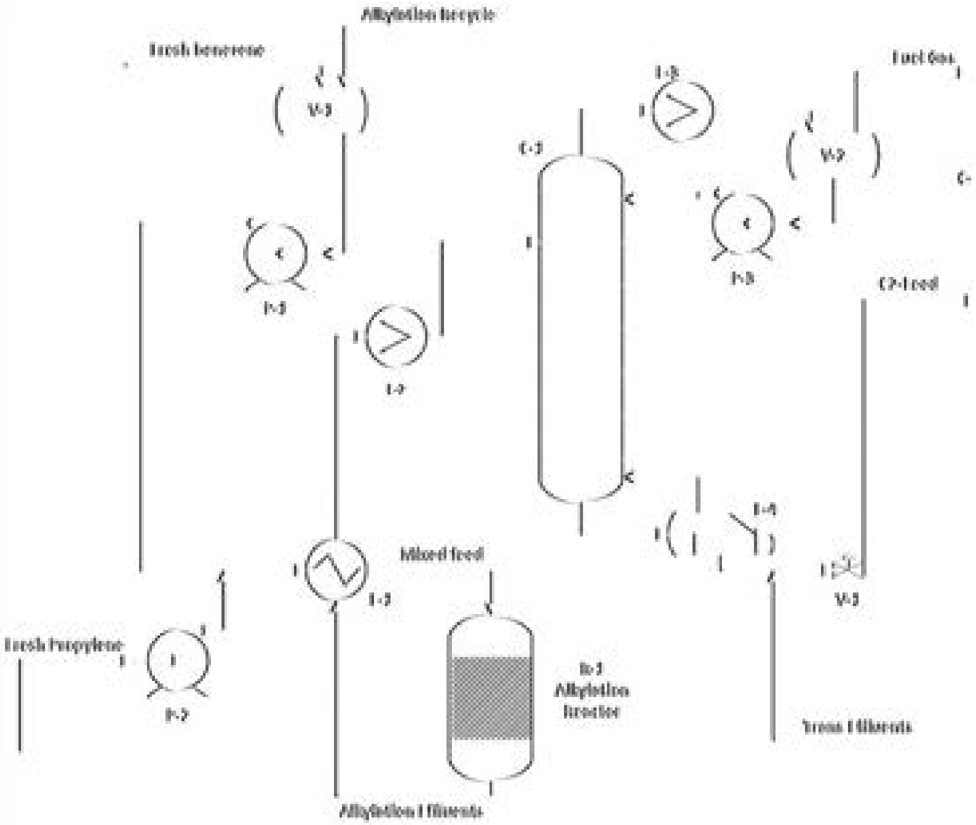
We can also apply multiple kernels to get different feature maps!



*

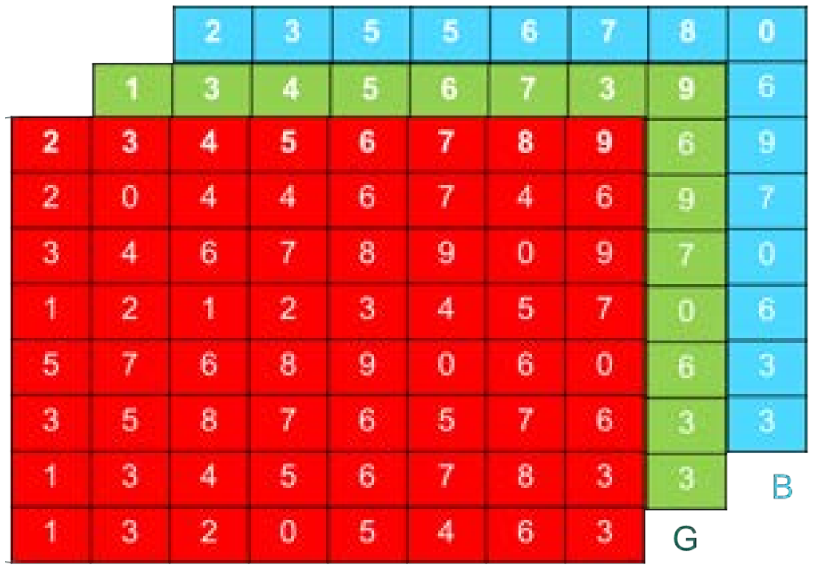
-1	-1	-1
0	0	0
1	1	1

=



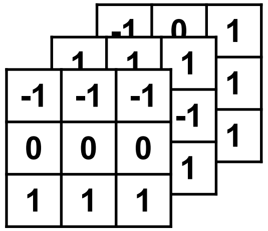
Resulting images are inverted for visibility!

A filter operating on multiple channels



input images R
[height, width, channels] = [8,8,3]

*



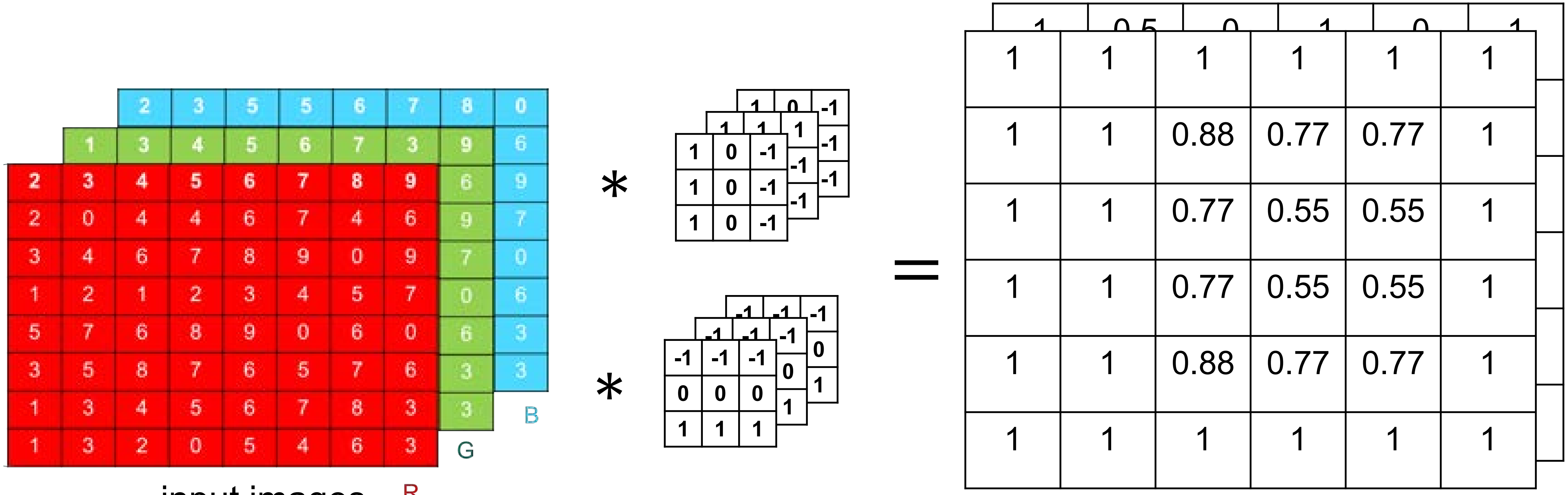
applied filter
[height, width, channels] = [3,3,1]

=

1	1	1	1	1	1
1	1	0.88	0.77	0.77	1
1	1	0.77	0.55	0.55	1
1	1	0.77	0.55	0.55	1
1	1	0.88	0.77	0.77	1
1	1	1	1	1	1

resulting feature map
[height, width, channels] = [6,6,1]

Multiple filters operating on multiple channels



input images
[height, width, channels] = [8,8,3]

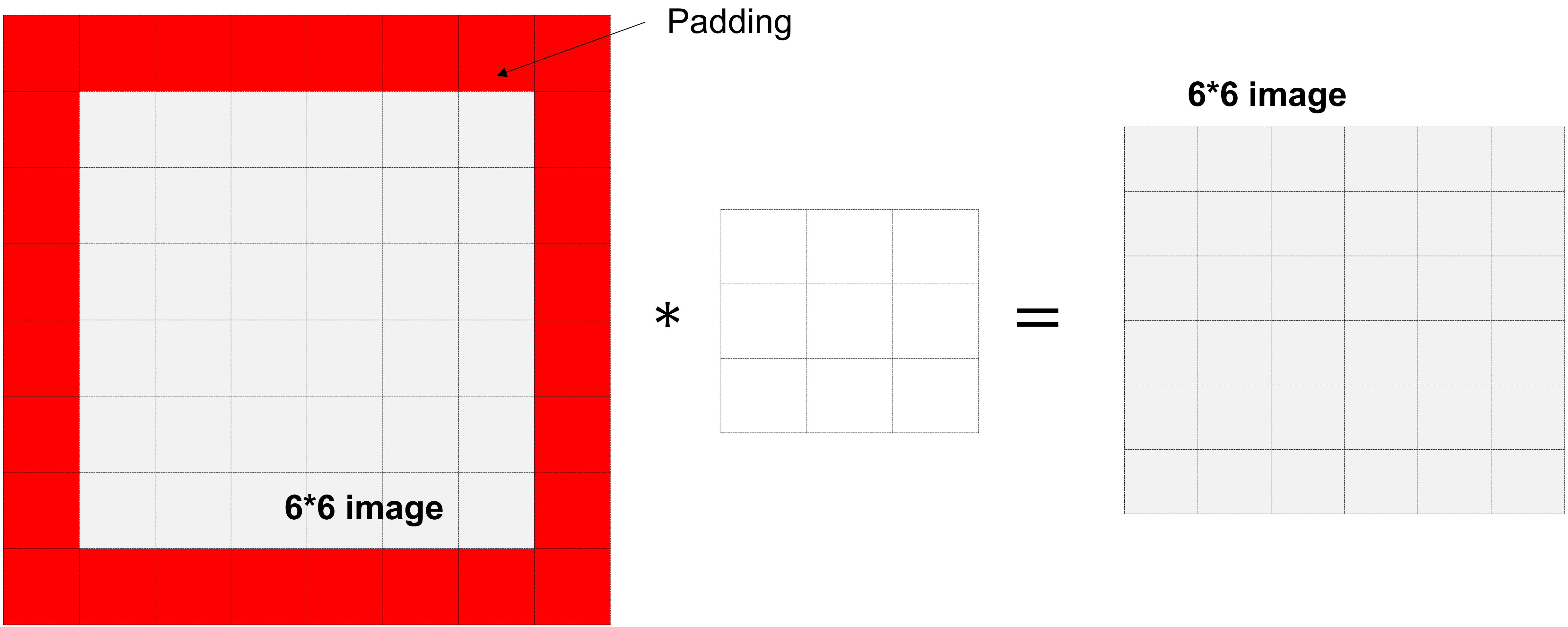
applied filters
[height, width, channels] = [3,3,2]

resulting feature map
[height, width, channels] = [6,6,2]

Agenda

- What is computer vision?
- What do machines see?
- **Important concepts in computer vision**
  - Filter
  - **Padding**
  - Stride
  - Pooling
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- Outlook on today's assignment

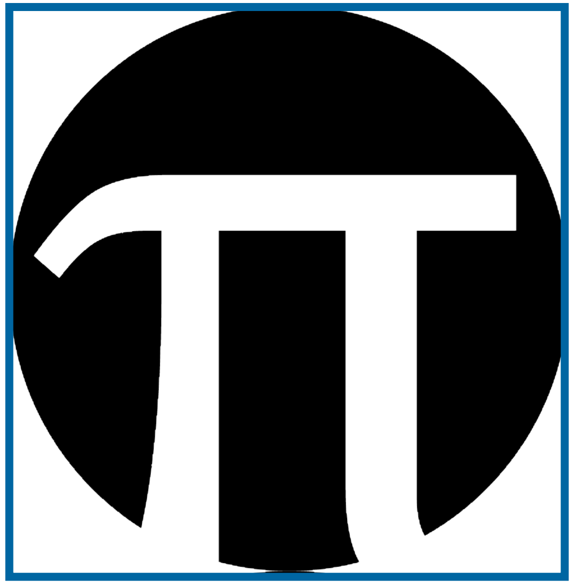
How can we avoid making our image smaller? Padding



How can we avoid making our image smaller? Padding

- We can avoid shrinking our images by artificially extending them

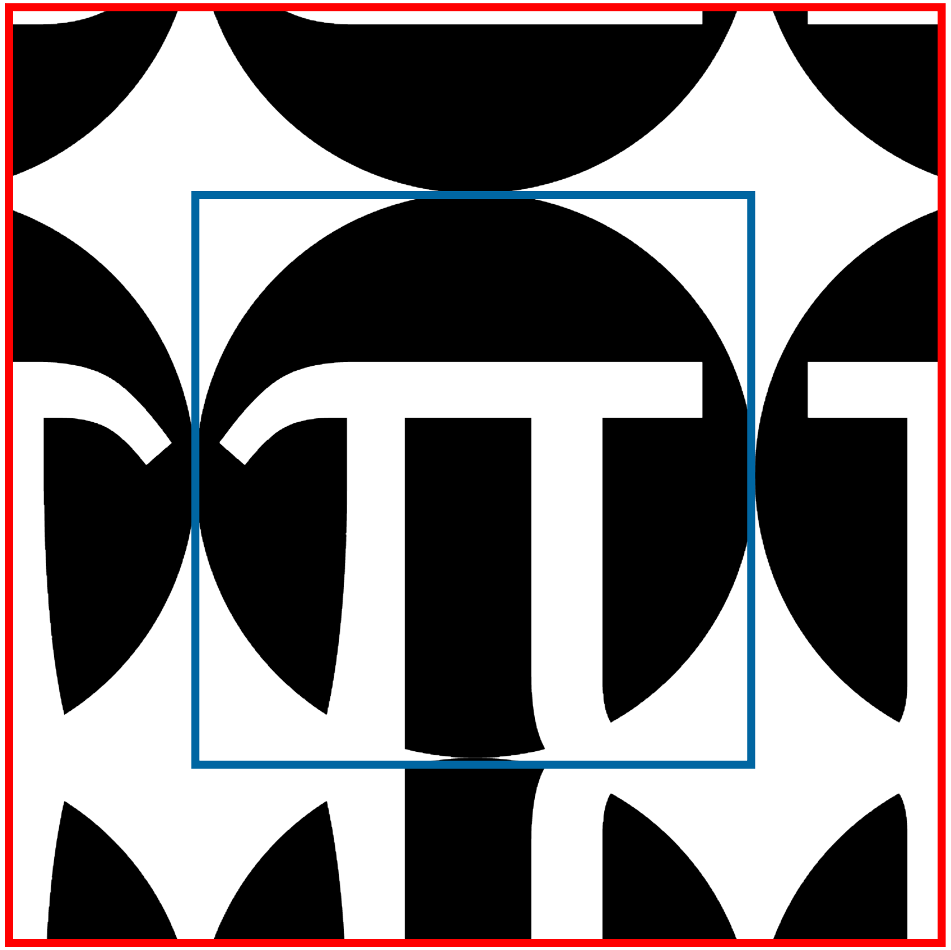
- There are several common padding techniques:
  - Mirror padding
  - Zero padding
  - Constant padding
  - Replicate padding
  - Circular padding



How can we avoid making our image smaller? Padding

- We can avoid shrinking our images by artificially extending them

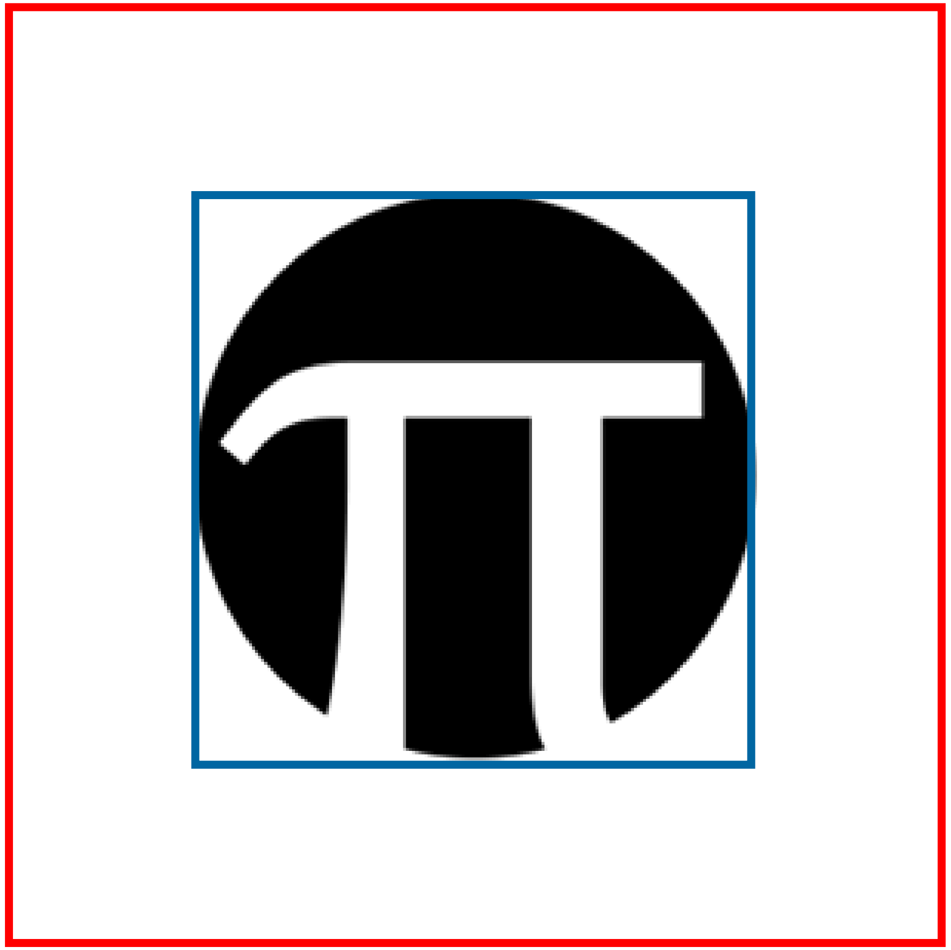
- There are several common padding techniques:
  - **Mirror padding**
  - Zero padding
  - Constant padding
  - Replicate padding
  - Circular padding



How can we avoid making our image smaller? Padding

- We can avoid shrinking our images by artificially extending them

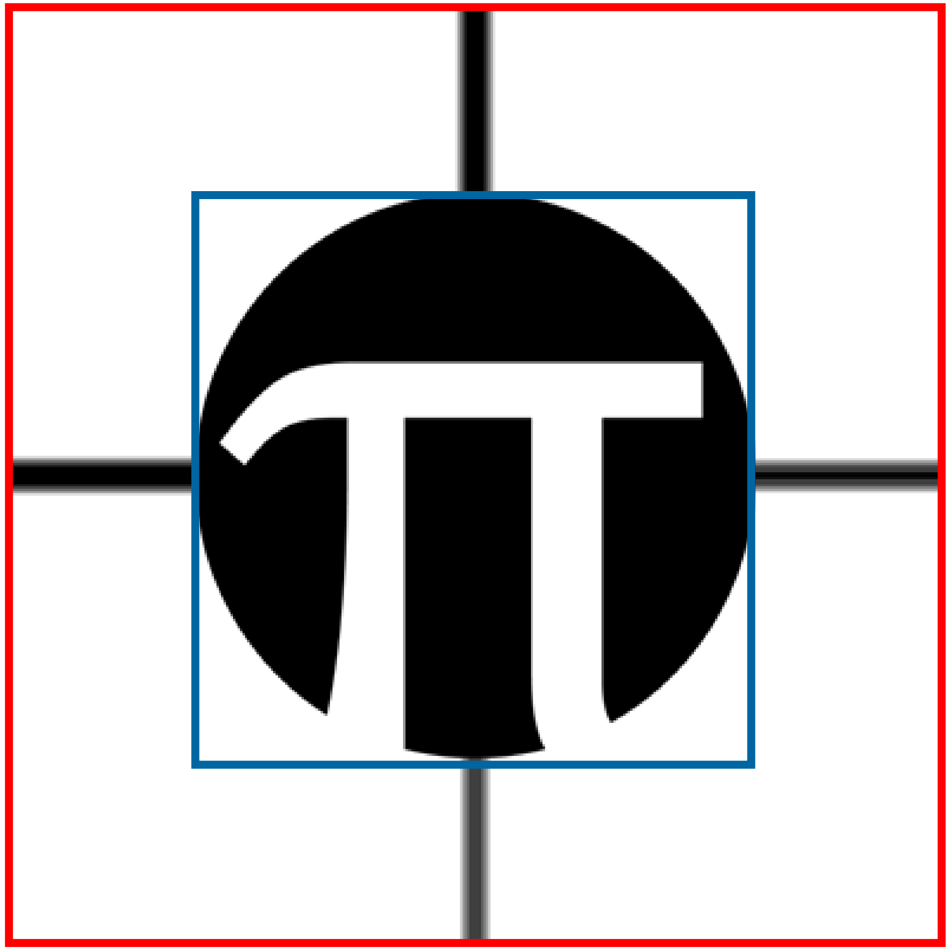
- There are several common padding techniques:
  - Mirror padding
  - **Zero padding**
  - Constant padding
  - Replicate padding
  - Circular padding



How can we avoid making our image smaller? Padding

- We can avoid shrinking our images by artificially extending them

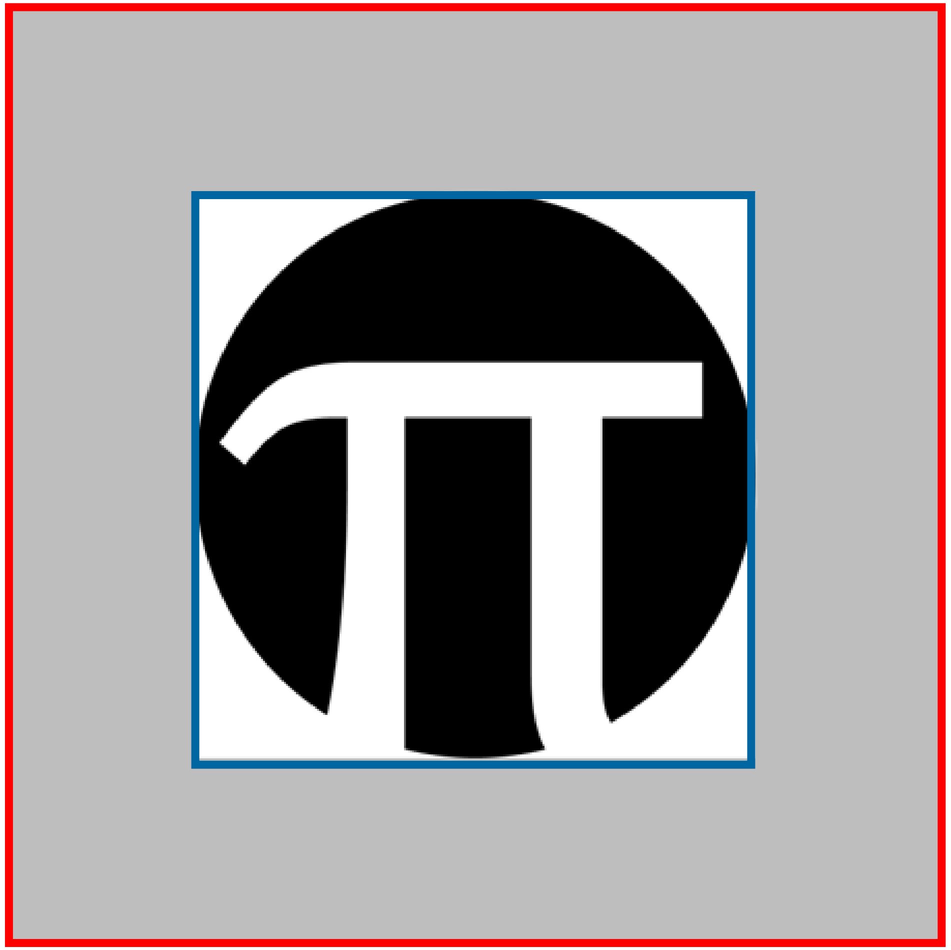
- There are several common padding techniques:
  - Mirror padding
  - Zero padding
  - **Replicate padding**
  - Constant padding
  - Circular padding



How can we avoid making our image smaller? Padding

- We can avoid shrinking our images by artificially extending them

- There are several common padding techniques:
  - Mirror padding
  - Zero padding
  - Replicate padding
  - **Constant padding**
  - Circular padding



How can we avoid making our image smaller? Padding

- We can avoid shrinking our images by artificially extending them

- There are several common padding techniques:
  - Mirror padding
  - Zero padding
  - Replicate padding
  - Constant padding
  - **Circular padding**

Agenda

- What is computer vision?
- What do machines see?
- **Important concepts in computer vision**
  - Filter
  - Padding
  - **Stride**
  - Pooling
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- Outlook on today's assignment

Do need to apply our kernel to every pixel? No

- Idea: We only apply the kernel every n-th time, where n is our **stride**

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

Stride = 1

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

Stride = 2

How can we reduce our image size quicker? Stride

- Idea: We only apply the kernel every n-th time, where n is our stride

Stride = 1

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

Stride = 1

Stride = 2

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

Stride = 2

How can we reduce our image size quicker? Stride

- Idea: We only apply the kernel every n-th time, where n is our stride

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

Stride = 1

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

Stride = 2

How can we reduce our image size quicker? Stride

- Idea: We only apply the kernel every n-th time, where n is our stride

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

Stride = 1

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

Stride = 2

What does stride look like when applied?

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

Some image: 6x6

*

¼	¼
¼	¼

moving average 2x2

=

1	1	1	1	1
1	1	¾	½	¾
1	1	½	0	½
1	1	¾	½	¾
1	1	1	1	1

Stride = 1

1	1	1
1	½	½
1	1	1

Stride = 2

How can we calculate the output dimension of feature maps?

- The output size (height and width) of the feature map is given by

$$output\ dimension = \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1$$

- where…
  - I_P:Input dimension (height, width of image)
  - F: Filter size (height and width of kernel)
  - P : Padding (Padding is usually applied symmetrically)
  - S : Stride

Bracket indicates the floor function (or greatest integer function) that returns the greatest integer less than the function argument.[2]

We expect you to know this formula for the exam (by heart)!

[2] https://en.wikipedia.org/wiki/Floor_and_ceiling_functions

Example calculation

Layer Type	Filter	Padding	Output Size
Input Layer	Input Image	None	224x224x3
Convolution	128 filters (3x3 stride 1)	1	-

1. Convolutions:
    - $I_p = 224$
    - $\text{F} = 3$
    - $\text{S} = 1$
    - $\text{P} = 1$

$$width = \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{224 - 3 + 2}{1} \right\rfloor + 1 = 224$$

$$height = \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{224 - 3 + 2}{1} \right\rfloor + 1 = 224$$

- Number of filters = 128

Output size = [height, width, number of channels] = 224x224x128

Height and width
of output feature maps

Channels

Agenda

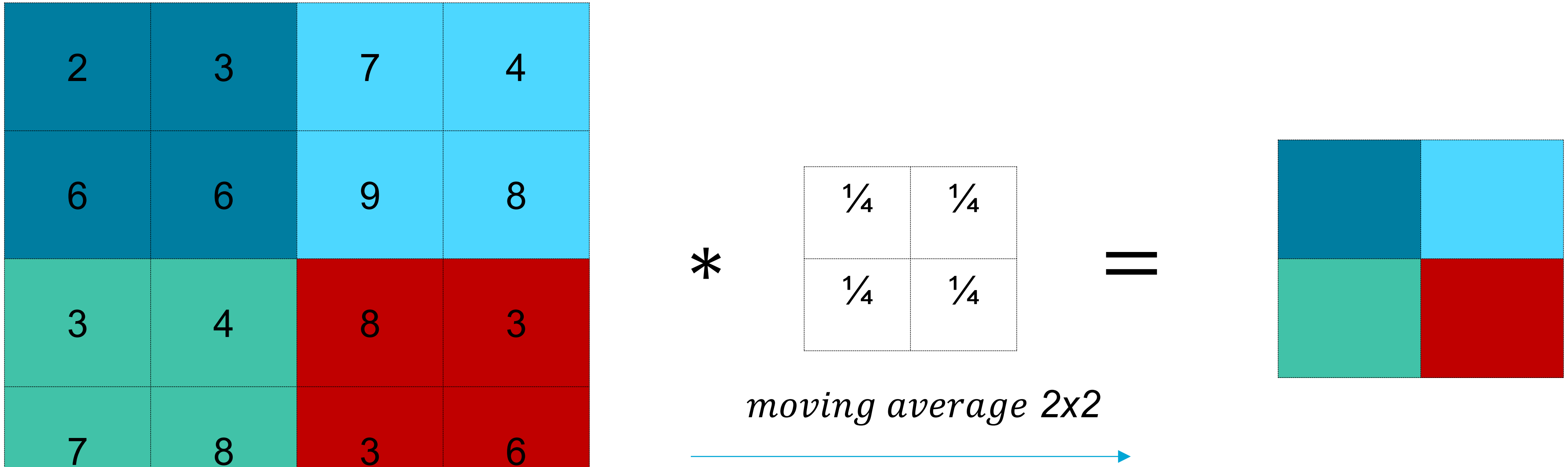
- What is computer vision?
- What do machines see?
- **Important concepts in computer vision**
  - Filter
  - Padding
  - Stride
  - **Pooling**
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- Outlook on today’s assignment

Motivation behind pooling

- There are can be many important features in a single image
- If we apply all these filters, we will end up with many feature maps
- So how can we reduce the dimensionality of our feature maps?
- Goal of pooling:
  - Keep important information
  - Reduce dimensionality

Image taken from freepik.com (worker-surrounded-with-glass-beakers-filled-with-colorful-liquid)

How can we summarize found feature maps? Pooling



Average pooling
with kernel size 2
and stride 2

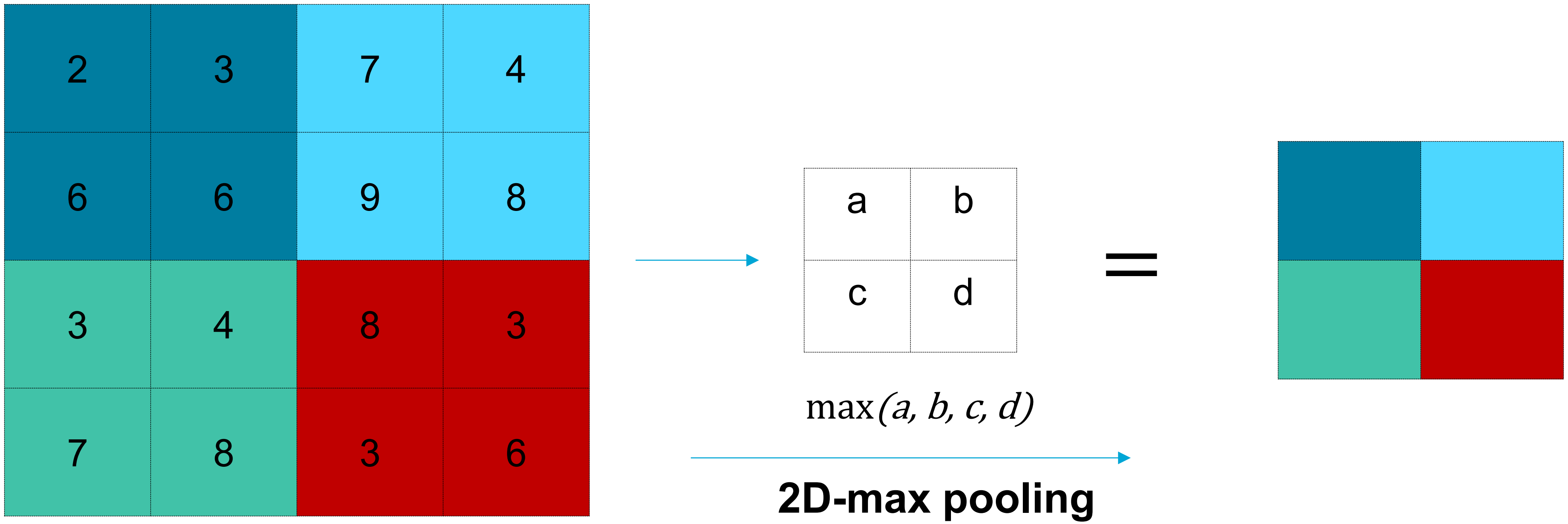
How can we summarize found feature maps? Pooling

$$
\begin{bmatrix} 2 & 3 & 7 & 4 \\ 6 & 6 & 9 & 8 \\ 3 & 4 & 8 & 3 \\ 7 & 8 & 3 & 6 \end{bmatrix} * \begin{bmatrix} \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} \end{bmatrix} = \begin{bmatrix} 4.25 & 7 \\ 5.6 & 5 \end{bmatrix}
$$

$moving\ average$ 2x2

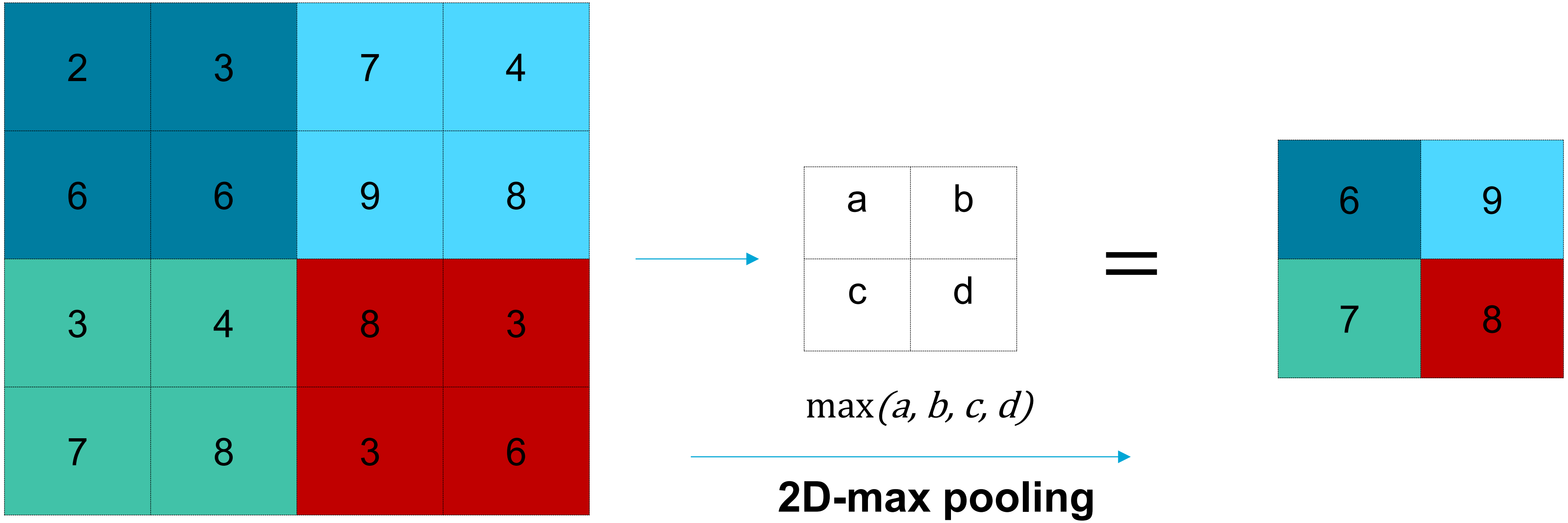
Average pooling
with kernel size 2
and stride 2

How can we summarize found feature maps? Pooling



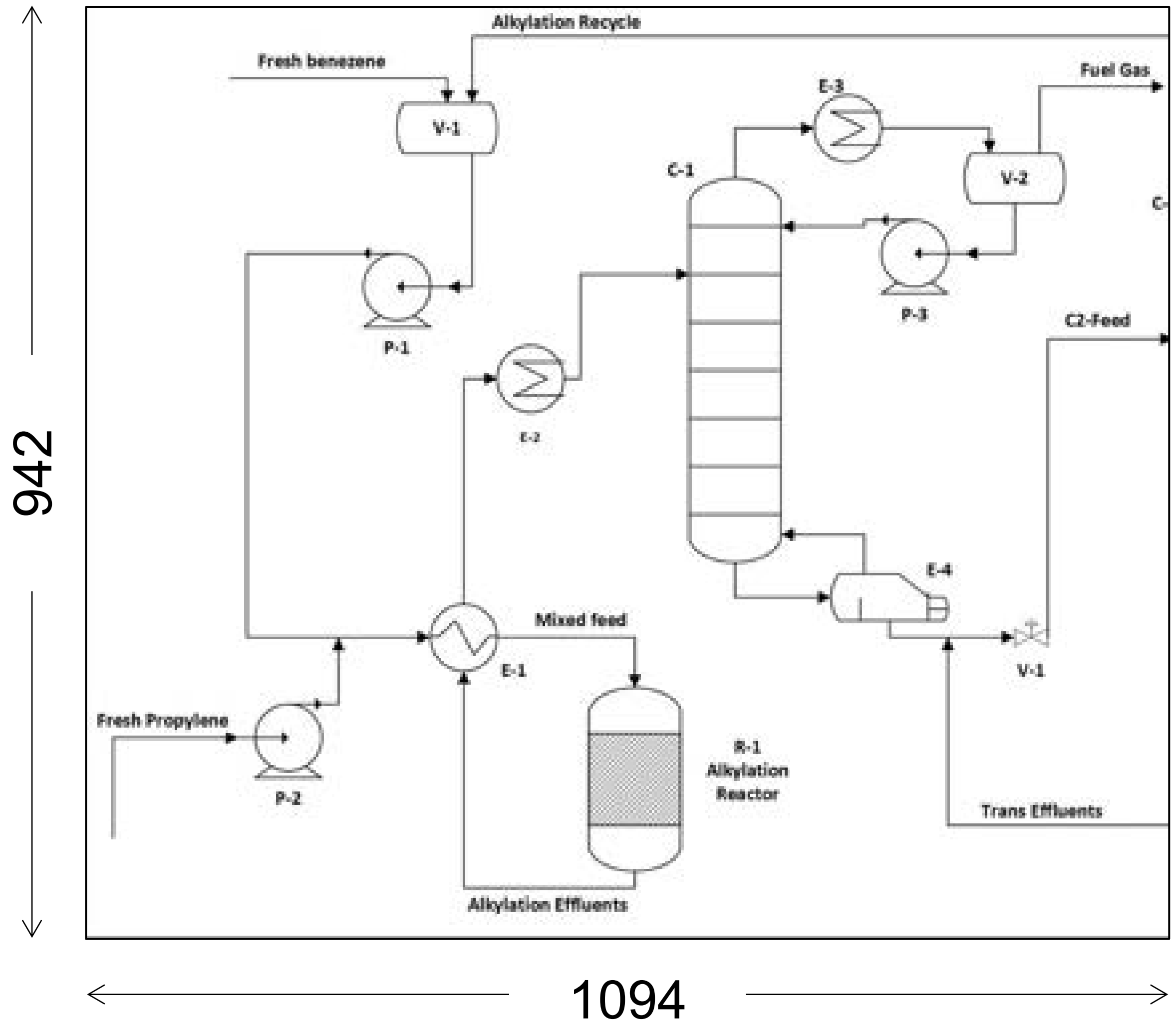
2D-max pooling
with kernel size 2
and stride 2

How can we summarize found feature maps? Pooling

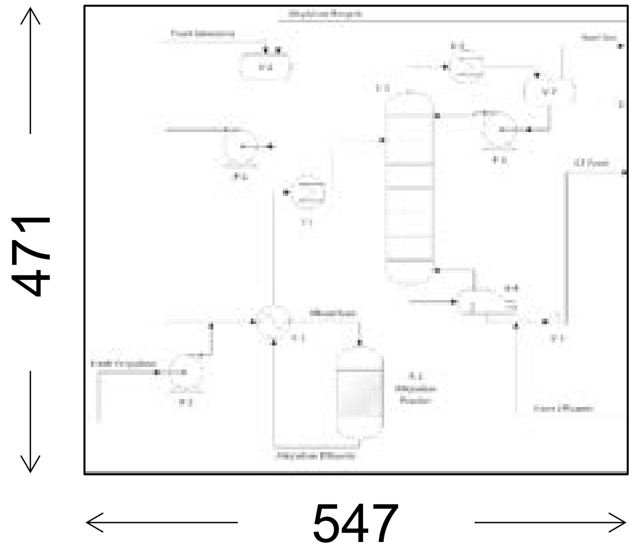


2D-max pooling
with kernel size 2
and stride 2

Example of applied max-pooling



2D-max pooling
with kernel size 2
and stride 2



Agenda

- What is computer vision?
- What do machines see?
- Important concepts in computer vision
  - Filter
  - Padding
  - Stride
  - Pooling
- **Elements in deep learning computer vision algorithms**
- How does a modern computer vision model look like?
- Outlook on today's assignment

Image classification tasks

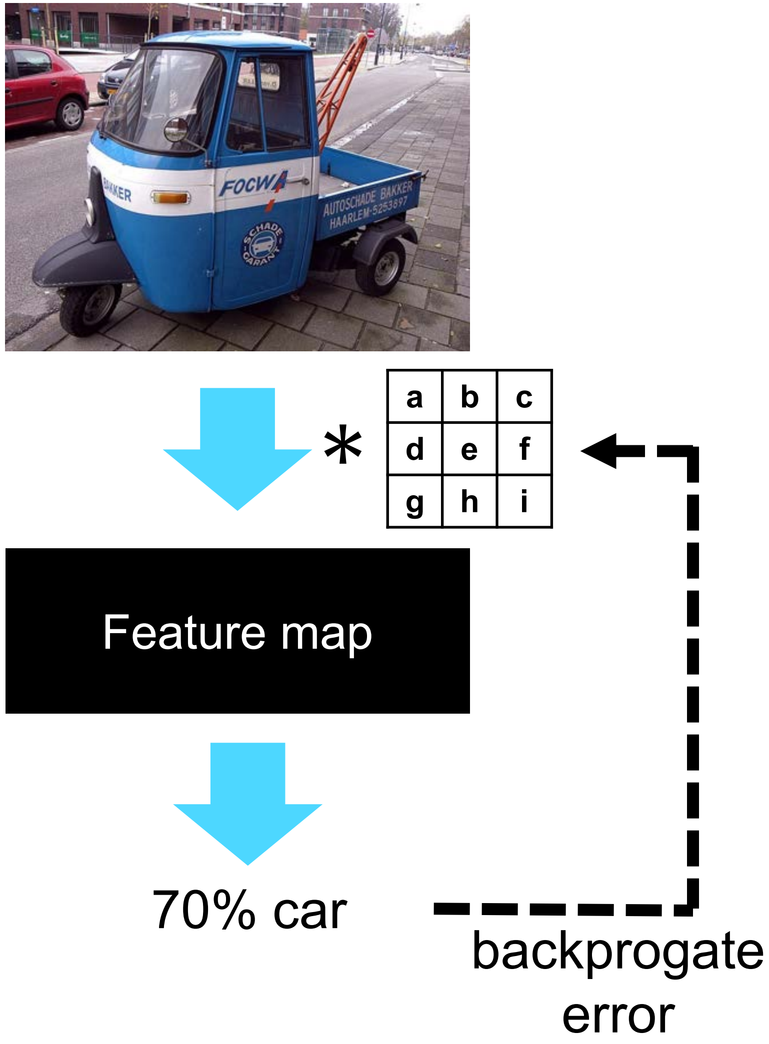
- In image classification, we aim to get probabilities for categories of image
- In our example, we have a dataset with two classes, “human” and “car”
- More general, for a model m and image I, we aim to get a prediction vector p
  - $p = m(I) = \begin{bmatrix} p_1(I) \\ p_2(I) \\ \dots \\ p_c(I) \end{bmatrix}$
  - containing a probability p_i for each class c of the dataset

Model m

70% car
30% human

Learning filters (aka convolutions) in convolutional neural networks

- For image classification, we train the convolutional neural network models *supervised*
- We use kernels with ***learnable* weight matrices** to extract features
- The model is trained by minimizing the error between prediction and ground truth (see slide 33 in Lecture 2)
- Since the kernels are learned, we do not need to manually define them

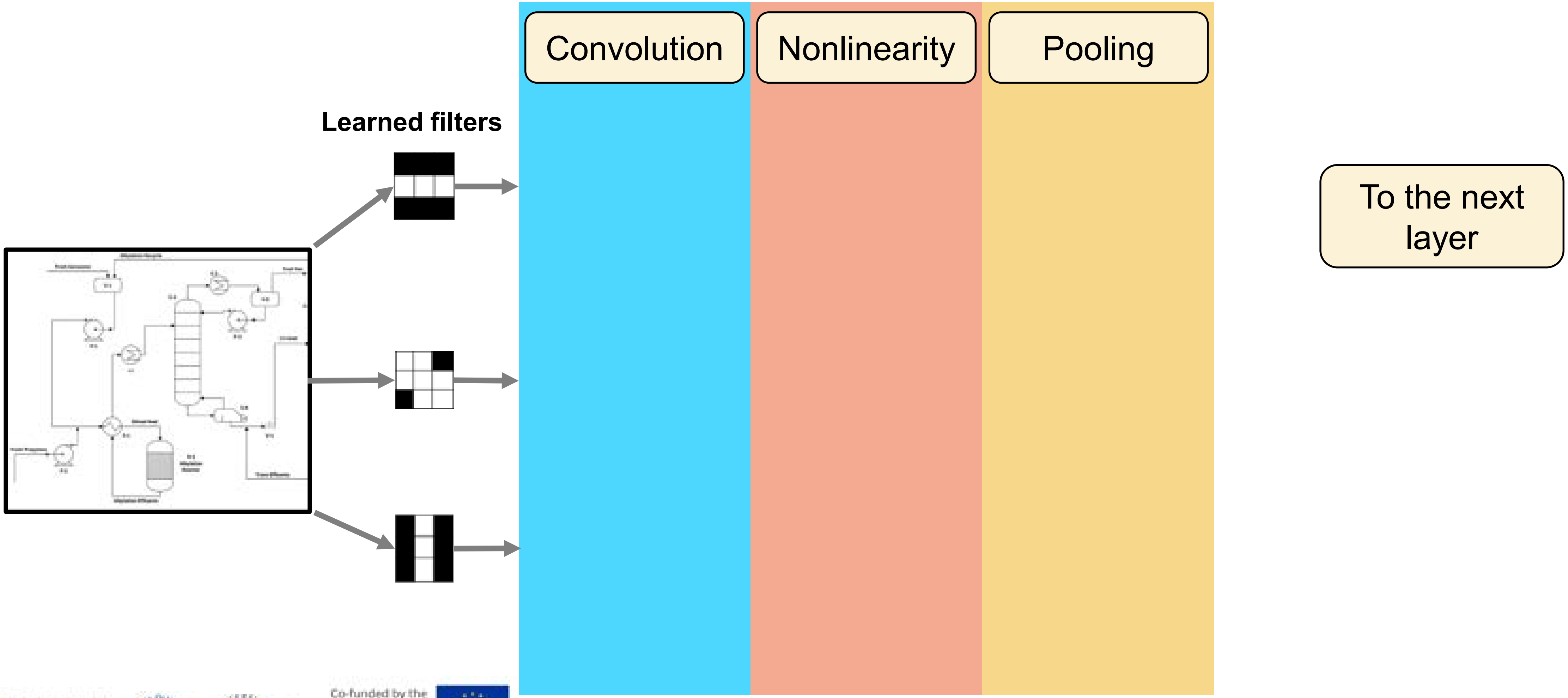


Modern computer vision models, i.e., convolutional neural networks (CNNs), consist of…

- …convolutions, used to extract features (edges, textures and patterns)
- …pooling, used to reduce dimensionality by downsampling
- …activation functions, used to introduce non-linearity
- …sampling techniques such as stride and padding

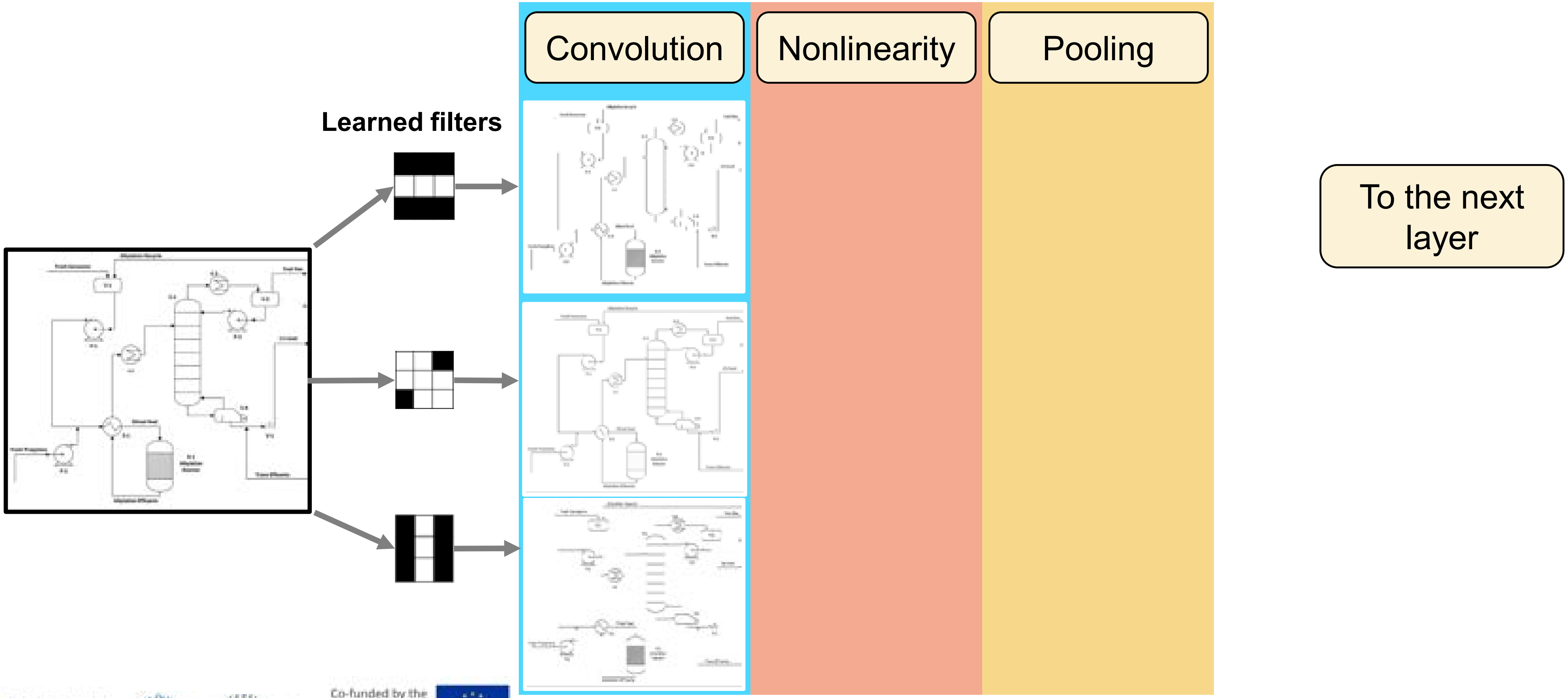
→ We do not need to manually define those filters. Instead, we let convolutional neural networks learn them!

How does this look like in a convolutional neural network?

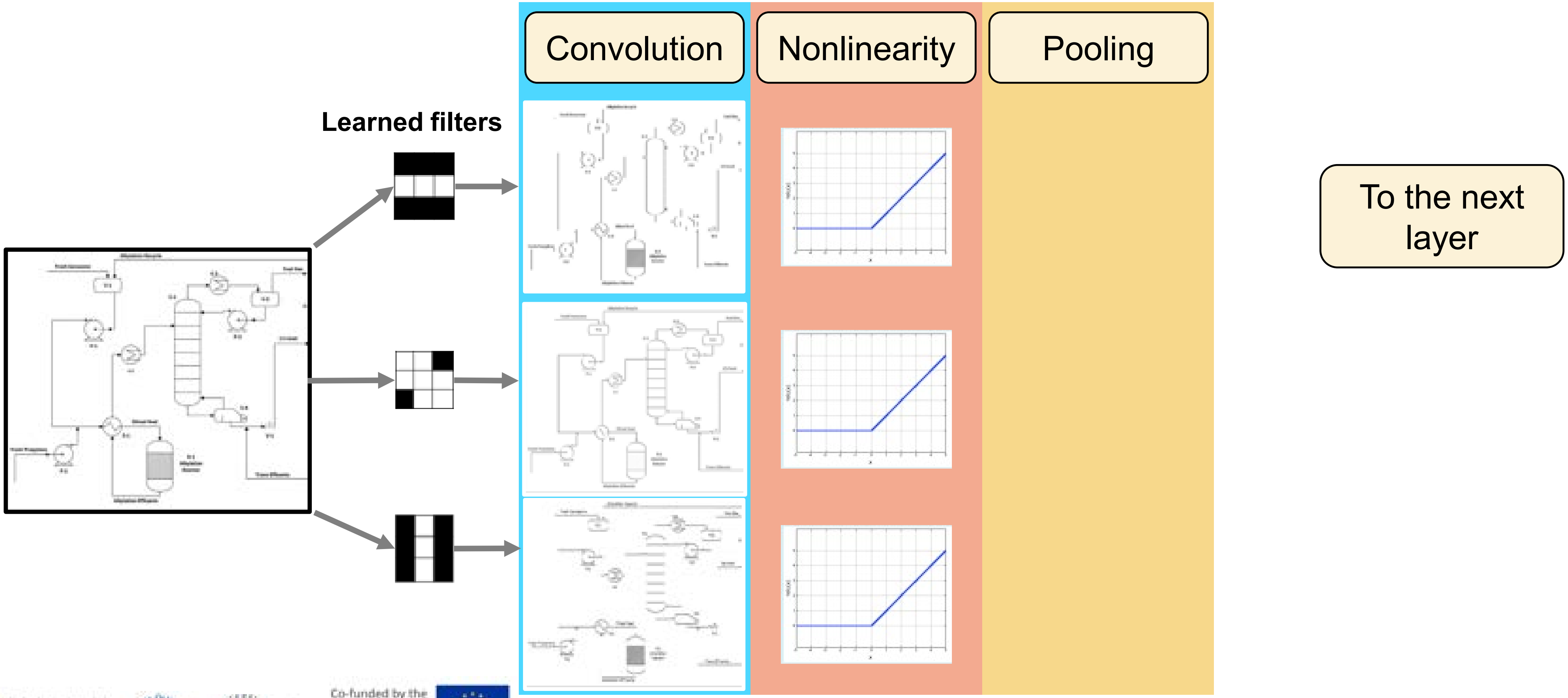


technische universität dortmund
TU Delft
Co-funded by the Erasmus+ programme of the European Union
MachineLearnAthon

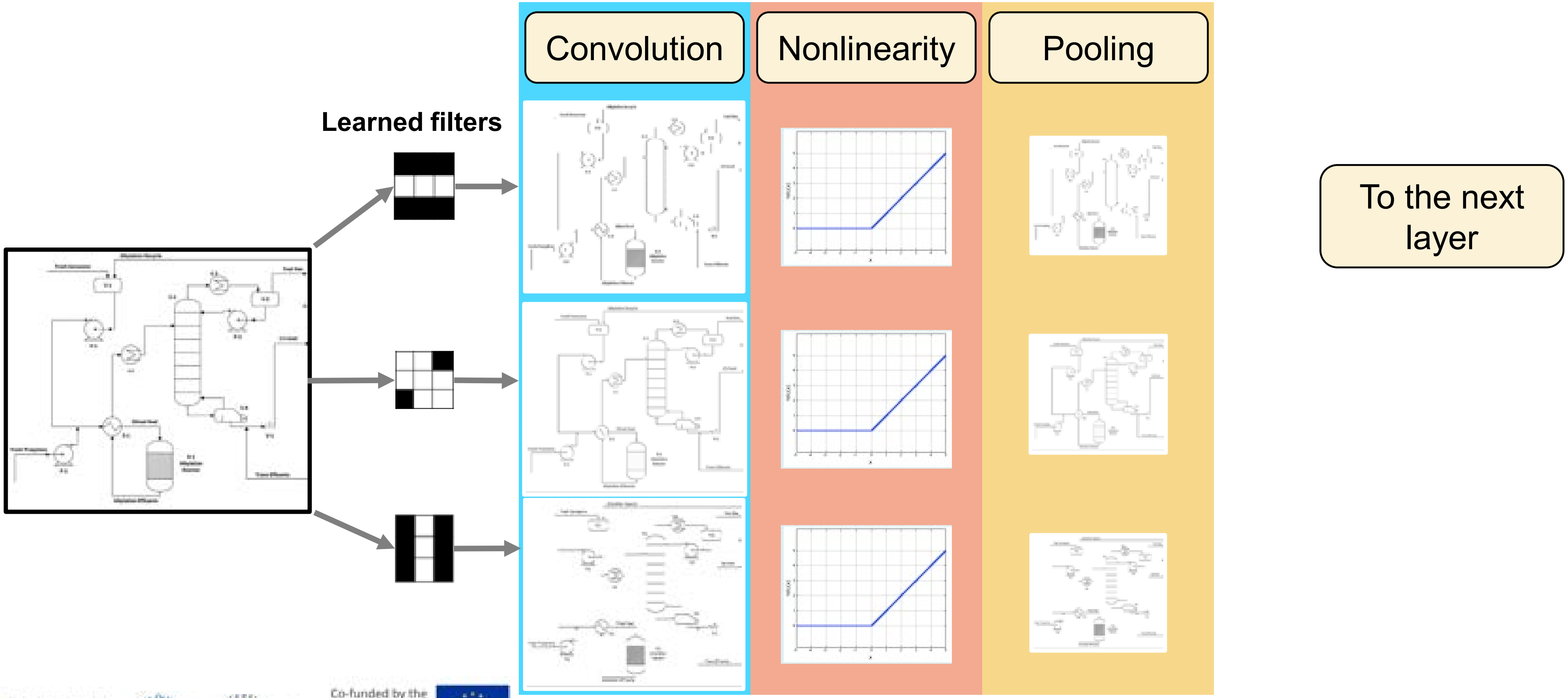
How does this look like in a convolutional neural network?



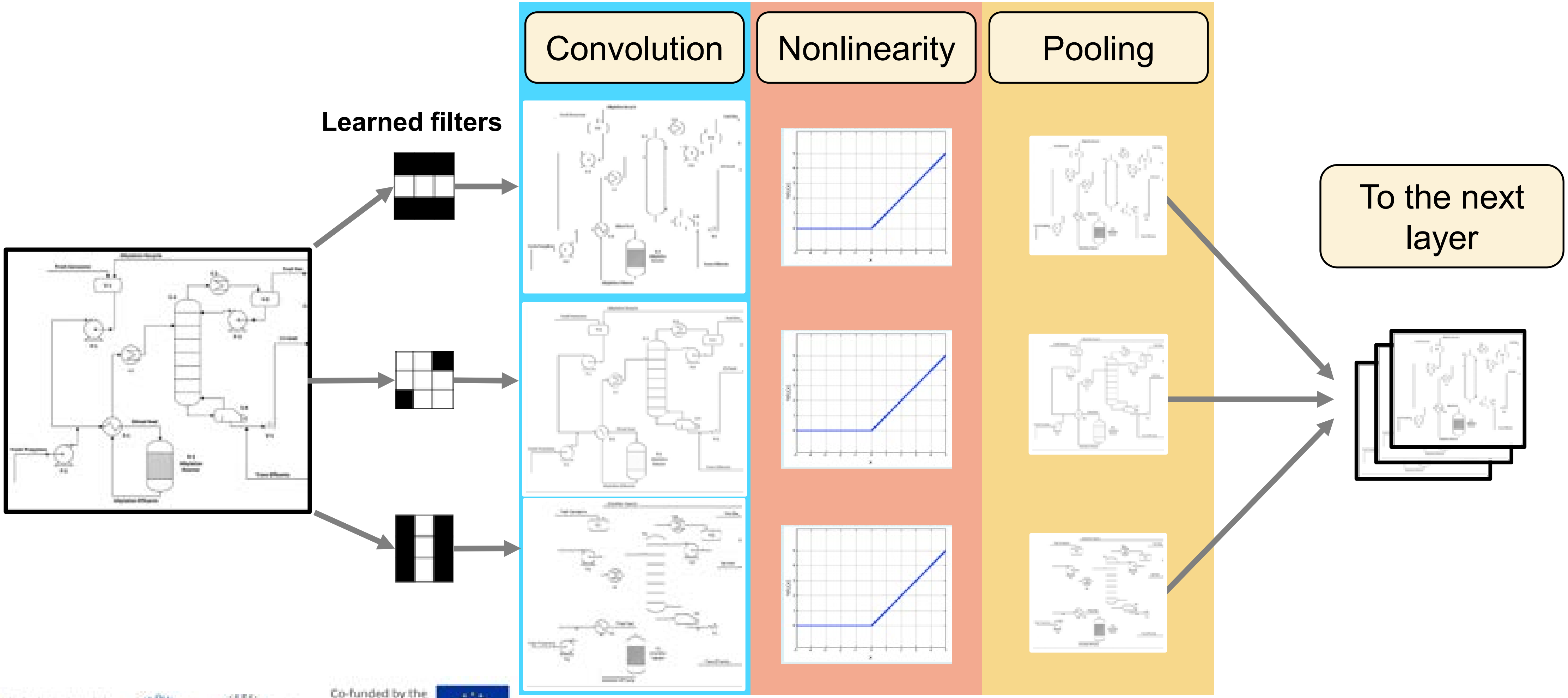
How does this look like in a convolutional neural network?



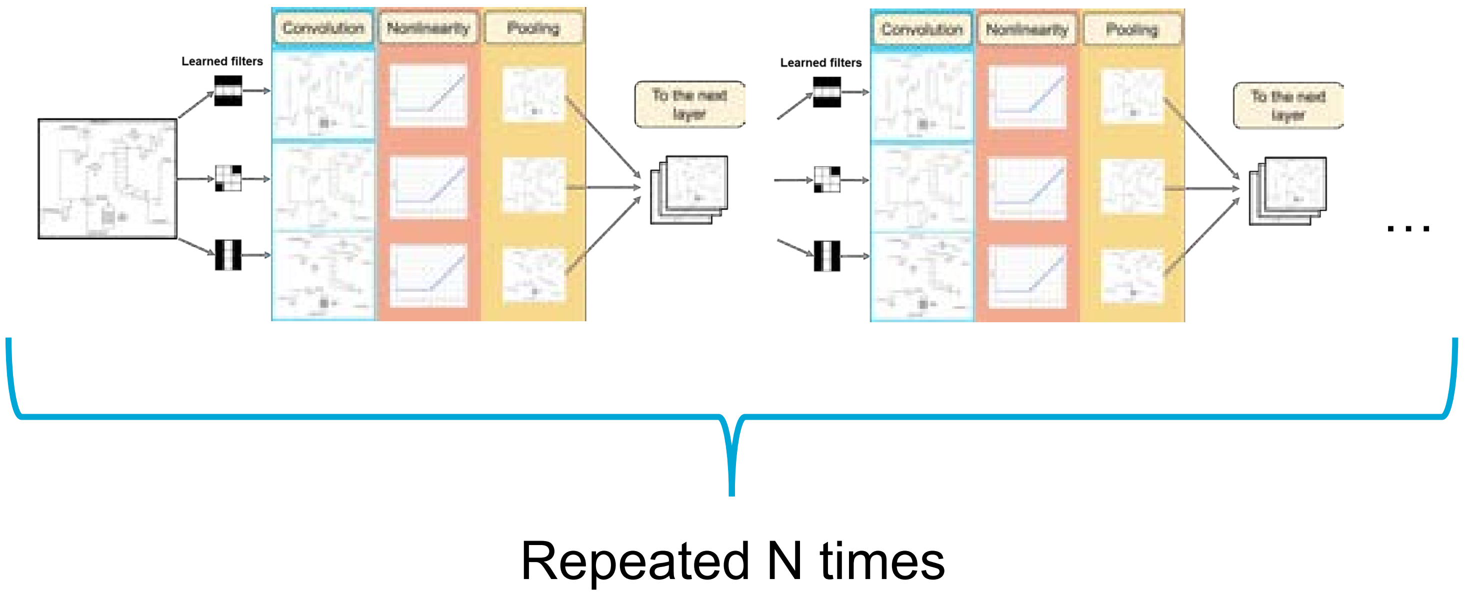
How does this look like in a convolutional neural network?

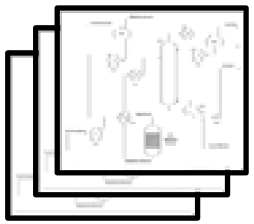


How does this look like in a convolutional neural network?



What do we do with the found feature maps?

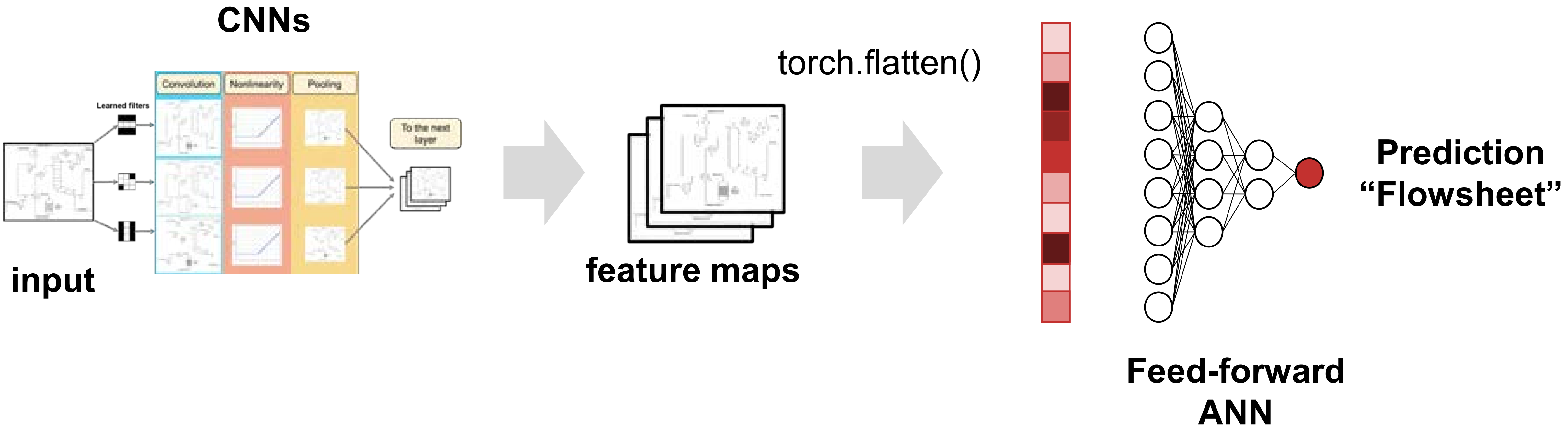




feature maps

What do we do with these feature maps?

We can flatten them into a vector and feed them to an feed forward neural networks!



The MLP can be a regression model or a classification model

Agenda

- What is computer vision?
- What do machines see?
- Important concepts in computer vision
  - Filter
  - Padding
  - Stride
  - Pooling
- Elements in deep learning computer vision algorithms
- **How does a modern computer vision model look like?**
- Outlook on today’s assignment

Let's look at an example: VGG16[1]

- VGG stands for Visual Geometry Group, it is a deep Convolutional Neural Net.
- The CNN is 16 layers 'deep' hence called VGG16.
- It is primarily used for *object detection* and *classification* algorithm.

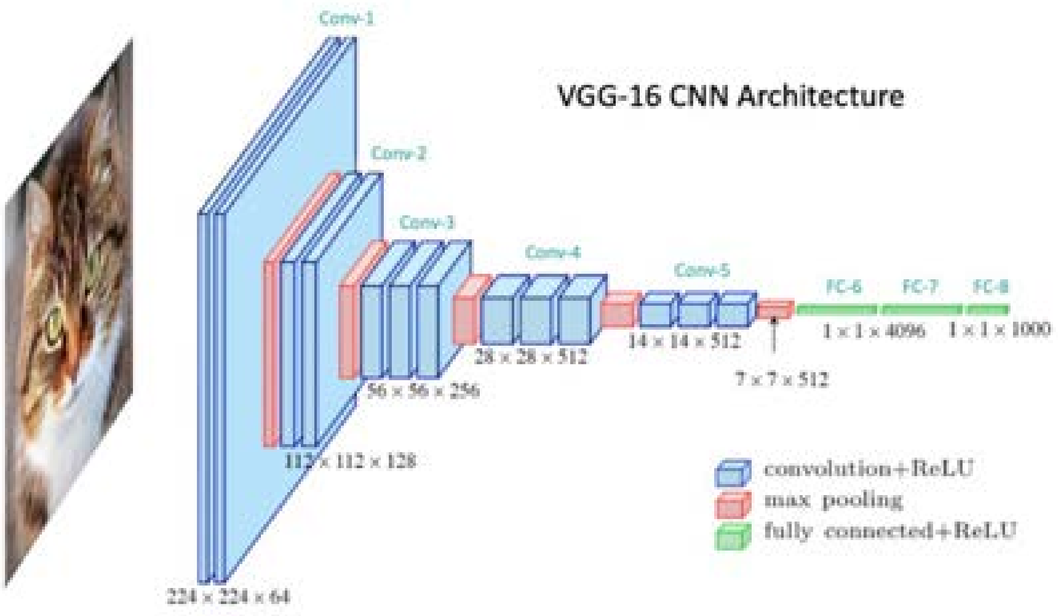
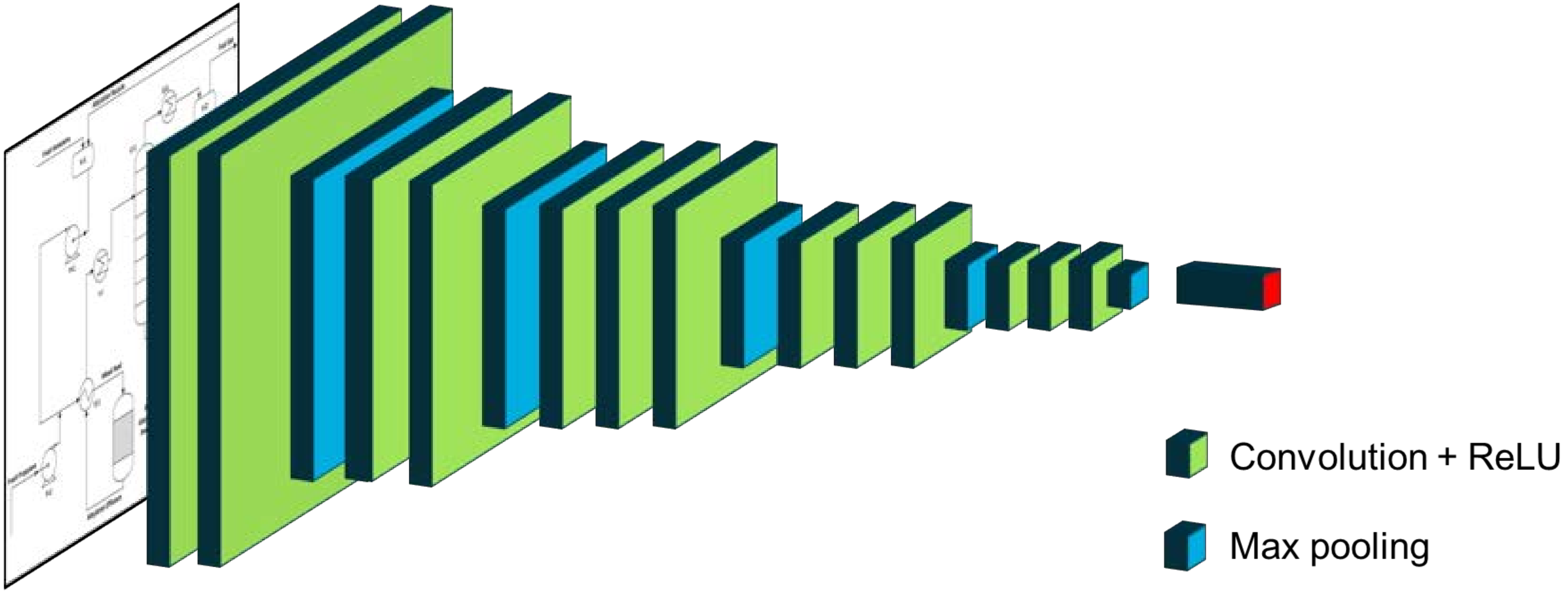


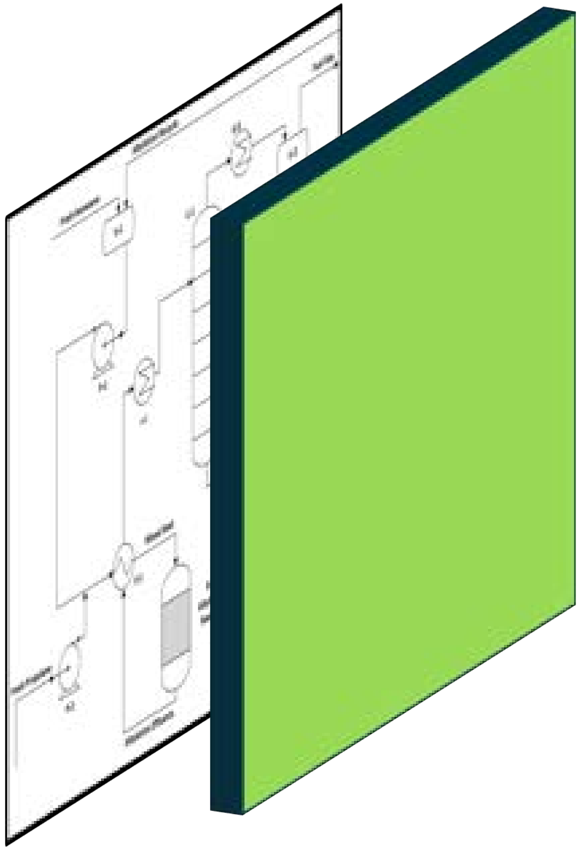
Image courtesy: https://learnopencv.com/tag/vgg16/

This is how it looks in a modern architecture – VGG161



Input image [224x224x3]

Exercise: What are the intermediate feature map dimensions?



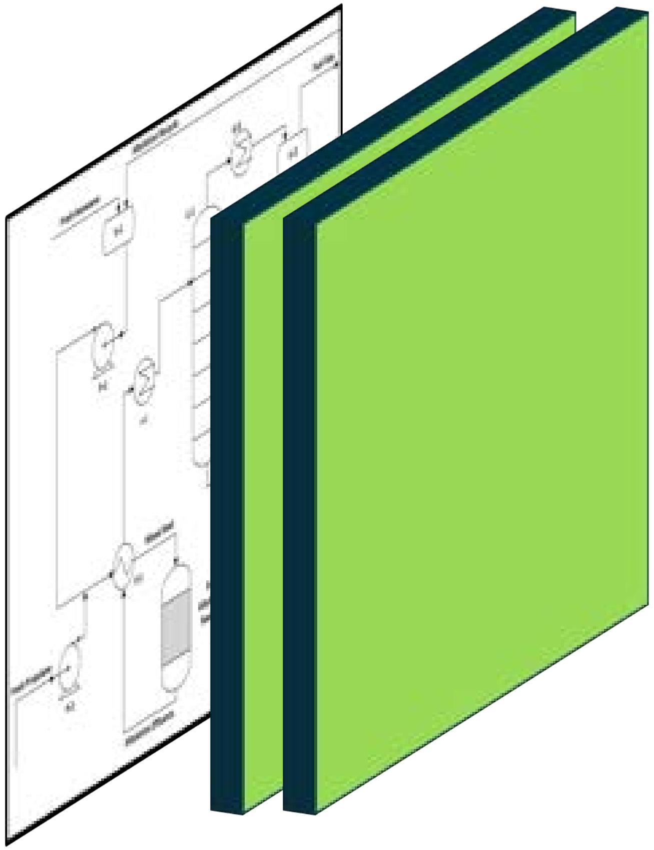
64 filters, kernel = [3,3], padding = 1

input feature map size = [224, 224, 3]

Output Dimension $= \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{224 - 3 + 2 * 1}{1} \right\rfloor + 1 = 224$

Resulting feature map size = [224,224,64]

Exercise: What are the intermediate feature map dimensions?



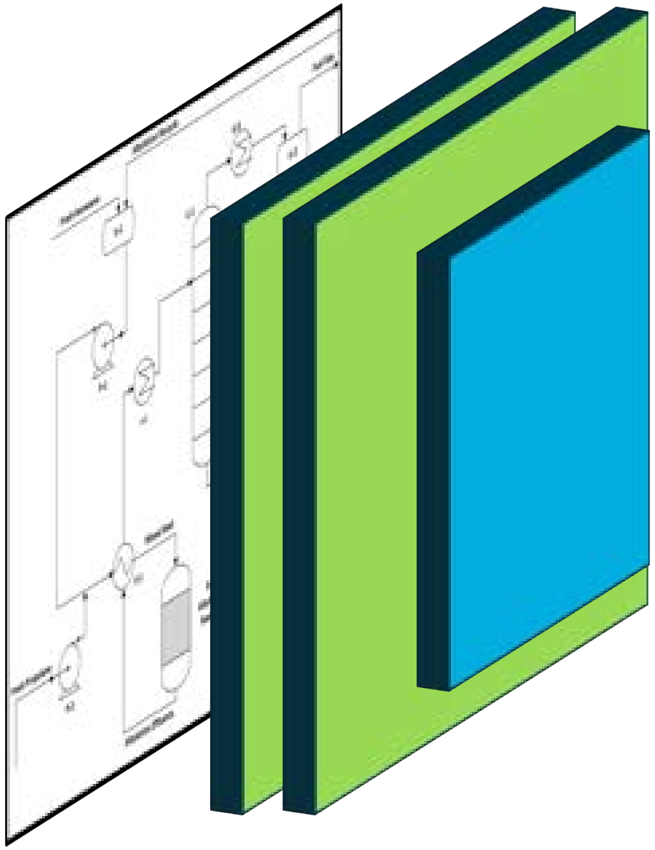
64 filters, kernel = [3,3], padding = 1

input feature map size = [224,224,64]

Output Dimension $= \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{224 - 3 + 2*1}{1} \right\rfloor + 1 = 224$

Resulting feature map size = [224,224,64]

Exercise: What are the intermediate feature map dimensions?



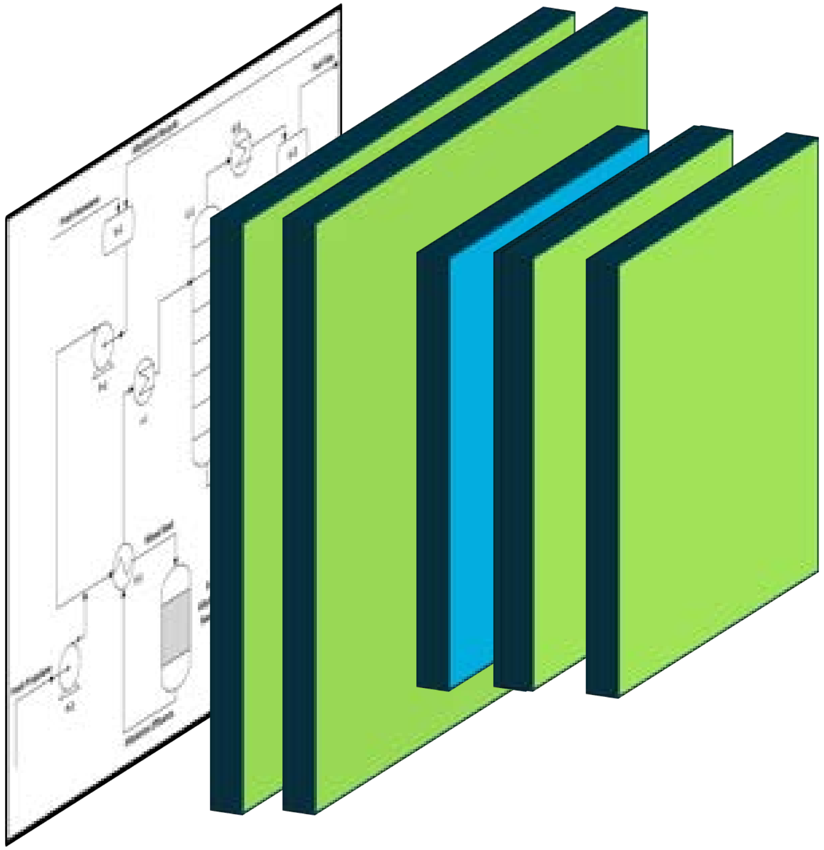
Max-pool size = [2,2], stride = [2,2]

input feature map size = [224,224,64]

Output Dimension $= \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{224-2+2*0}{2} \right\rfloor + 1 = 112$

Resulting feature map size = [112, 112, 64]

Exercise: What are the intermediate feature map dimensions?



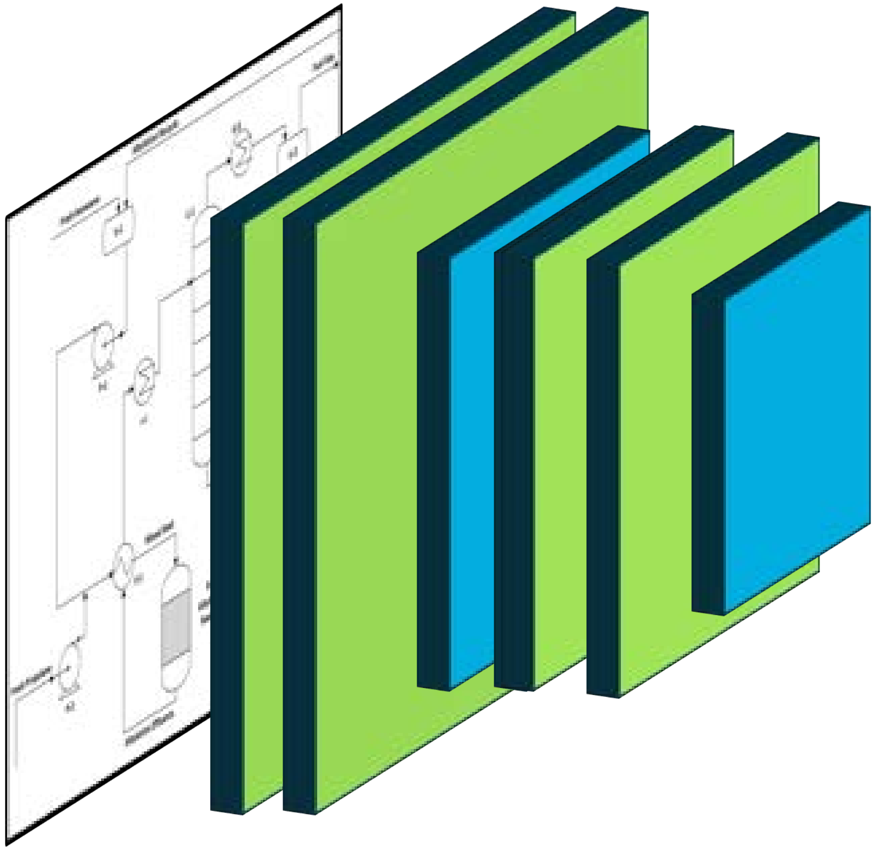
128 filters, kernel = [3,3], padding = 1

input feature map size = [112,112,64]

$$\text{Output Dimension} = \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{112-3+2*1}{1} \right\rfloor + 1 = 112$$

Resulting feature map size = [112, 112, 128]

Exercise: What are the intermediate feature map dimensions?



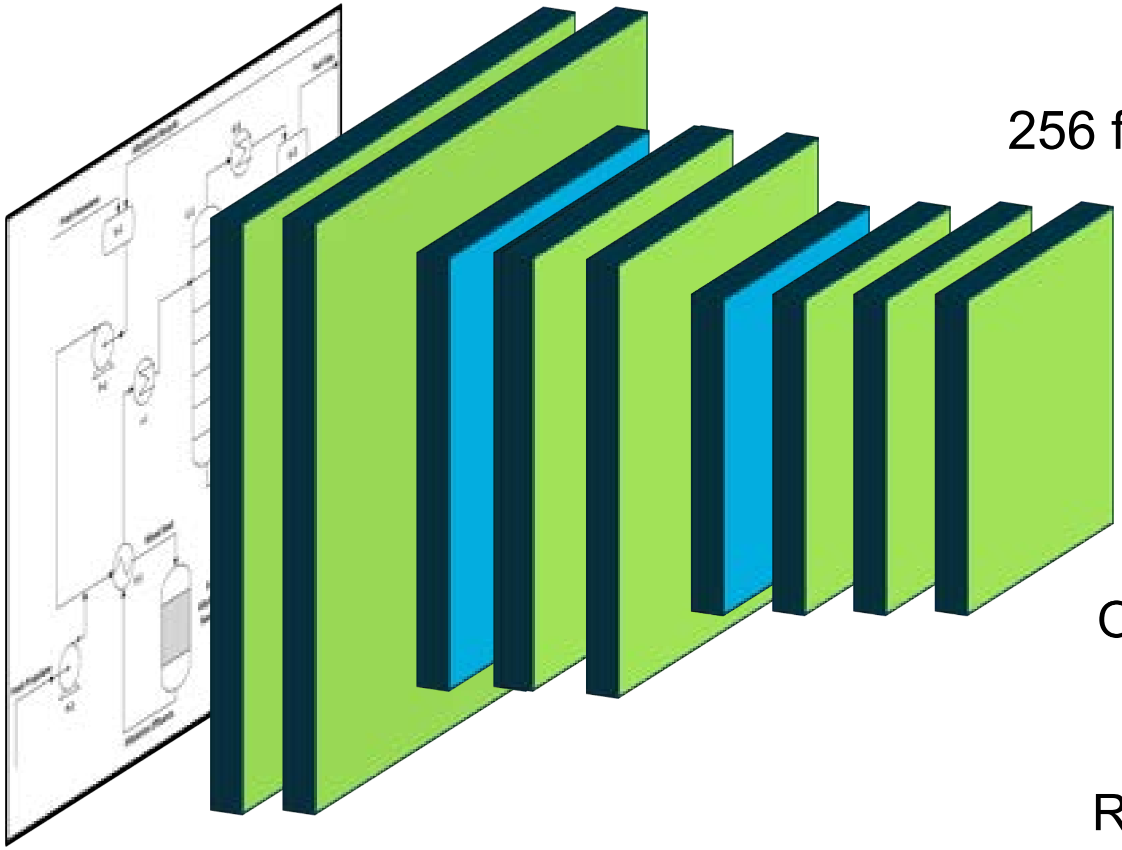
Max-pool size = [2,2], stride = [2,2]

input feature map size = [112,112,128]

Output Dimension $= \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{112-2+2*0}{1} \right\rfloor + 1 = 56$

Resulting feature map size = [56, 56, 128]

Exercise: What are the intermediate feature map dimensions?



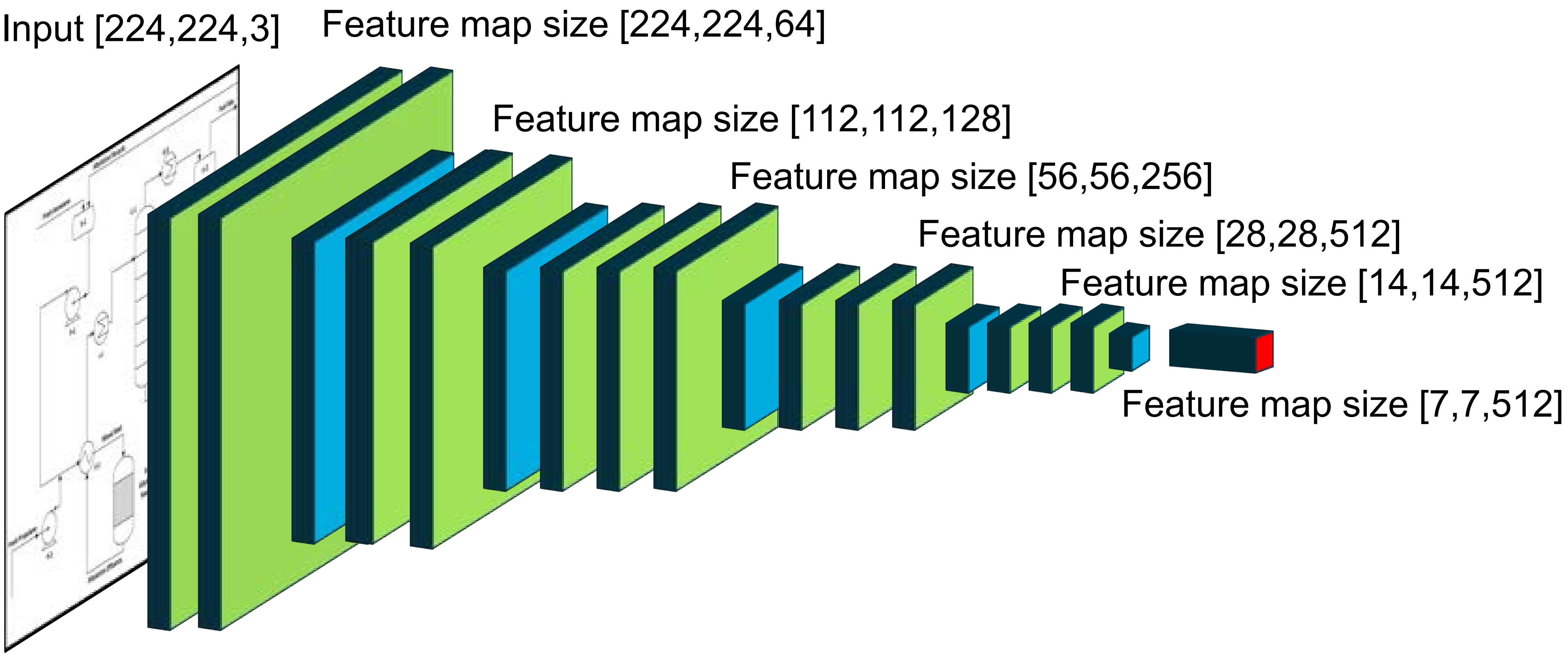
256 filters, kernel = [3,3], padding = 1

input feature map size = [112,112,128]

Output Dimension $= \left\lfloor \frac{I_p - F + 2P}{S} \right\rfloor + 1 = \left\lfloor \frac{112-3+2*1}{1} \right\rfloor + 1 = 56$

Resulting feature map size = [56, 56, 256]

…and so on, until we reach the final dimensionality



VGG16: Architecture

VGG16 already exists within PyTorch, so we import by a function call.

```python
import torch
import torch.nn as nn
from torchvision import models
from torchsummary import summary

vgg = models.vgg16() #
summary(vgg, (3, 224, 224))
```

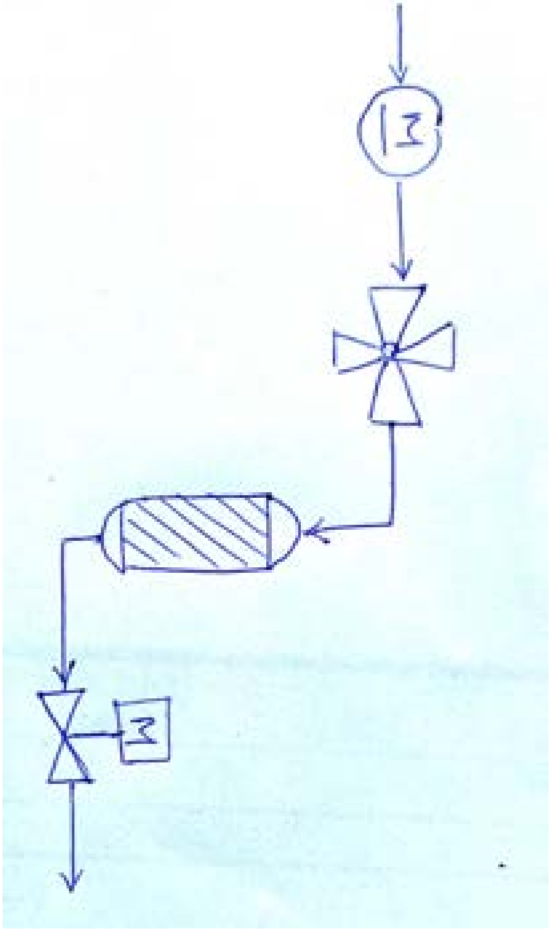
Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 224, 224]	1,792
ReLU-2	[-1, 64, 224, 224]	0
Conv2d-3	[-1, 64, 224, 224]	36,928
ReLU-4	[-1, 64, 224, 224]	0
MaxPool2d-5	[-1, 64, 112, 112]	0
Conv2d-6	[-1, 128, 112, 112]	73,856
ReLU-7	[-1, 128, 112, 112]	0
Conv2d-8	[-1, 128, 112, 112]	147,584
ReLU-9	[-1, 128, 112, 112]	0
MaxPool2d-10	[-1, 128, 56, 56]	0
Conv2d-11	[-1, 256, 56, 56]	295,168
ReLU-12	[-1, 256, 56, 56]	0
Conv2d-13	[-1, 256, 56, 56]	590,080
ReLU-14	[-1, 256, 56, 56]	0
Conv2d-15	[-1, 256, 56, 56]	590,080
ReLU-16	[-1, 256, 56, 56]	0
MaxPool2d-17	[-1, 256, 28, 28]	0
Conv2d-18	[-1, 512, 28, 28]	1,180,160
ReLU-19	[-1, 512, 28, 28]	0
Conv2d-20	[-1, 512, 28, 28]	2,359,808
ReLU-21	[-1, 512, 28, 28]	0
Conv2d-22	[-1, 512, 28, 28]	2,359,808

Agenda

- What is computer vision?
- What do machines see?
- Important concepts in computer vision
  - Filter
  - Padding
  - Stride
  - Pooling
- Elements in deep learning computer vision algorithms
- How does a modern computer vision model look like?
- **Outlook on today's assignment**

Outlook: Assignment 5 on computer vision

- In Assignment 5, you will…
- Implement computer vision operations by yourself
- Test different architectures for classification
- Try out VGG-16!
- Work with a real dataset



Thank you very much for your attention!