Deep Learning Bootcamp: Convolutional Neural Networks

Technische Hochschule Ingolstadt

KI-basierte Optimierung in der Automobilproduktion

Technische Hochschule Ingolstadt

Motivation

Guiding Neuroscientific Principles

Hubel & Wiesel

Nobel Prize in Physiology or Medicine in 1981

1959

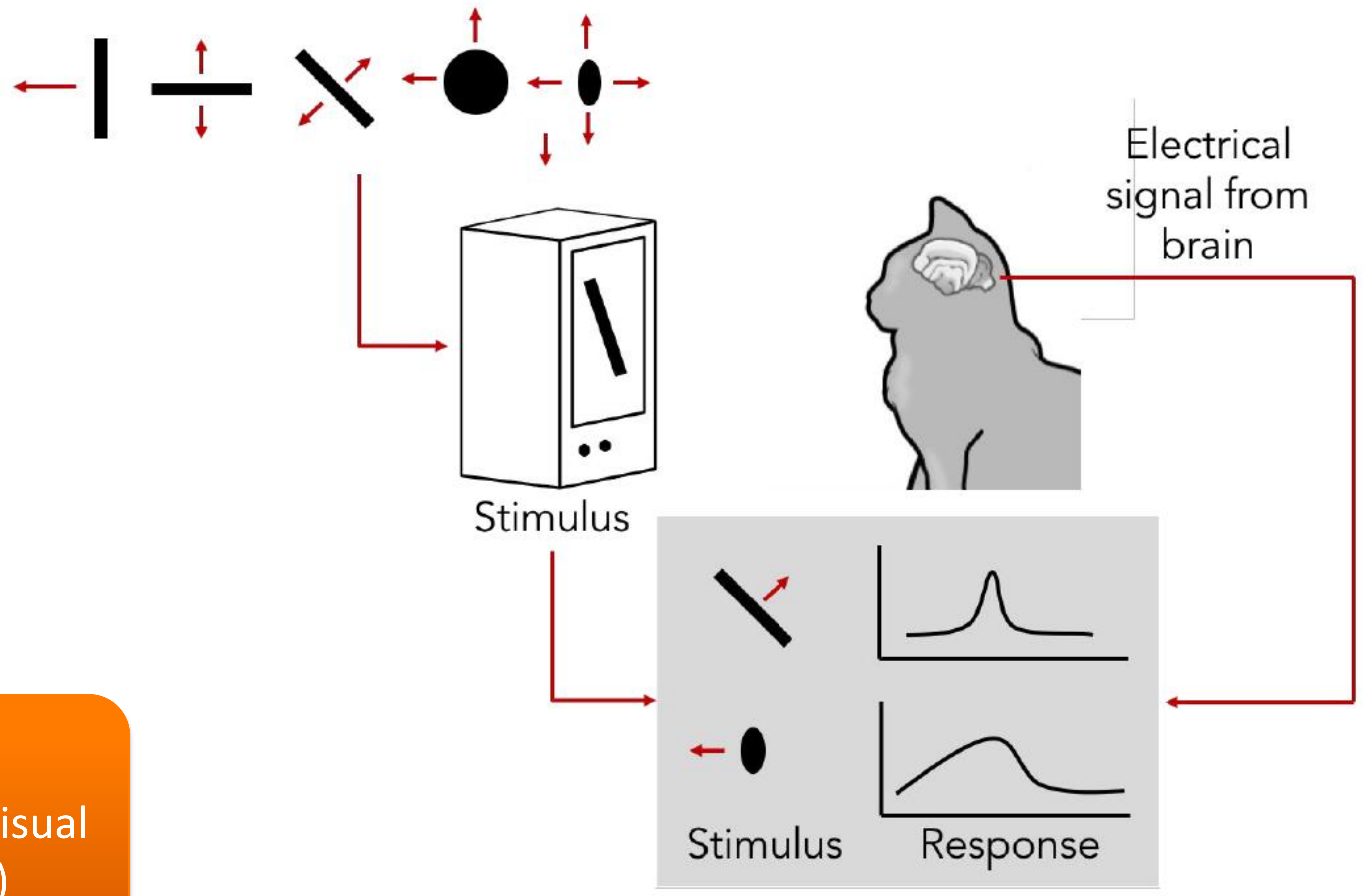
RECEPTIVE FIELDS OF SINGLE NEURONES IN THE CAT'S STRIATE CORTEX

1962

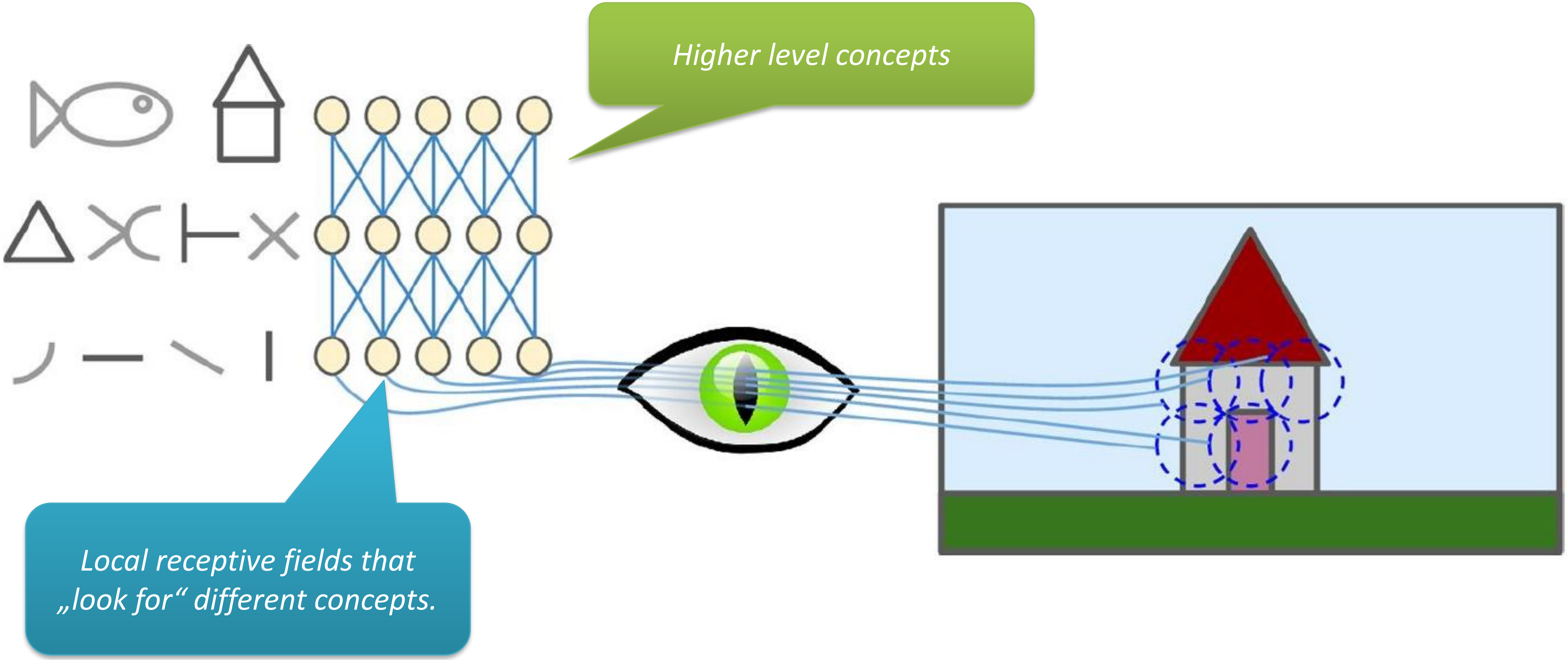
RECEPTIVE FIELDS, BINOCULAR INTERACTION AND FUNCTIONAL ARCHITECTURE IN THE CAT'S VISUAL CORTEX

Concept neurons:

Some neurons only react to certain patterns in the visual field (e.g., „horizontal lines“, „vertical lines“, etc.)



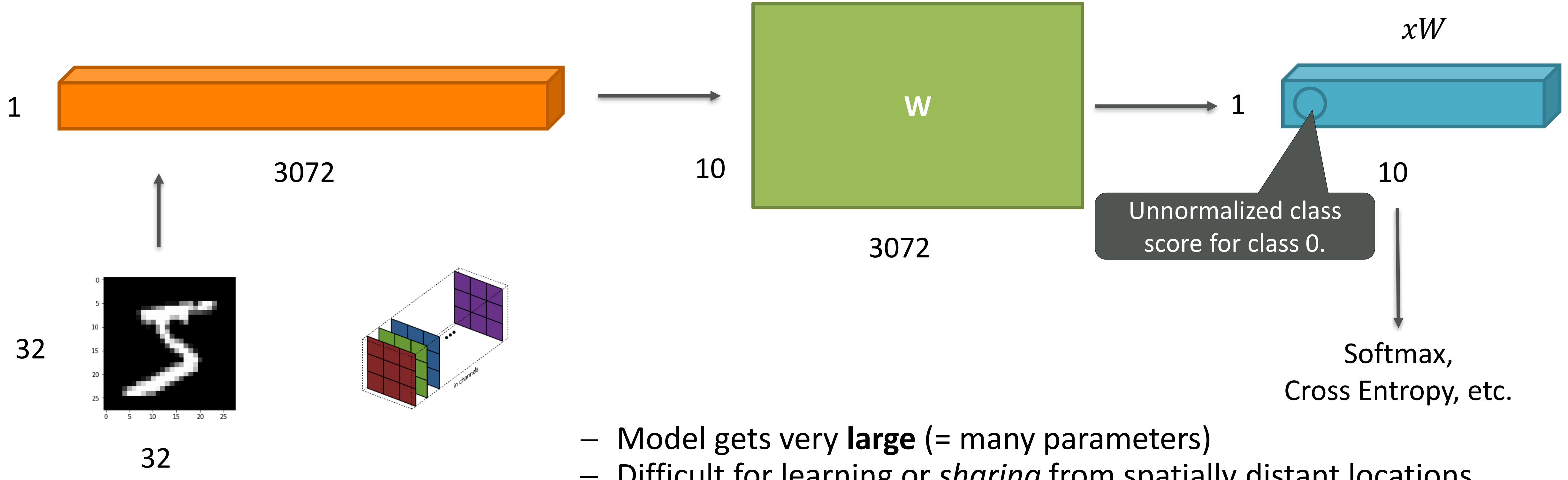
A Cartoon Impression Of Our Visual Cortex



[Hands-On Machine Learning with Scikit-Learn and Tensorflow, Géron, 2017]

Why Not Using Fully-Connected Layers?

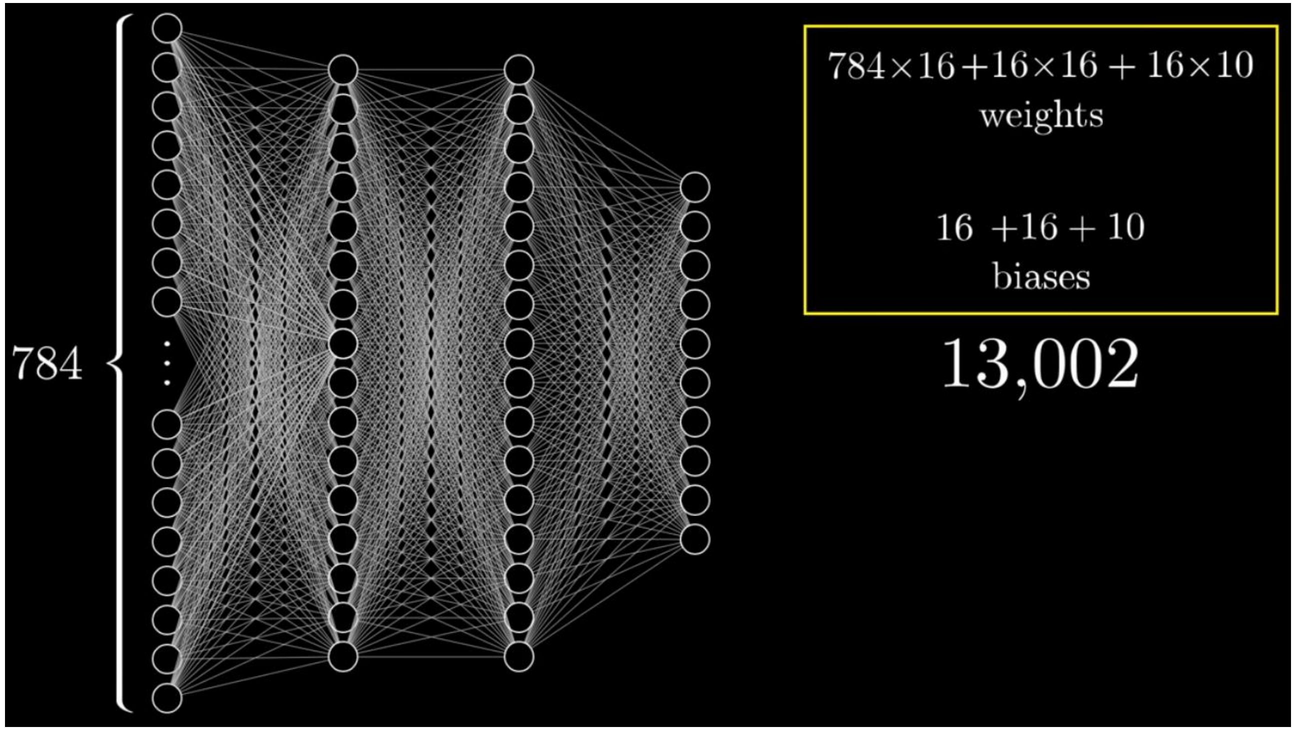
With feedforward nets, we would stretch a $32 \times 32 \times 3$ image to a 1×3072 vector



– Model gets very **large** (= many parameters)
– Difficult for learning or *sharing* from spatially distant locations (reusing concepts)!

Feedforward Networks – Problems

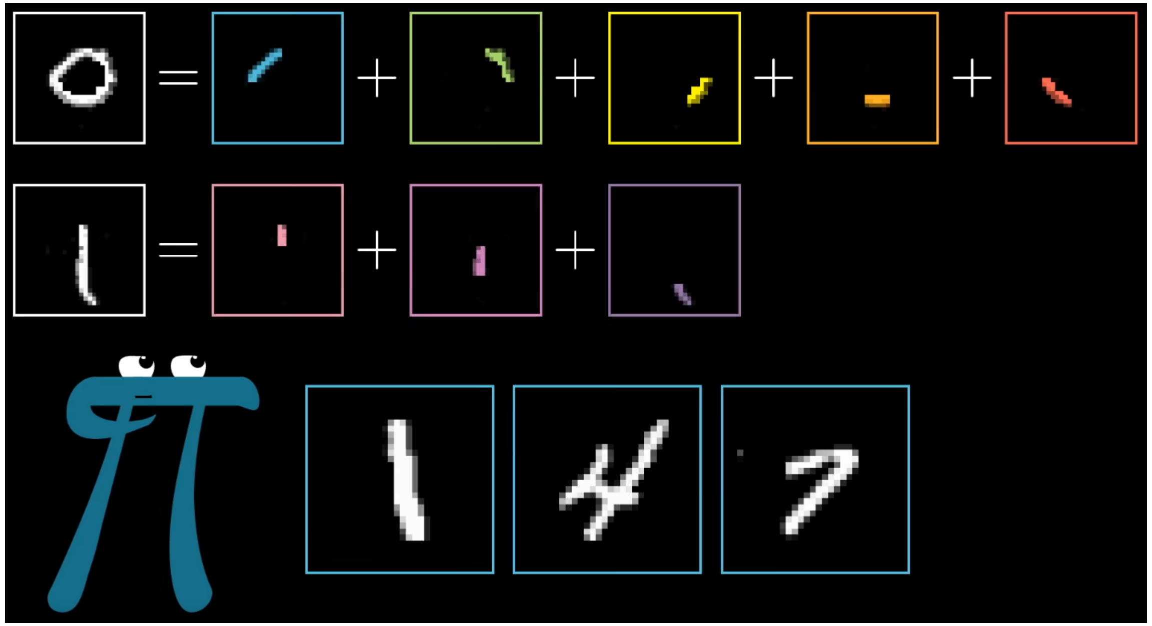
- Feedforward Networks become too large very fast
- Reminder: Each neuron is connected with every neuron from the previous and following layer
- If we have larger images, FF Networks are very hard to handle



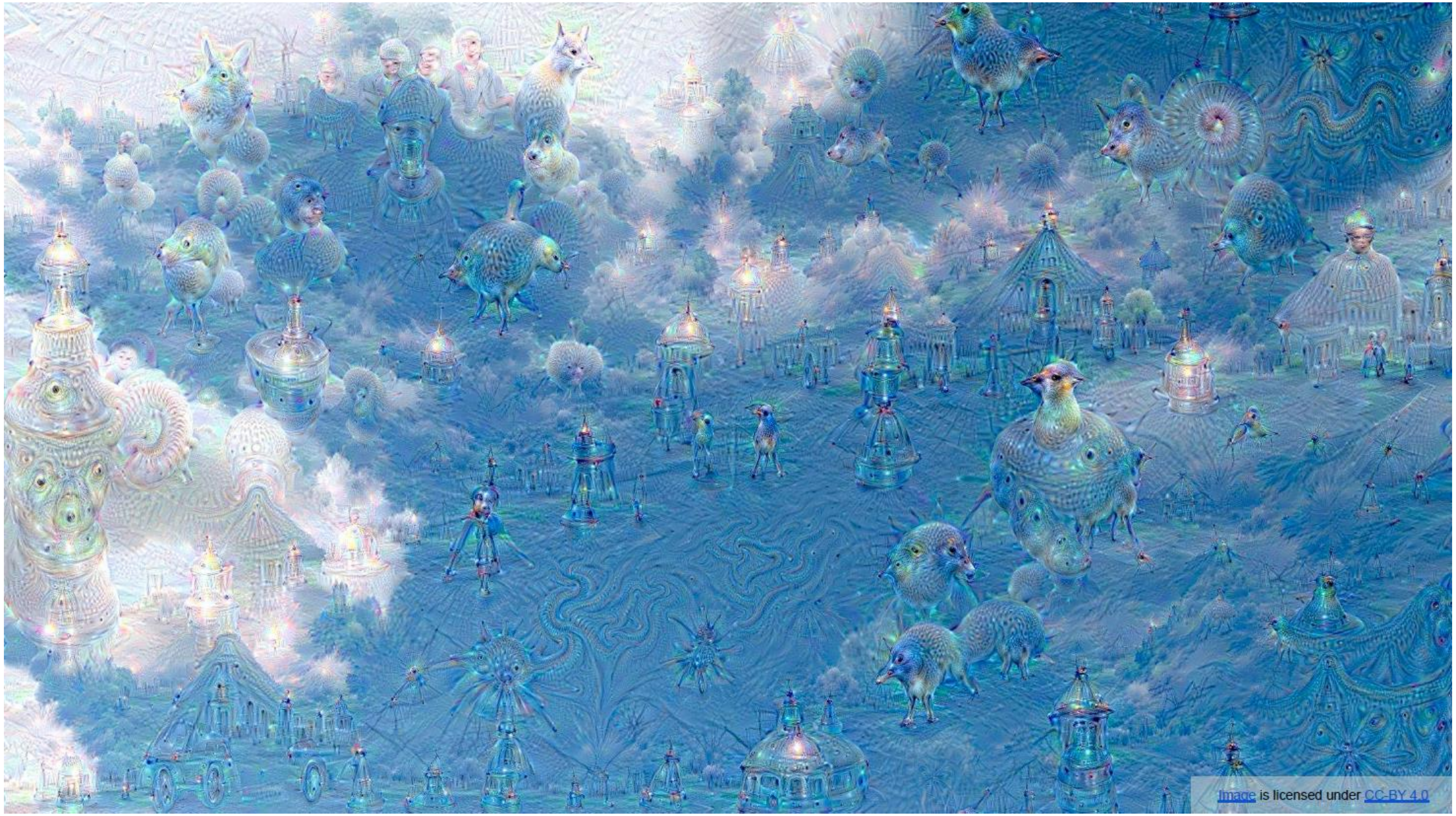
Example: Images of size (128, 128, 3)
Input dim: 128*128*3 = 49 152
Second layer: 4096 Neurons
Output layer: 10 Neurons
Overall weights and biases: = 201 420 810
(needs ca. 0,8 GB memory)

Feedforward Networks – Problems

- Neurons that detect certain features can only detect them in exactly the same spot
- But: In images, the same features can be found in different regions
- We want to learn features independently of their location

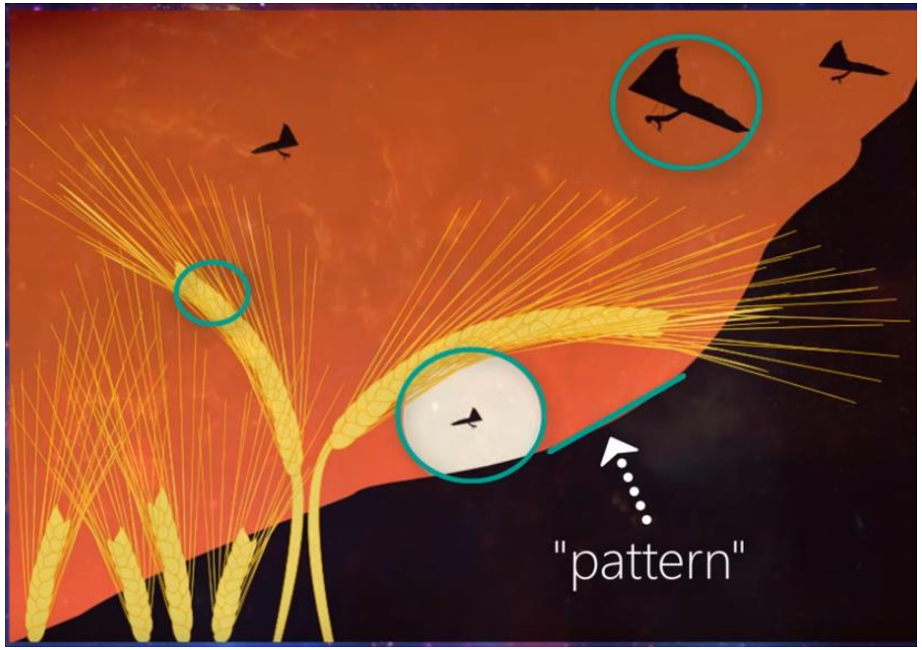


CNNs – Biological Motivation



Convolutions

Self-Study Time



deeplizard – CNNs explained:
https://www.youtube.com/watch?v=YRhxdVk_sIs

Tasks:

- Write a short summary of the video for your own notes
- Note 2 core intuitions explained by deeplizard

Discrete Convolutions as feature extractors

Filter kernel

-1	1

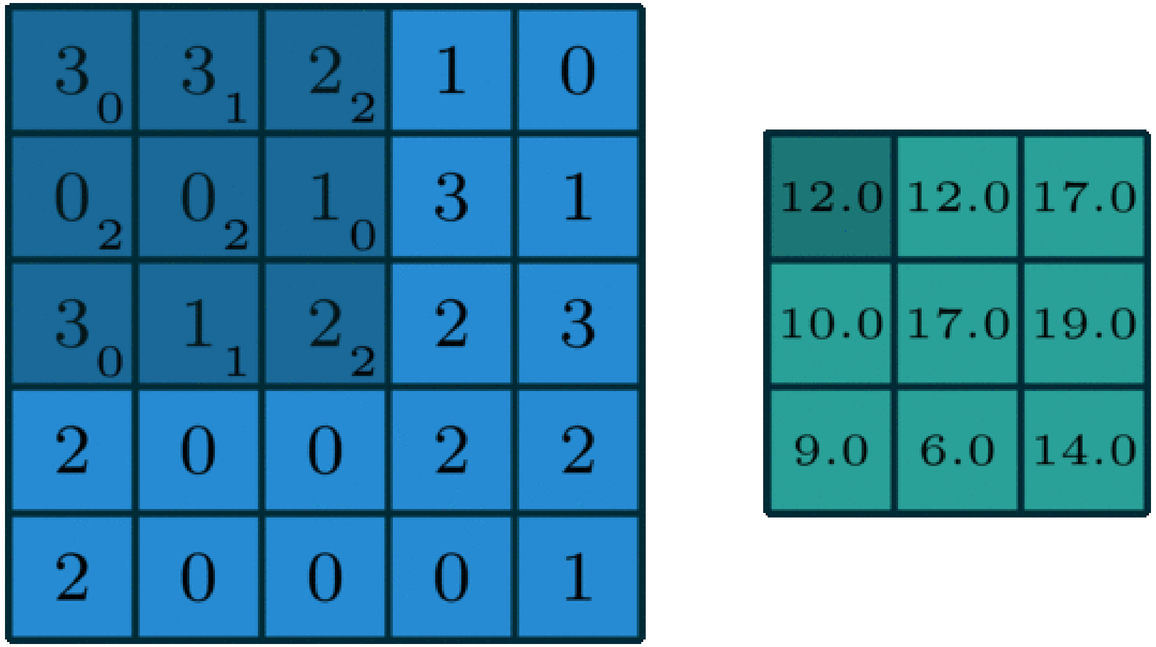
-1 1

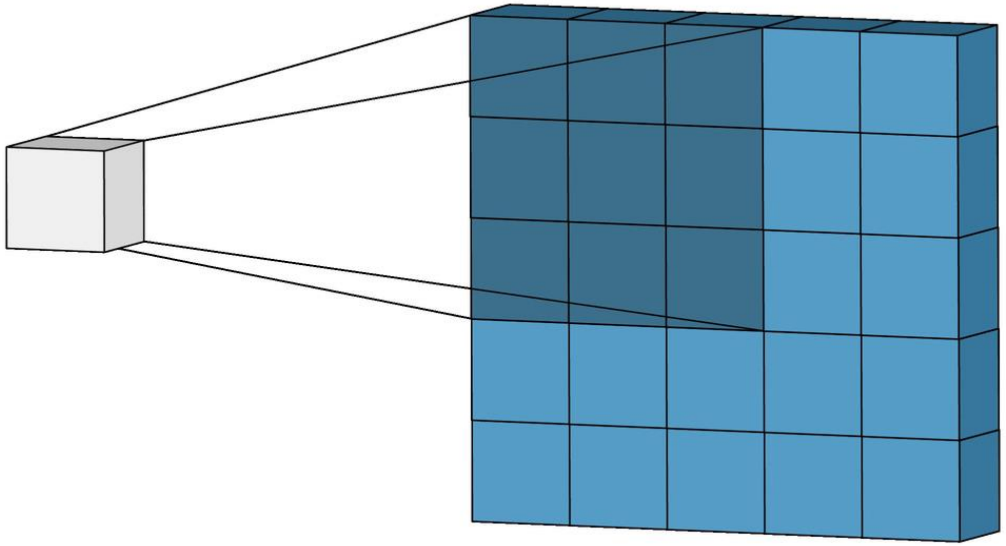
4	7	11	3	12	5

3	4	-8	9	-7

Convolutions: Core Idea

- Exploit **grid-like spatial structure** (1D time series, 2D images, 3D videos, etc.)
- Restrict input locations to neighborhoods (receptive fields)





[https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1]

What Are Convolutions, Actually?

In pure mathematics and signal processing (input signal x and filter w):

$$y(t) = (x * w)(t) = \int_{-\infty}^{\infty} x(\tau) \cdot w(t - \tau)\, d\tau$$

As a discrete approximation:

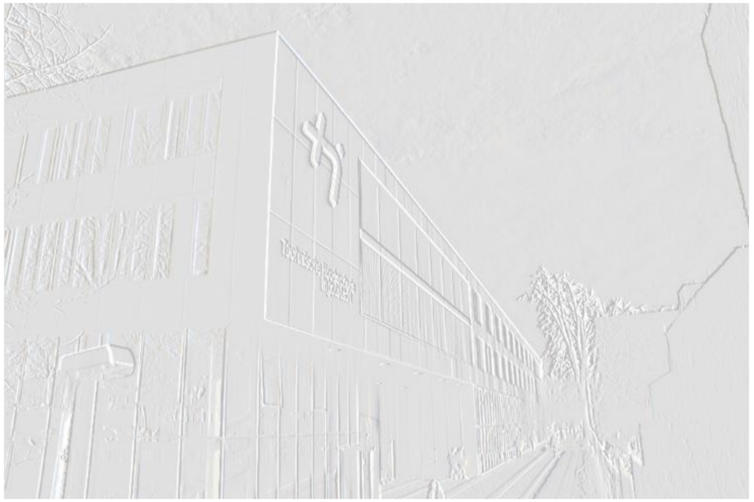
$$(x * w)(t) = \sum_{\tau \in \mathbb{Z}} x(\tau) \cdot w(t - \tau) = \sum_{\tau \in \mathbb{Z}} x(t - \tau) \cdot w(\tau)$$

Convolution: A Sobel Filter For Edge Detection

I

$$G_x$$

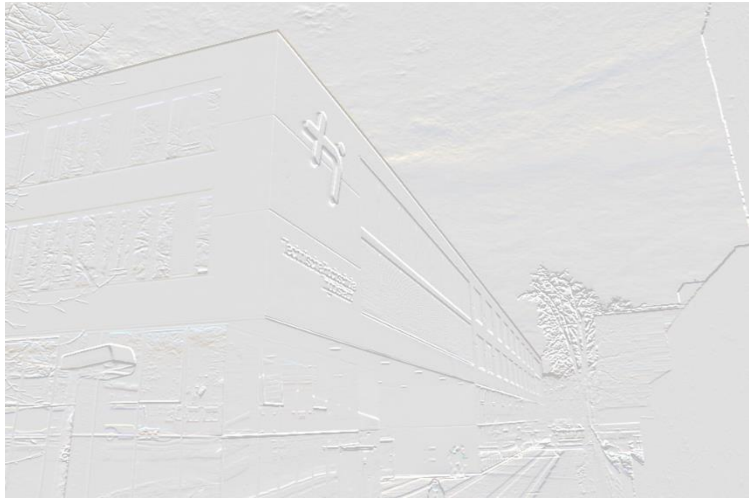
$$* \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$



$I * G_x$

$$G_y$$

$$* \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$



$I * G_y$

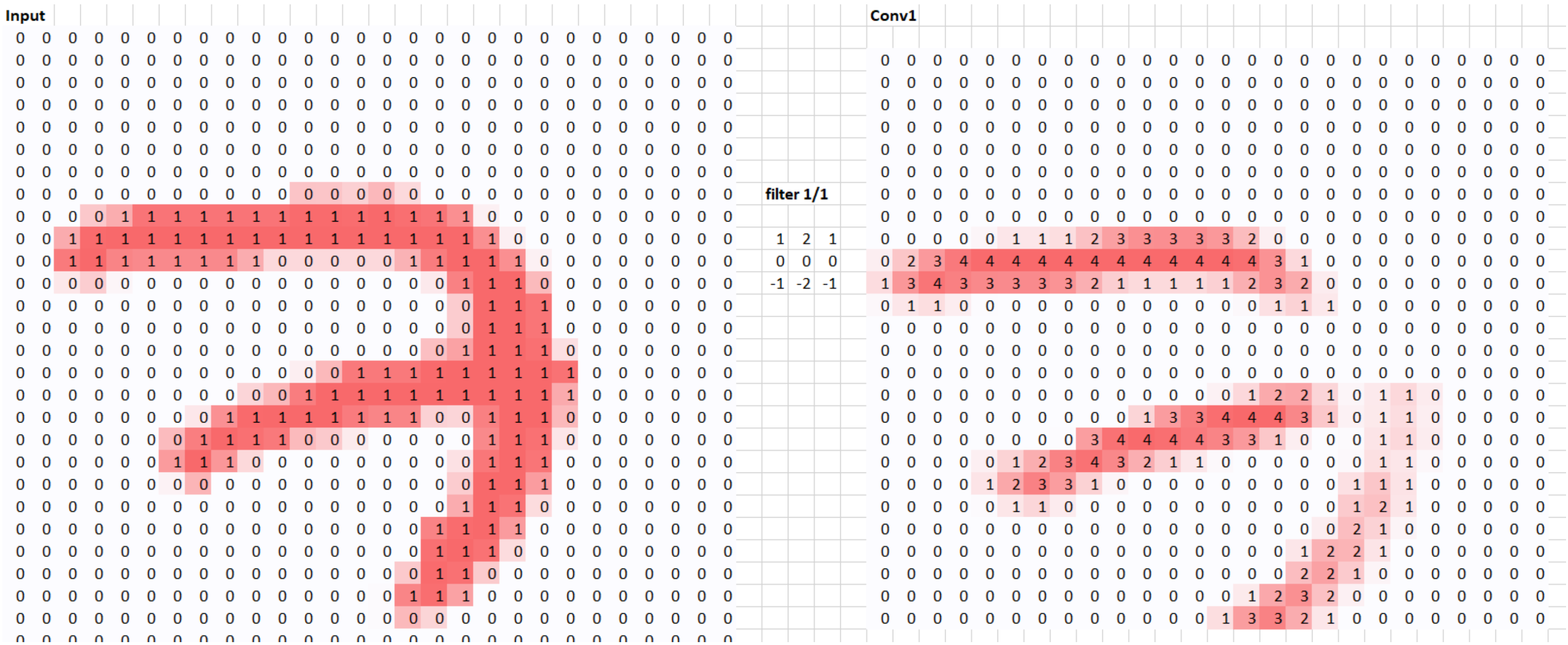
Task

Play around with the excel sheet!

https://moodle.thi.de/mod/resource/view.php?id=339801

Search for different filters and try out their effects, e.g on Wikipedia:

https://en.wikipedia.org/wiki/Kernel_(image_processing)



Convolutional Layers

Example

```
layers.Conv2D(filters=16, kernel_size=(5, 5), strides=(1,1) activation="relu")
```

How many different kernels are applied?

How big is the receptive field of the kernels?

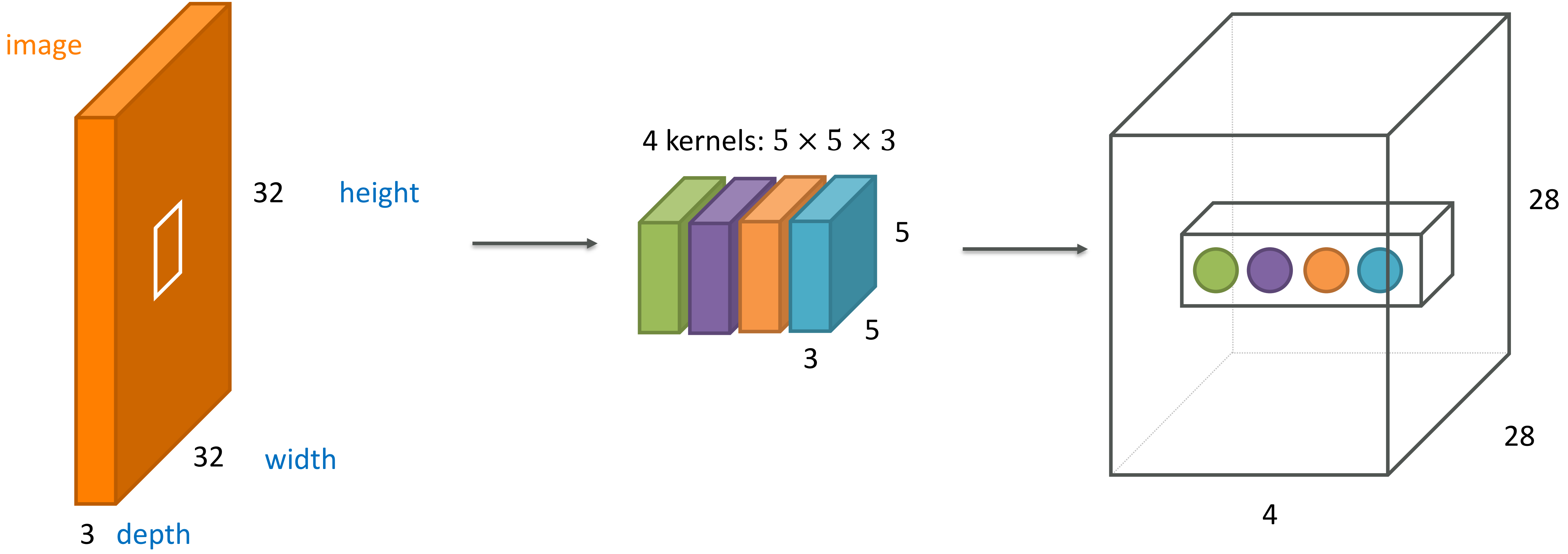
What‘s the step size of the convolution?

```
layers.MaxPooling2D(pool_size=(2, 2))
```

How big is the area that should be summarized into one value

Convolutional Layer

Each kernel looks at the same region of the input, but for different things



Refinements: Stride

„Stride“ defines the step size of the convolution operator (default 1)

Stride 1: Shifts filter by 1 position

3	3	2	1
0	0	1	3
3	1	2	2

3	3	2	1
0	0	1	3
3	1	2	2

3	3	2	1
0	0	1	3
3	1	2	2

Stride 2: Shifts filter by 2 positions

3	3	2	1
0	0	1	3
3	1	2	2

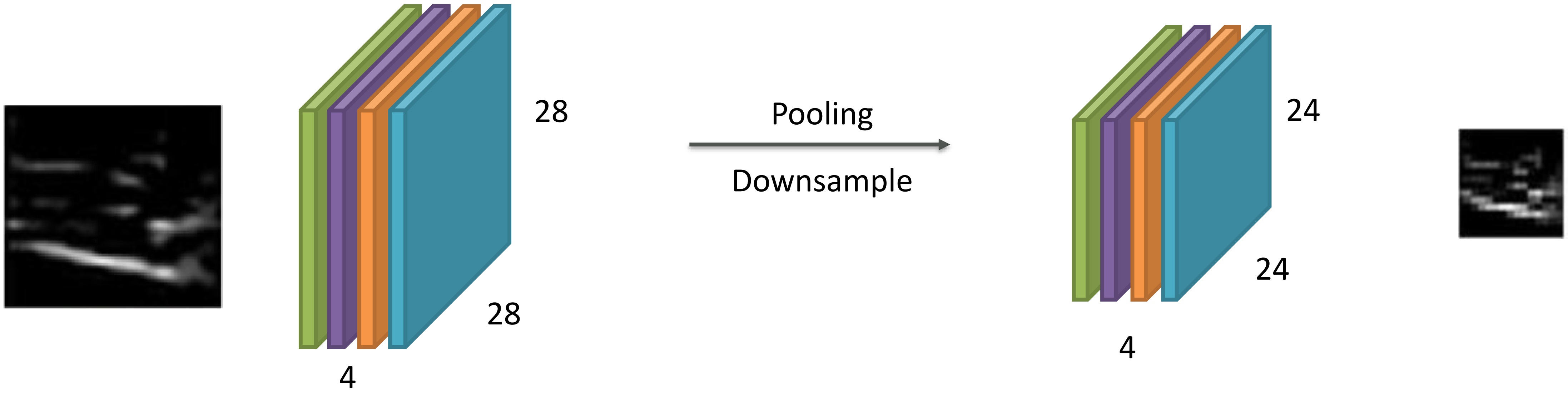
3	3	2	1
0	0	1	3
3	1	2	2

Pooling Layers

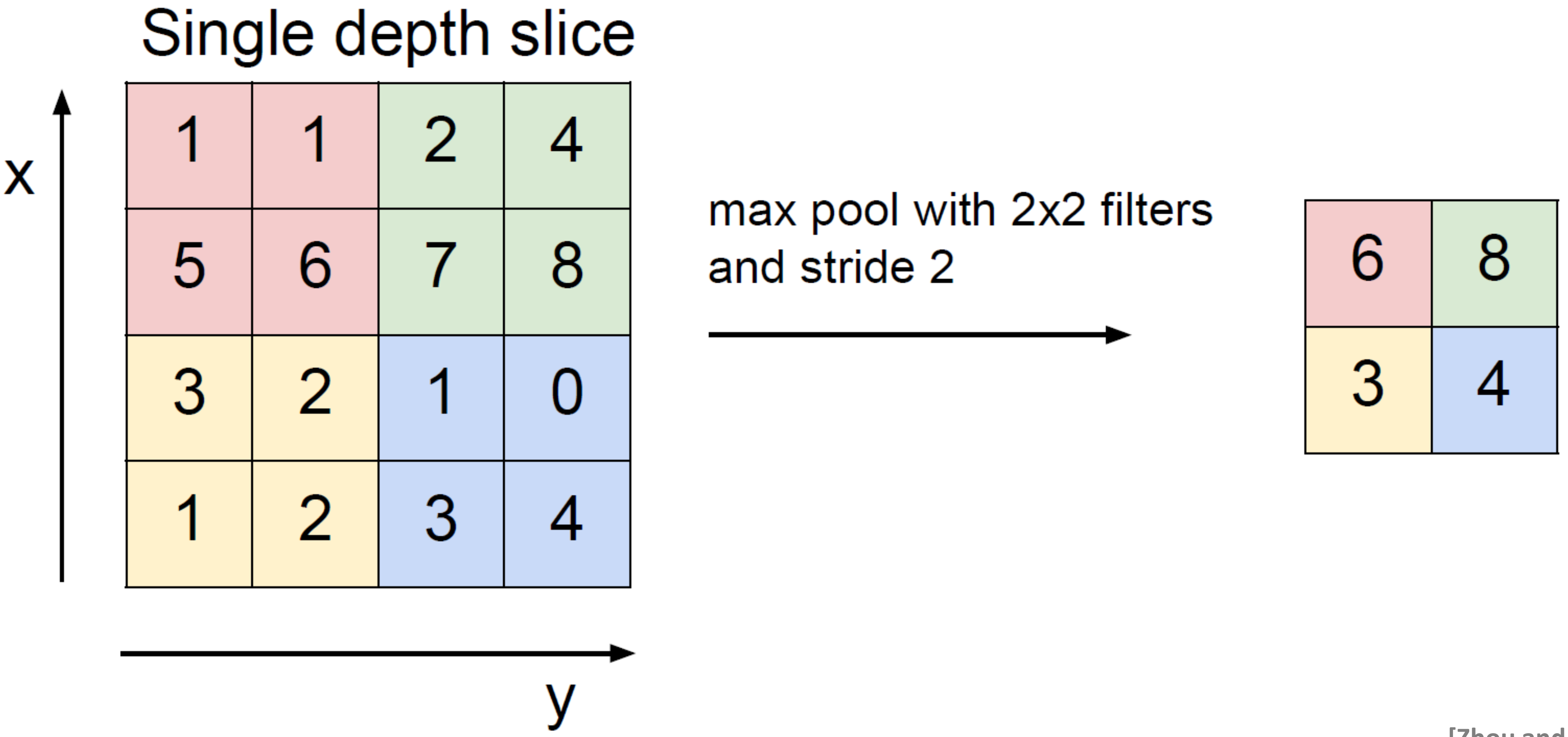
Perform a *downsampling* of an image (similar procedure to convolution)

- reduce spatial dimensions; does not change the depth!
- calculate summary statistics (e.g. average, max, …) of a region

→ memory and computational savings, reduction of noise

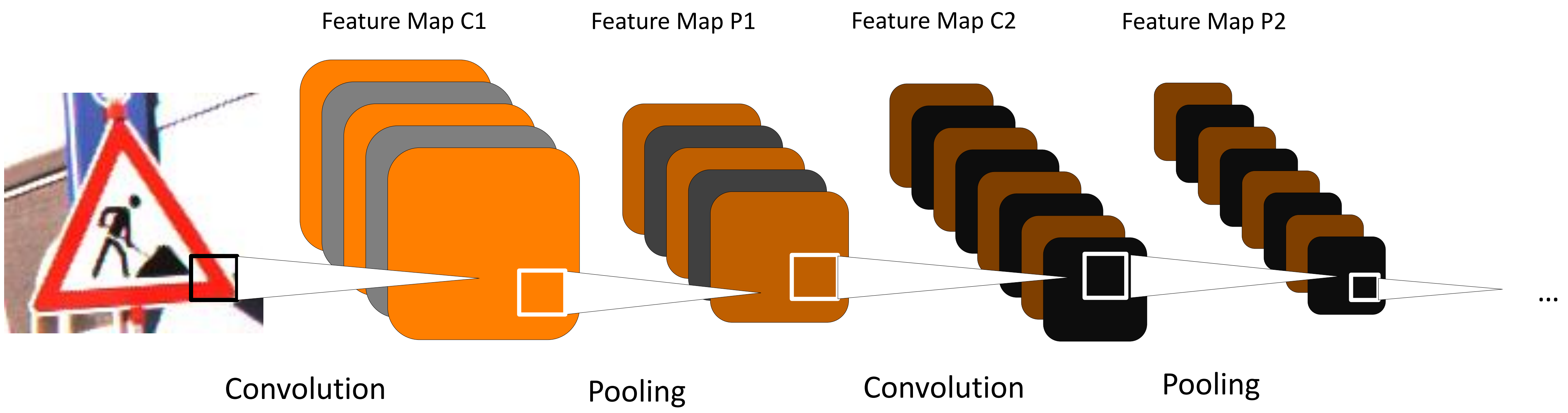


Max-Pooling



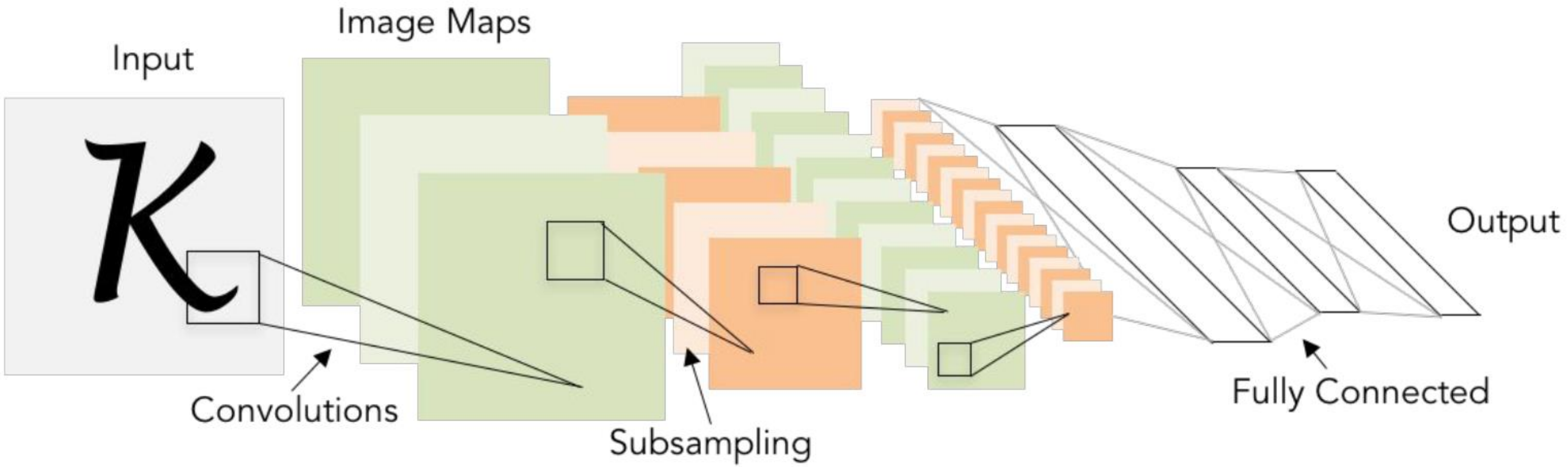
[Zhou and Chellappa, 1988]

Convolution And Pooling Layers Working Together



LeNet

The first successful application of ConvNets to classify digits



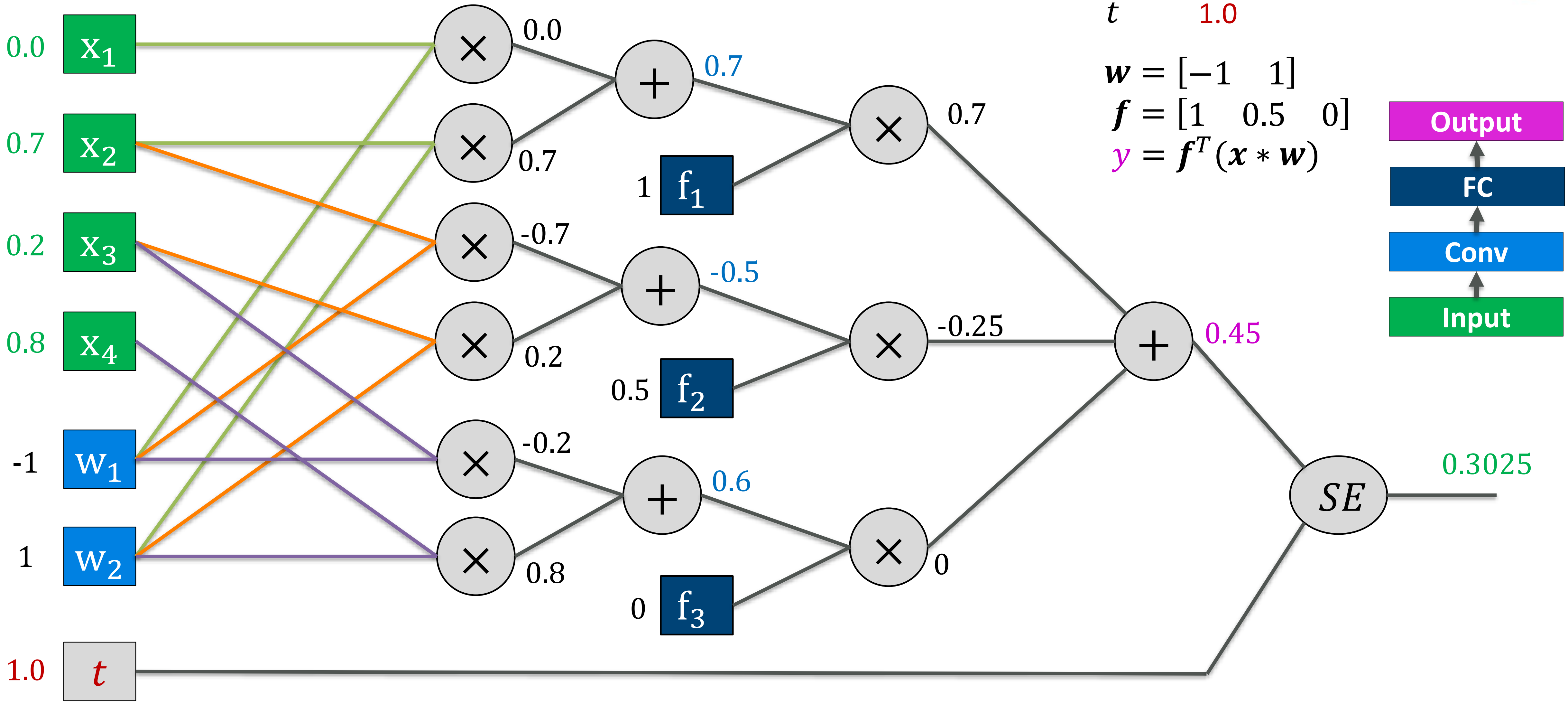
[LeCun et al., 1998]

The Computational Graph Of A 1D Convolution

$\boldsymbol{x}$ | 0.0 | 0.7 | 0.2 | 0.8

t 1.0

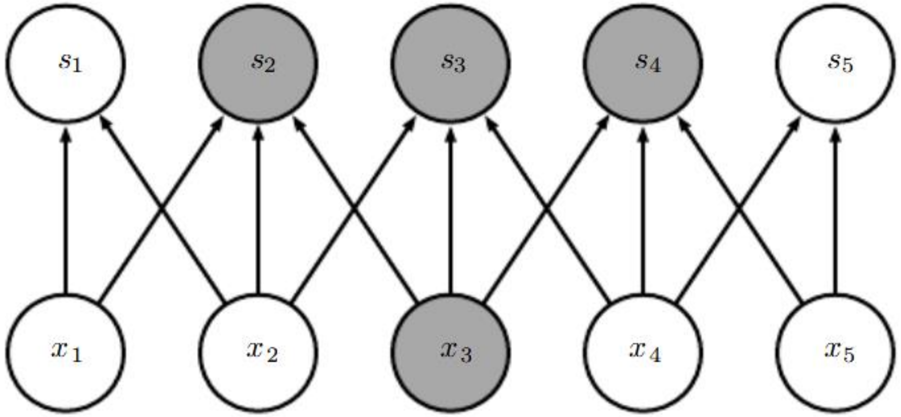
$$\boldsymbol{w} = [-1 \quad 1]$$
$$\boldsymbol{f} = [1 \quad 0.5 \quad 0]$$
$$y = \boldsymbol{f}^T(\boldsymbol{x} * \boldsymbol{w})$$

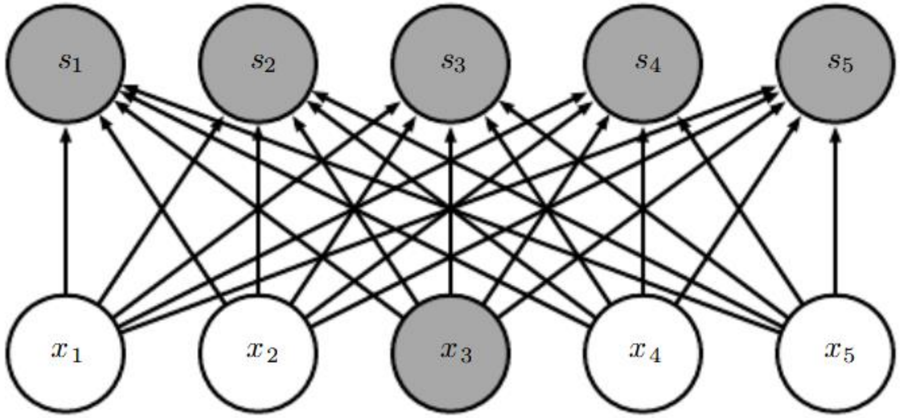


Benefits & Potentials

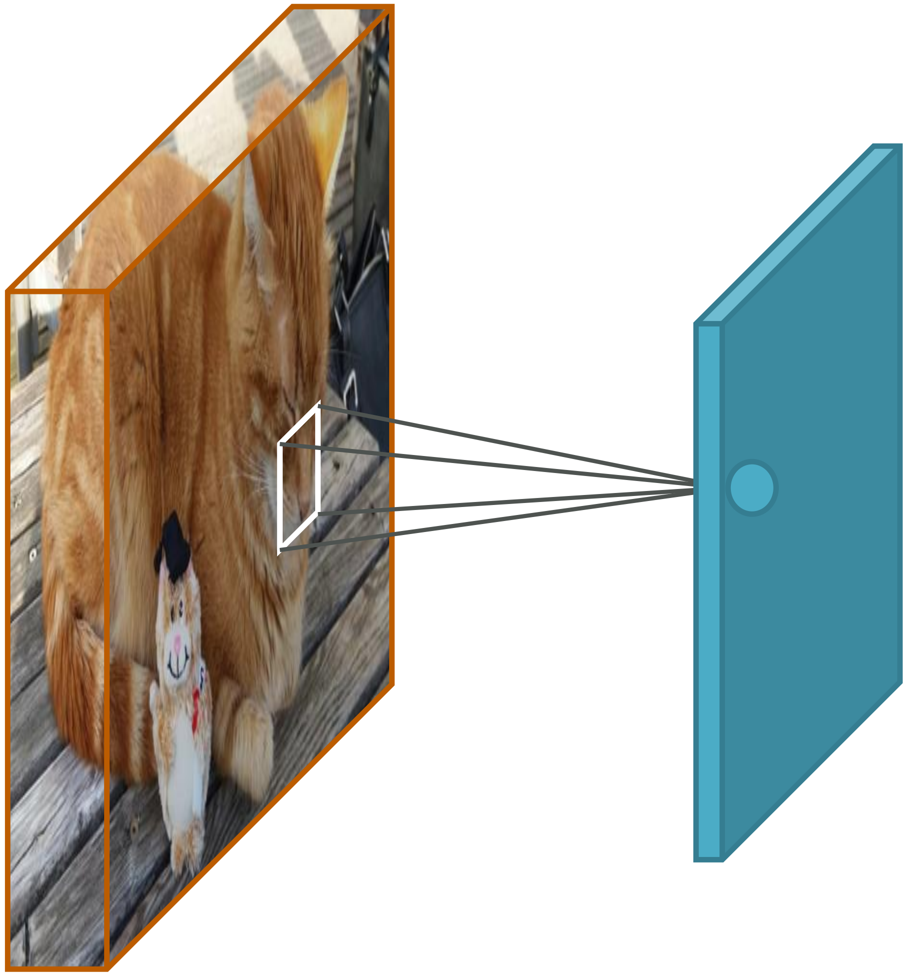
Sparse Connectivity

Convolution Layer





Fully-connected Layer



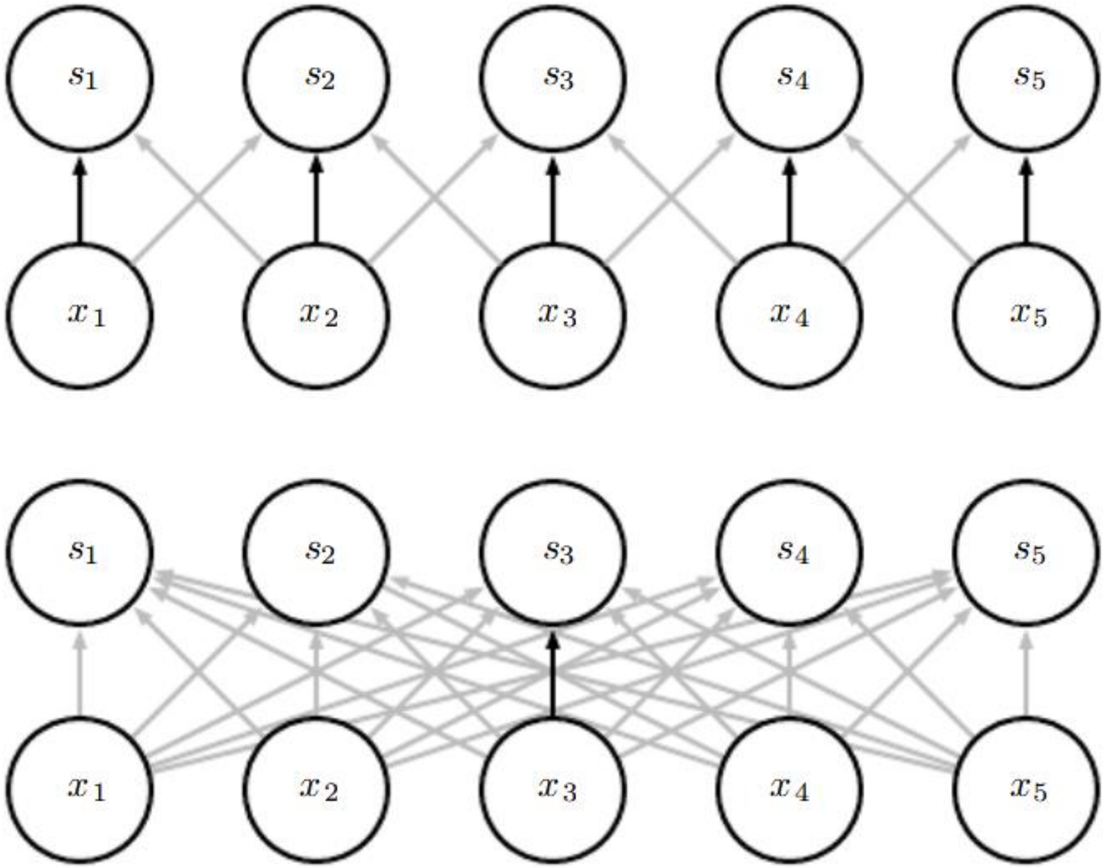
[Deep Learning, Goodfellow et al., 2016]

How many parameters does a Conv-Layer have?

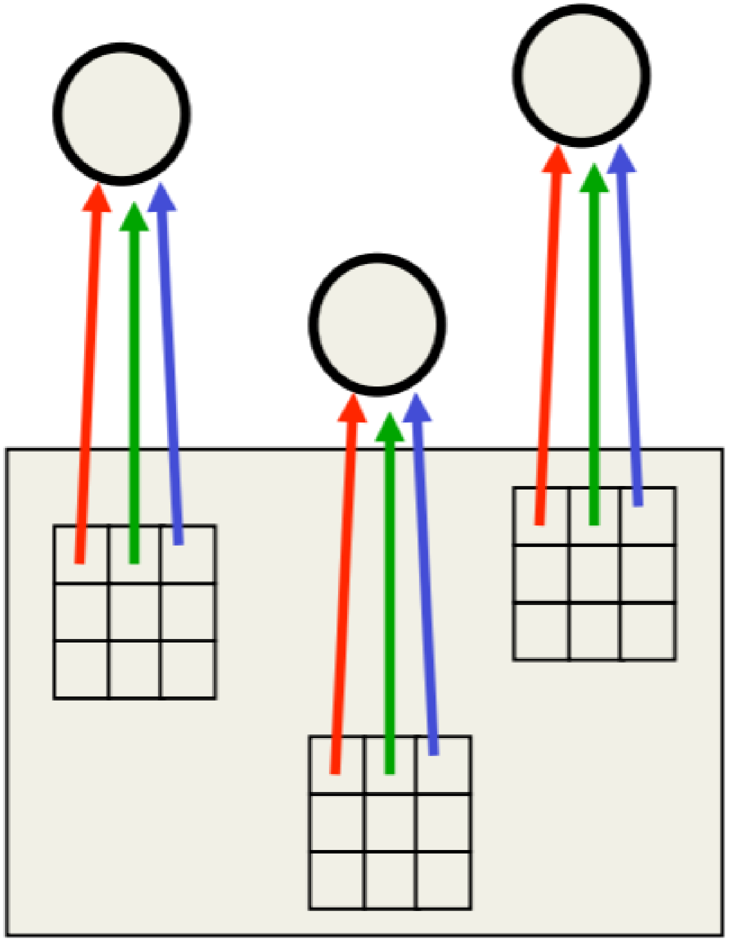
- Example: `layers.Conv2D(filters=16, kernel_size=(5, 5), strides=(1,1) activation="relu")`
- Assumption: 3 input channels
- Formula: $filter_{input} \times filters \times kernelsize_x \times kernelsize_y + filters$
- Here: $3 * 16 * 5 * 5 + 16 = 1216$ weights
- Additional benefit: There‘s no connection between all neurons of neighboring layers

Parameter Sharing

Convolution Layer



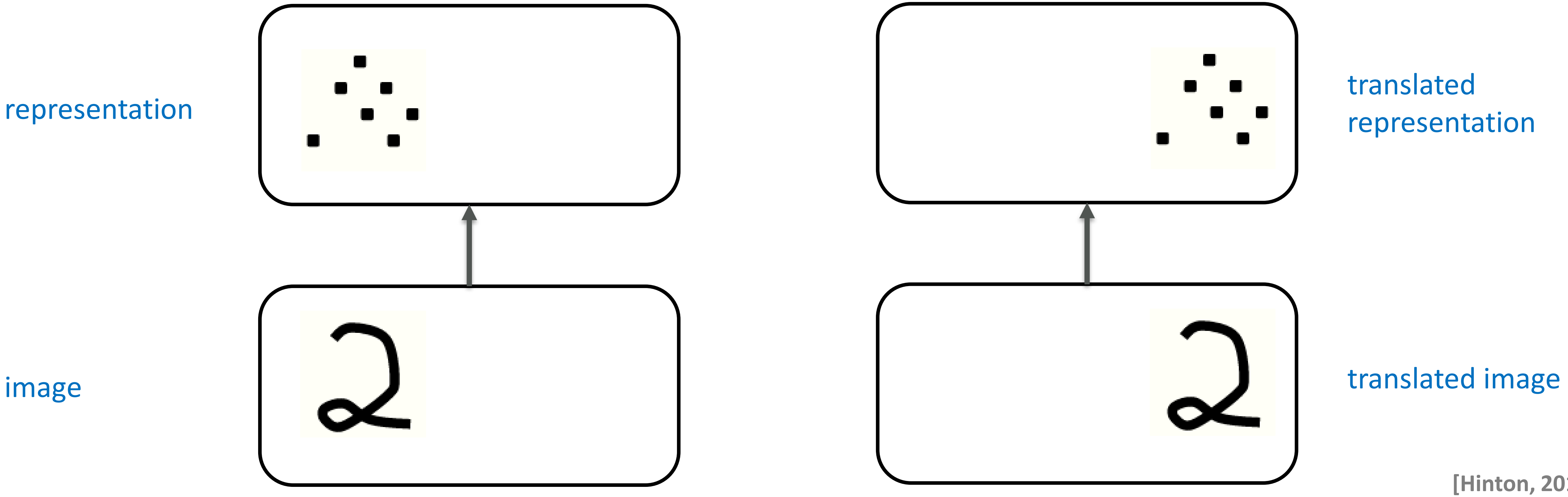
Fully-connected Layer



[Deep Learning, Goodfellow et al., 2016]

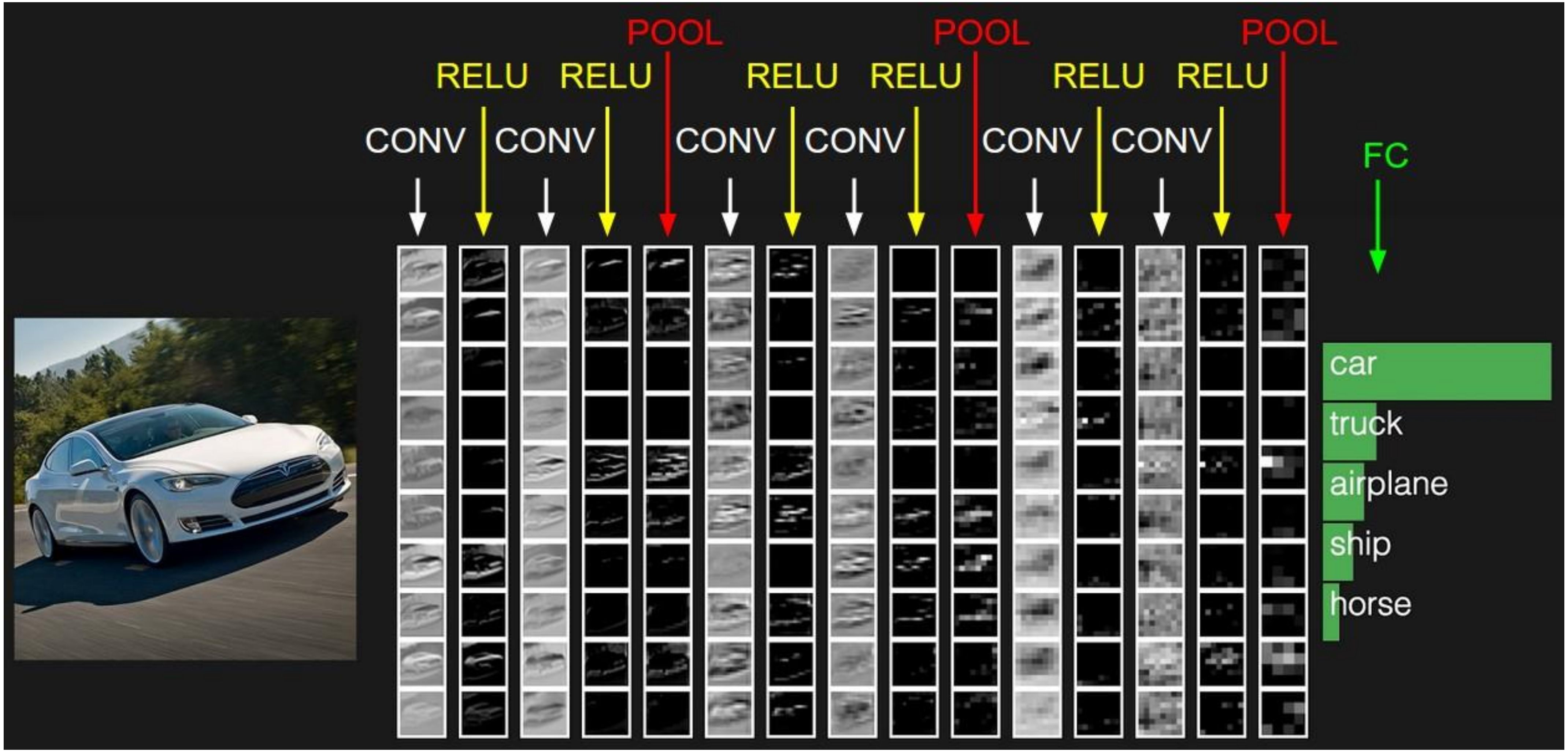
Equivariance To Translation

The same pattern produces the same output at different places in the input.
Formally, $f \circ g = g \circ f$

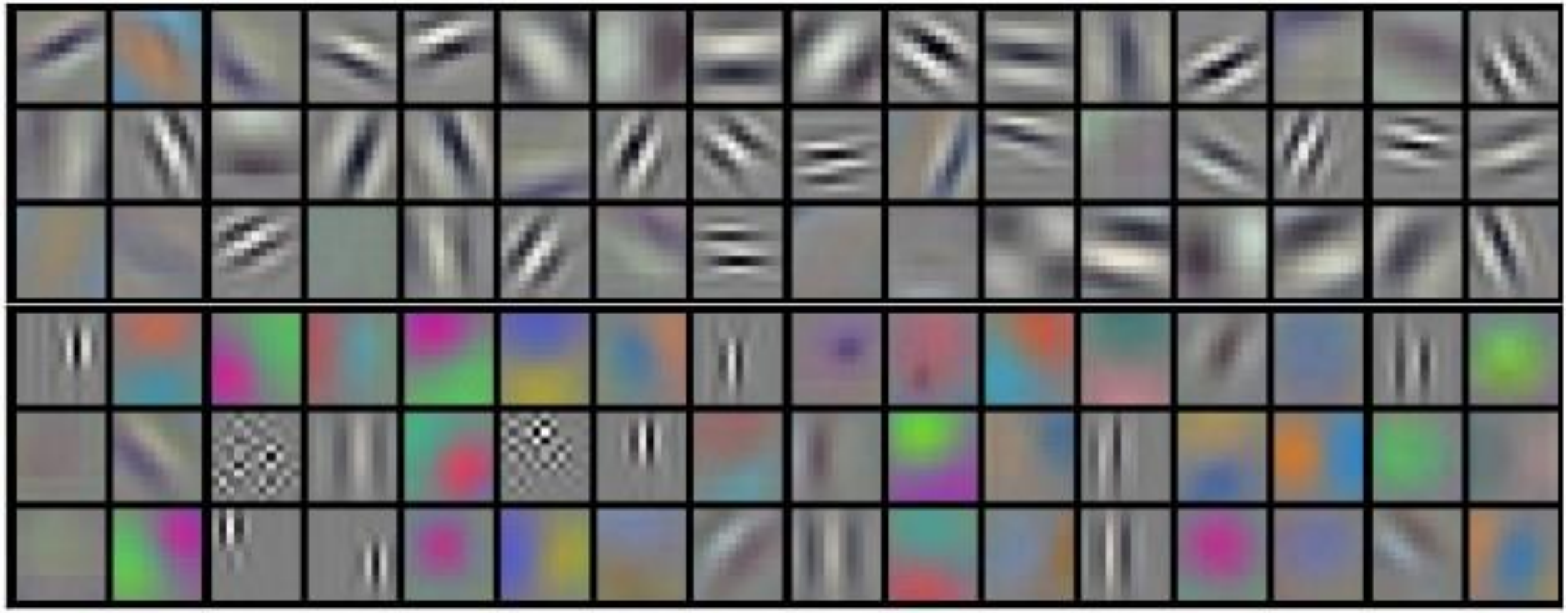


[Hinton, 2012]

A CNN In Action

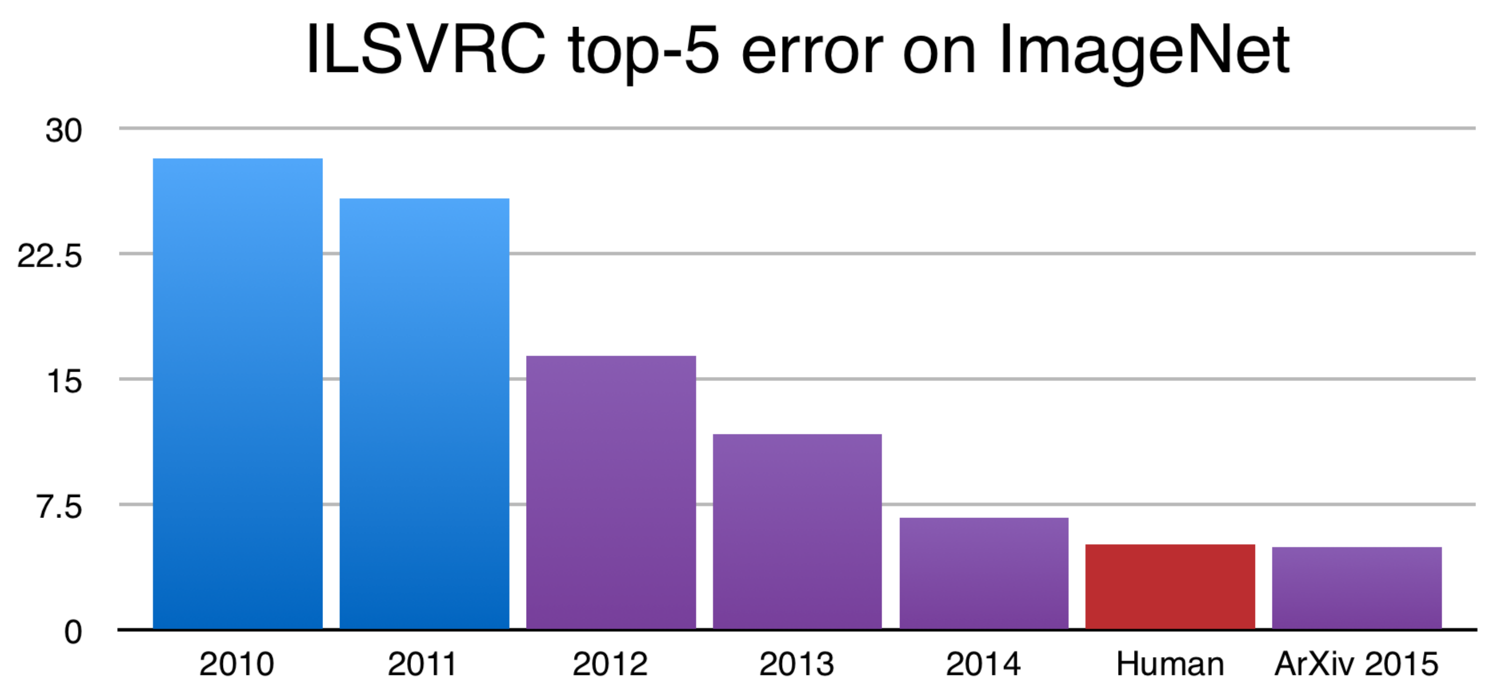


Learned Kernels



[Krizhevsky et al., 2012]

Recent Advances In Image Recognition Due To Convolutional Nets



[https://devblogs.nvidia.com/mocha-jl-deep-learning-julia/image1/]

ImageNet **L**arge **S**cale **V**isual **R**ecognition **C**hallenge

Since 2012: Deep Learning/Convolutional Networks (- 10% Top-5 Error)

Example

Explore ConvNetJS by yourself!

https://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html