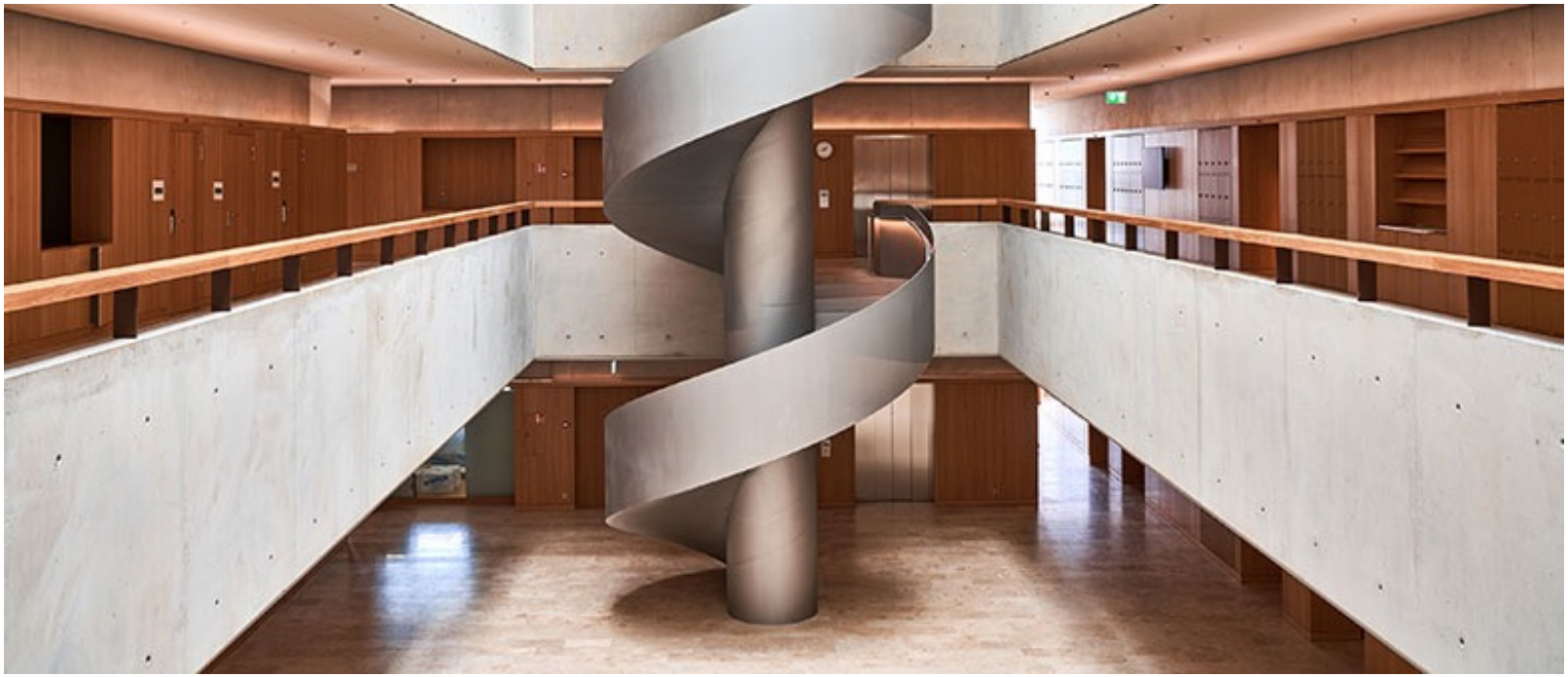
HSLU Hochschule Luzern



CAS Machine Learning

Deep Learning: From Core Foundations to Applications

Javier Montoya, Dr. sc. ETH
javier.montoya@hslu.ch

Core Foundation

Artificial Intelligence

- History
- Perceptron
- Neurons and Layers
- Activation Functions

Neural Networks

- Architectures
- Optimization: Forward/Backprop.
- Loss Functions

Training in Practice

- Regularization
- Train/Validation/Test

Deep Learning meets Computer Vision

Foundations

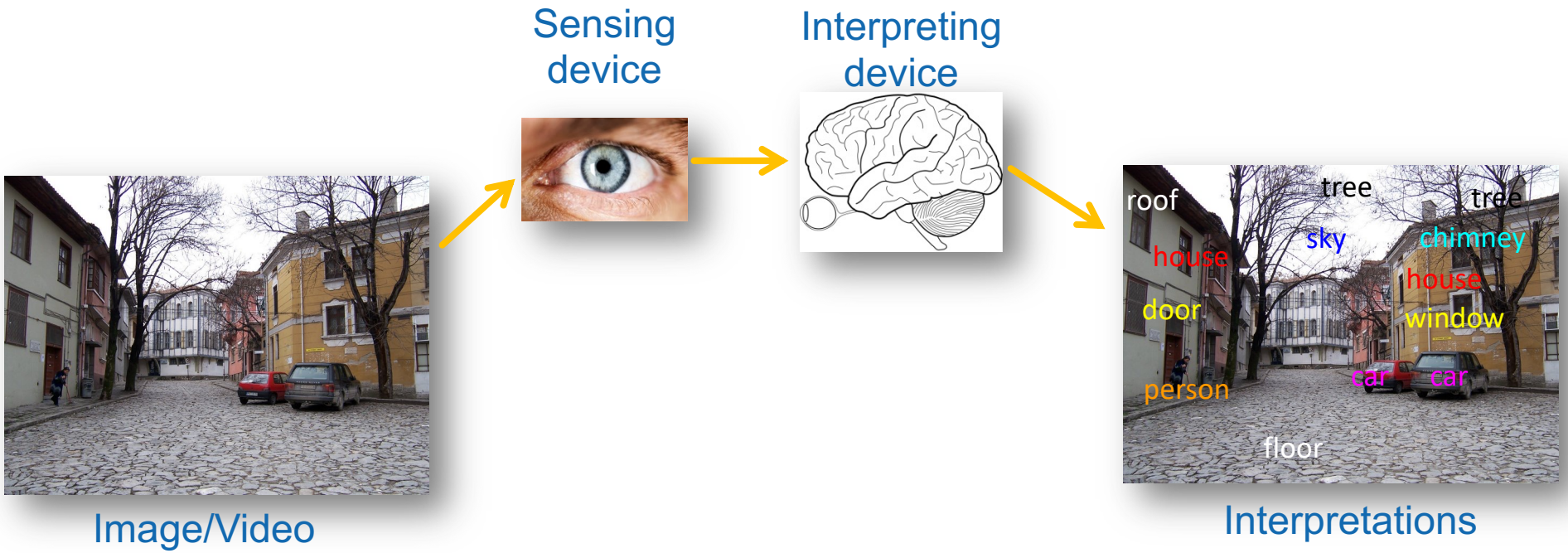
- What's Computer Vision?
- Challenges

Convolutional Neural Networks

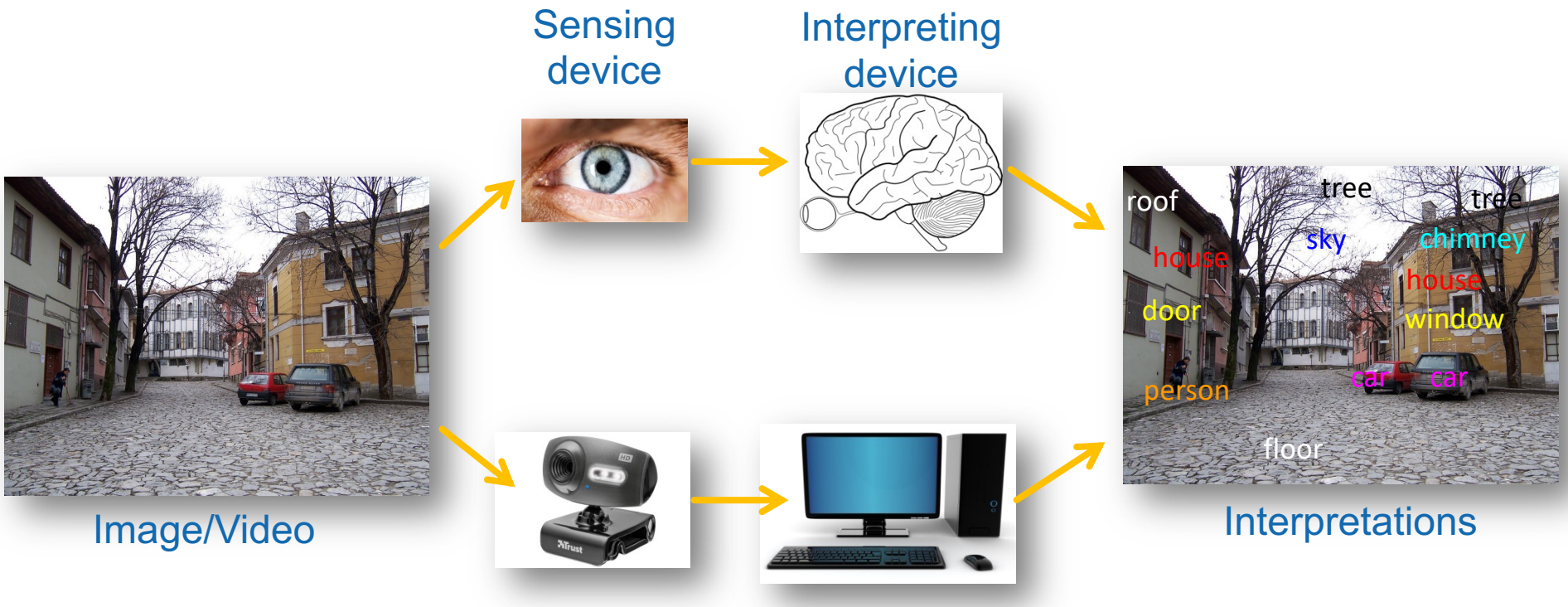
- Architectures
- Layers
- Loss Functions
- Common Tasks

Deep Learning meets Computer Vision

Computer Vision



Computer Vision

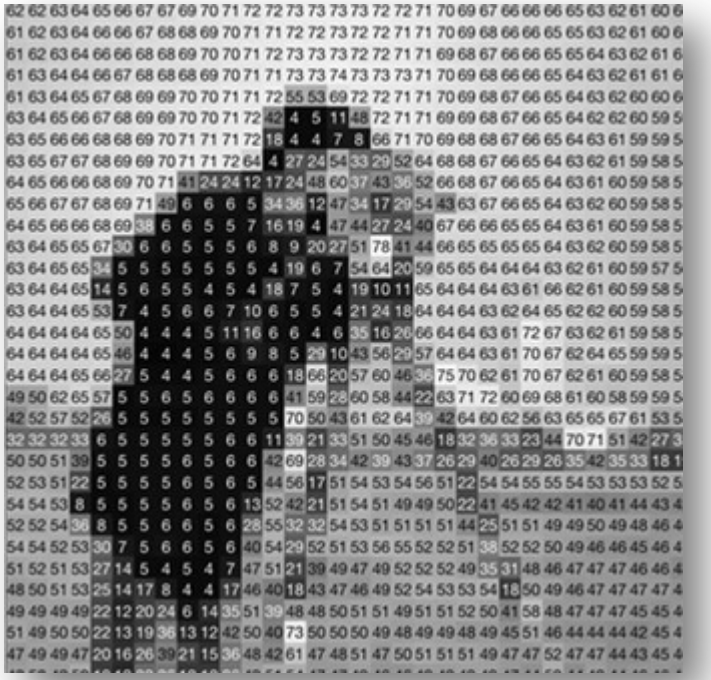


Digital Representation of Images

What humans see

Digital Representation of Images

What humans see



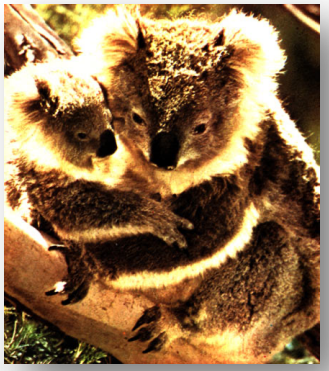
What computers see

Computer Vision: Challenges

Koala

Computer Vision: Challenges

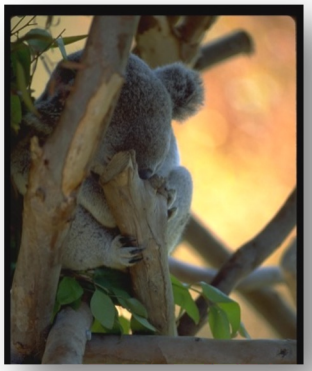
Koala



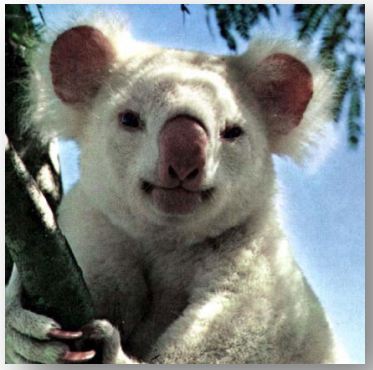
Illumination

Object Pose

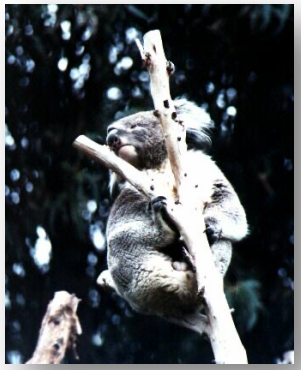
Clutter



Oclussions



Intra-class appeareance



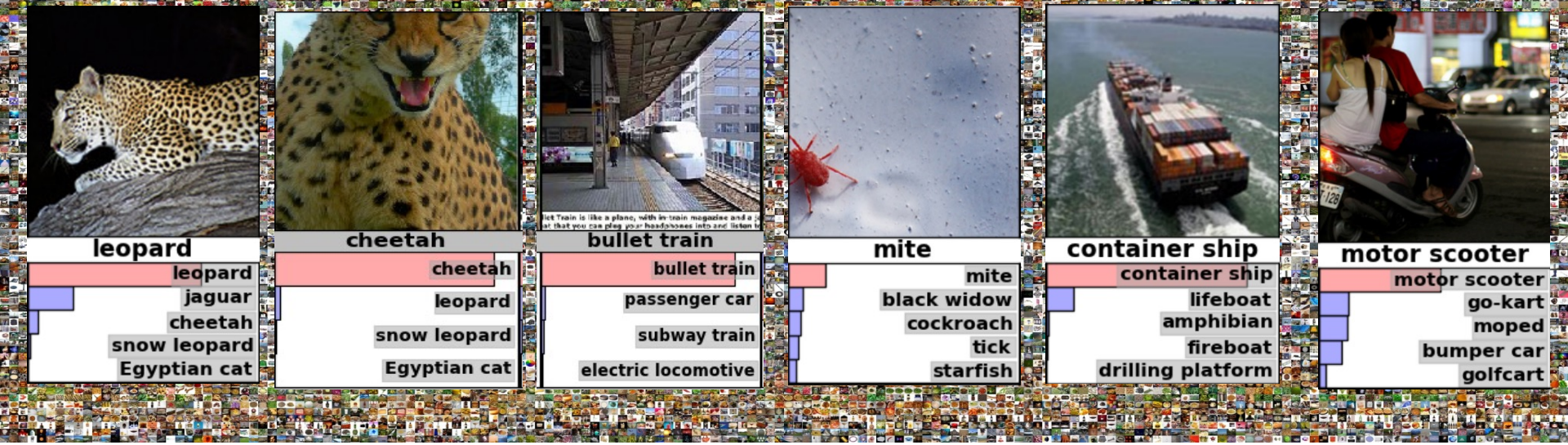
Viewpoint

Imagenet Challenge

The Image Classification Challenge:

1 000 Object Classes

1 431 167 Images

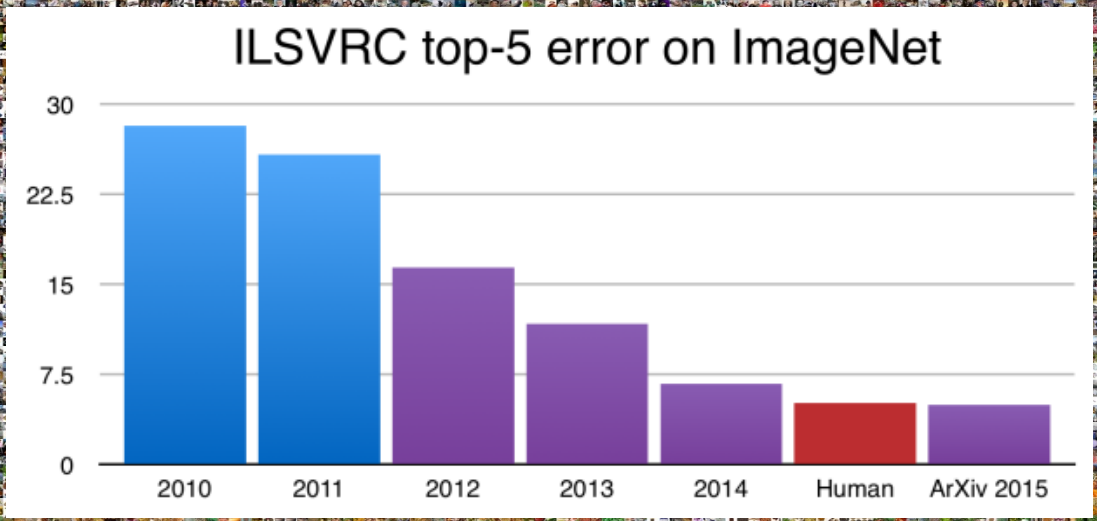


Imagenet Challenge

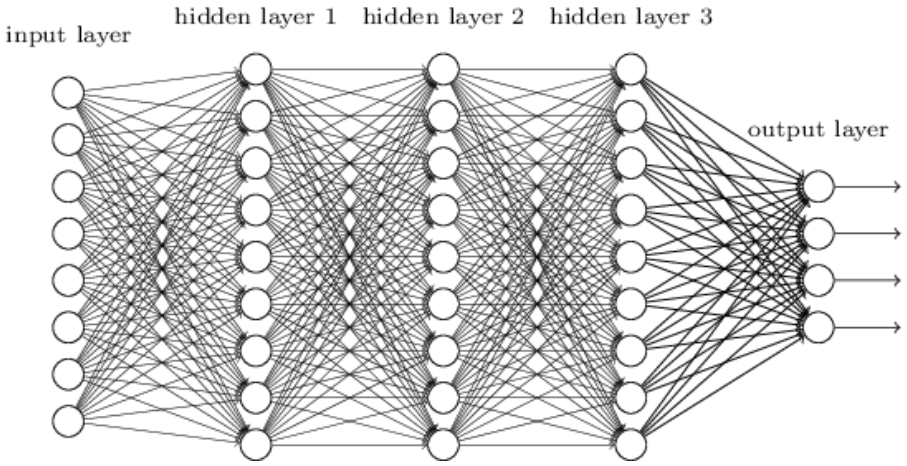
The Image Classification Challenge:

1 000 Object Classes

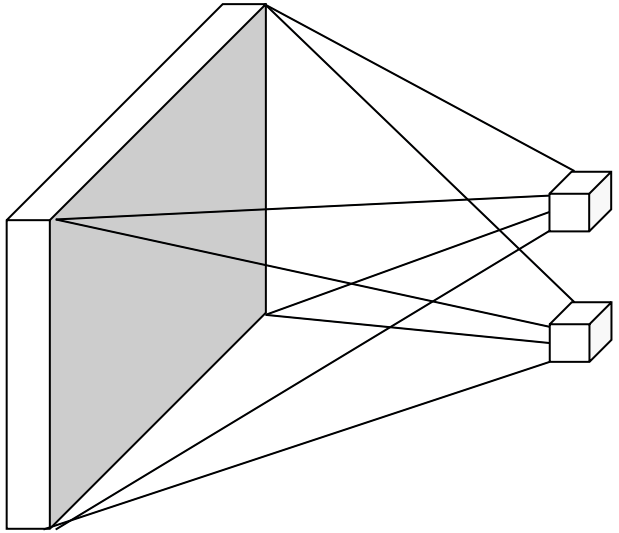
1 431 167 Images



Convolutional Neural Networks



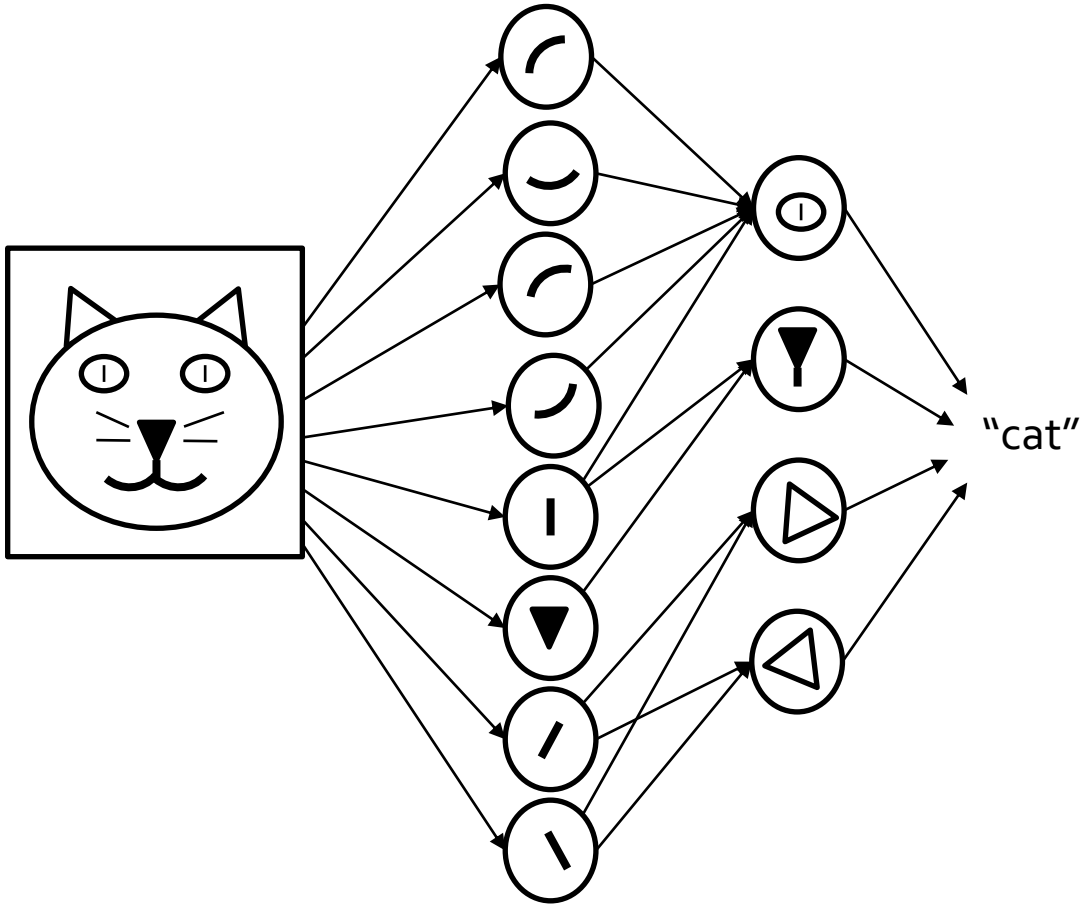
Multilayer Perceptron



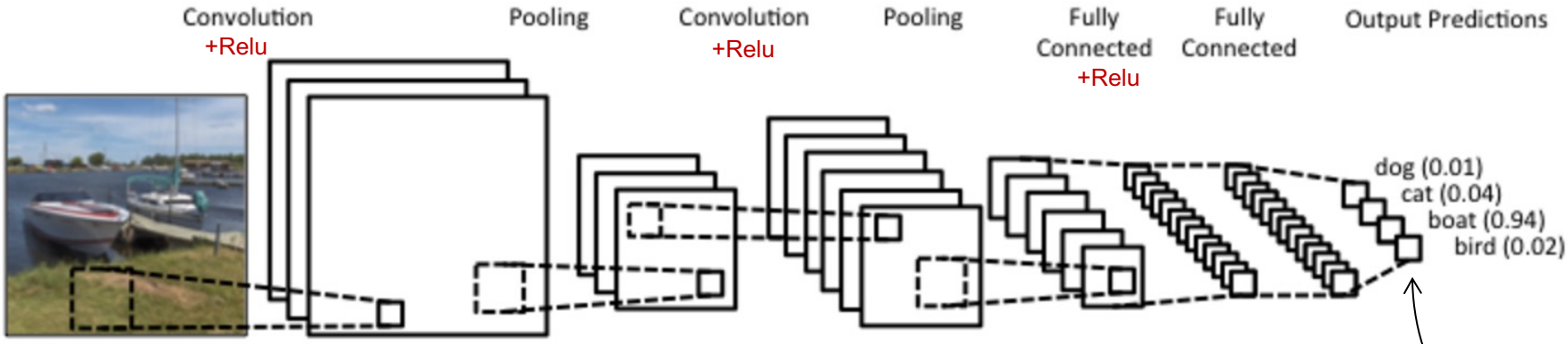
Convolutional Neural Network (CNN)

Convolutional Neural Networks

- We use convolutional layers to capture local spatial information.
- We can use a sequence of those layers.
- The stacking of successive layers give us a hierarchical representation.



Convolutional Neural Networks



Softmax layer:

$$P(c \mid \mathbf{x}) = \frac{\exp(\mathbf{w}_c \cdot \mathbf{x})}{\sum_{k=1}^{C} \exp(\mathbf{w}_k \cdot \mathbf{x})}$$

Convolution

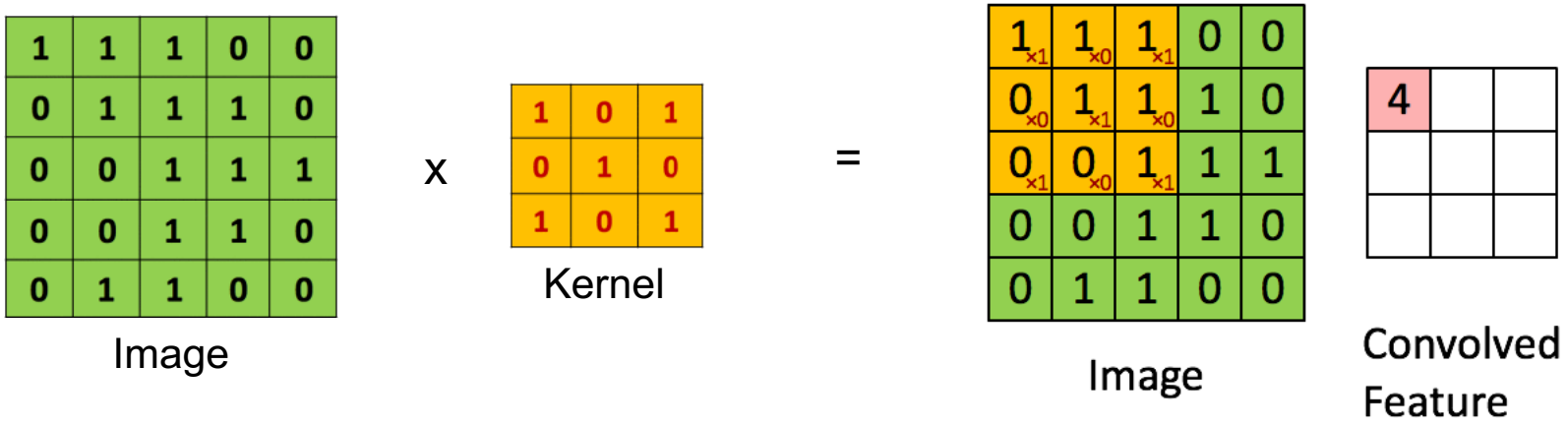
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Image

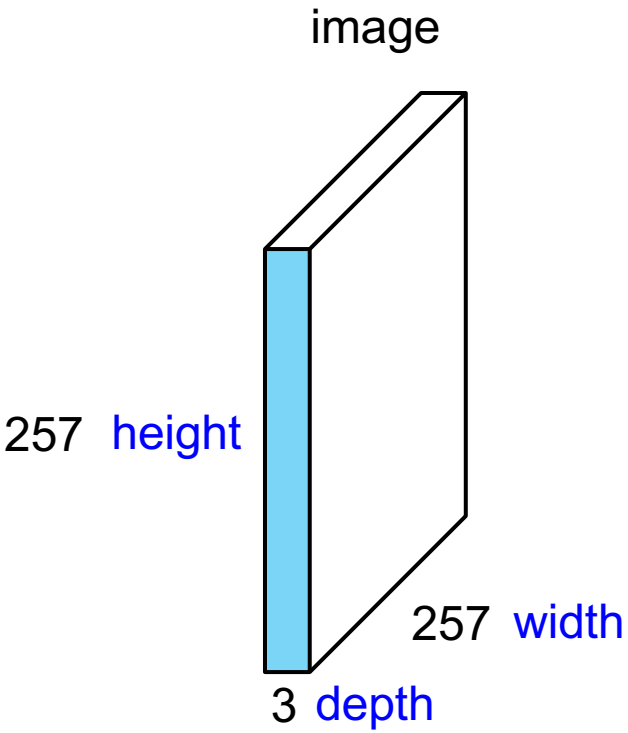
1	0	1
0	1	0
1	0	1

Kernel

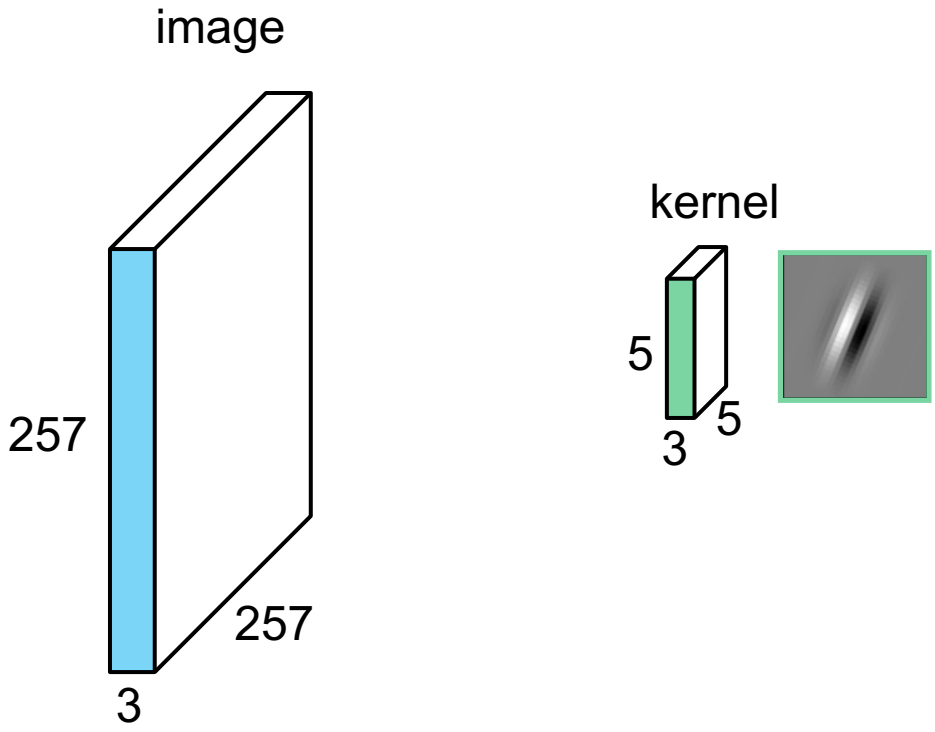
Convolution



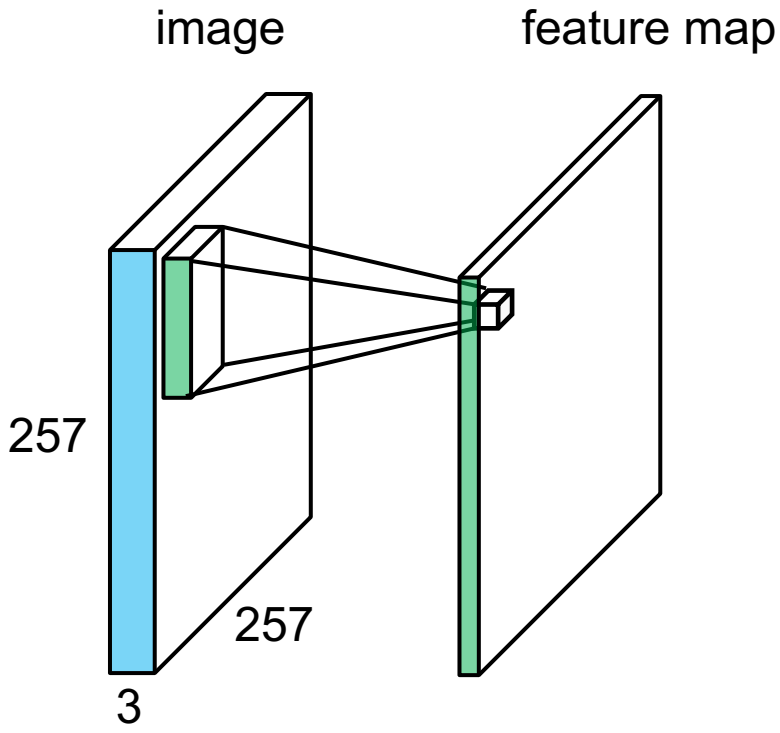
Convolutional Neural Networks



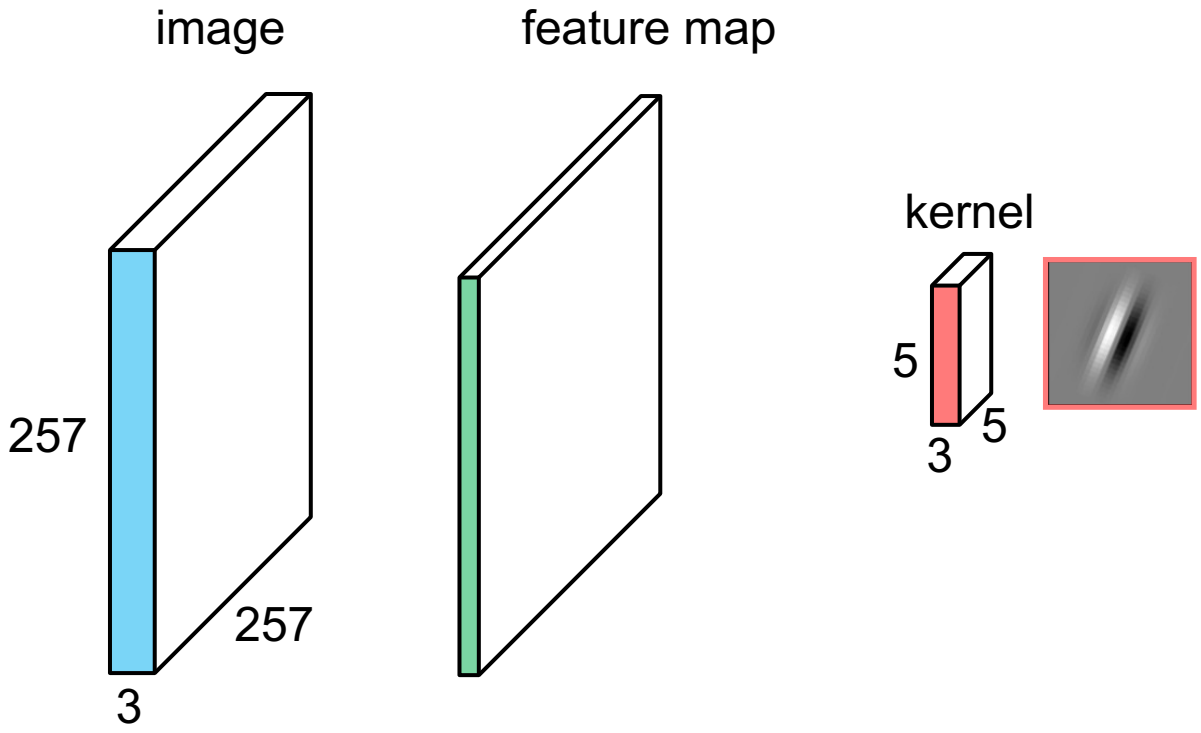
Convolutional Neural Networks



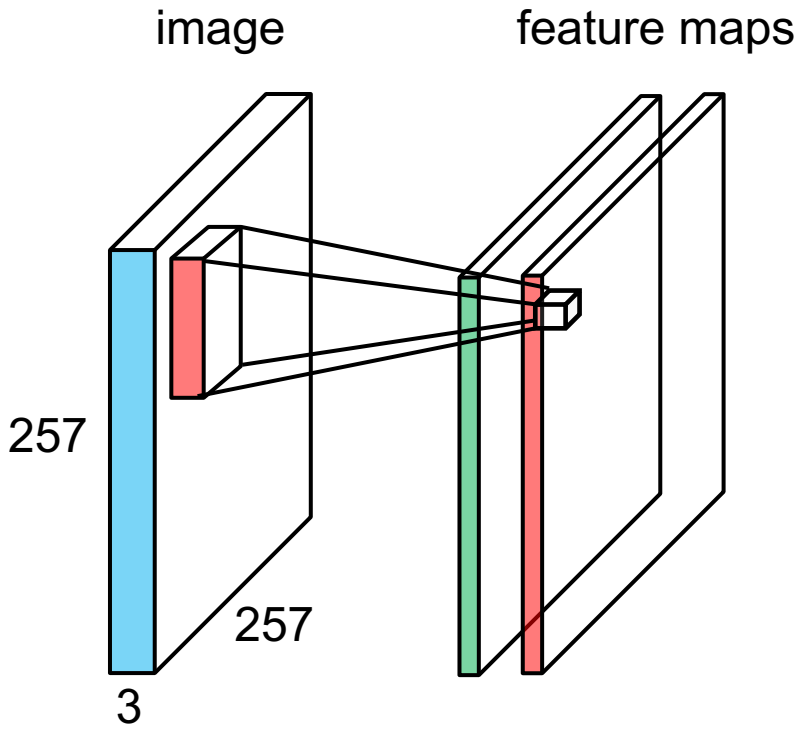
Convolutional Neural Networks



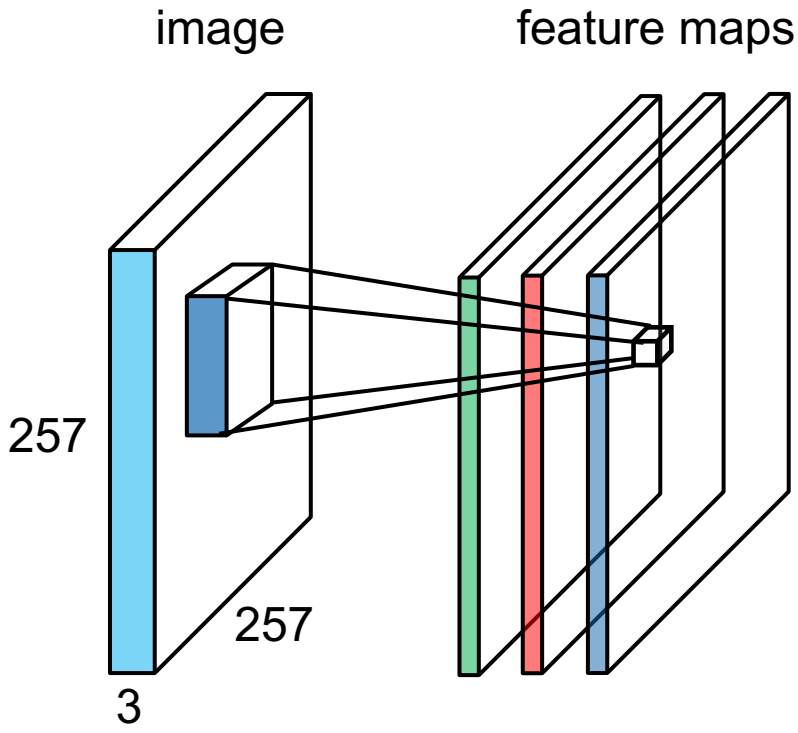
Convolutional Neural Networks



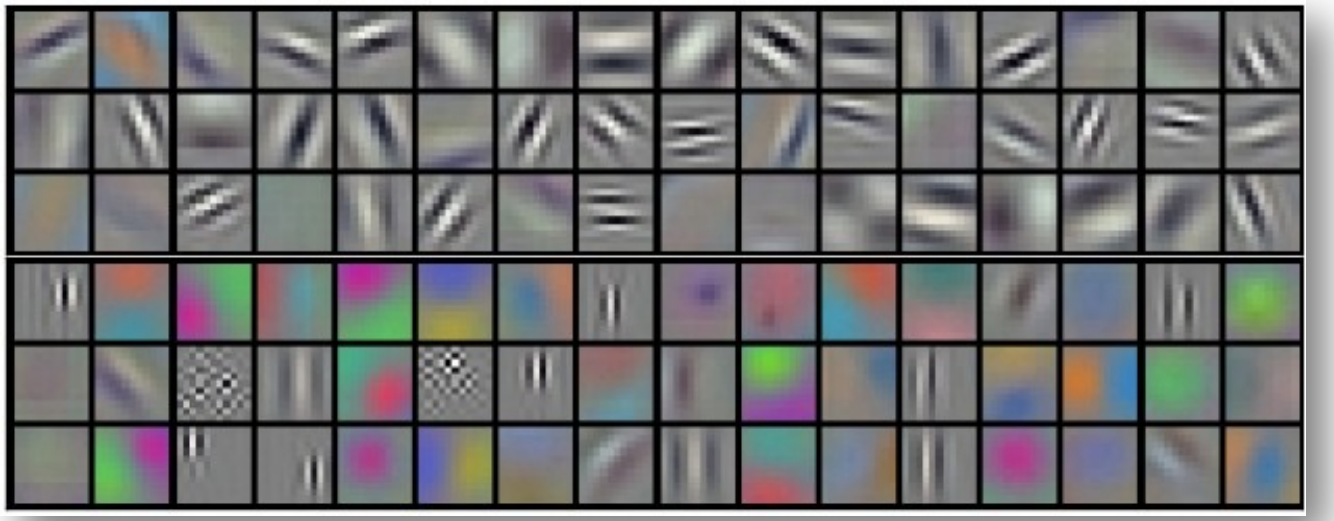
Convolutional Neural Networks



Convolutional Neural Networks

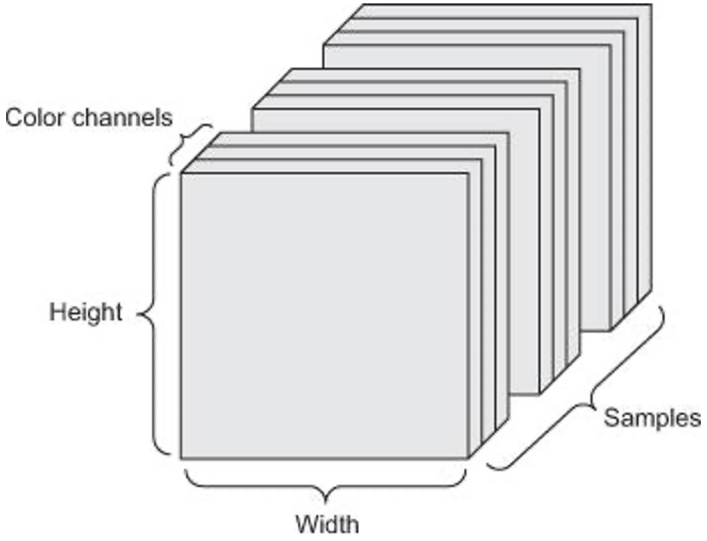


Convolutional Neural Networks

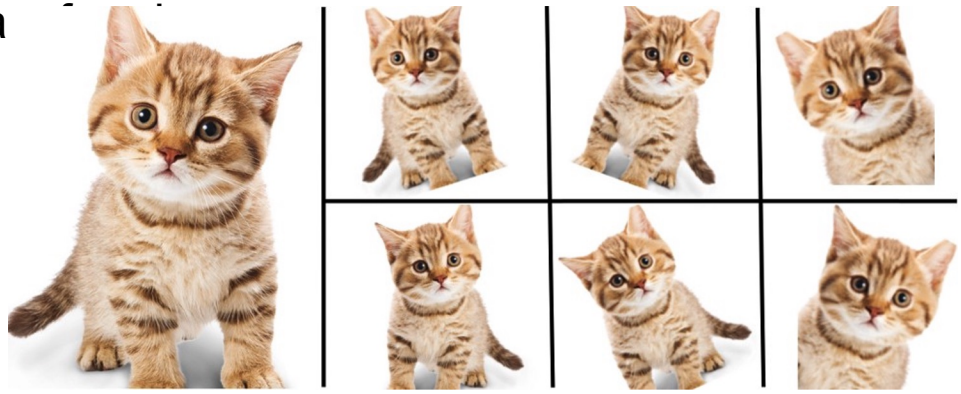


Examples of learned kernels/weights

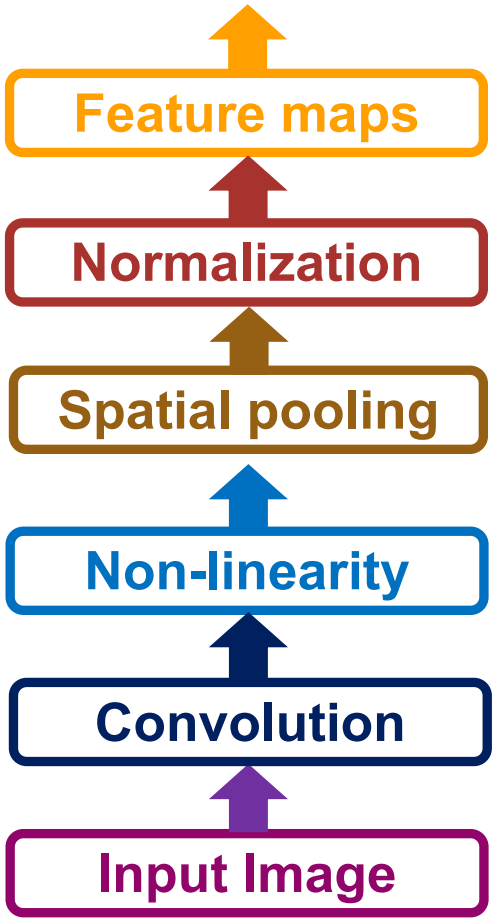
Convolutional Neural Networks

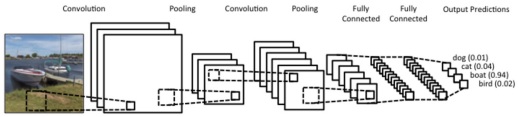


- Color image mini-batches are 4D tensors:
  width × height × color channels × samples
- Plenty of big datasets for training exist, e.g., ImageNet with 1,2 million images in 1000 classes
- Data augmentation for small datasets: generate more training data by tra existing data
- E.g., shifting, rotation, cropping, Scaling, adding noise, etc …

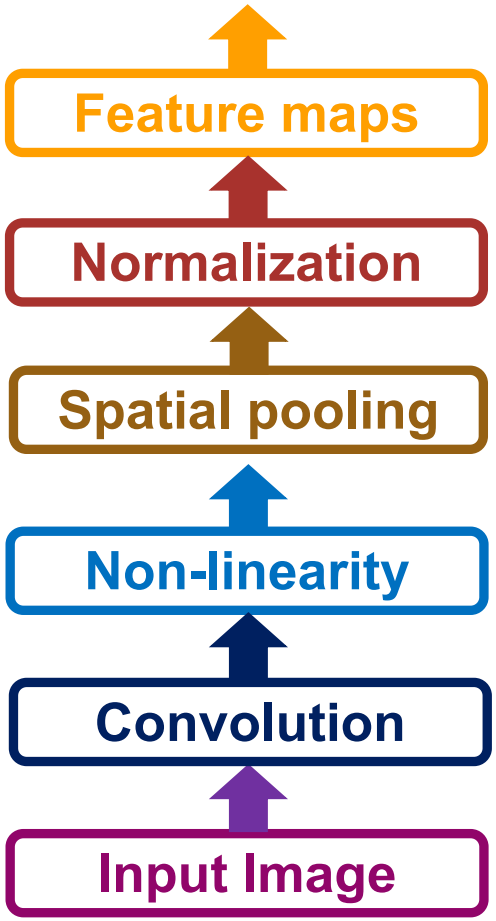


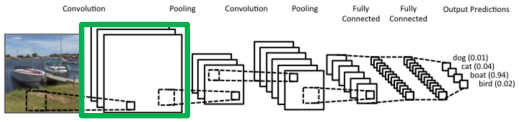
Convolutional Neural Networks





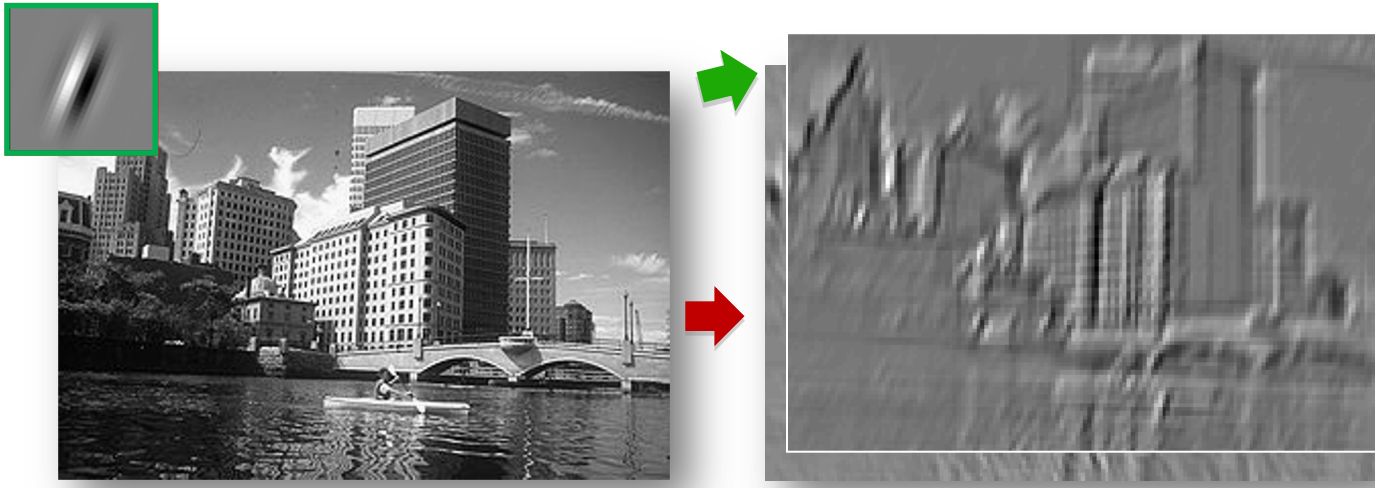
Convolutional Neural Networks





Convolution

- Apply learned filter weights.
- One feature map per filter.

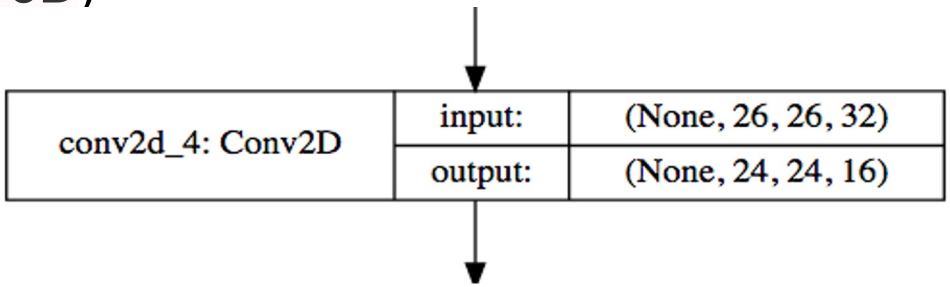


Convolutional Neural Networks

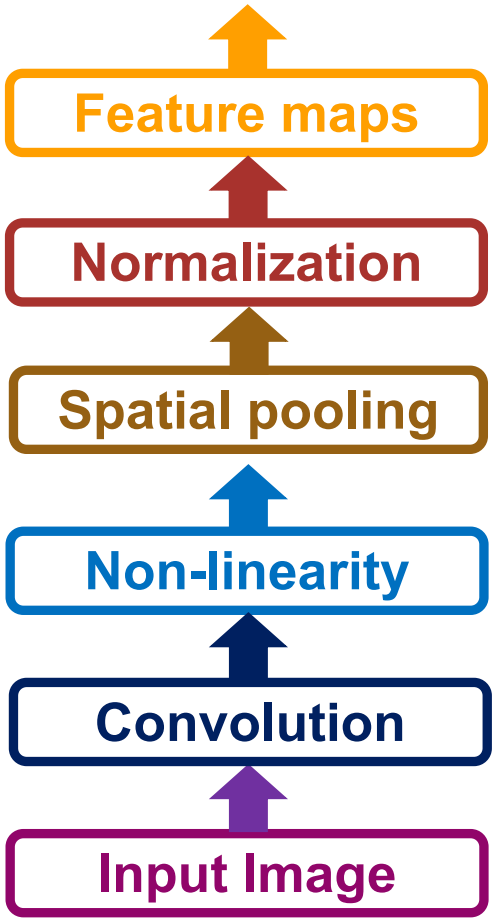
- Input: tensor of size $N \times W_i \times H_i \times C_i$
- Hyperparameters:
  - K: number of filters
  - w, h: kernel size
  - padding: how to handle image borders
  - activation function
- Output: tensor of size $N \times W_o \times H_o \times K$
- In `tf.keras`:

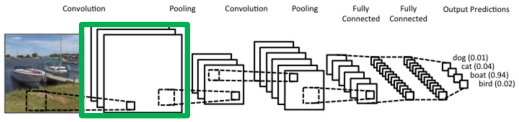
```
layers.Conv2D(filters, kernel_size,
              padding, activation)
```

(there is also `Conv1D` and `Conv3D`)



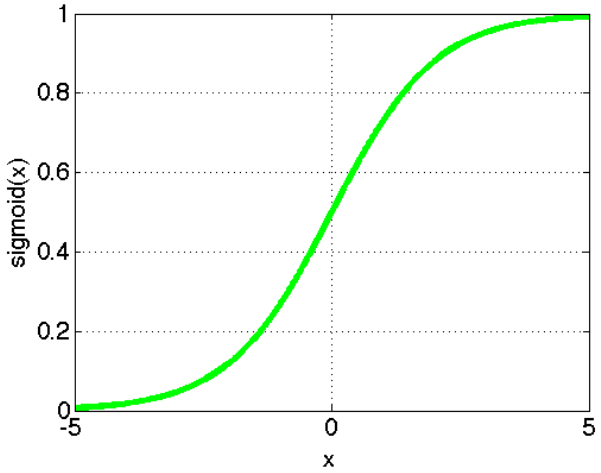
Convolutional Neural Networks



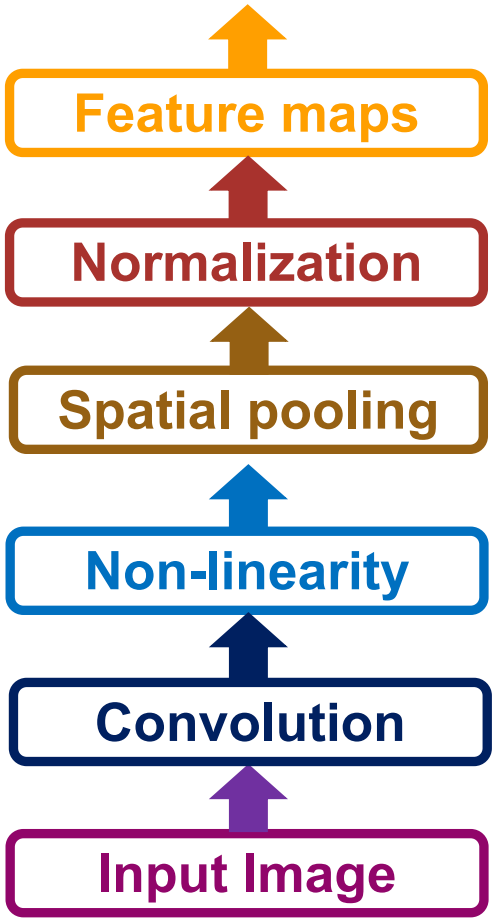


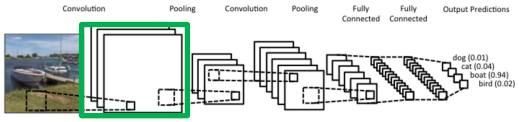
Activation Functions

- Per-feature independent.
- Sigmoid: 1/(1+exp(-x))



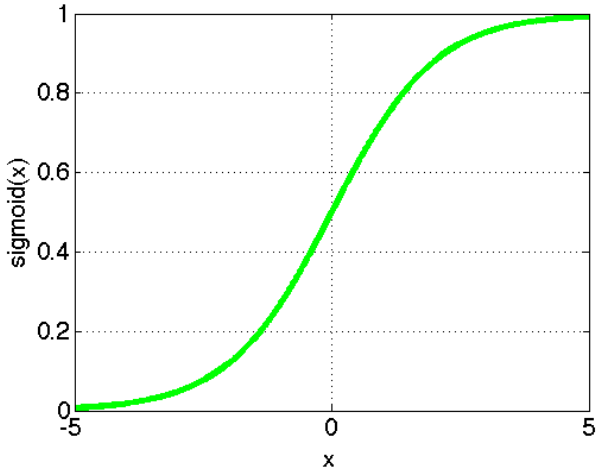
Convolutional Neural Networks

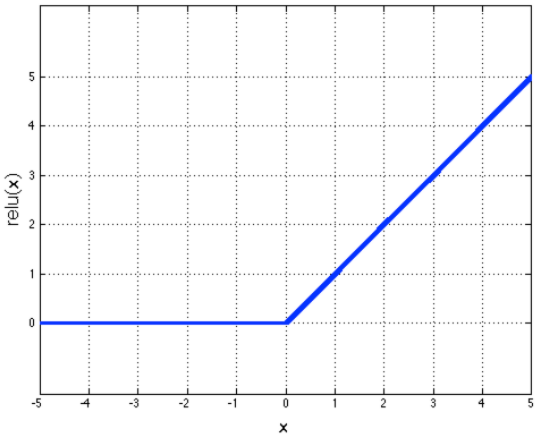




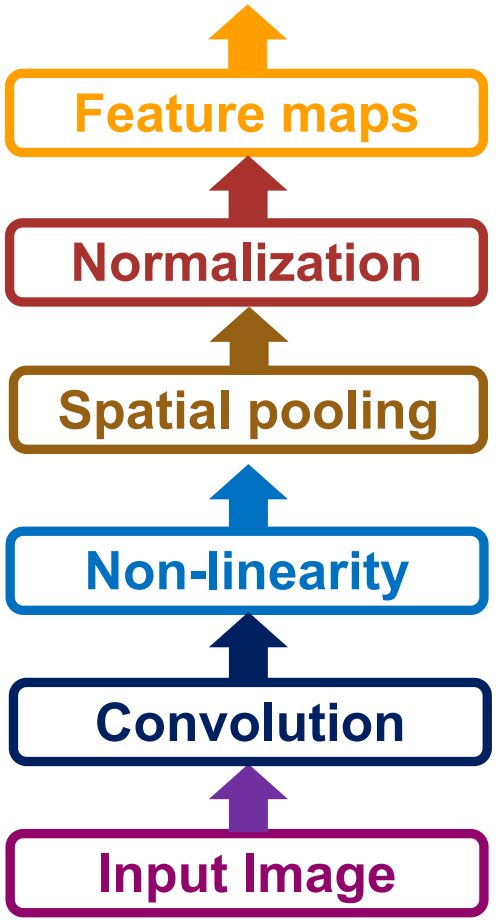
Activation Functions

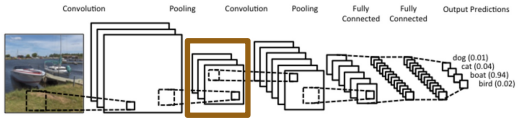
- Per-feature independent.
- Sigmoid: 1/(1+exp(-x))
- Relu: max(0,x)





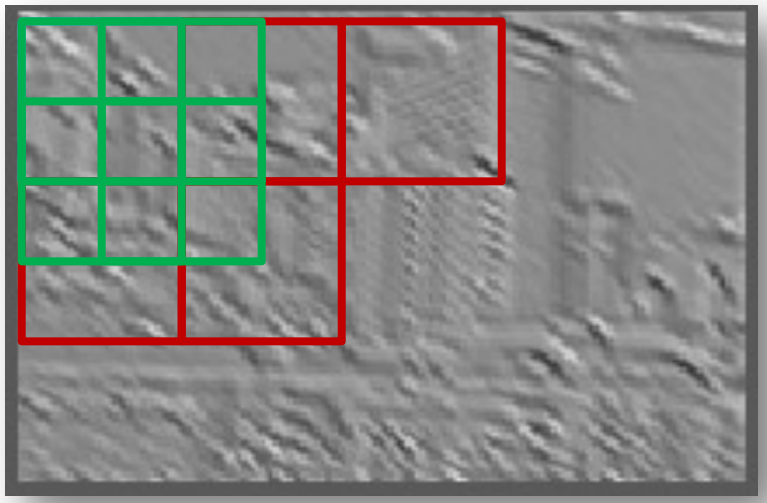
Convolutional Neural Networks



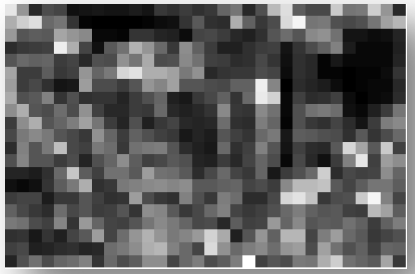


Spatial Pooling

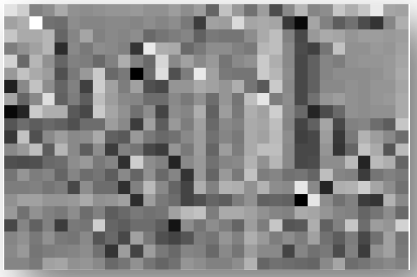
- Provides invariance to small transformations.
- Larger receptive fields (see more of the input).



Max



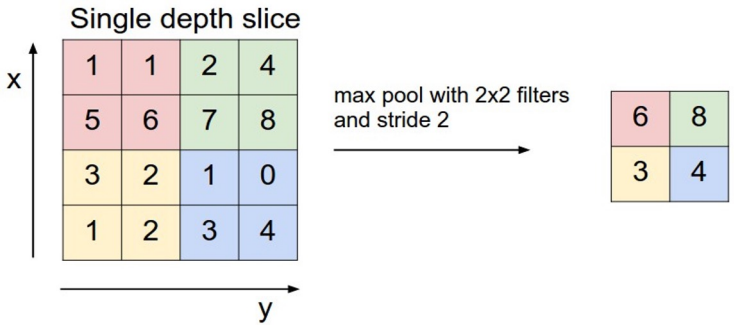
Sum

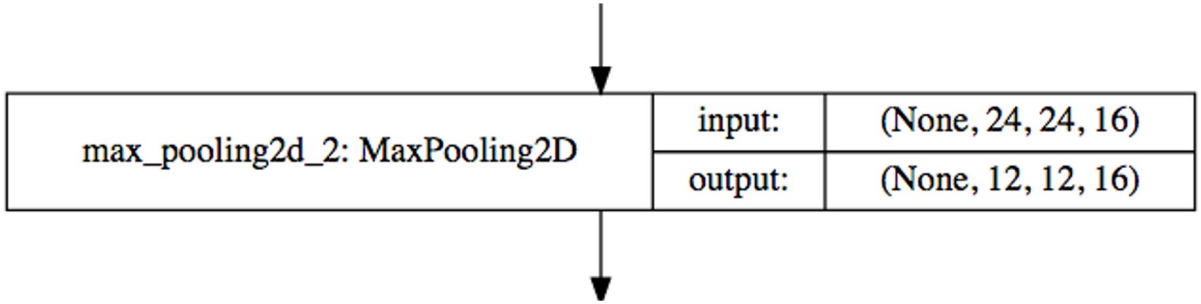


Convolutional Neural Networks

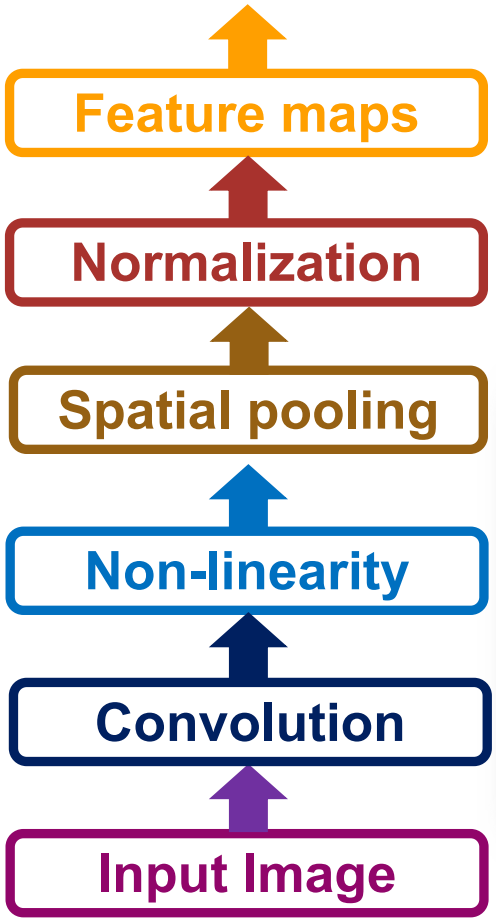
- Used to reduce the spatial resolution
  - independently on each channel
  - reduce complexity and number of parameters
- MAX operator most common
  - sometimes also AVERAGE
- In `tf.keras`:

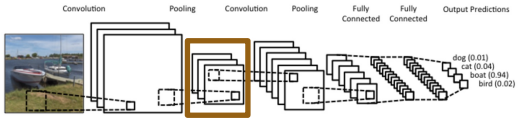
```
layers.MaxPooling2D(pool_size)
layers.AveragePooling2D(pool_size)
```





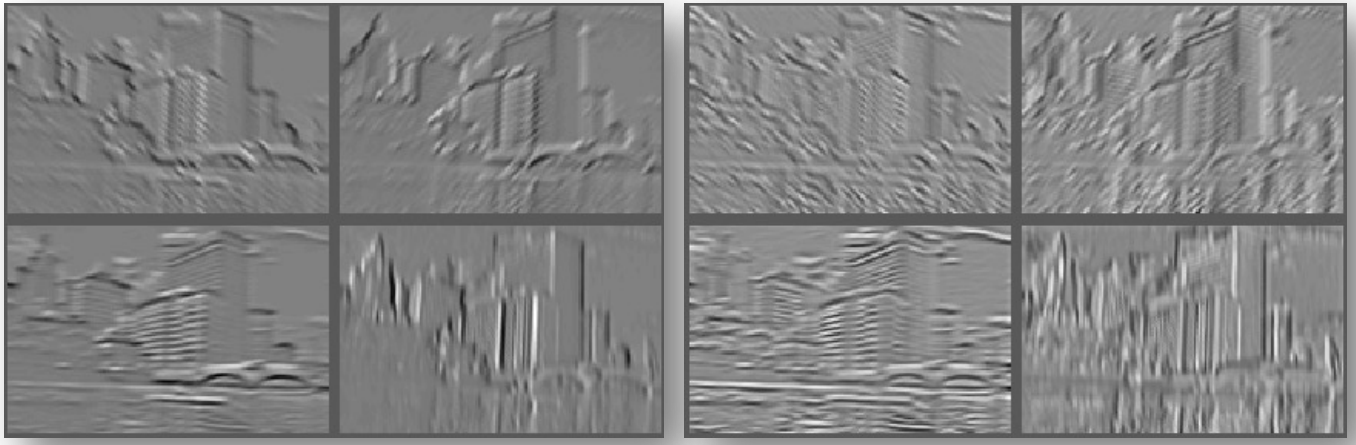
Convolutional Neural Networks





Normalization:

- Equalizes/normalizes feature maps.
- Suppresses low-frequencies/Highlight peaks.

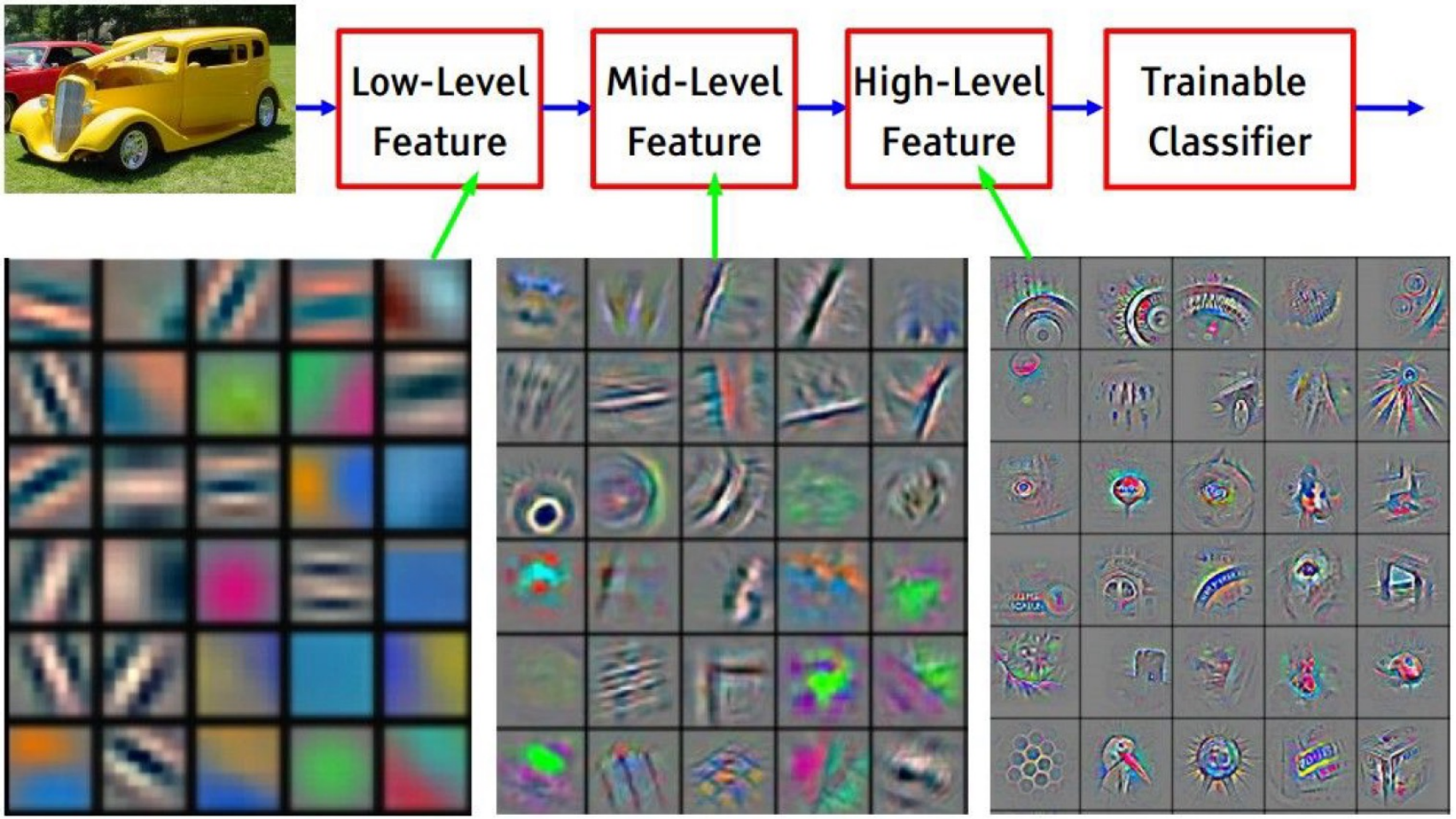


Convolutional Neural Networks

- Flatten
  - flattens the input into a vector (typically before dense layers)
- Dropout
  - similar as with dense layers
- In `tf.keras`:

```
layers.Flatten()
layers.Dropout(rate)
```

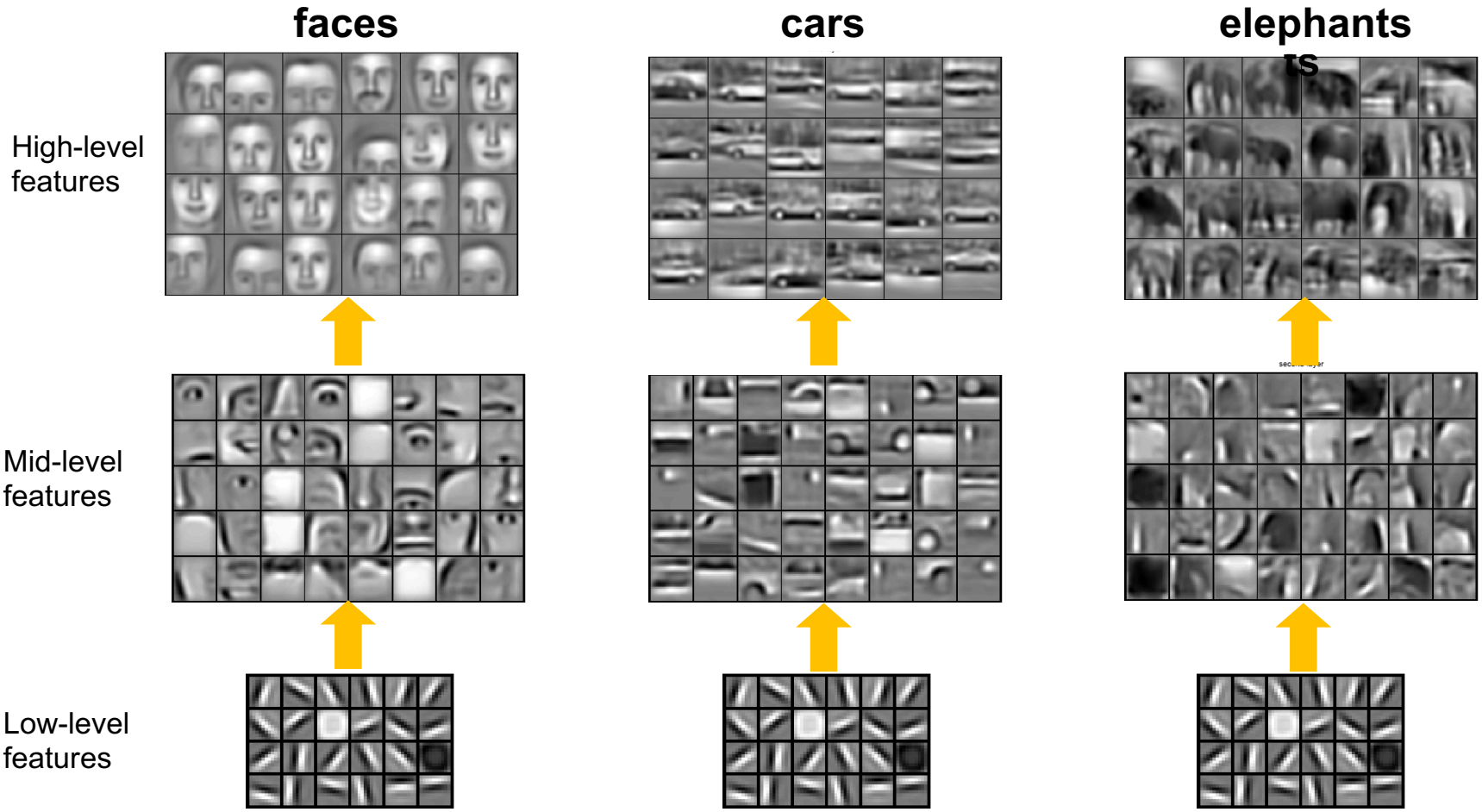
Convolutional Neural Networks



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

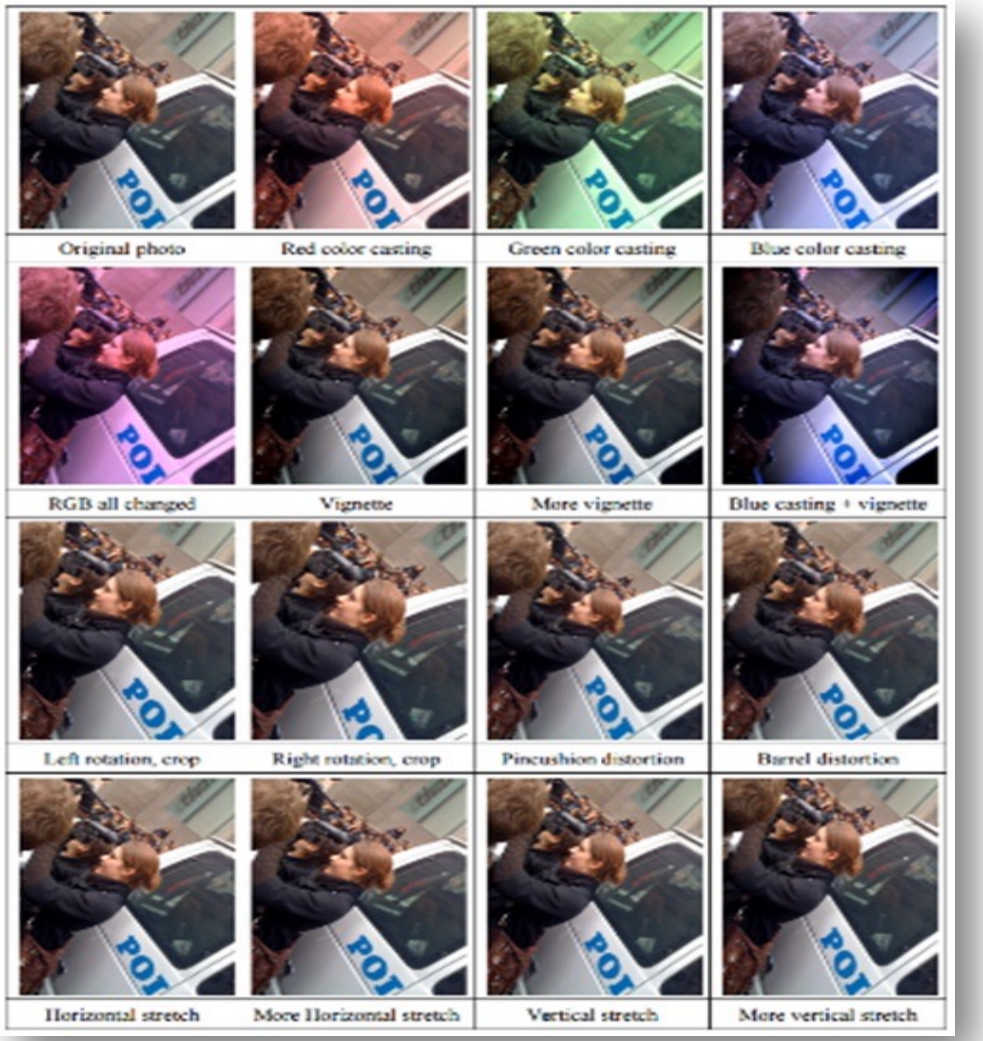
Convolutional Neural Networks

Examples of learned object parts from object categories

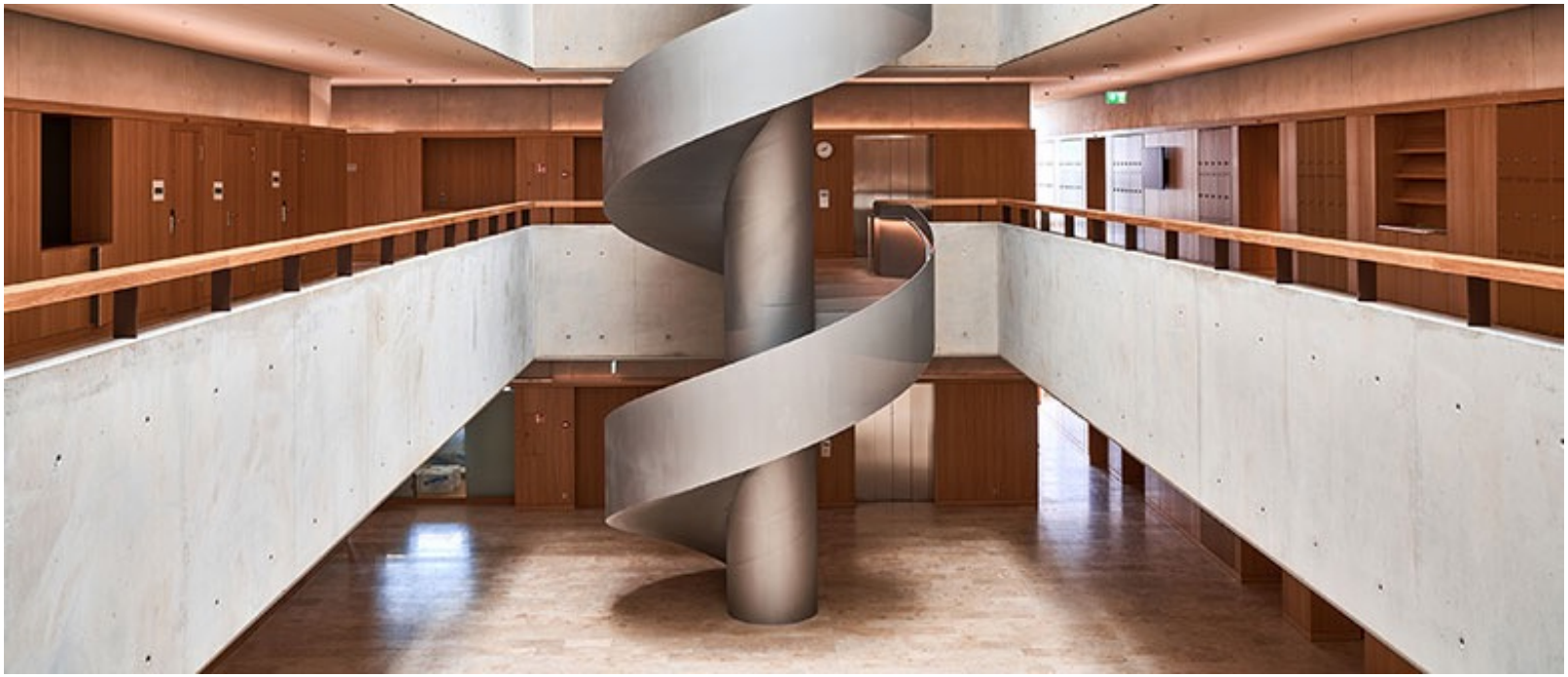


Convolutional Neural Networks

- Create *virtual* training samples
  - Horizontal flip
  - Random crop
  - Color casting
  - Geometric distortion



HSLU Hochschule Luzern



CAS Machine Learning

Deep Learning: From Core Foundations to Applications

Javier Montoya, Dr. sc. ETH
javier.montoya@hslu.ch