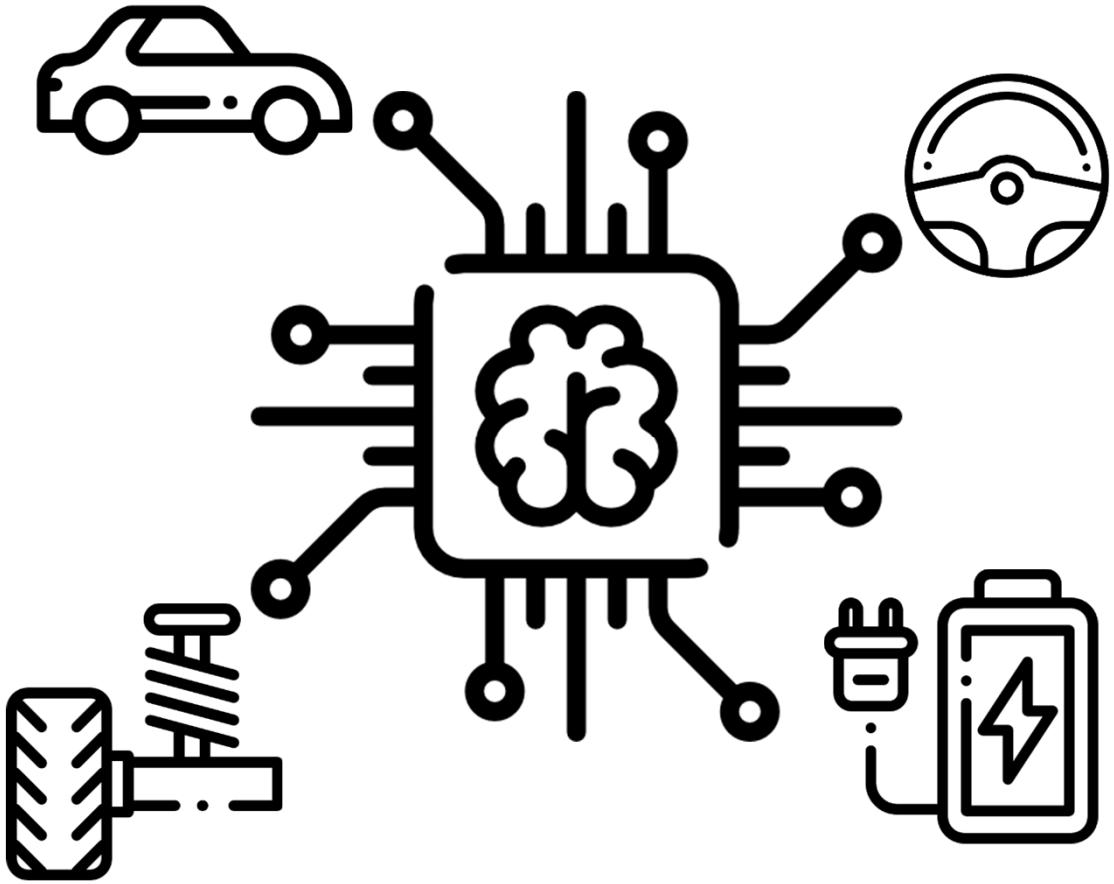
Lehrstuhl für Fahrzeugtechnik
Fakultät für Maschinenwesen
Technische Universität München

Artificial Intelligence in Automotive Technology

Johannes Betz / Prof. Dr.-Ing. Markus Lienkamp / Prof. Dr.-Ing. Boris Lohmann



Lecture Overview

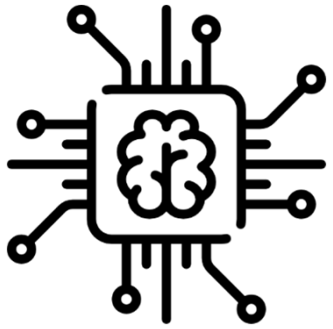
Overall Introduction for the Lecture 18.10.2018 – Betz Johannes	**6 Pathfinding: From British Museum to A*** 29.11.2018 – Lennart Adenaw	**11 Reinforcement Learning** 17.01.2019 – Christian Dengler
1 Introduction: Artificial Intelligence 18.10.2018 – Betz Johannes	**P6:** 29.11.2018 – Lennart Adenaw	**P11** 17.01.2019 – Christian Dengler
P1: 18.10.2018 – Betz Johannes	**7 Introduction: Artificial Neural Networks** 06.12.2018 – Lennart Adenaw	**12 AI-Development** 24.01.2019 – Johannes Betz
2 Perception 25.10.2018 – Betz Johannes	**P7** 06.12.2018 – Lennart Adenaw	**P12** 24.01.2019 – Johannes Betz
P2: 25.10.2018 – Betz Johannes	**8 Deep Neural Networks** 13.12.2018 – Jean-Michael Georg	**13 Free Discussion** 31.01.2019 – Betz/Adenaw
3 Supervised Learning: Regression 08.11.2018 – Alexander Wischnewski	**P8** 13.12.2018 – Jean-Michael Georg	
P3: 08.11.2018 – Alexander Wischnewski	**9 Convolutional Neural Networks** 20.12.2018 – Jean-Michael Georg	
4 Supervised Learning: Classification 15.11.2018 – Jan-Cedric Mertens	**P9** 20.12.2018 – Jean-Michael Georg	
P4: 15.11.2018 – Jan-Cedric Mertens	**10 Recurrent Neural Networks** 10.01.2019 – Christian Dengler	
5 Unsupervised Learning: Clustering 22.11.2018 – Jan-Cedric Mertens	**P10** 10.01.2019 – Christian Dengler	

Deep Neural Networks

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

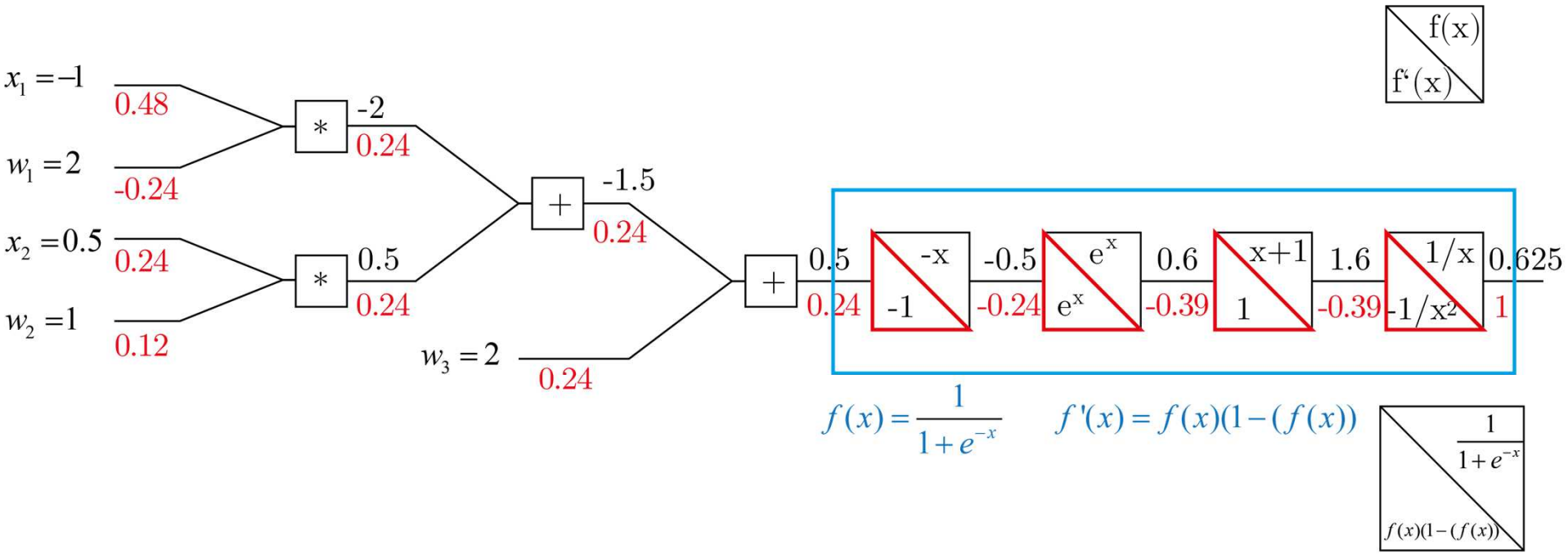
Agenda

1. Chapter: Convolutional Neural Networks
    - 1.1 Motivation
    - 1.2 Introduction
    - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
    - 2.1 Optimizer
    - 2.2 Data Formatting

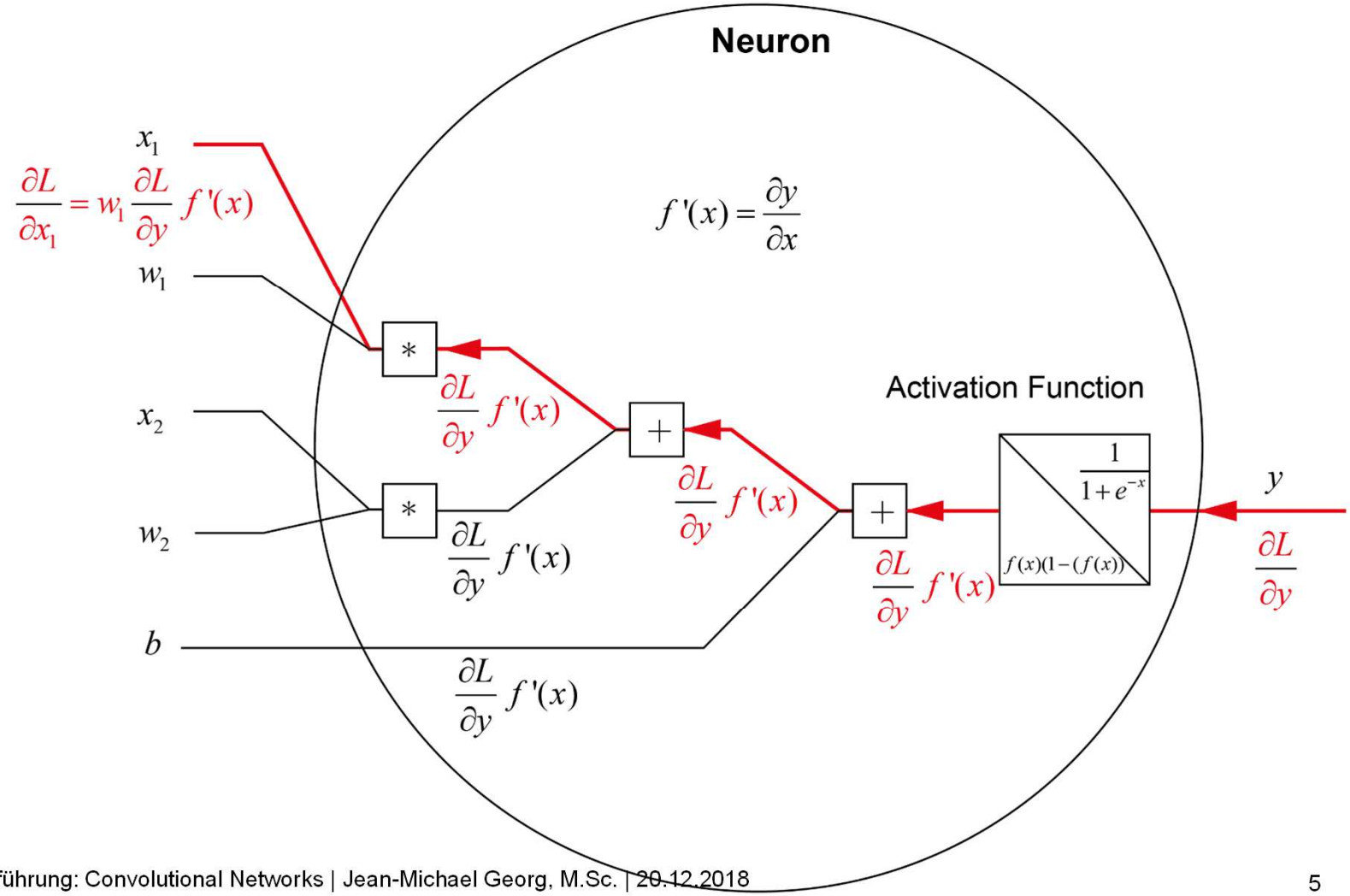


Computational Graph – Complex Example

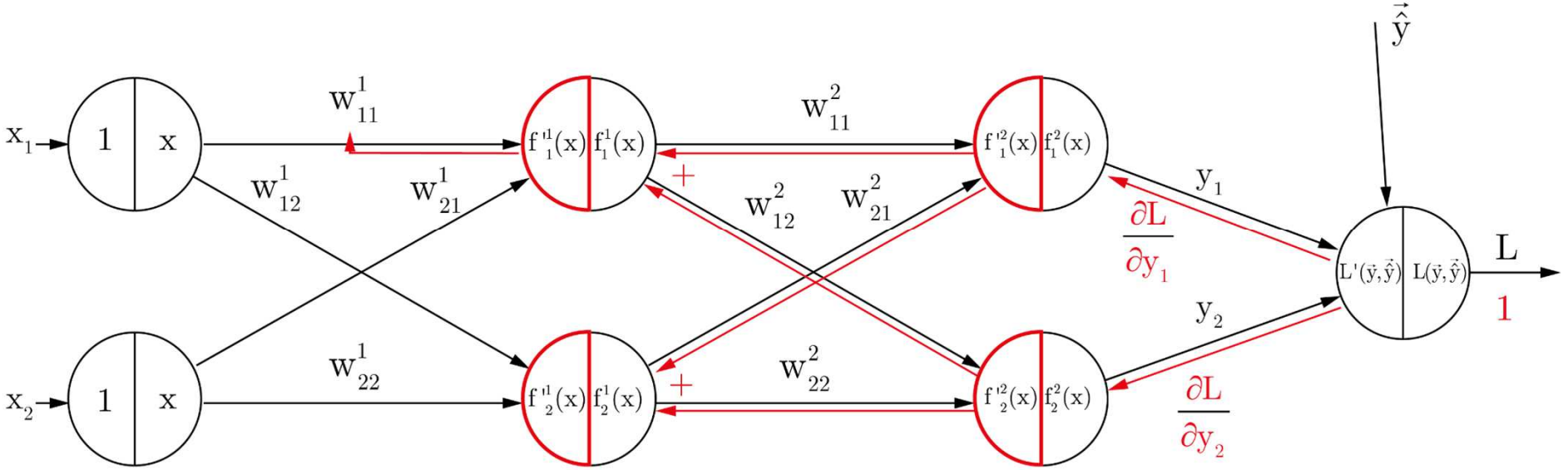
$$f(x_1, x_2) = \frac{1}{1 + e^{-(x_1 w_1 + x_2 w_2 + w_3)}}$$



Computational Graph – Neuron

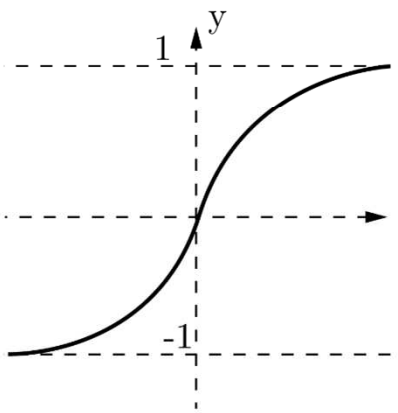


Backpropagation– Neural Network 2 Hidden Layers



Activation Functions

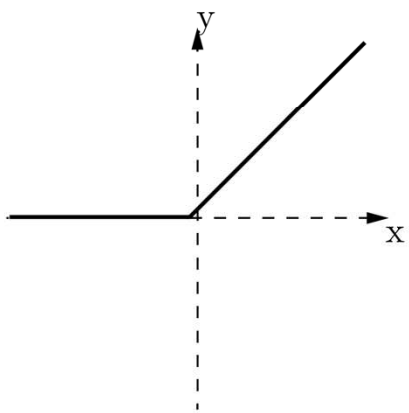
tanh



$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

$$f'(x) = \frac{4}{(e^{x} + e^{-x})^{2}}$$

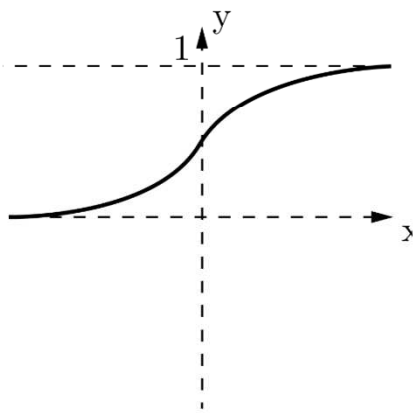
ReLu



$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

$$f'(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

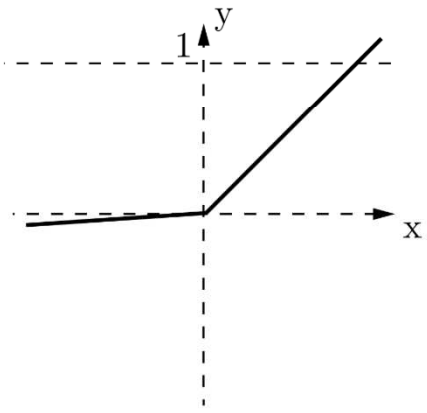
Sigmoid



$$f(x) = \frac{1}{1 + e^{-x}}$$

$$f'(x) = f(x)(1 - f(x))$$

leaky/parametric Relu



$$f(x) = \begin{cases} x, & x \geq 0 \\ ax, & x < 0 \end{cases}$$

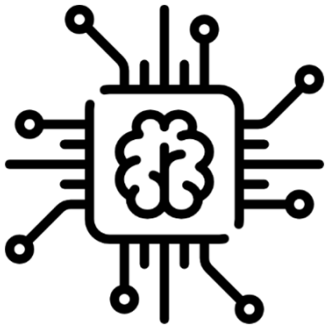
$$f'(x) = \begin{cases} 1, & x \geq 0 \\ a, & x < 0 \end{cases}$$

Deep Neural Networks

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

Agenda

1. Chapter: Convolutional Neural Networks
   - 1.1 Motivation
   - 1.2 Introduction
   - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
   - 2.1 Optimizer
   - 2.2 Data Formatting



Video in our everyday life





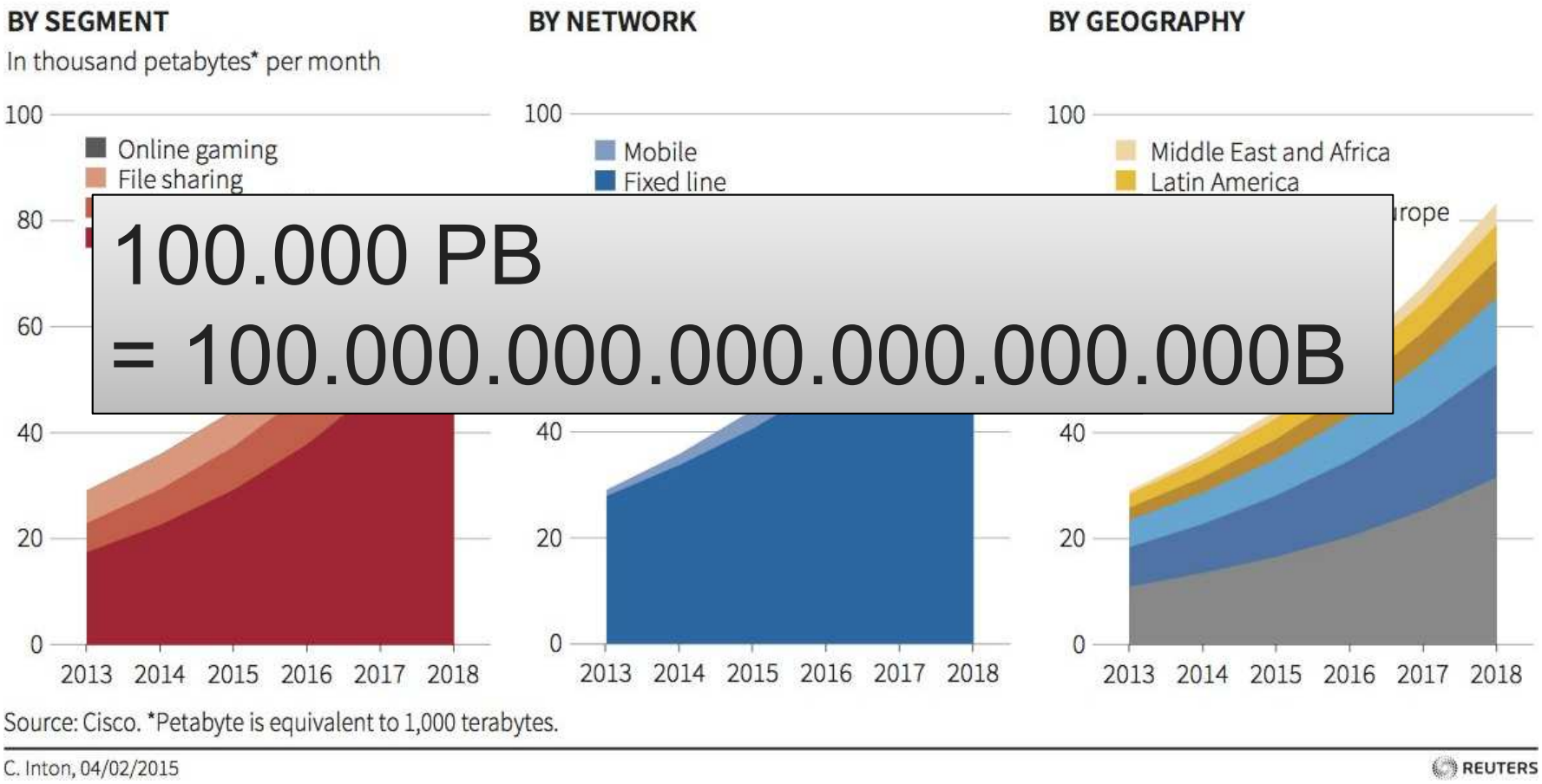






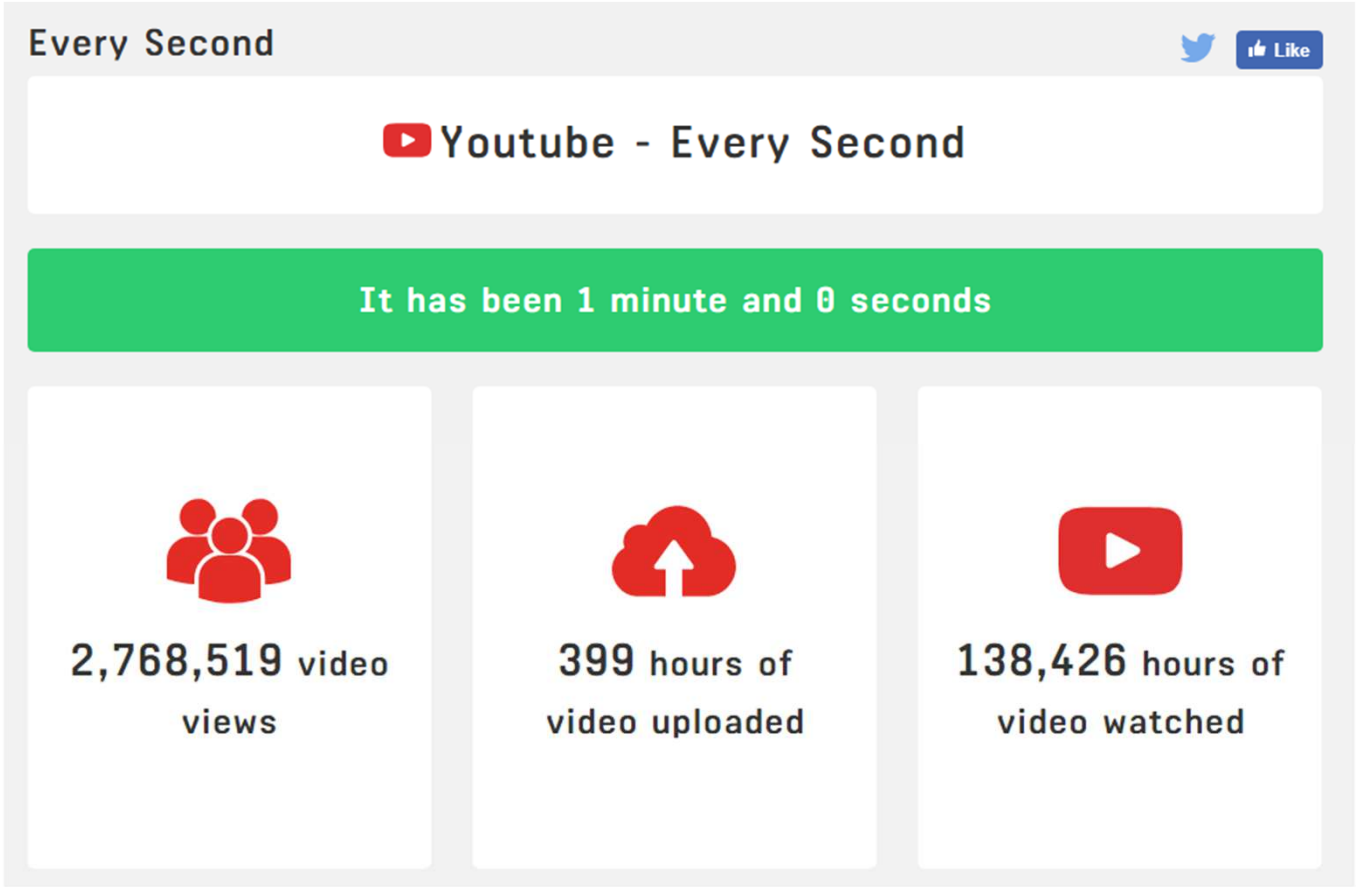

Consumer internet traffic

Internet video traffic will rise from 60 to 75 percent of total consumer internet traffic by 2018, according to estimates by Cisco.



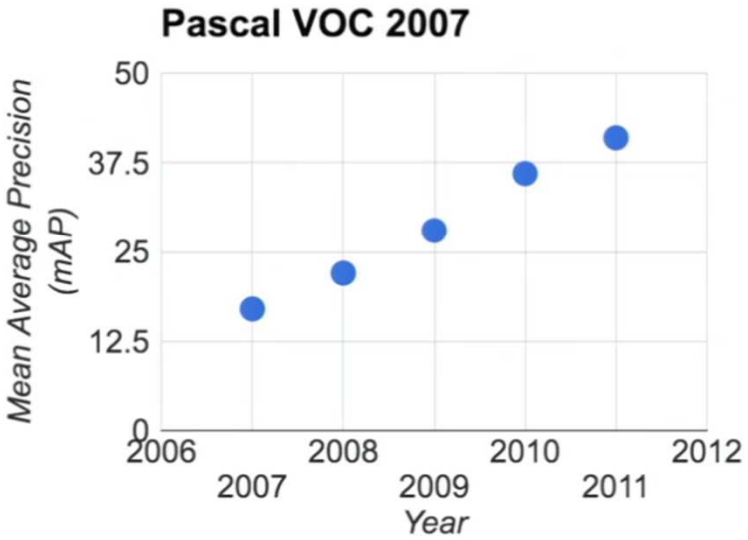
Youtube Every Second

- http://www.everysecond.io/youtube



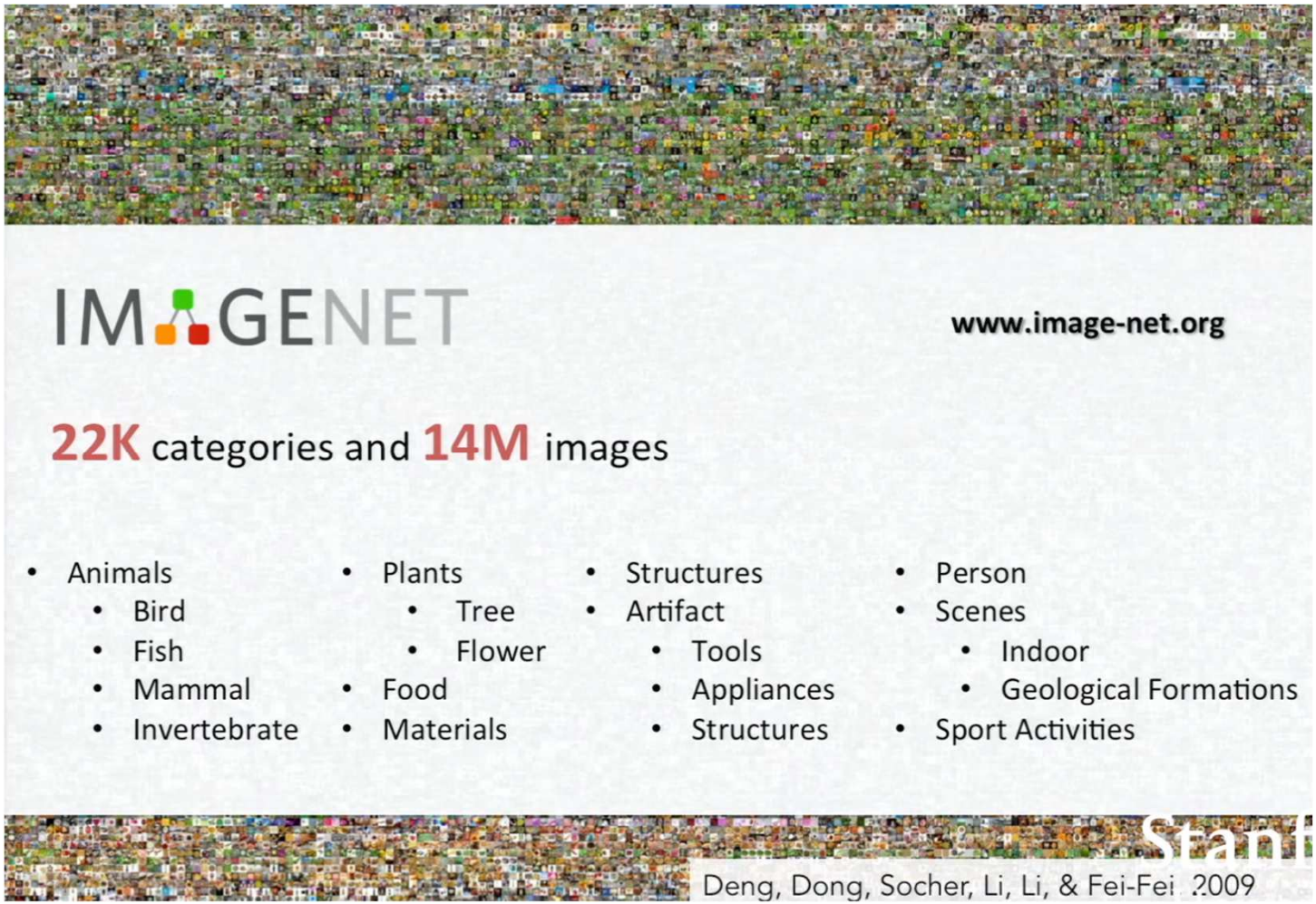
Cameras in autonomous cars

PASCAL Visual Object Challenge

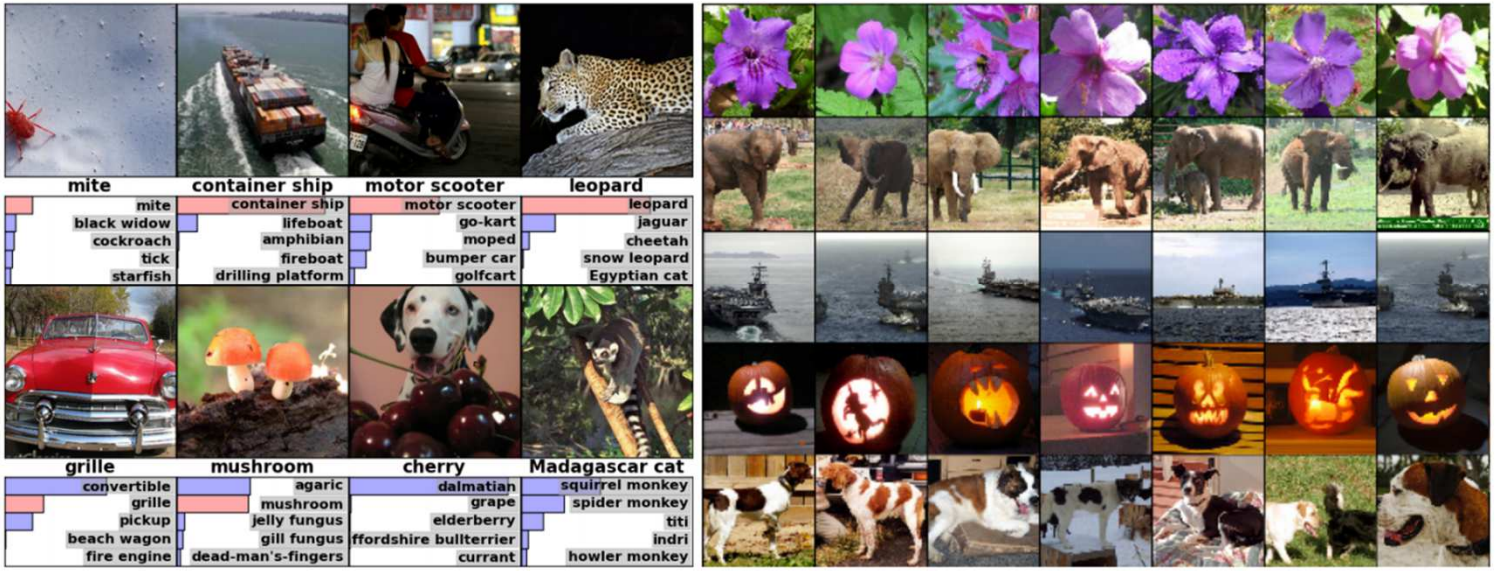


[Everingham et al. 2006-2012]

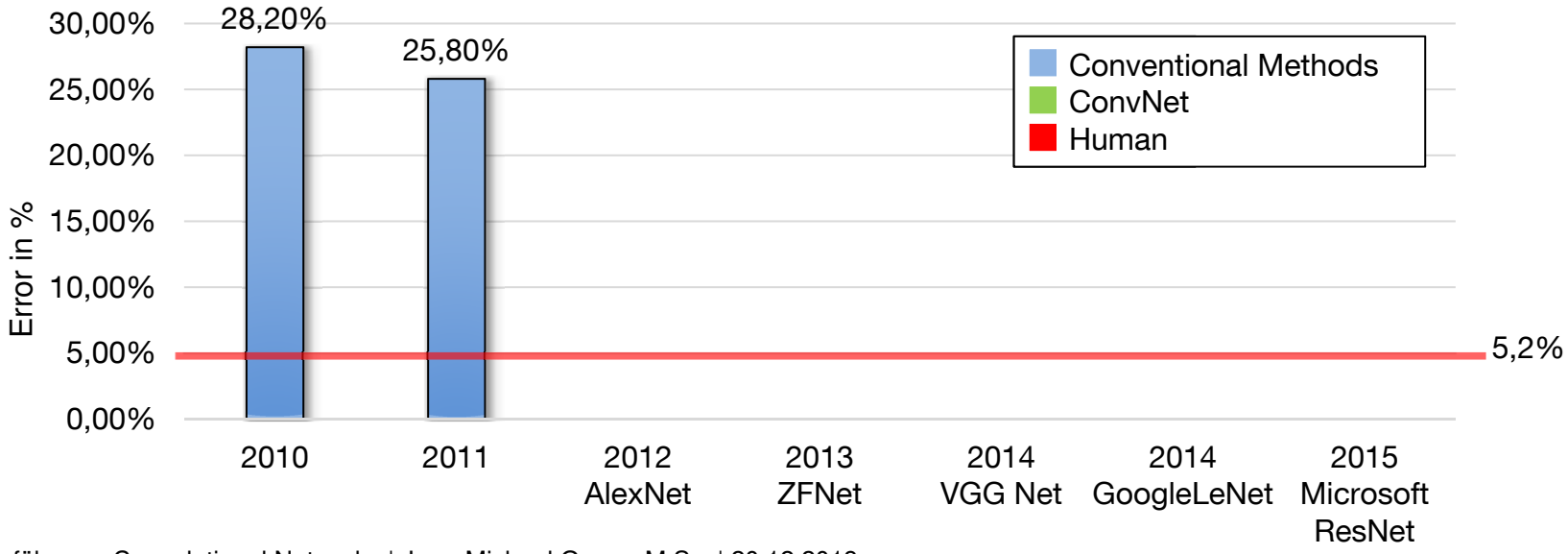
Imagenet


IMAGENET
www.image-net.org
22K categories and 14M images
• Animals
• Bird
• Fish
• Mammal
• Invertebrate
• Plants
• Tree
• Flower
• Food
• Materials
• Structures
• Artifact
• Tools
• Appliances
• Structures
• Person
• Scenes
• Indoor
• Geological Formations
• Sport Activities
Deng, Dong, Socher, Li, Li, & Fei-Fei 2009

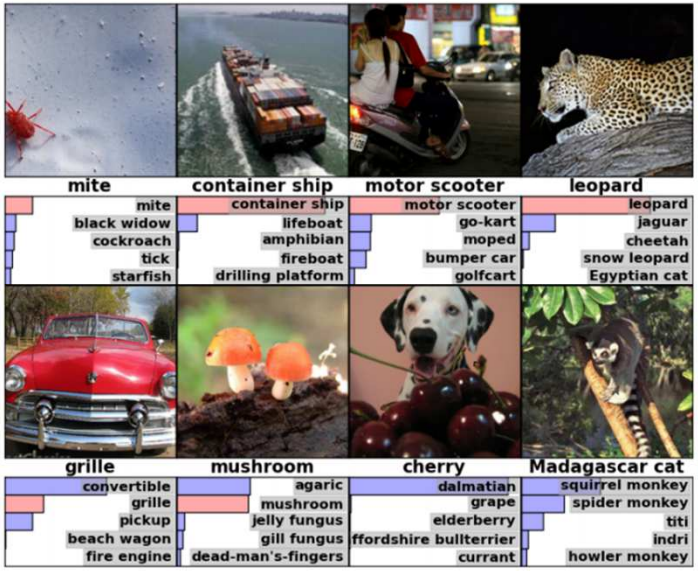
ImageNet Wettbewerb - Bildklassifizierung

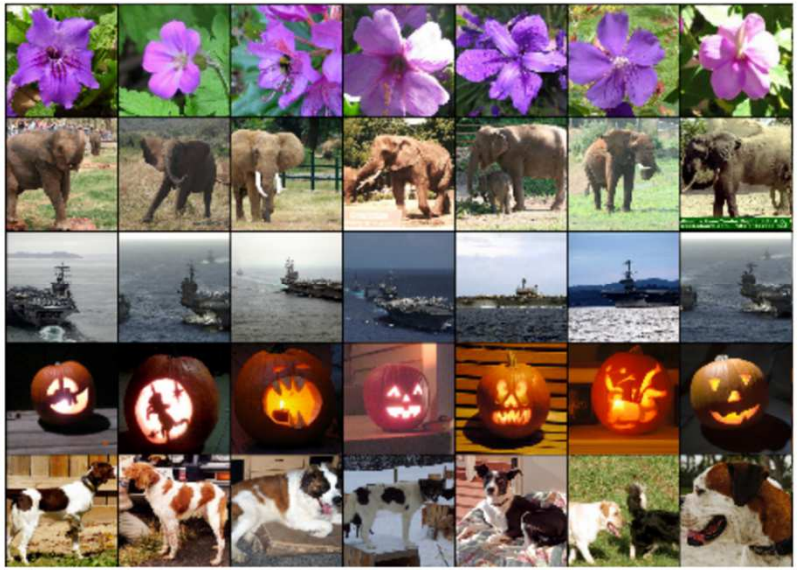


Results

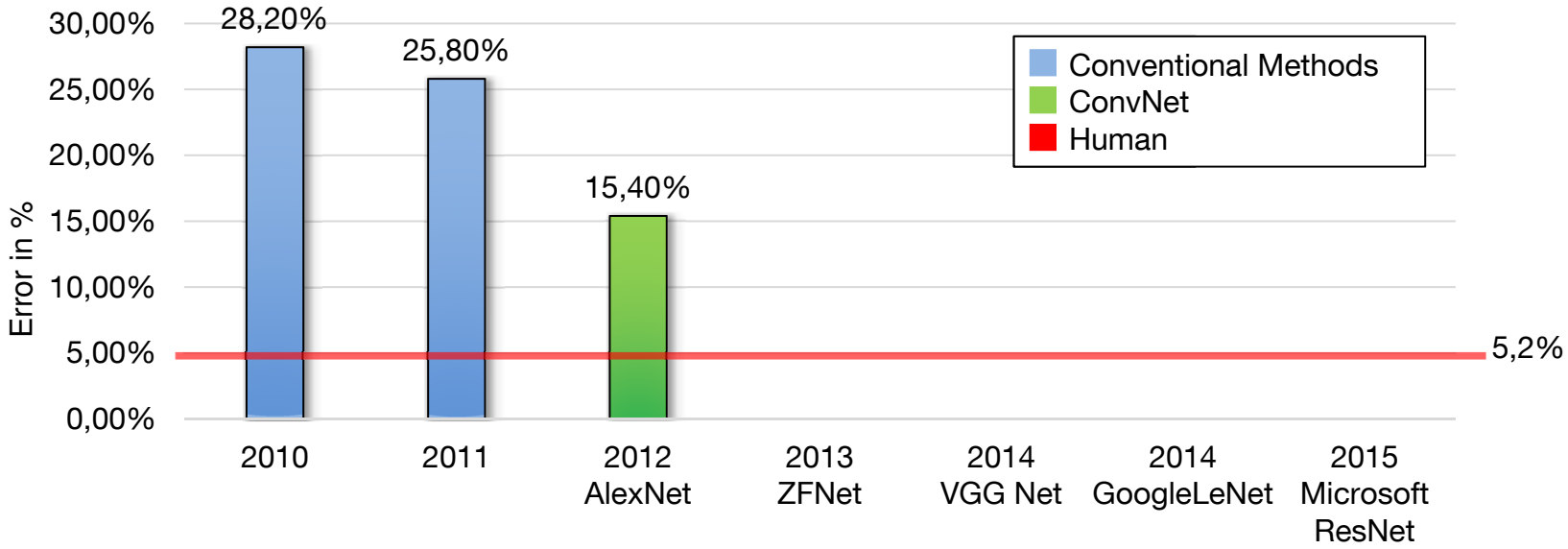


ImageNet Wettbewerb - Bildklassifizierung

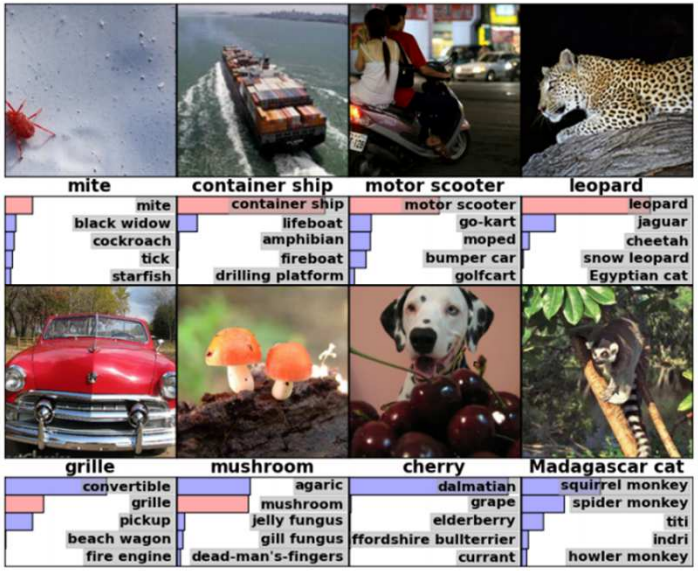


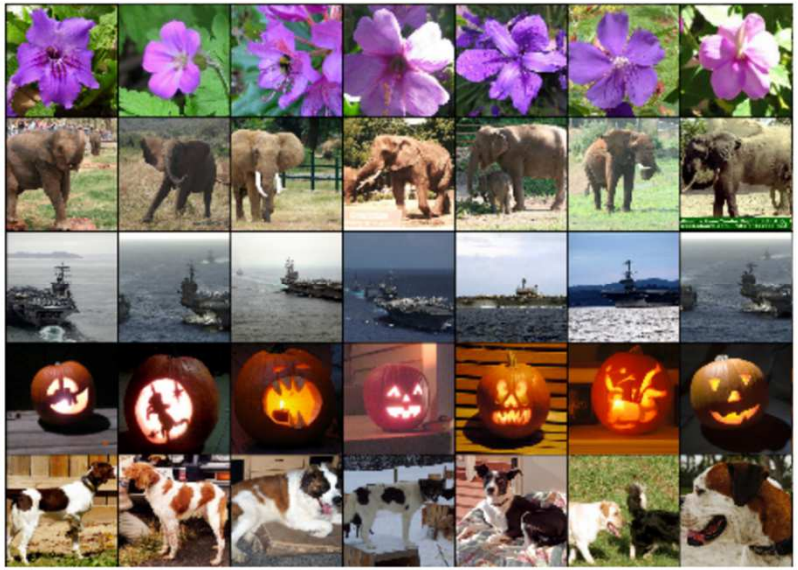


Results

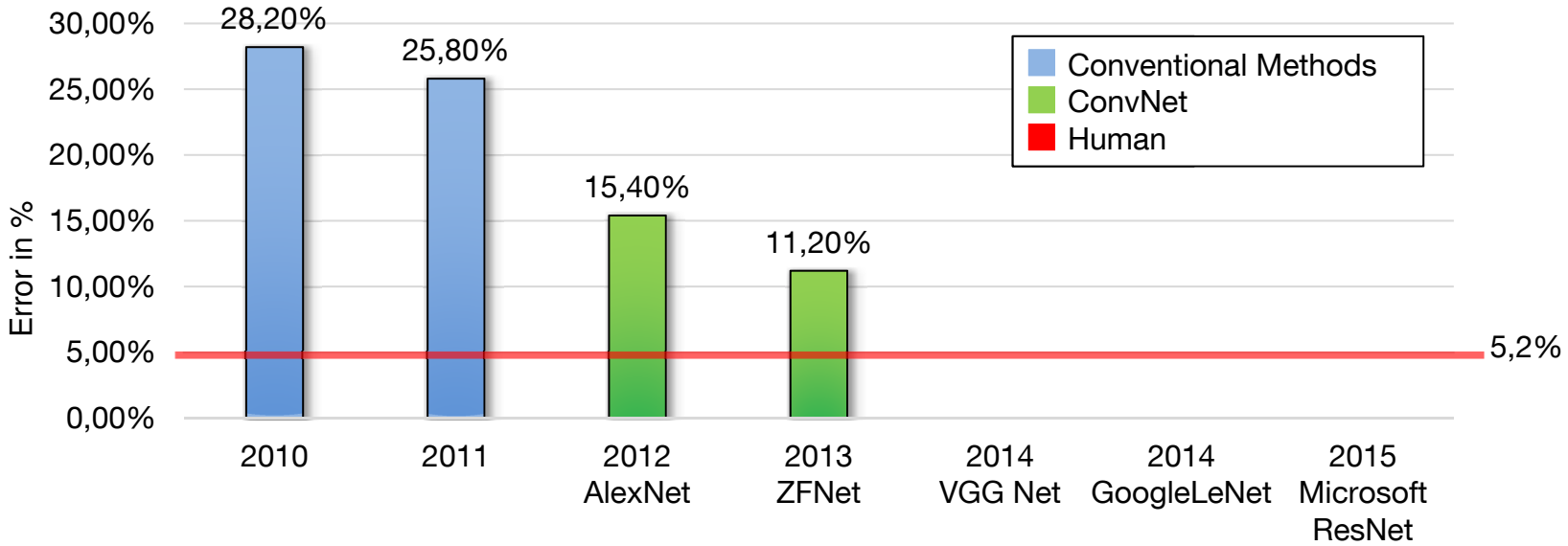


ImageNet Wettbewerb - Bildklassifizierung

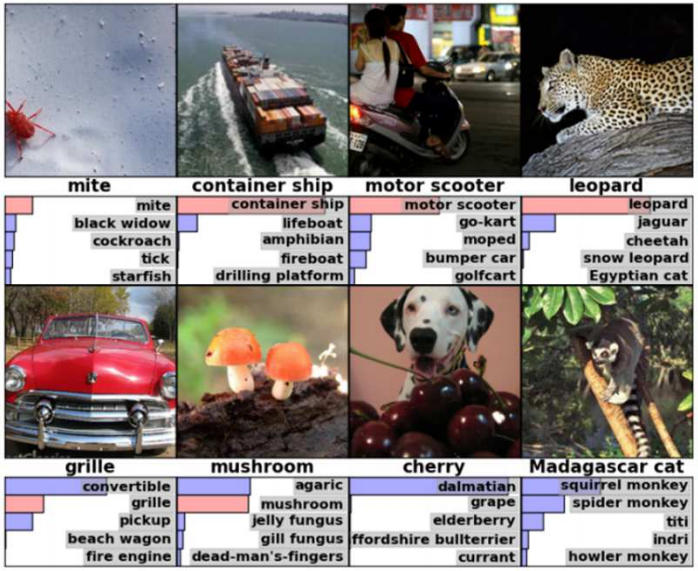


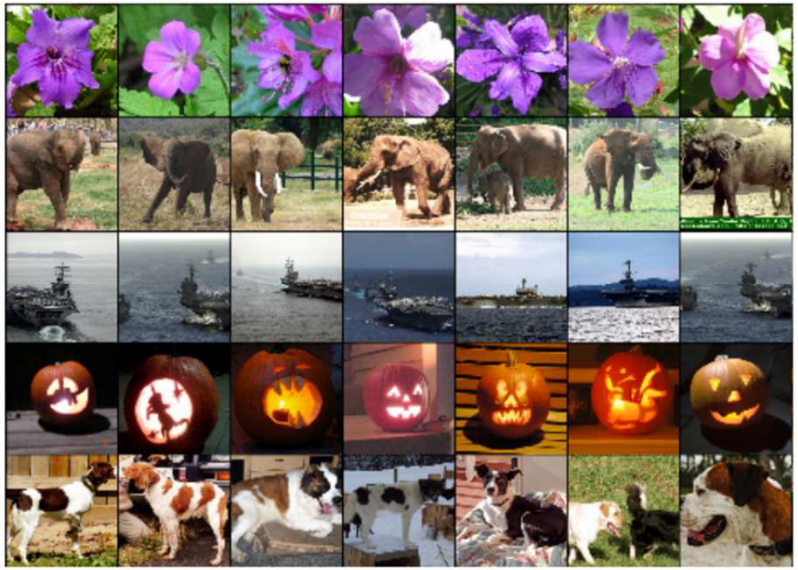


Results

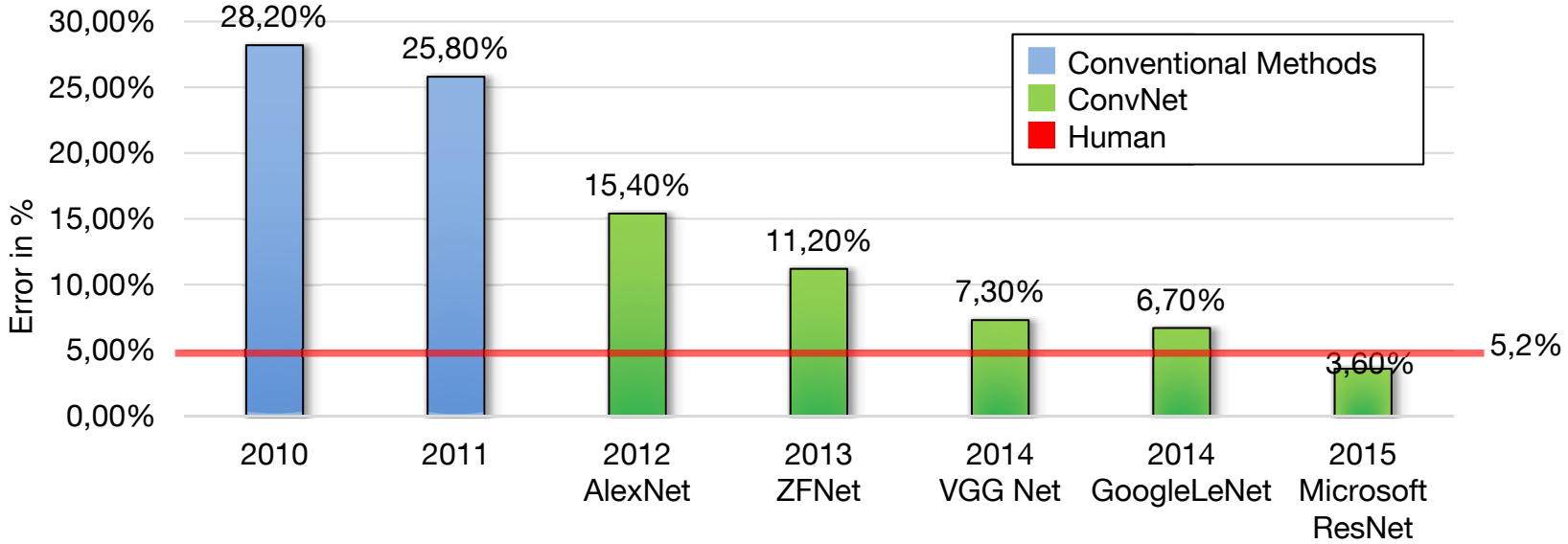


ImageNet Wettbewerb - Bildklassifizierung

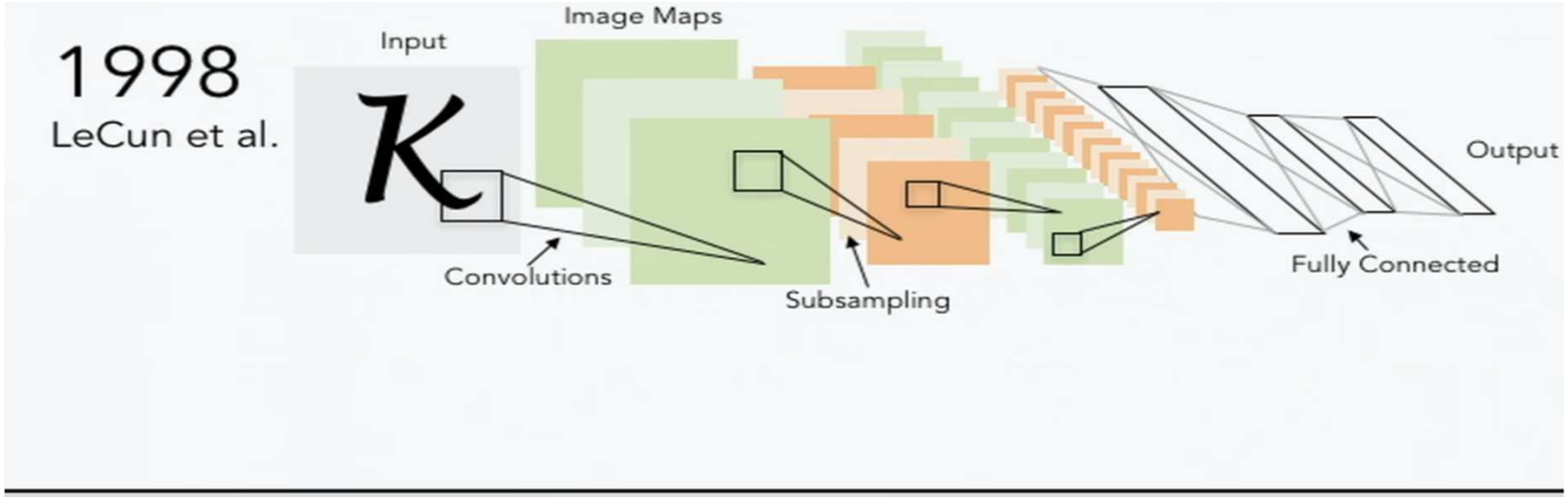




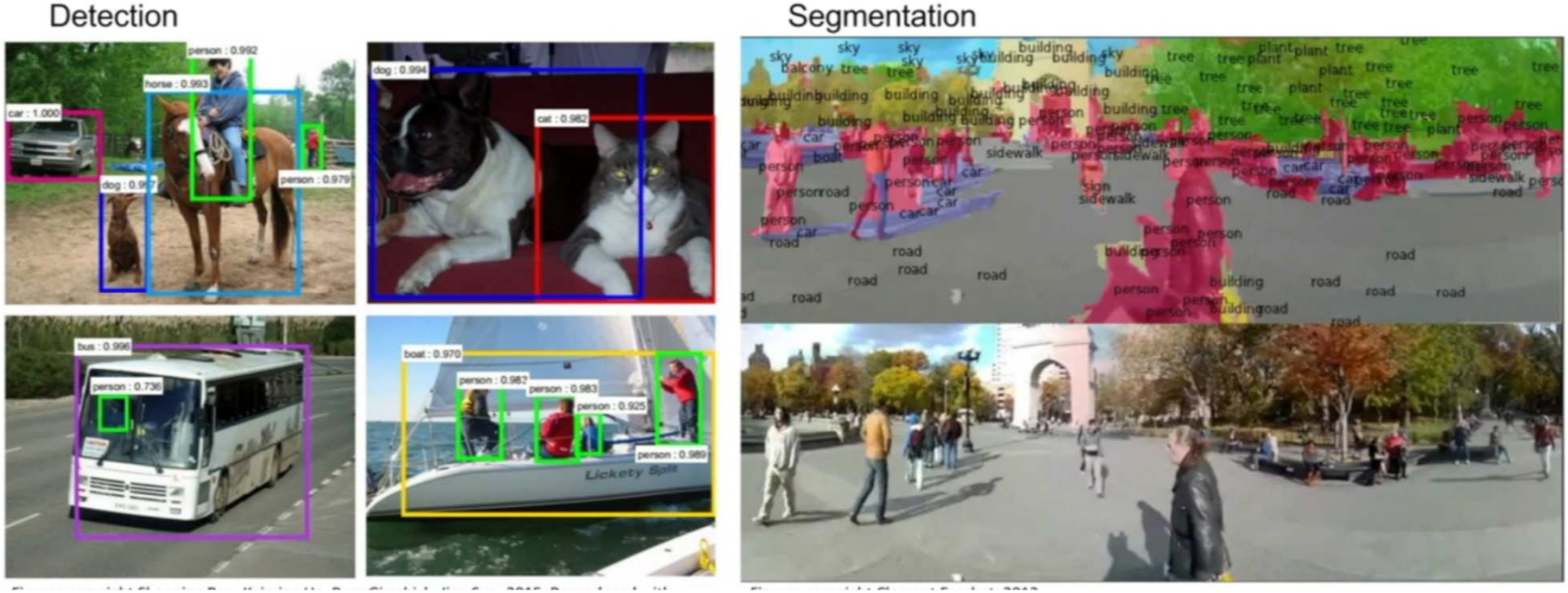
Results



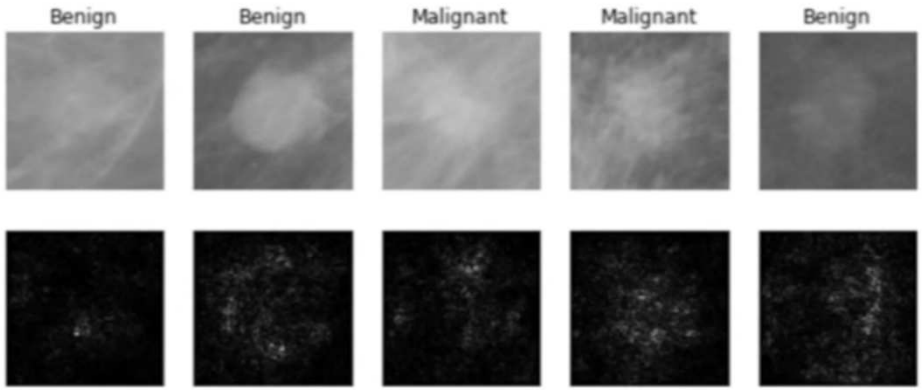
ConvNets exited before 2012…



What else can ConvNets to today ?



What else can ConvNets to today ?



[Levy et al. 2016]

[Dieleman et al. 2014]

[Sermanet et al. 2011]
[Ciresan et al.]

Most important: Convnets can make you rich!

SPIEGEL ONLINE SPIEGEL+ Anmelden

Menü | Politik Meinung Wirtschaft Panorama Sport Kultur Netzwelt Wissenschaft mehr ▾

NETZWELT Schlagzeilen | DAX 11.066,41 | TV-Programm | Abo

Nachrichten > Netzwelt > Web > Christie's > Künstliche Intelligenz: Christie's erzielt mit KI-Gemälde 432.500 Dollar

KI-Kunstwerk versteigert

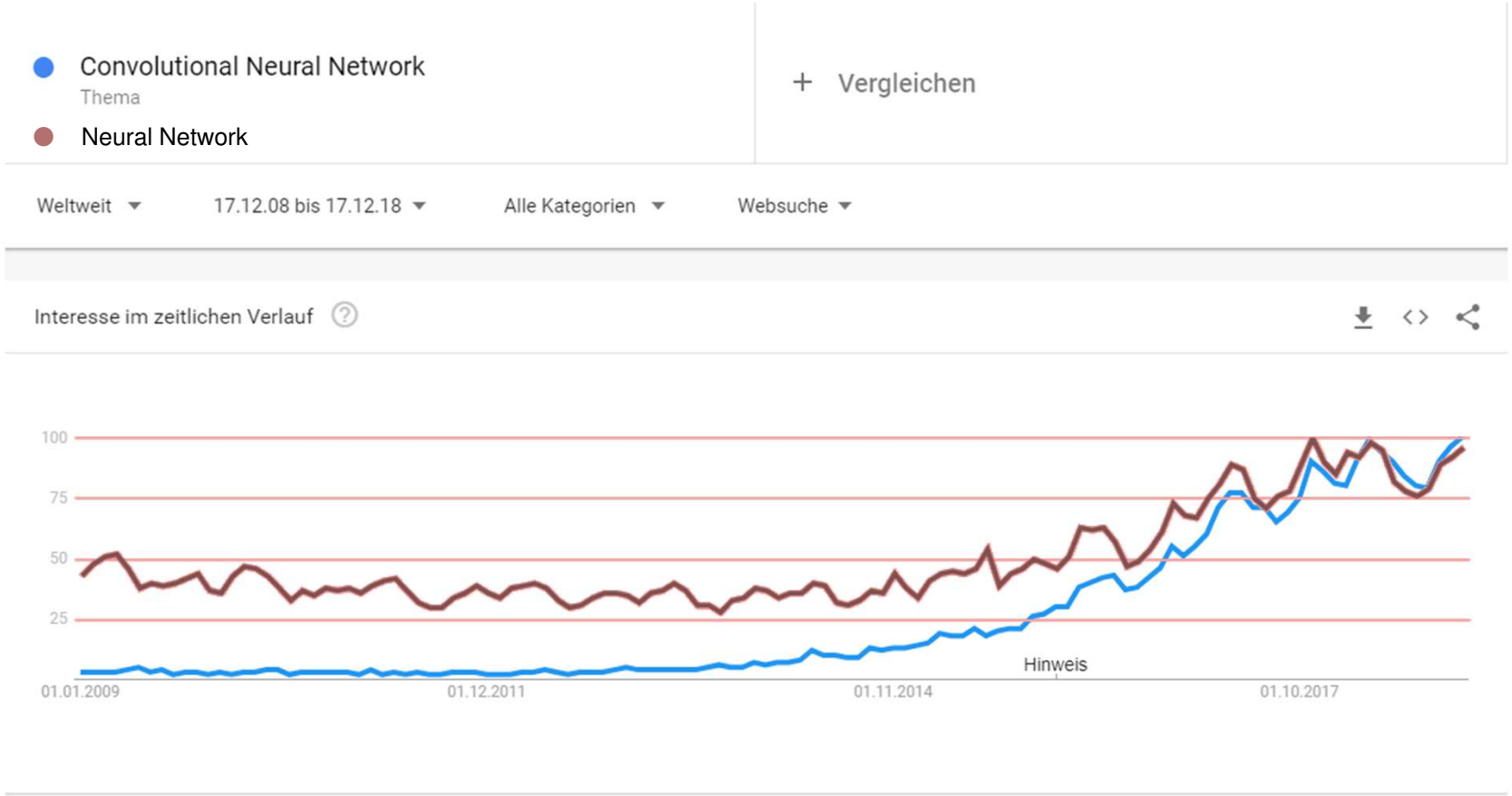
Gemälde von "min G max D Ex[log(D(x))]+Ez[log(1-D(G(z)))]" erzielt 432.500 Dollar

Das Werk "Edmond de Belamy" wurde von einem künstlichen neuronalen Netzwerk geschaffen, nun hat das Auktionshaus Christie's es versteigert - und für das KI-Gemälde deutlich mehr bekommen als ursprünglich geschätzt.

26.10.2018

Convolutional Neural Networks

Google Trend

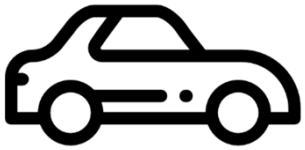


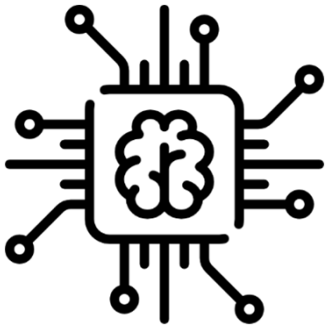
Deep Neural Networks

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

Agenda

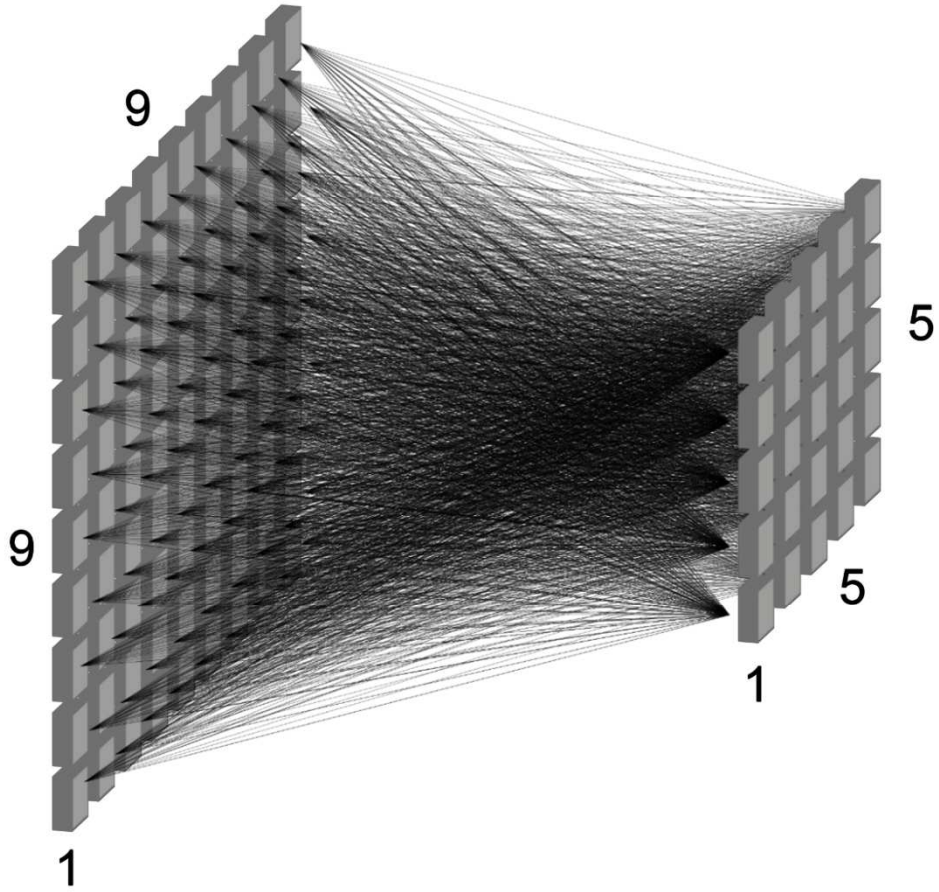
1. Chapter: Convolutional Neural Networks
    - 1.1 Motivation
    - 1.2 Introduction
    - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
    - 2.1 Optimizer
    - 2.2 Data Formatting



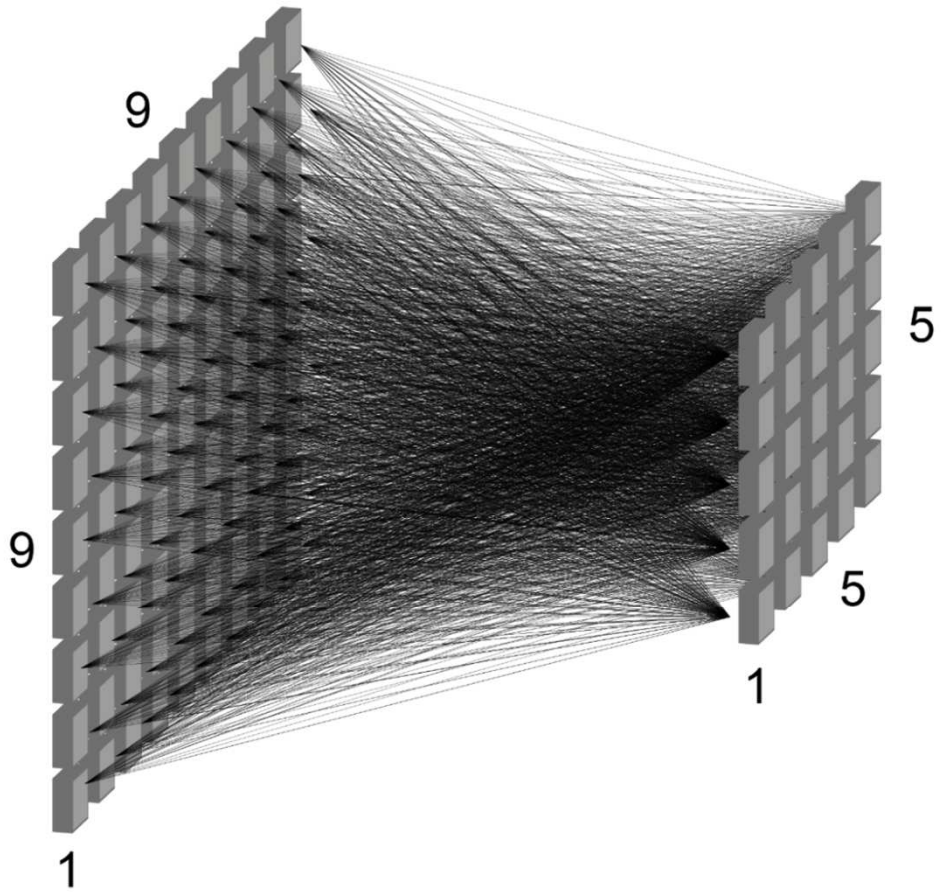


But…
…what are Convolutional Neural Networks?
...how do they work?
...and why do we need them?

Fully Connected Layer



Fully Connected Layer



$$\vec{x} \in K^{1\times 81}$$

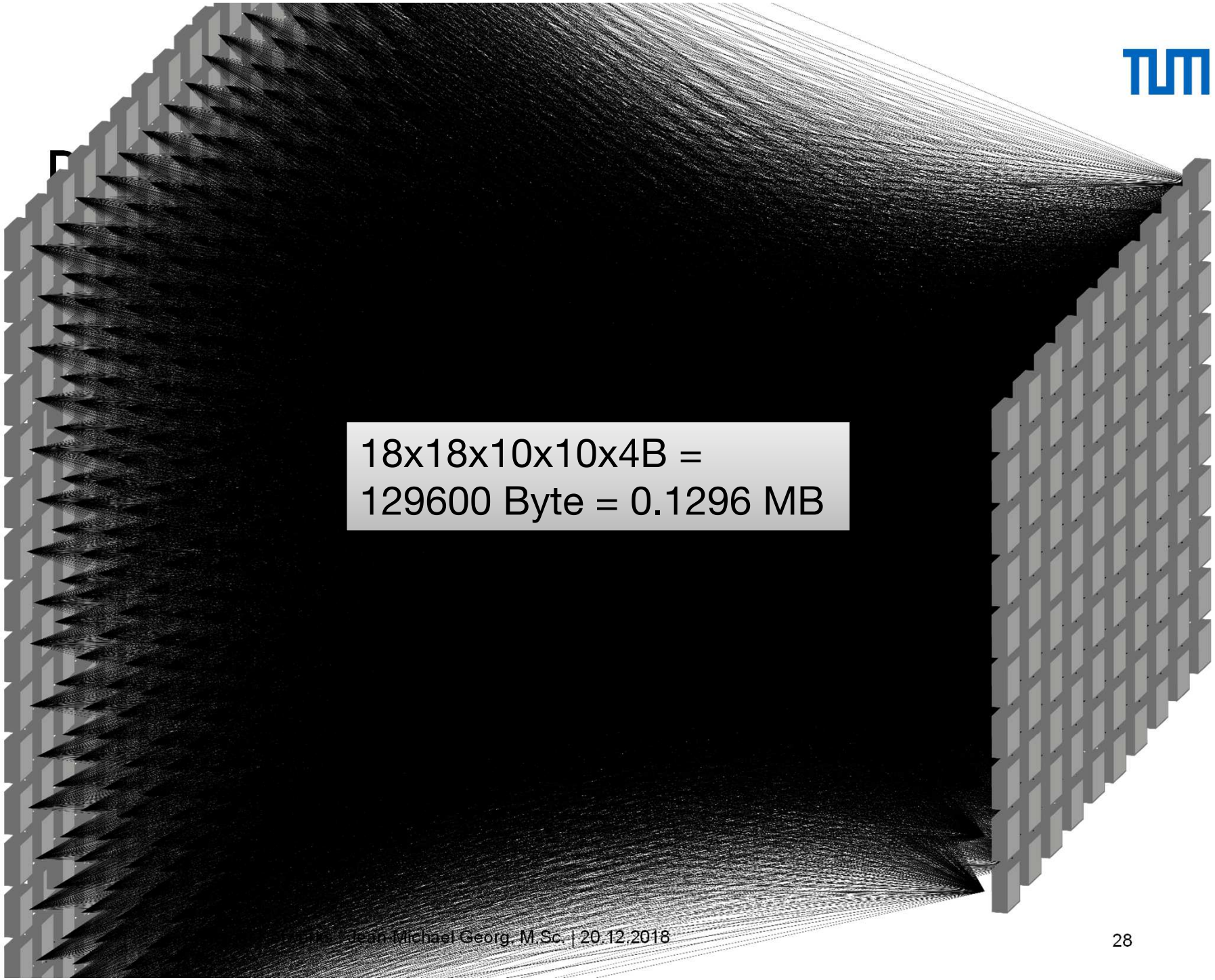
$$\vec{y} \in K^{1\times 25}$$

$$W \in K^{81x25}$$

$$W = \begin{pmatrix} w_{11} & \cdots & w_{1,25} \\ \vdots & \ddots & \vdots \\ w_{81,1} & \cdots & w_{81,25} \end{pmatrix}$$

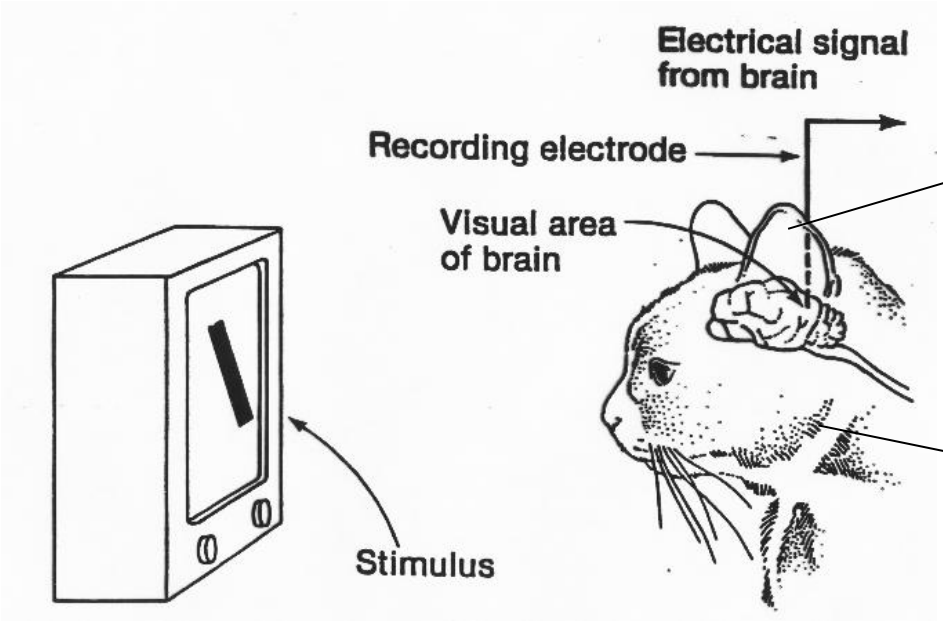
$$\Rightarrow 9 \cdot 9 \cdot 5 \cdot 5 = 2025 \text{ Parameter}$$

2025x4B =
8100 Byte = 8 kB

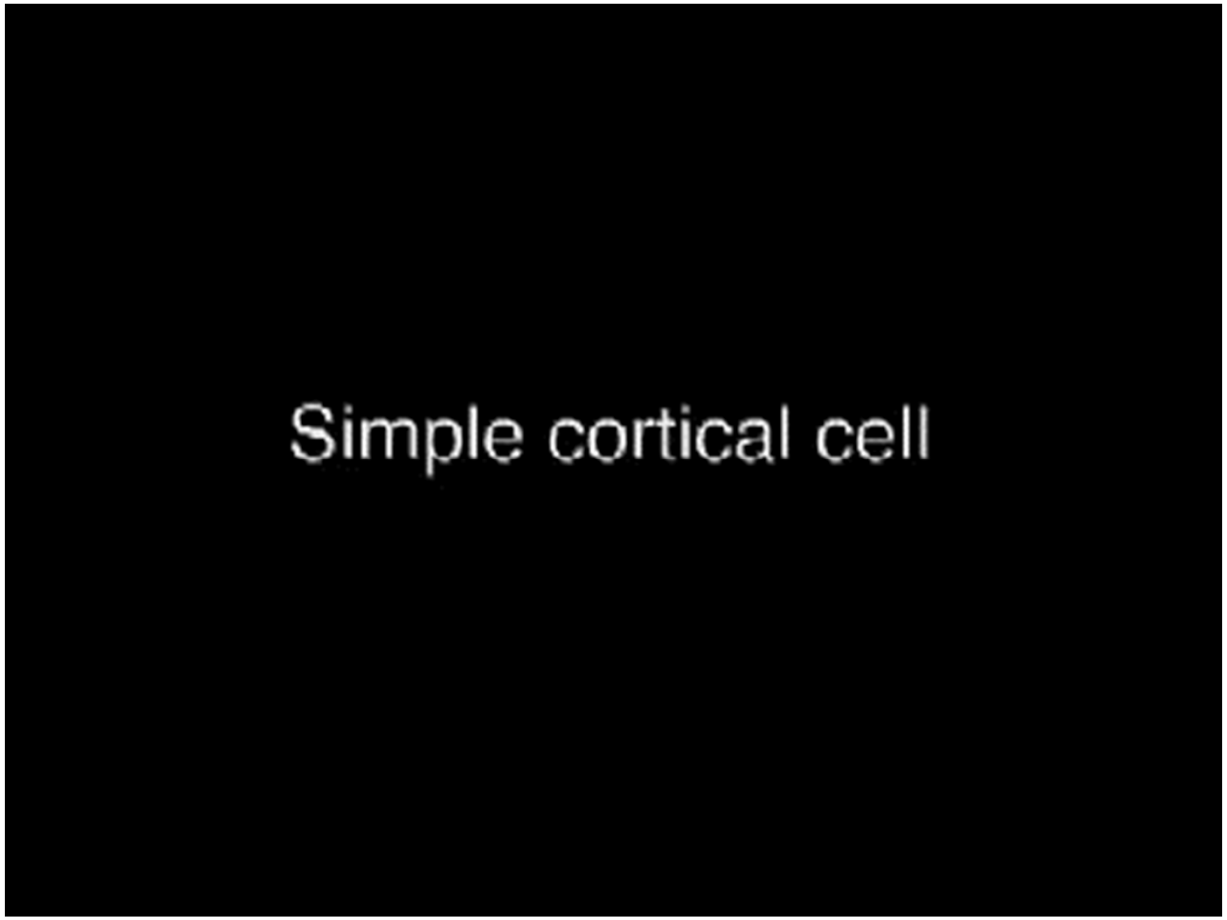

18x18x10x10x4B =
129600 Byte = 0.1296 MB

FullHD x FullHD?
1920x1080x3 x 1920x1080x3x 4B =
1.54729341e+14 B =
154.293 GB =
154 TB =
Internet Traffic of 12800 people in 2018
=> For images we need something else

Hubel and Wiesel Experiment 1959



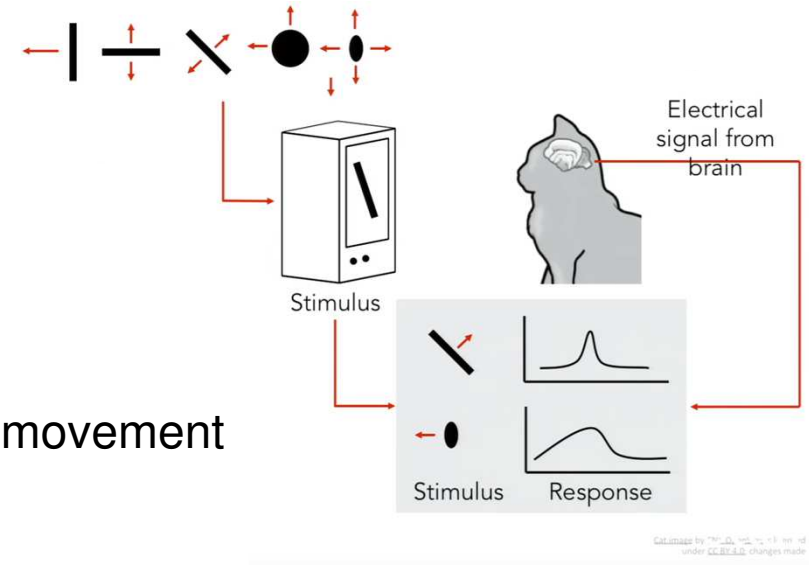
Hubel and Wiesel Experiment Video



Hubel and Wiesel

Results

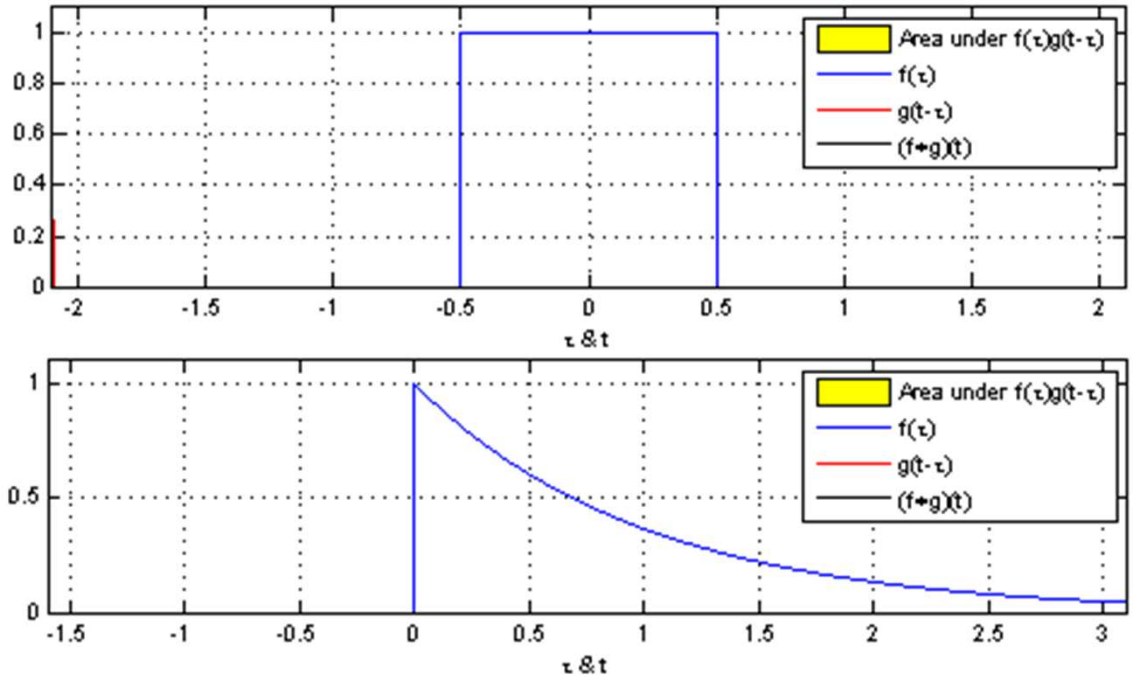
- **Simple cell:**
  - Responses to light orientation
- **Complex cell:**
  - Responses to light orientation and movement
- **Hypercomplex cells:**
  - Repsonse to movement with and end point



Convolution Layer

Math and other domains

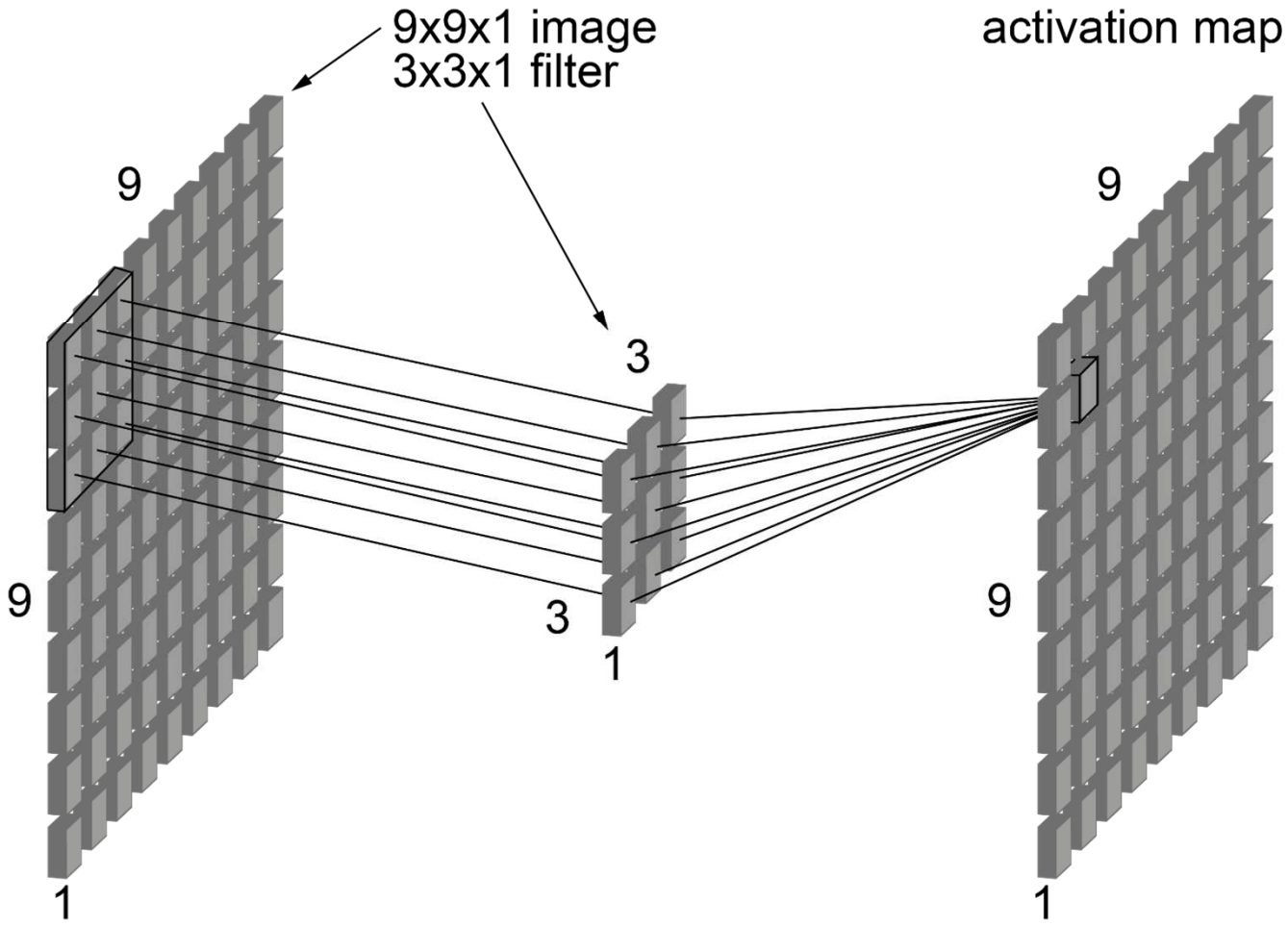
Definition: $$(f * g)(x) = \int_{\mathbb{R}^n} f(\tau) g(x - \tau) \mathrm{d}\tau$$



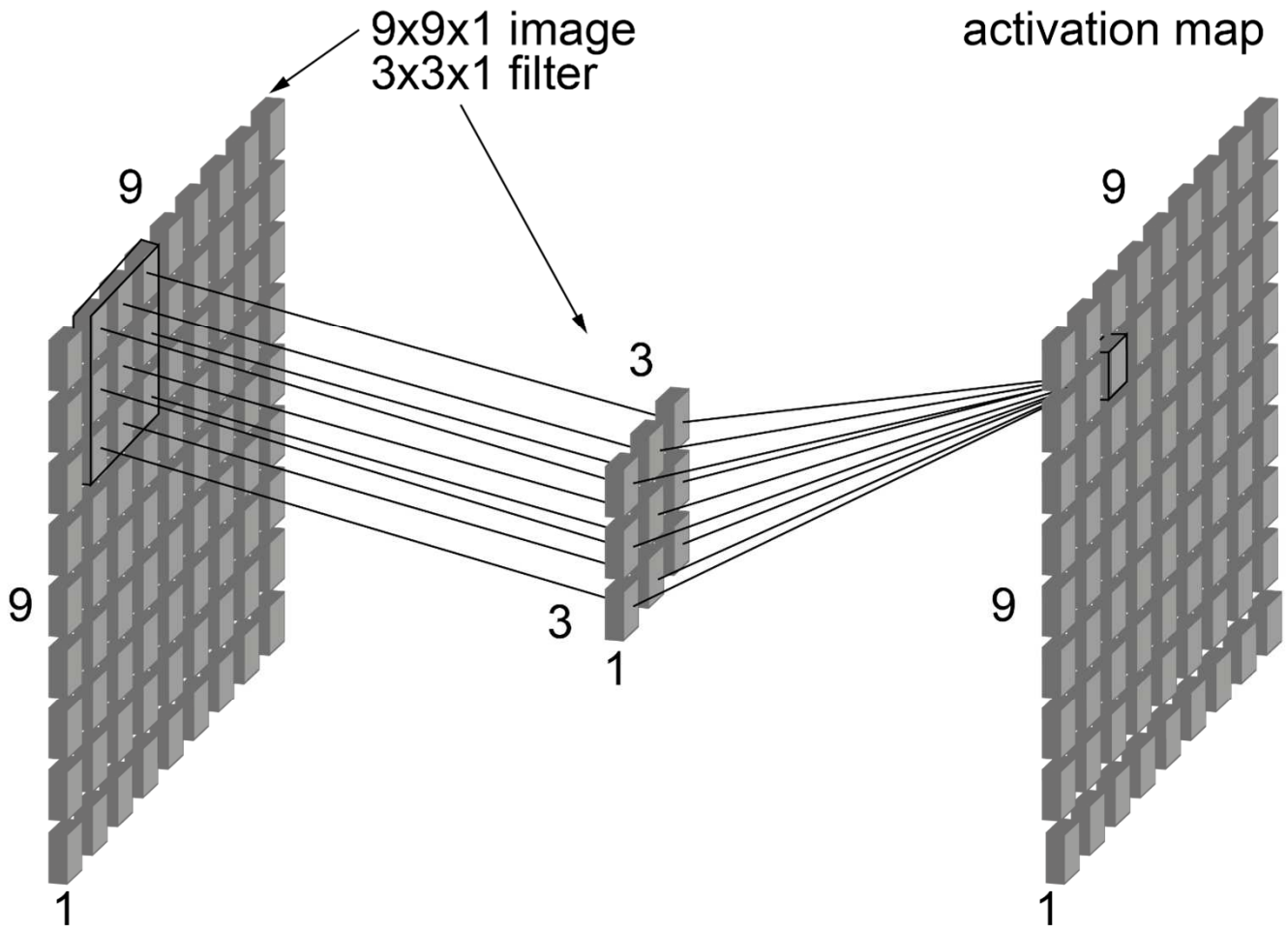
$$f(x, y) * g(x, y) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f(n_1, n_2) \cdot g(x - n_1, y - n_2)$$

https://en.wikipedia.org/wiki/Convolution

Convolution Layer

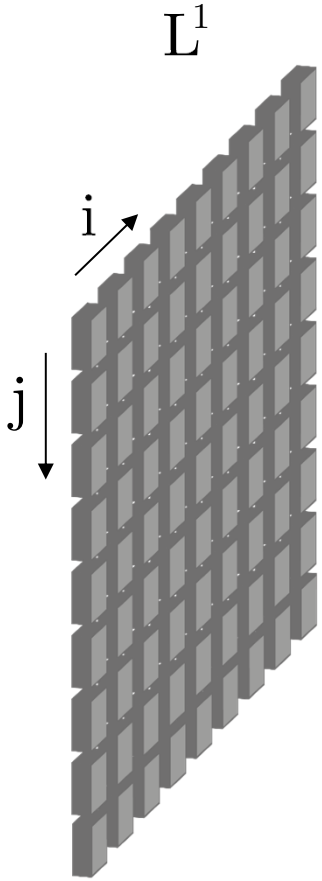


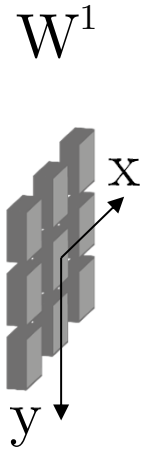
Convolution Layer



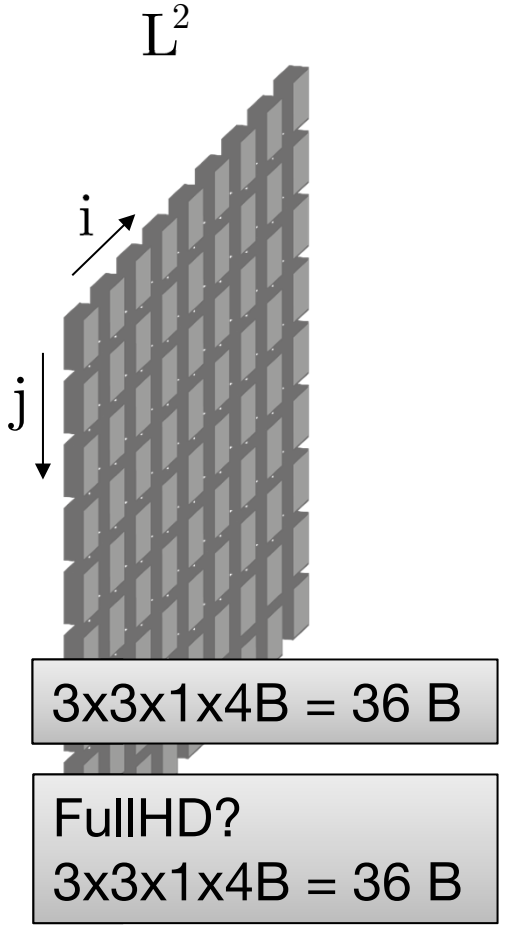
Convolution Layer

$$L^2_{i,j} = f\left(\sum_{x=-1}^{1}\sum_{y=-1}^{1} L^1_{x+i,y+j} \cdot w_{x,y} + b_{i,j}\right)$$

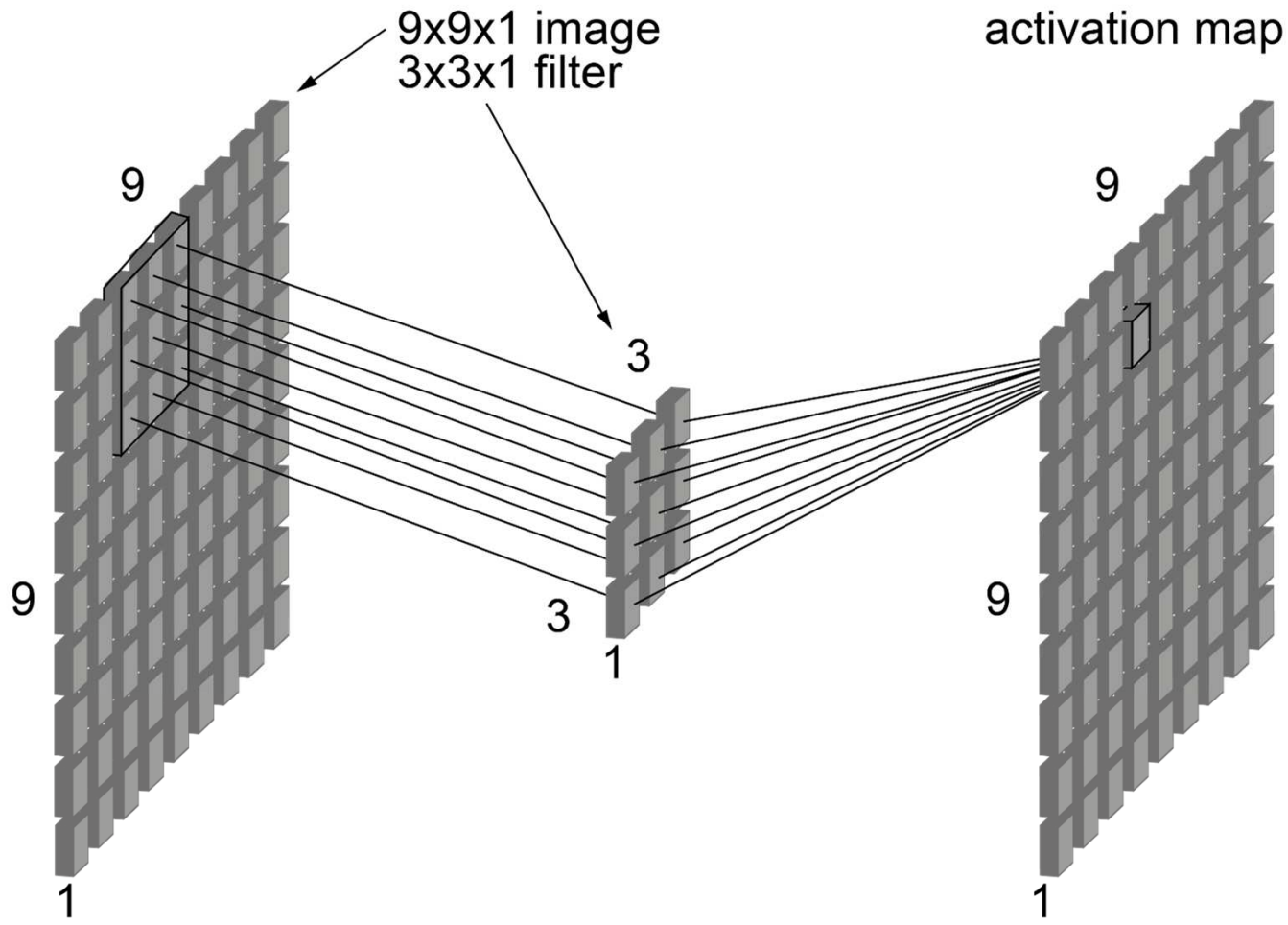




$$W^1 = \begin{pmatrix} w_{-1,-1} & w_{0,-1} & w_{1,-1} \\ w_{-1,0} & w_{0,0} & w_{1,0} \\ w_{-1,1} & w_{0,1} & w_{1,1} \end{pmatrix}$$

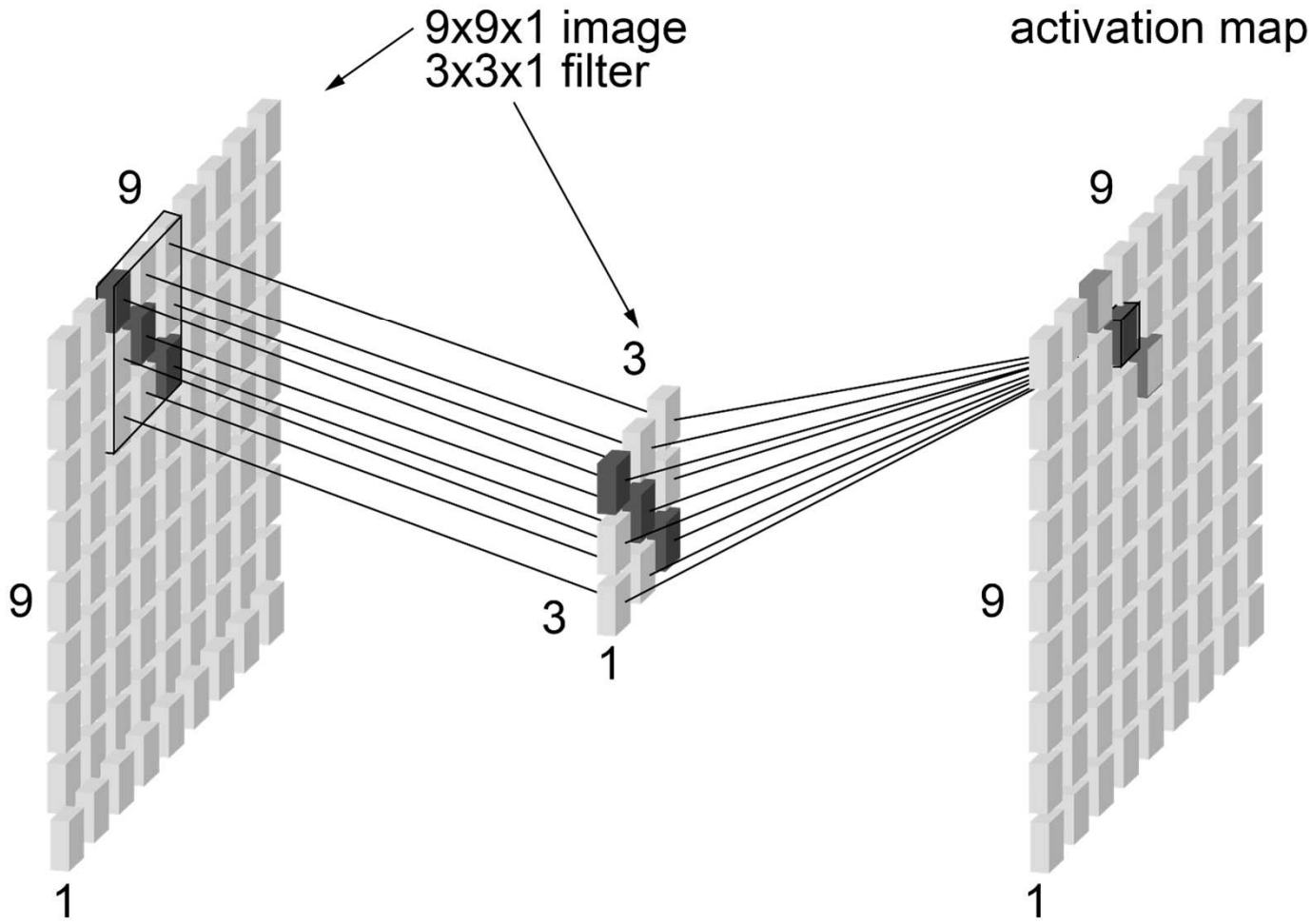


Convolution Layer

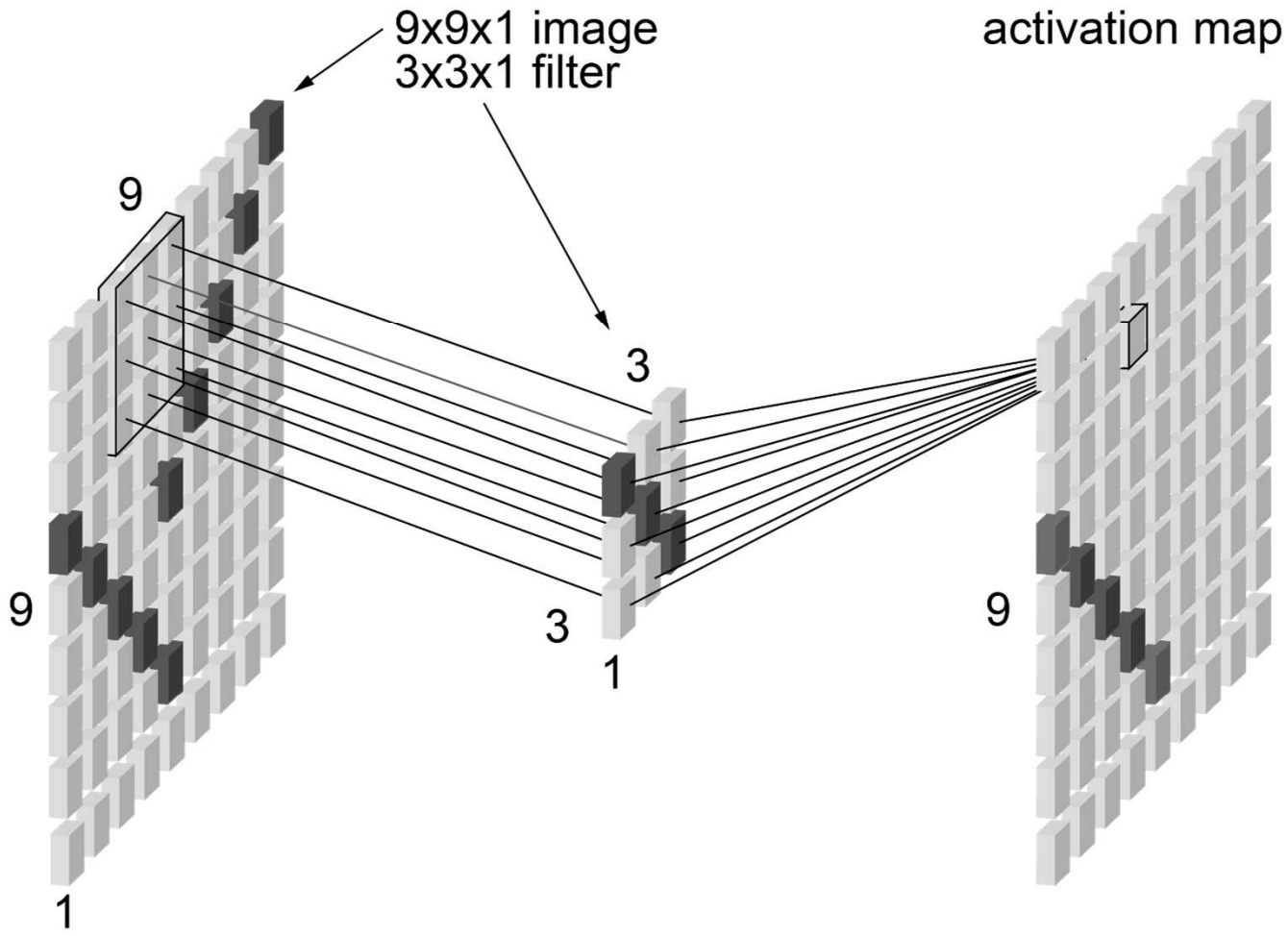


Convolution Layer

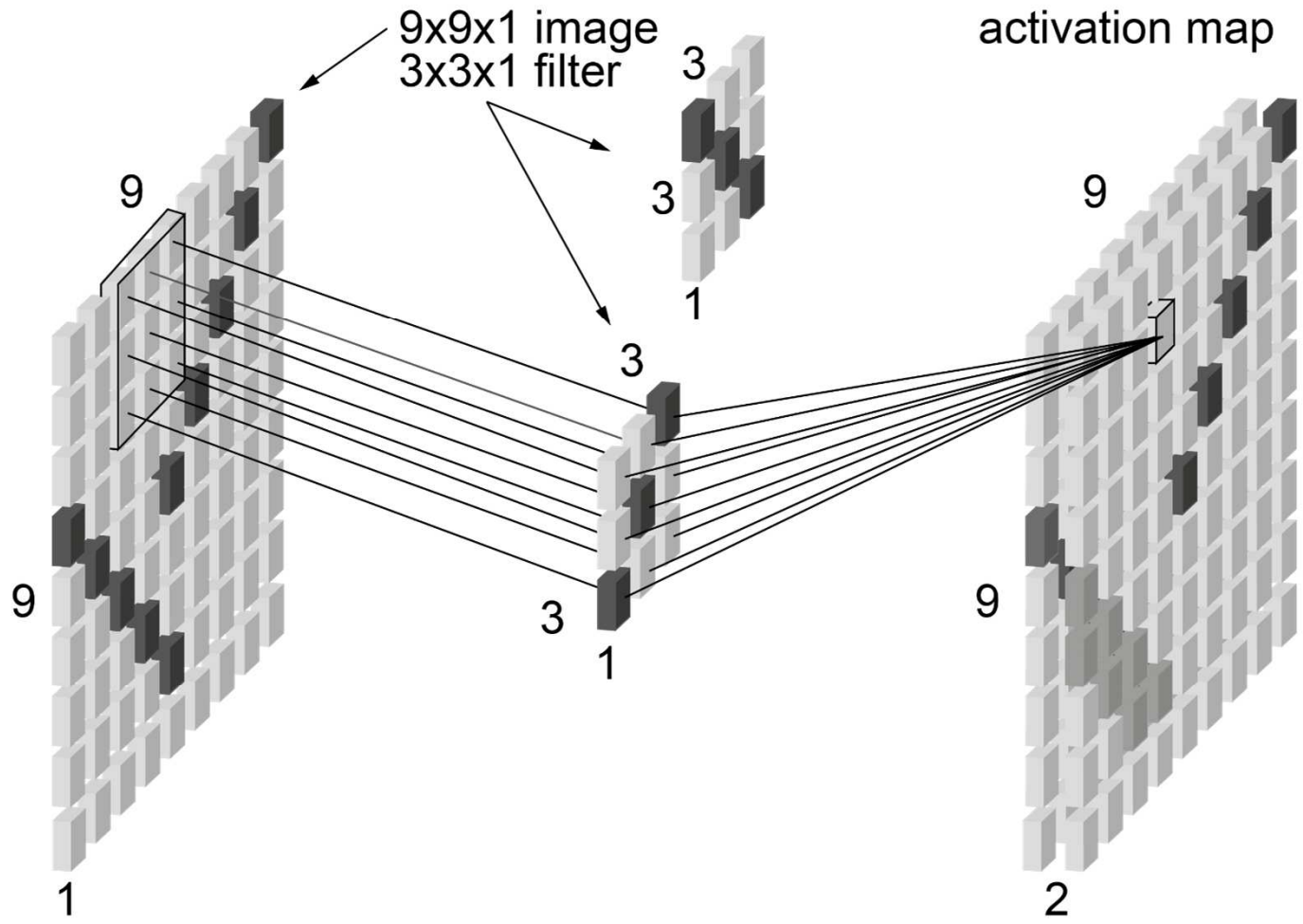
$$L^2_{i,j} = f\left(\sum_{x=-1}^{1}\sum_{y=-1}^{1} L^1_{x+i,y+j} \cdot w_{x,y} + b_{i,j}\right)$$



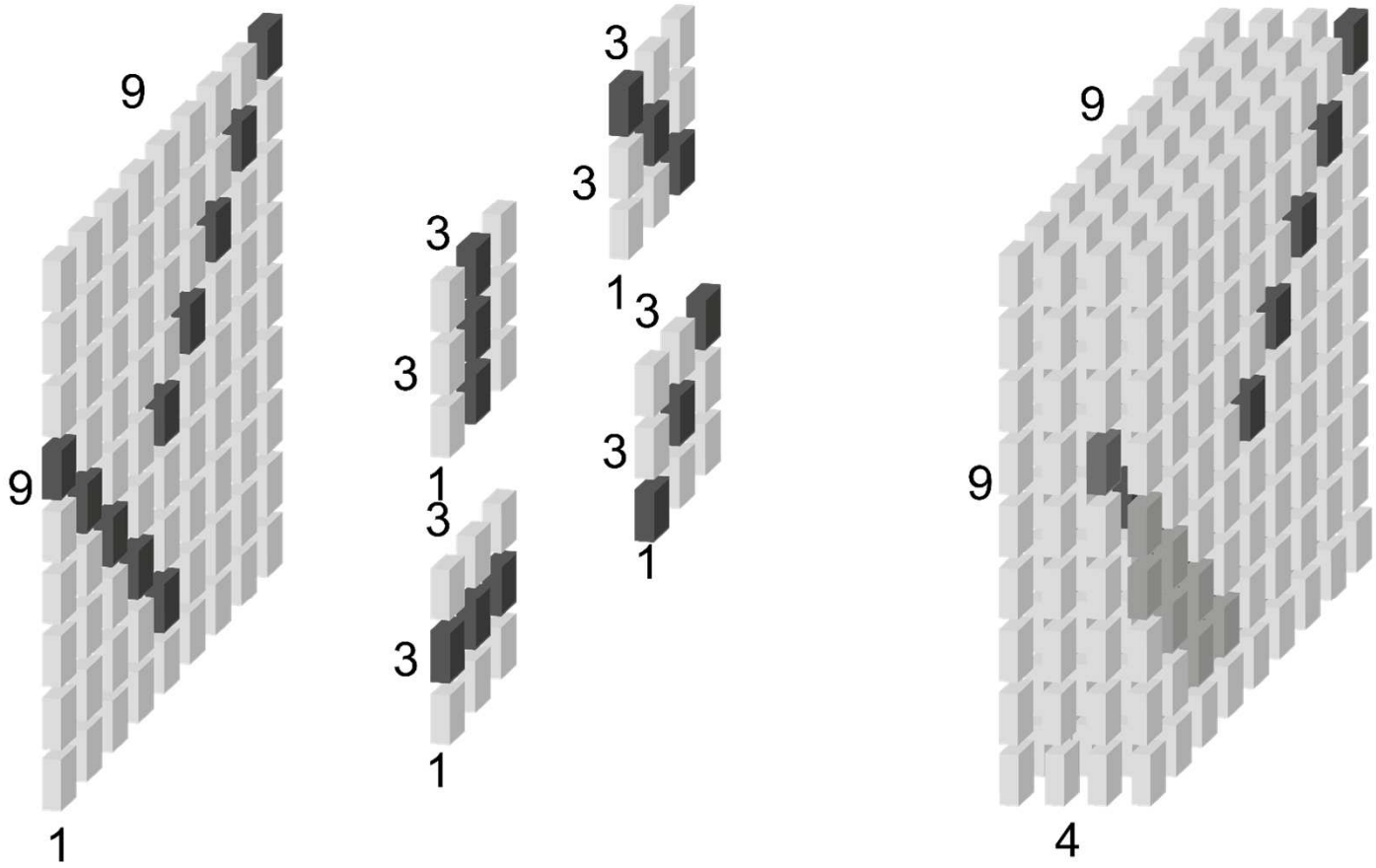
Convolution Layer



Convolution Layer

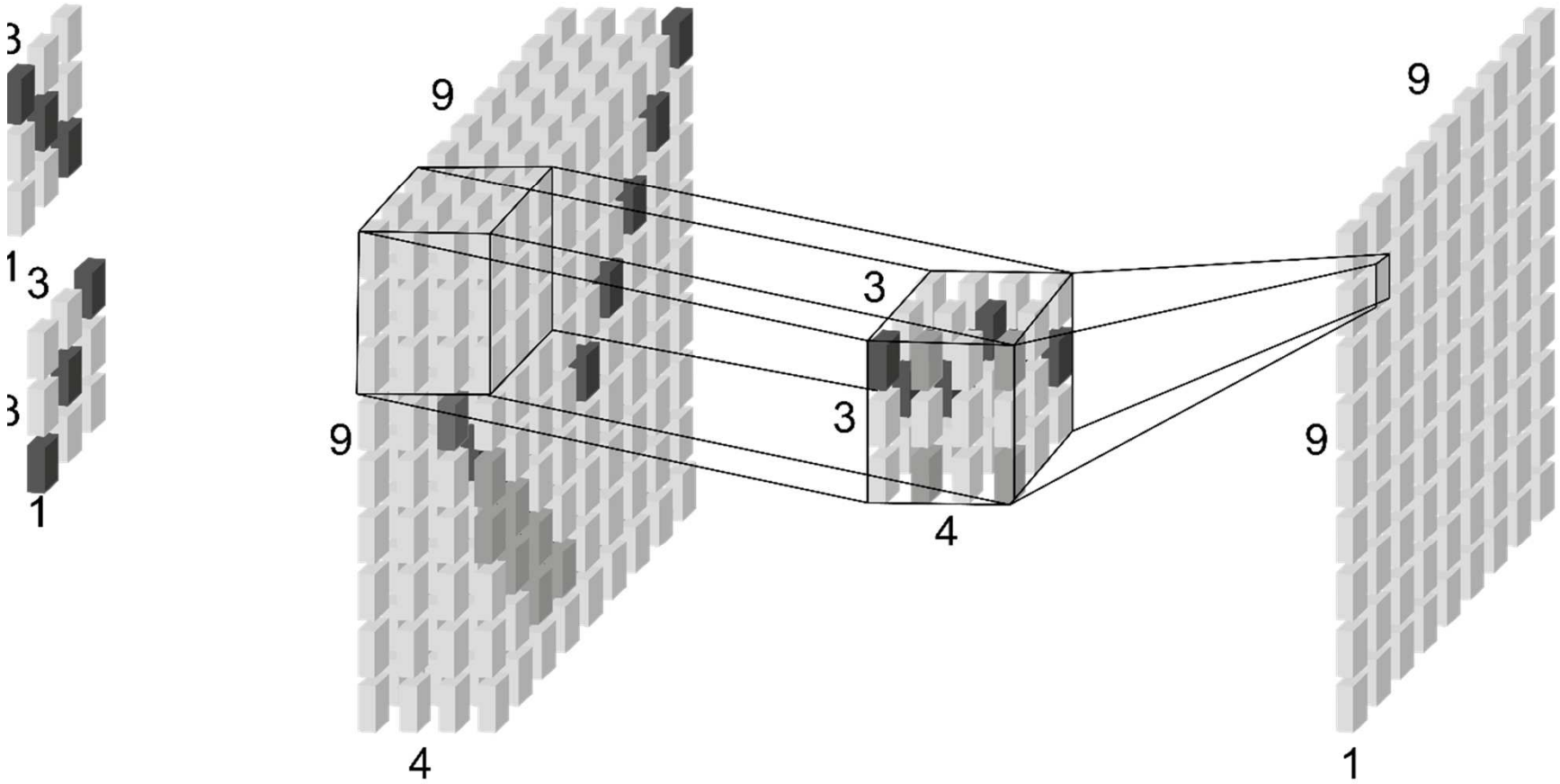


Convolution Layer



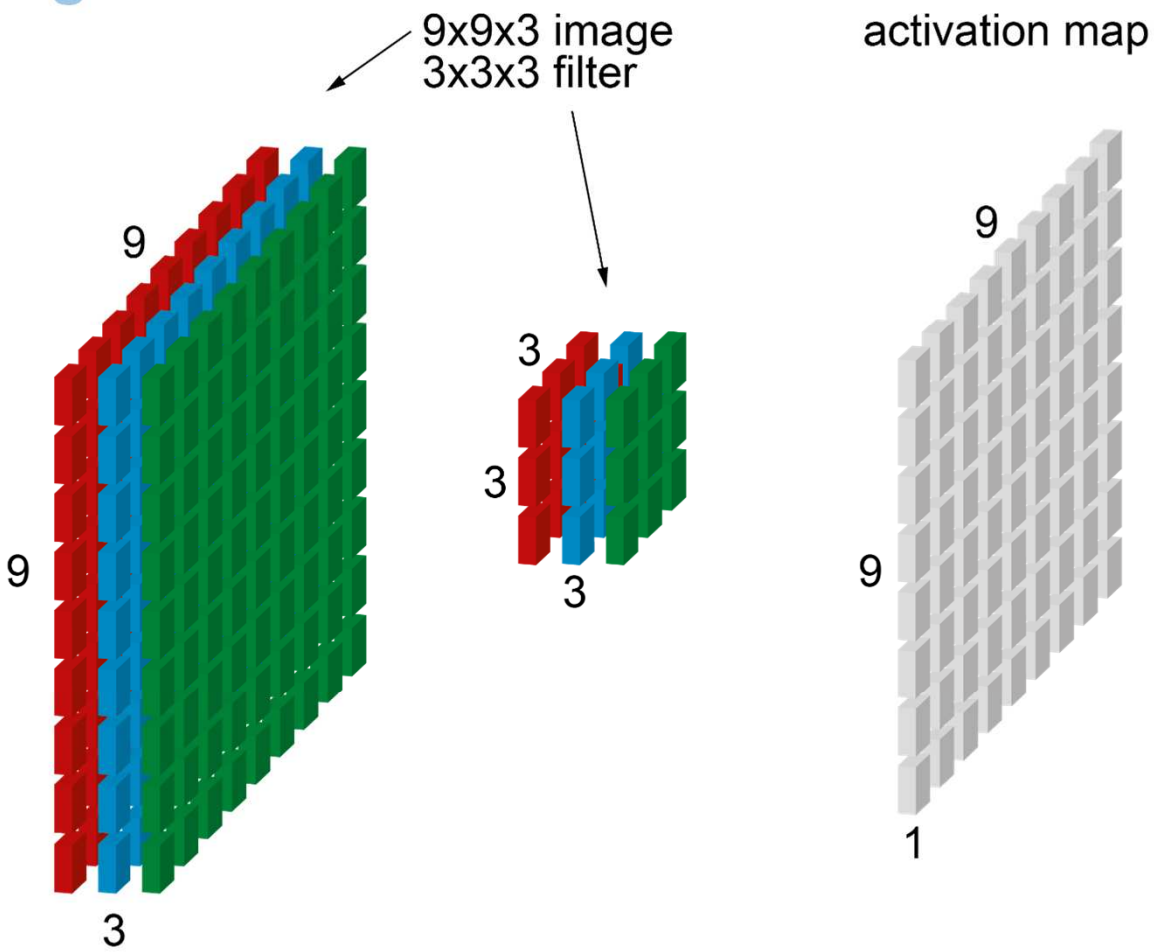
Convolution Layer

$$L^2_{i,j,k} = f(\sum_{x=-1}^{1} \sum_{y=-1}^{1} \sum_{z=1}^{4} L^1_{x+i,y+j,z} \cdot w^k_{i,j,z} + b^k_{i,j})$$

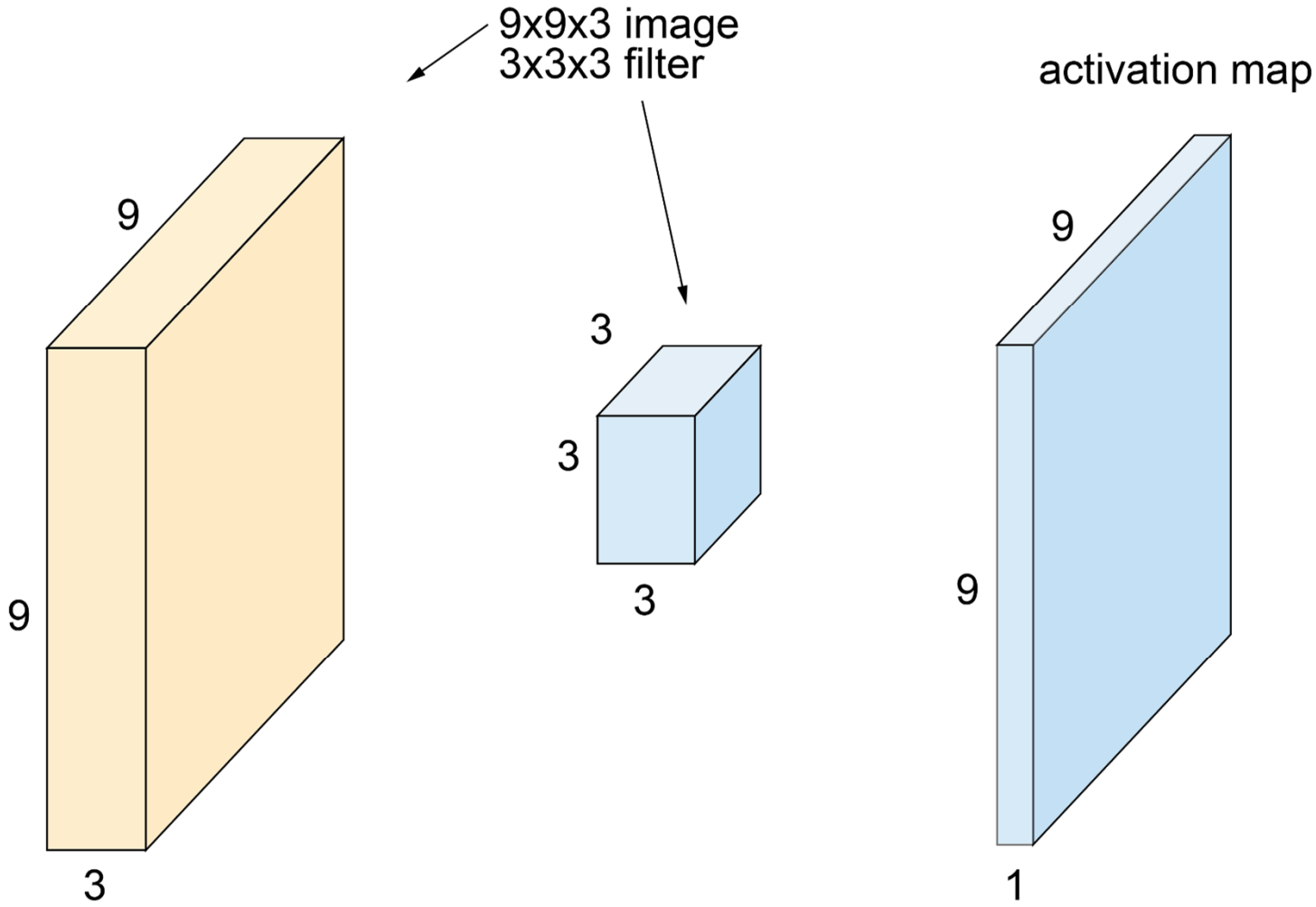


Convolution Layer

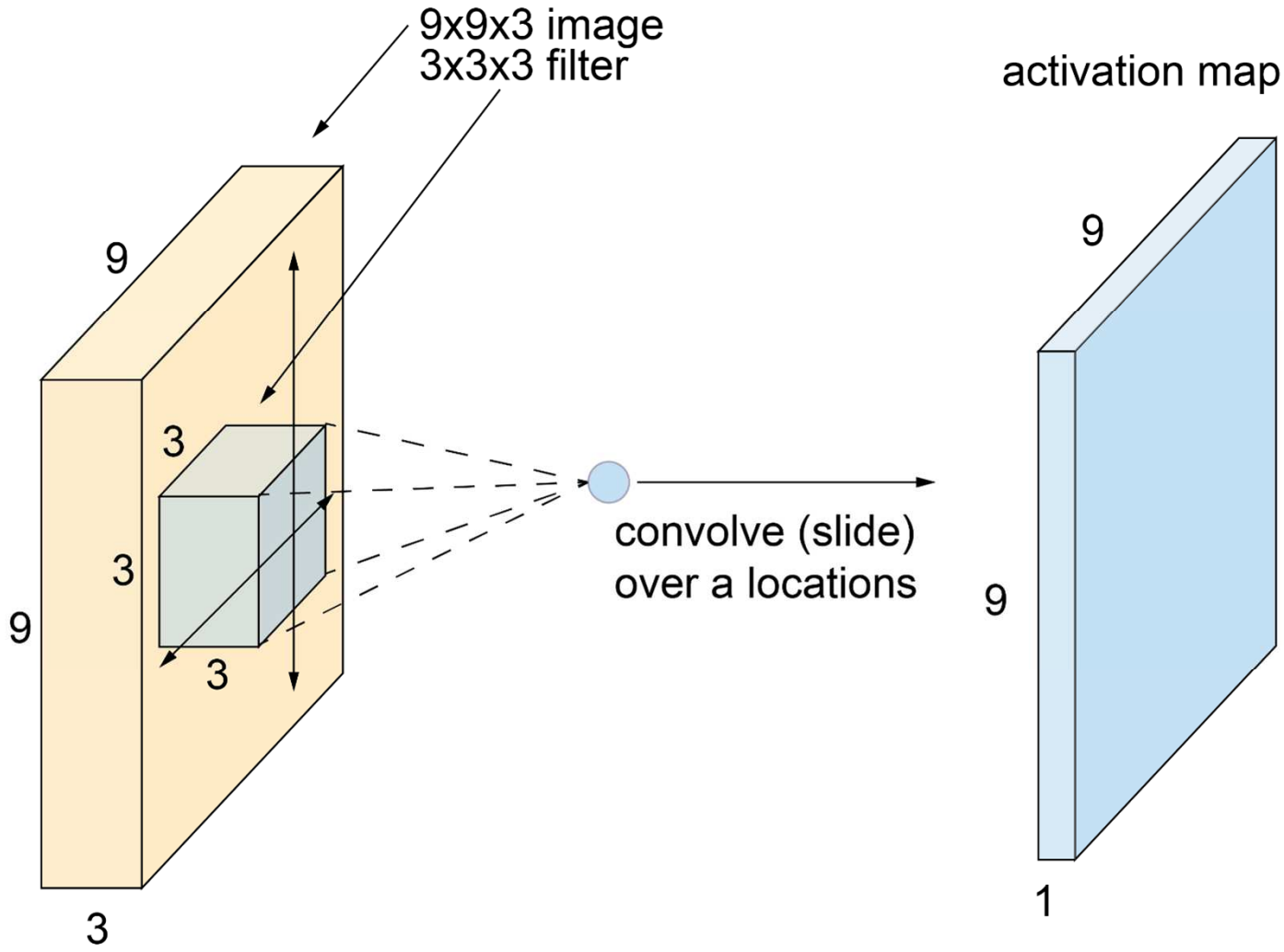
RGB Image



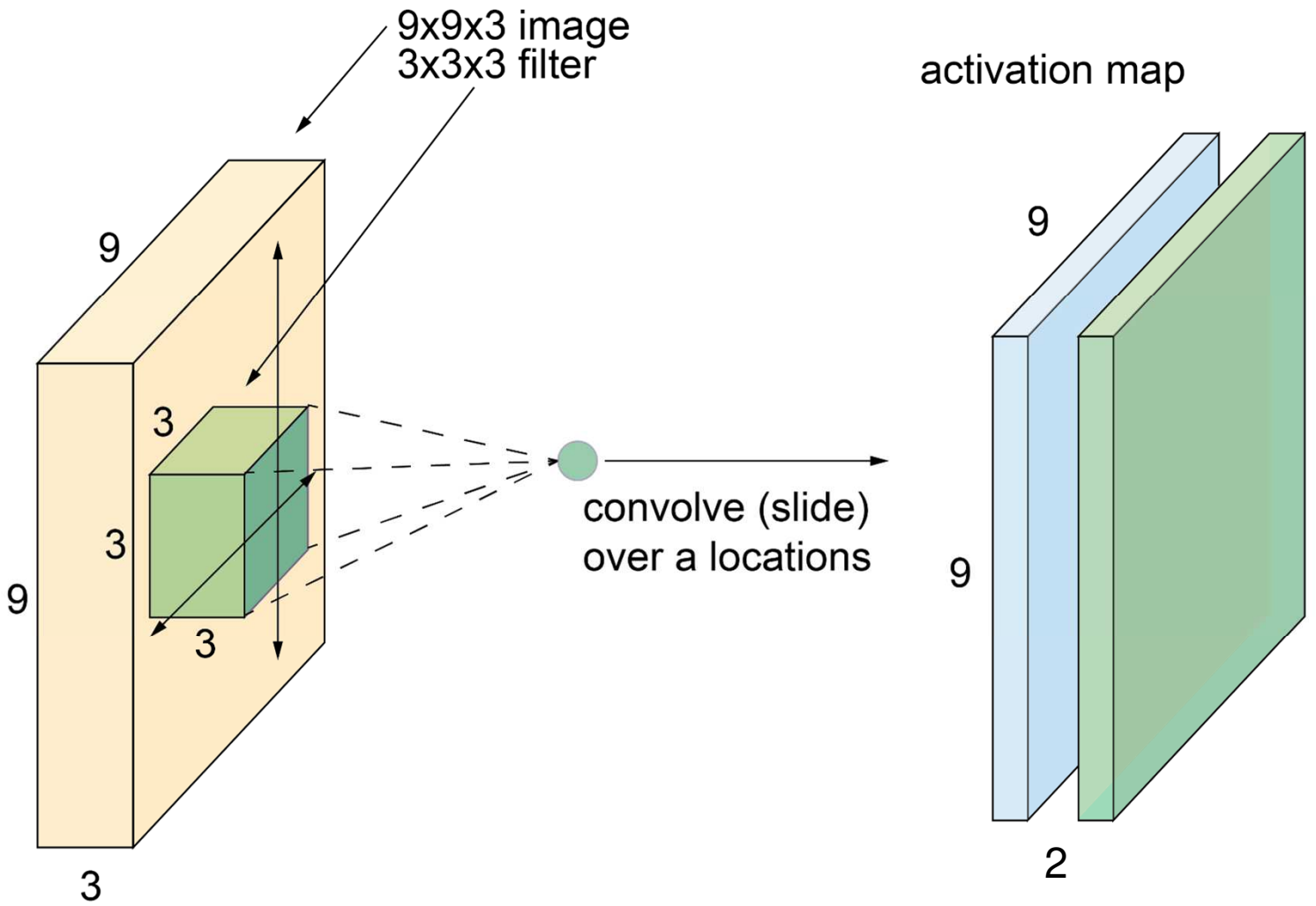
Convolution Layer



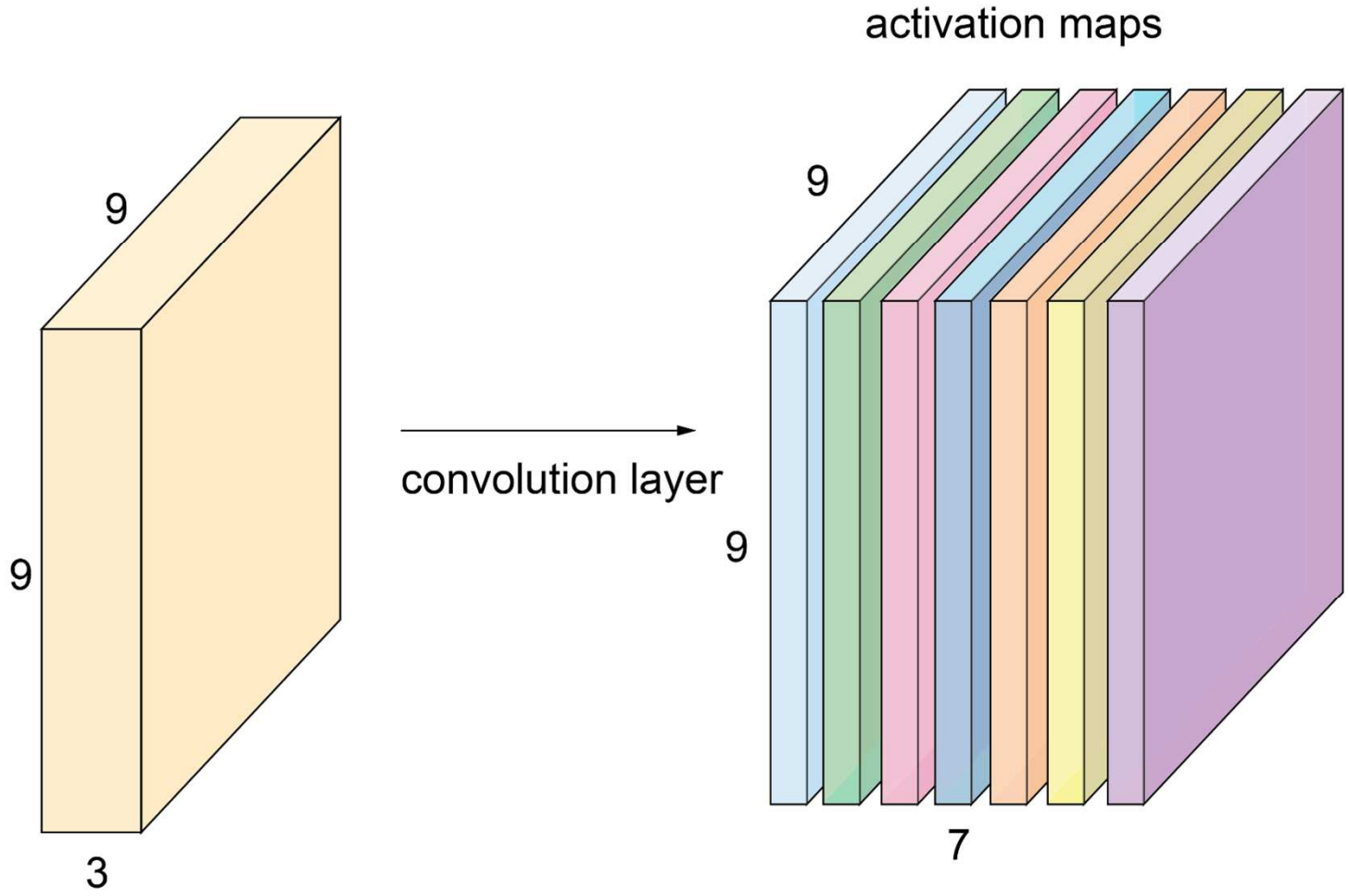
Convolution Layer



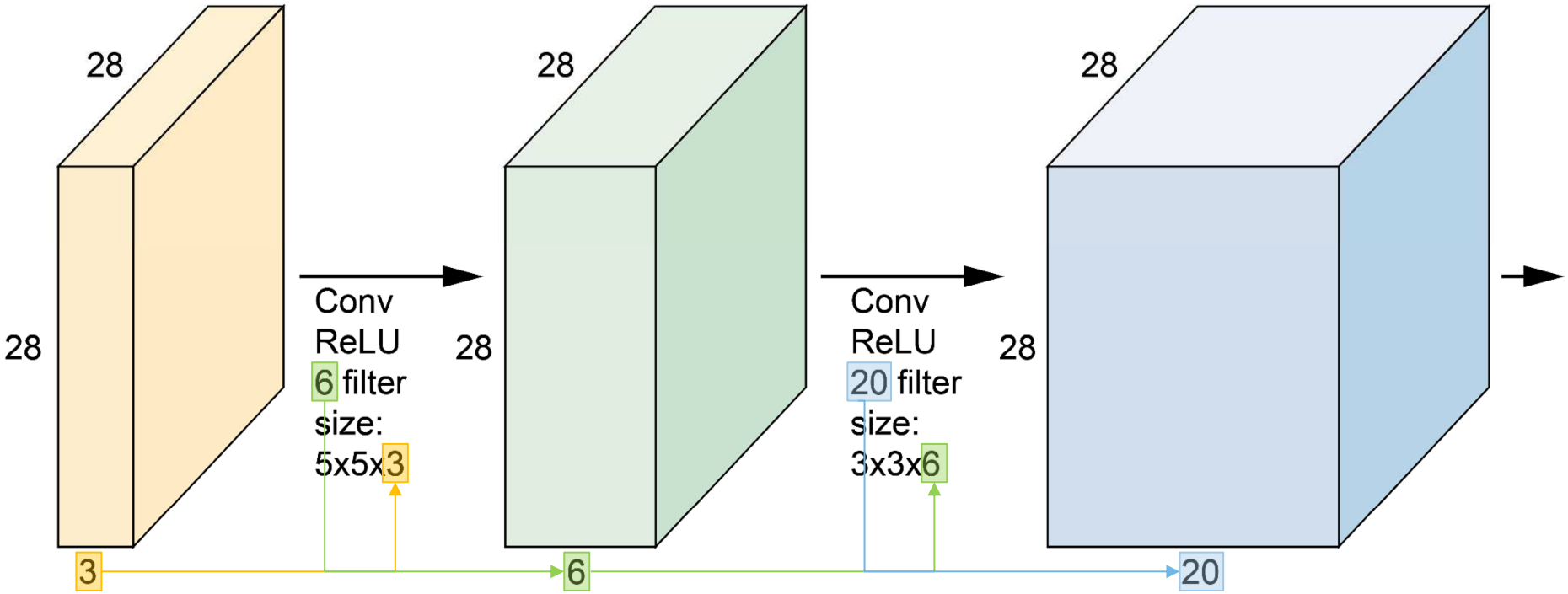
Convolution Layer



Convolution Layer

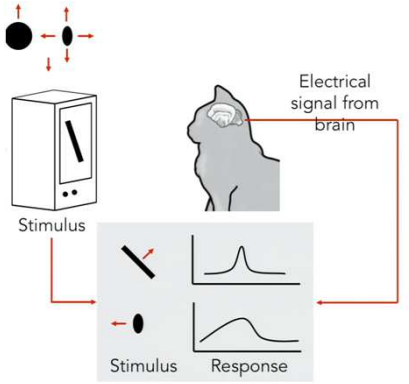


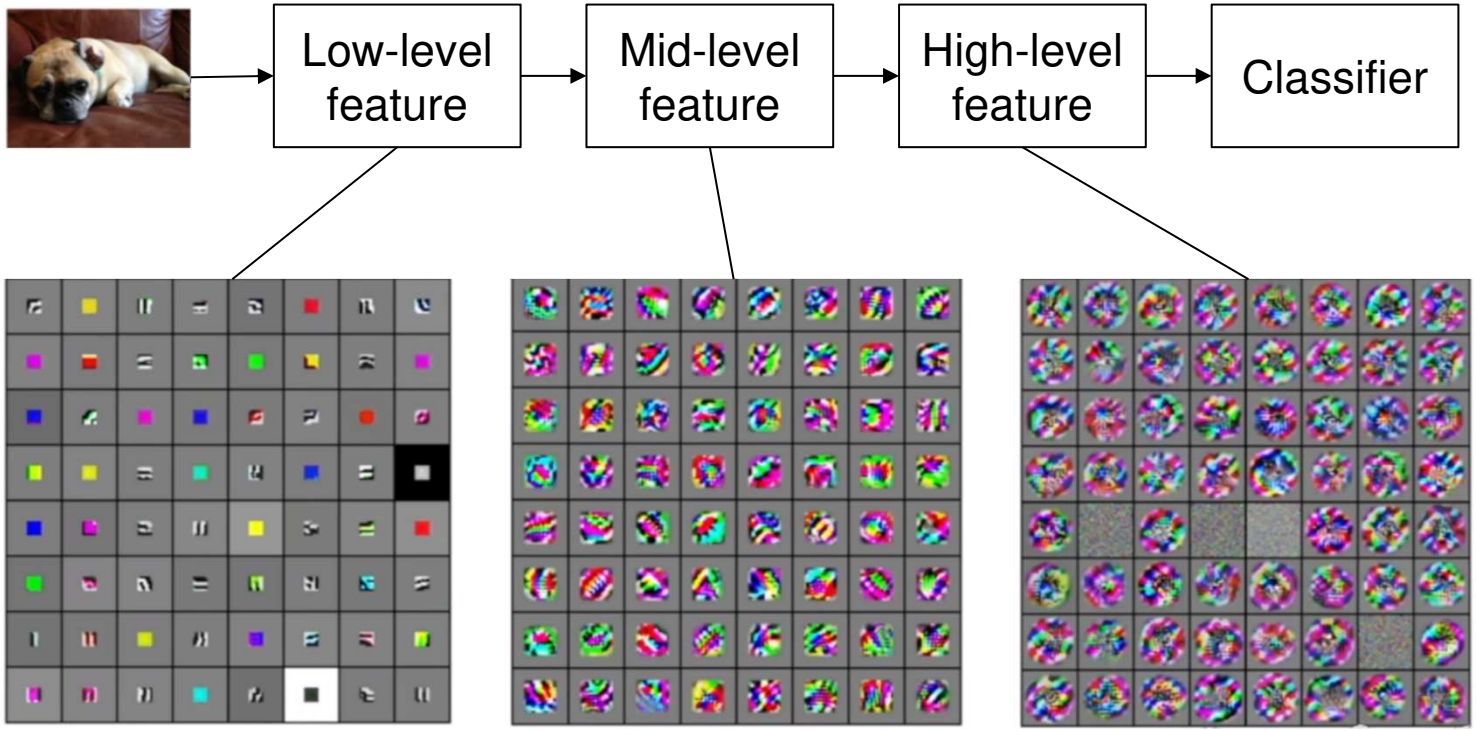
Convolution Layer



Convolution Layer

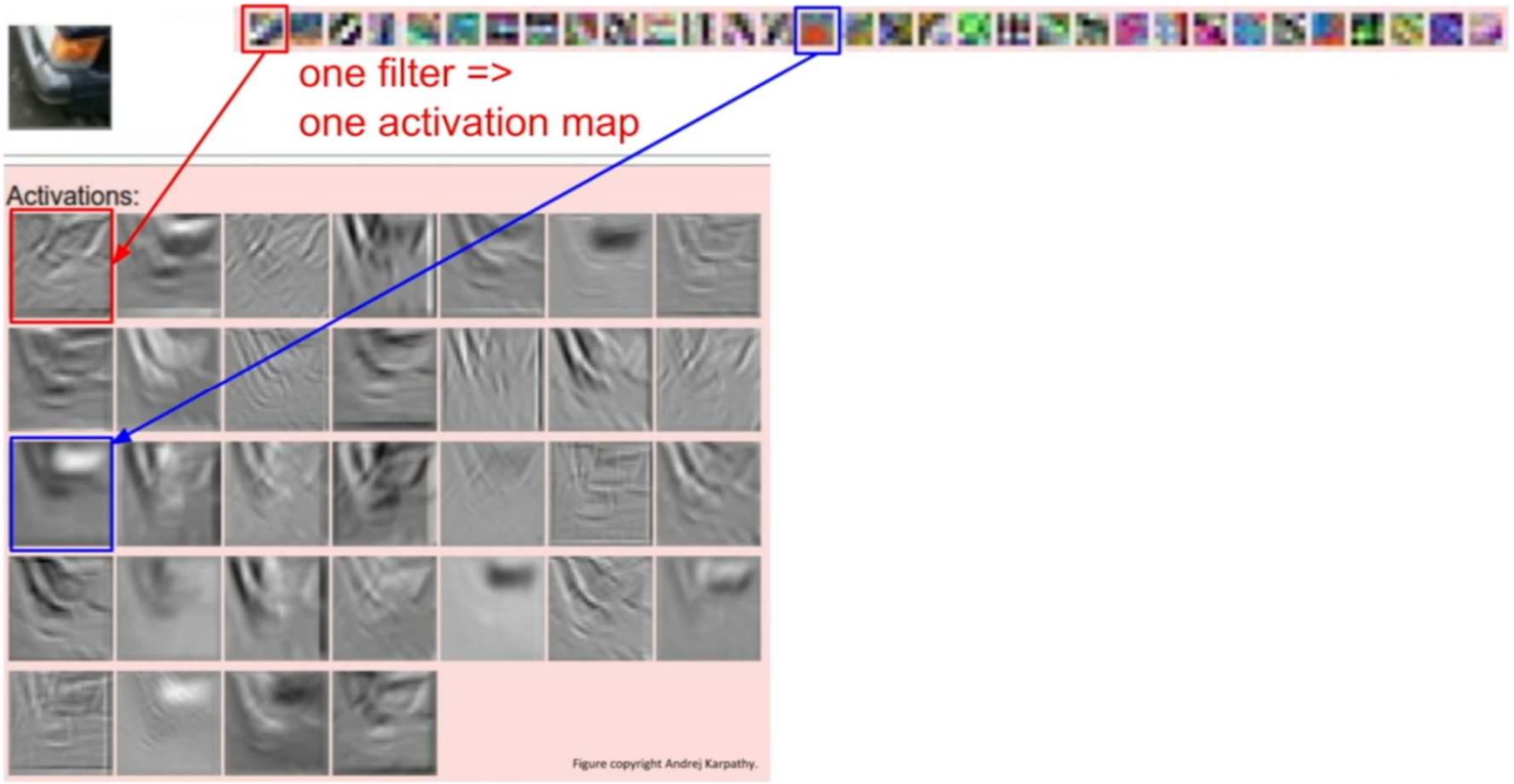
Filter Visualisation





Convolution Layer

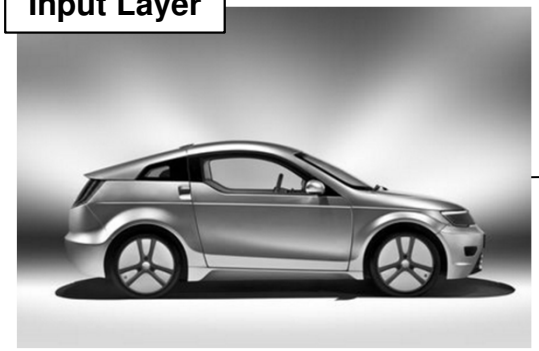
Activation Map Visualisation



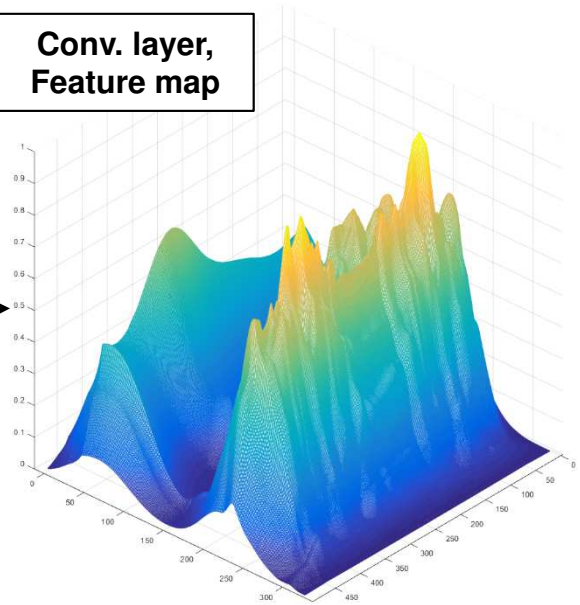
Convolution Layer

Visualisation

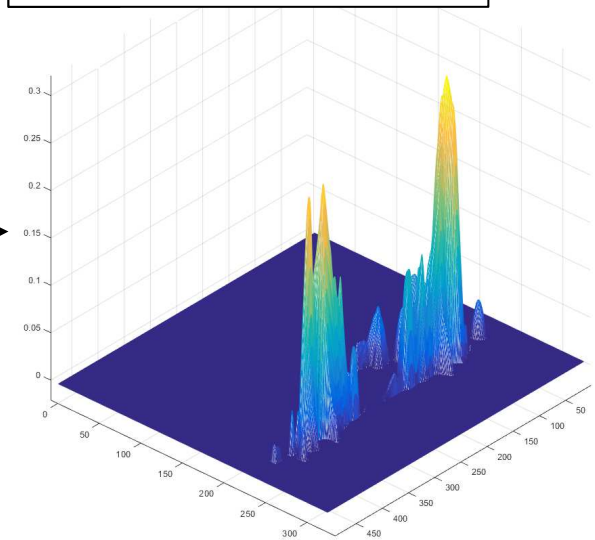
Input Layer



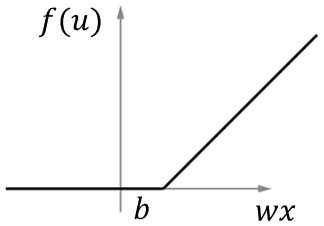
Conv. layer, Feature map



Relu Aktivierungfuntkion b=-0.7



Feature 1

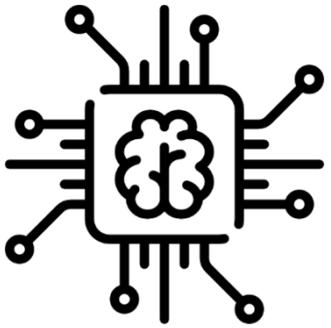


Deep Neural Networks

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

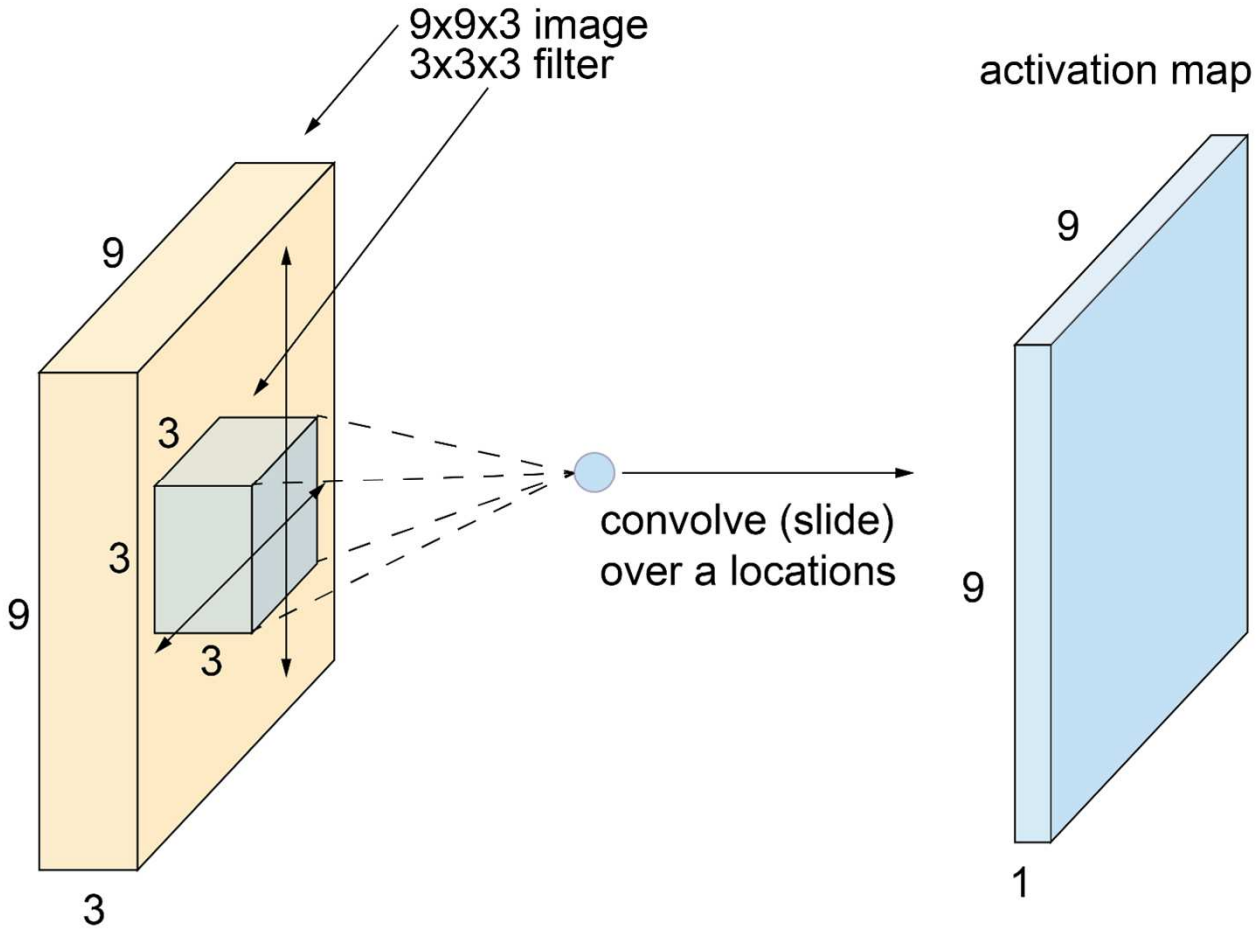
Agenda

1. Chapter: Convolutional Neural Networks
    - 1.1 Motivation
    - 1.2 Introduction
    - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
    - 2.1 Optimizer
    - 2.2 Data Formatting



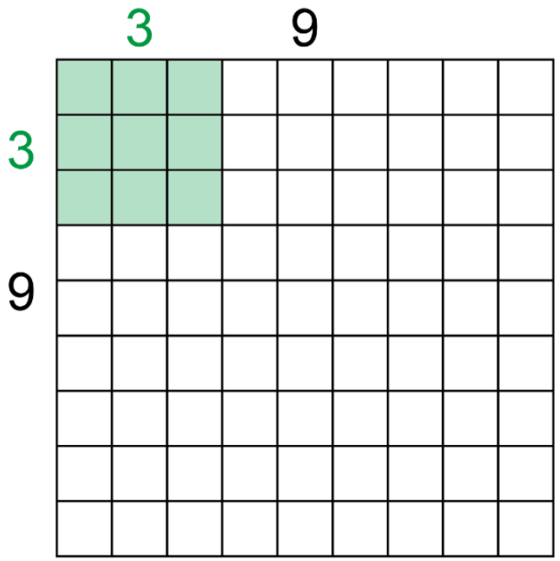
Convolution Layer

Dimensions



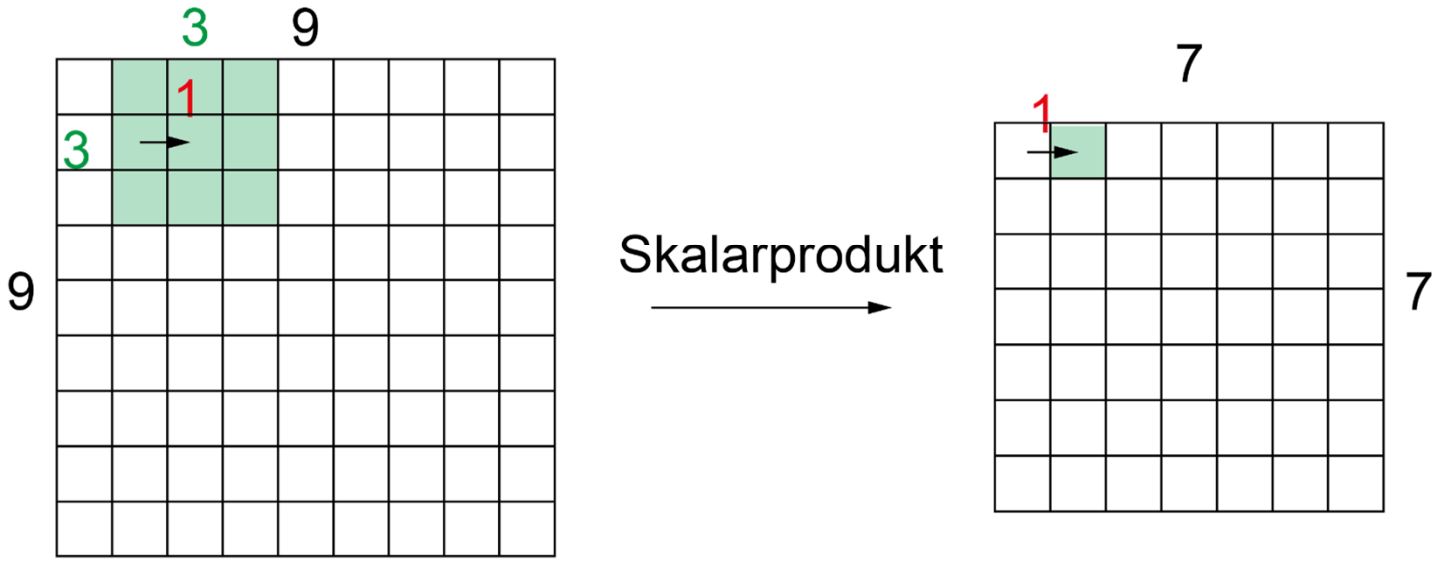
Convolution Layer

Dimensions



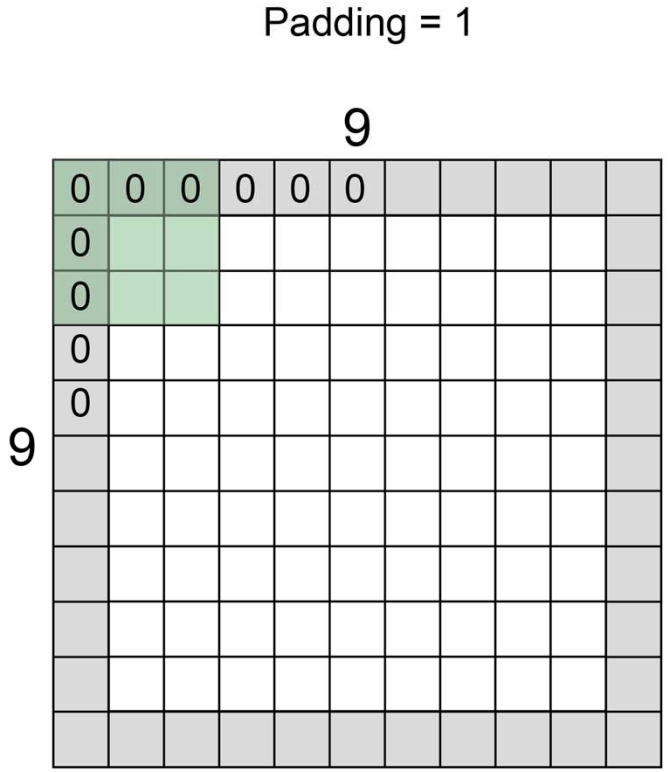
Convolution Layer

Dimensions



Convolution Layer

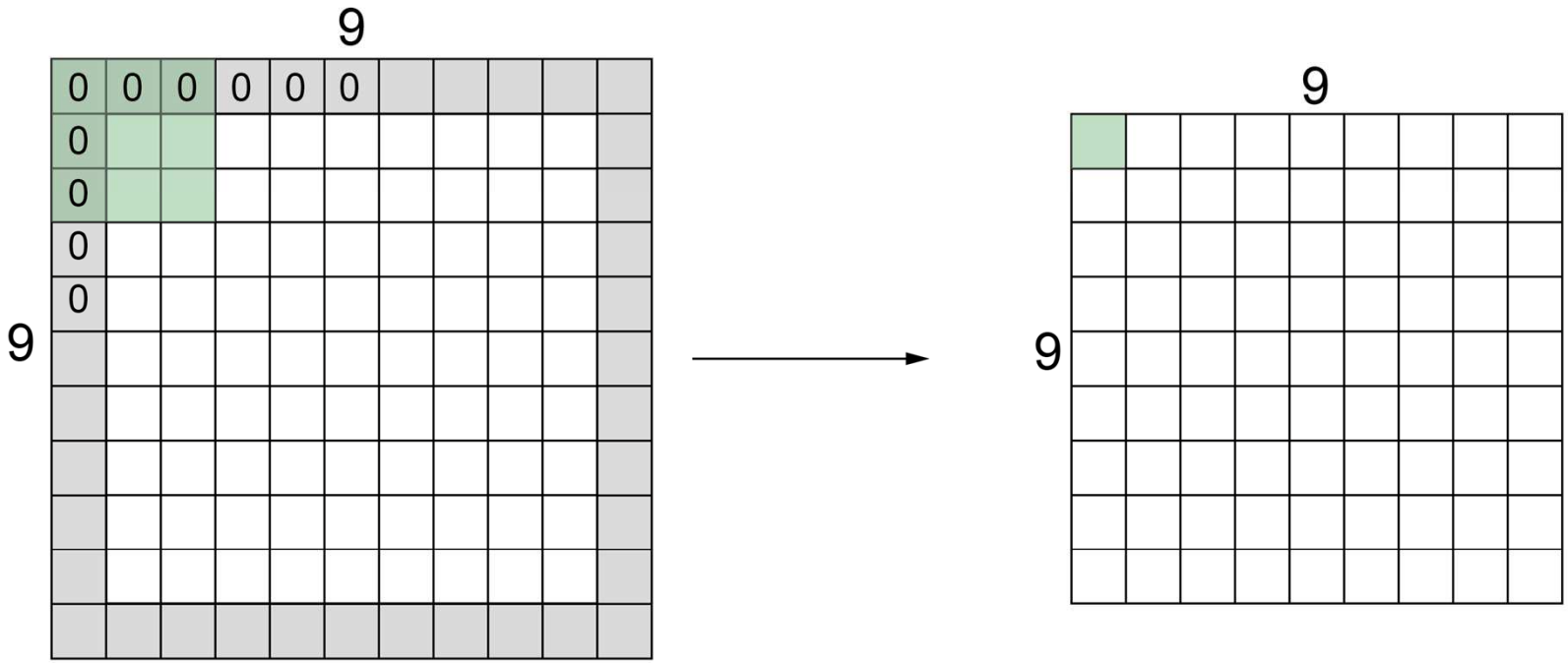
Dimensions: Padding



input 9x9
3x3 filter, applied with stride 1
what is the output?

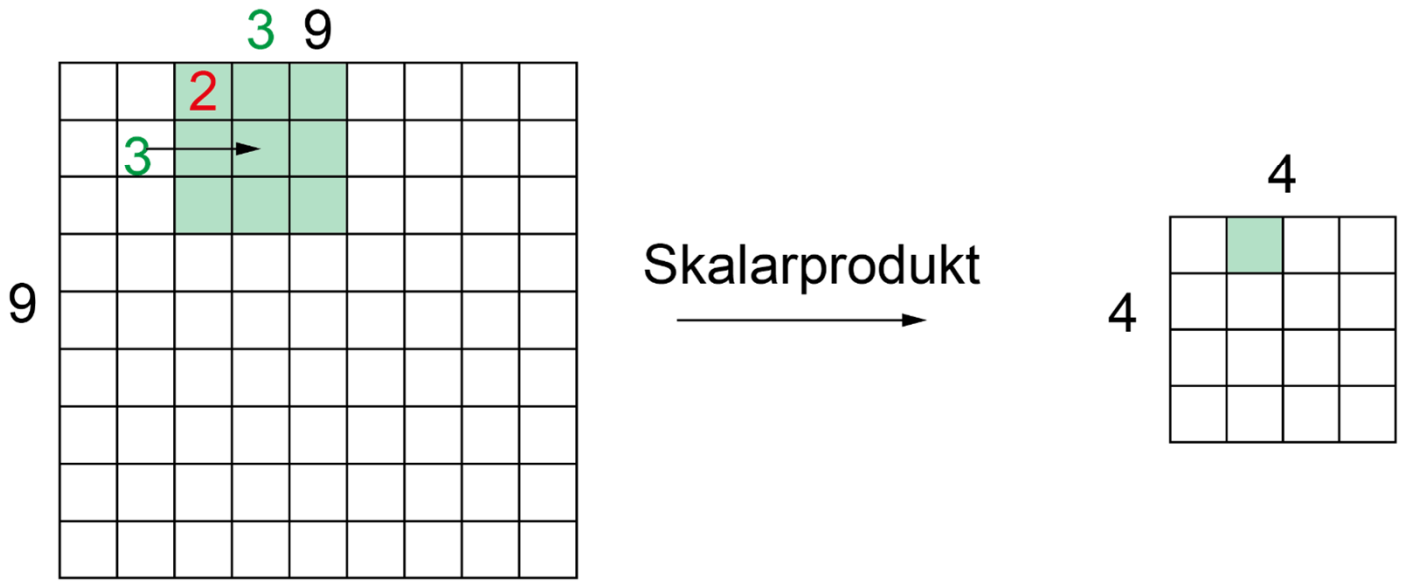
Convolution Layer

Dimensions: Padding



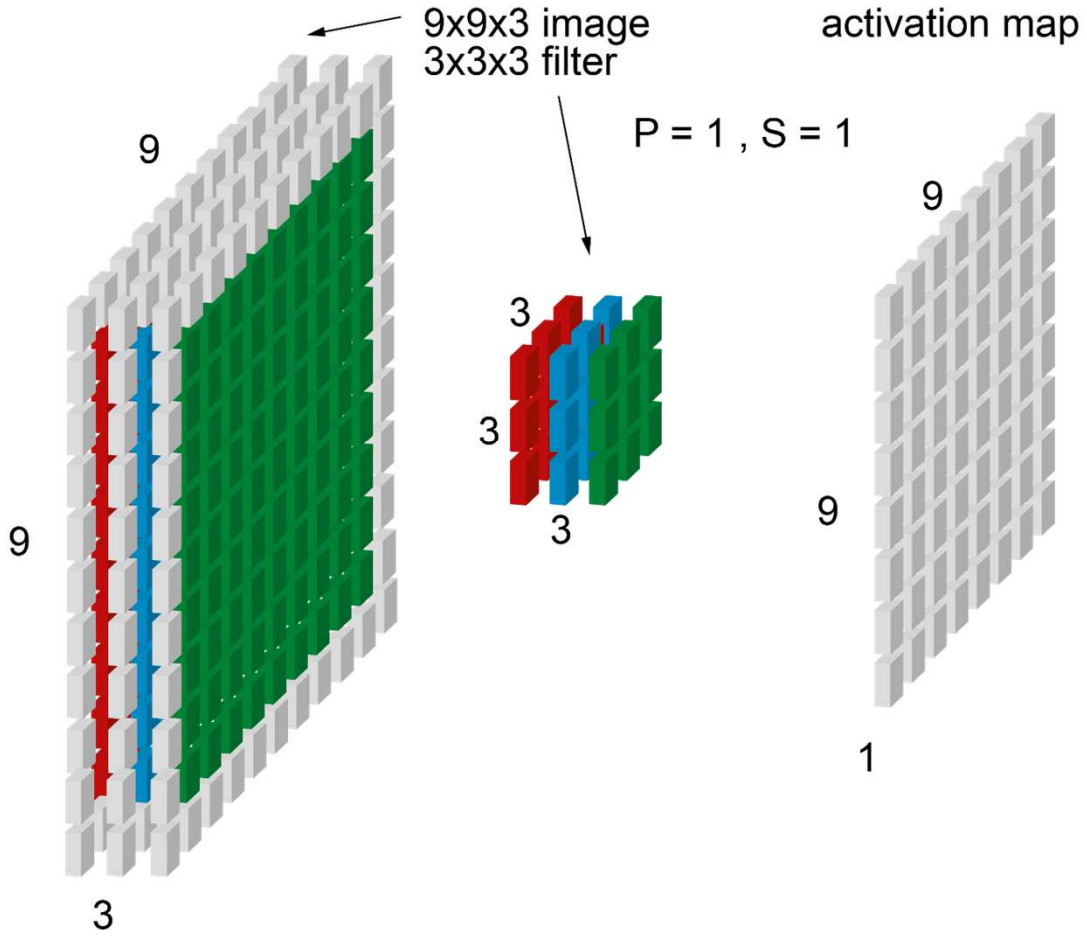
Convolution Layer

Dimensions: Stride



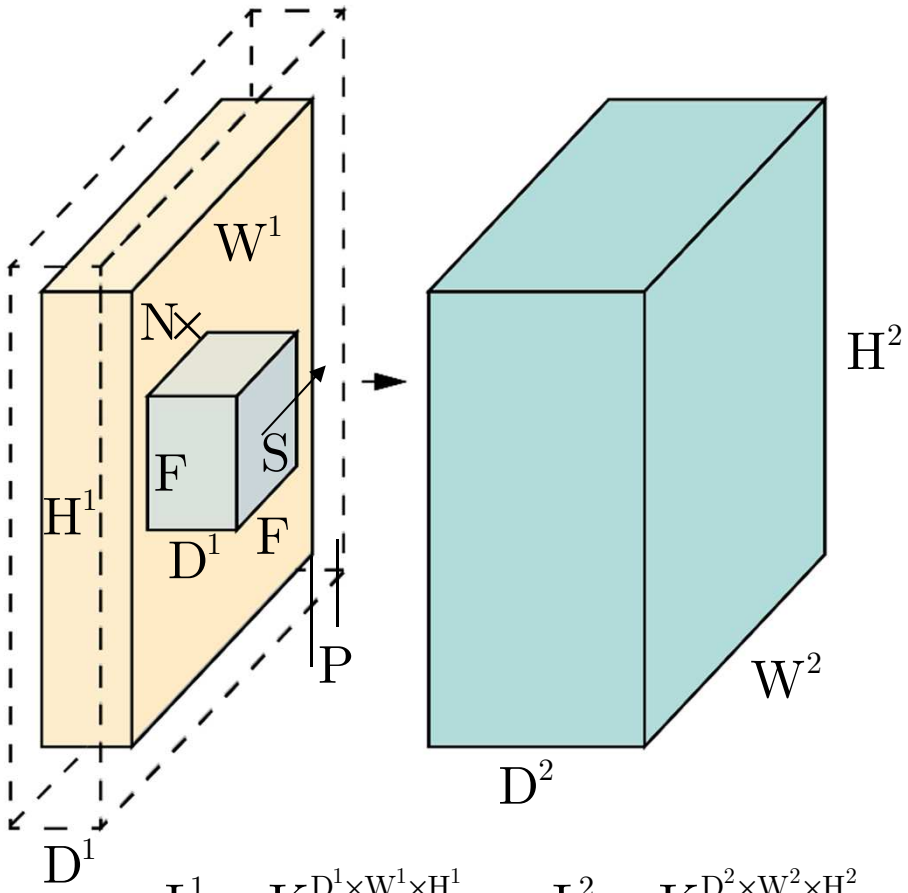
Convolution Layer

Padding with RGB Image



Convolution Layer

Dimensions Formula



$$L^1 \in K^{D^1 \times W^1 \times H^1} \rightarrow L^2 \in K^{D^2 \times W^2 \times H^2}$$

N : Number of Filters

F : Filtersize

S : Stride

P : Padding

New Dimensions:

$$W^2 = \frac{W^1 + 2P - F}{S} + 1$$

$$H^2 = \frac{H^1 + 2P - F}{S} + 1$$

$$D^2 = N$$

Keep Dimensions:

$$F = 3 \Rightarrow P = 1$$

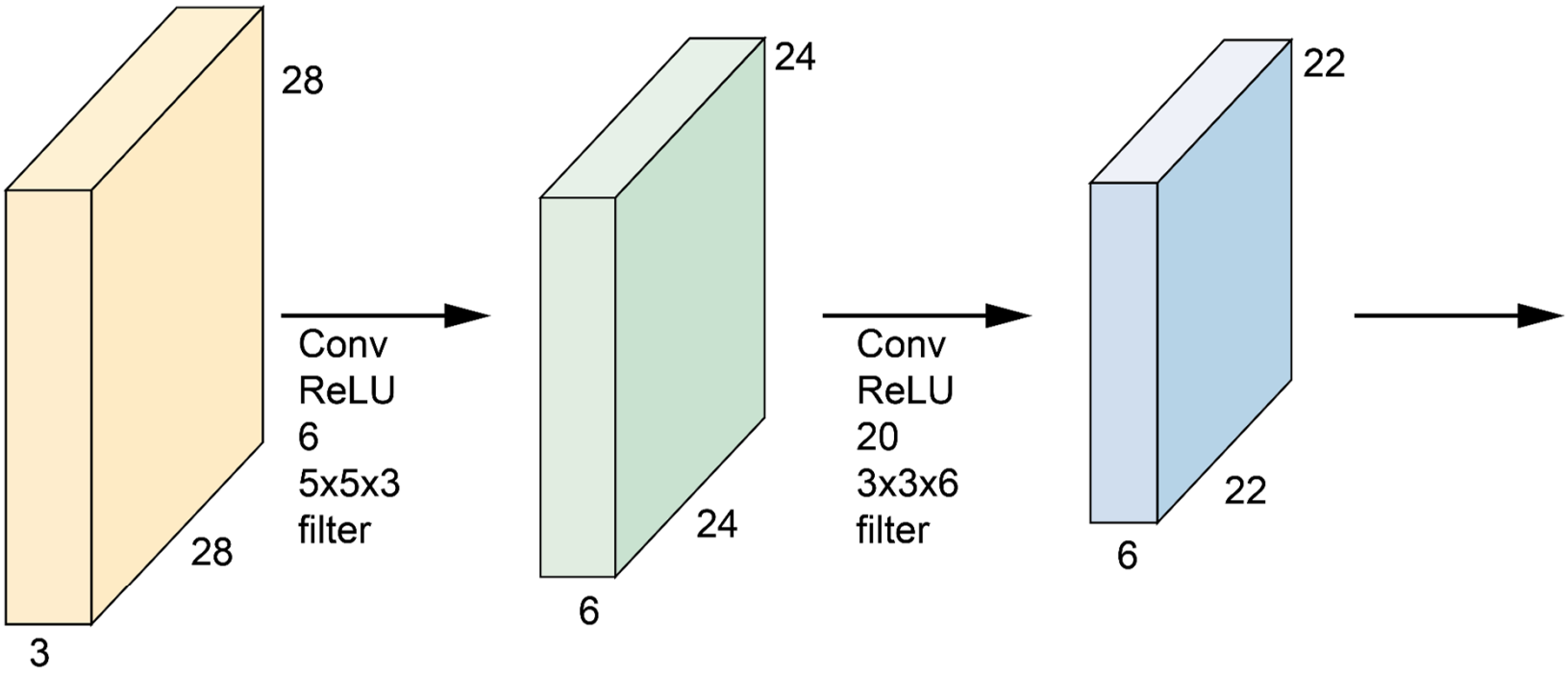
$$F = 5 \Rightarrow P = 2$$

$$F = 7 \Rightarrow P = 3$$

Convolution Layer

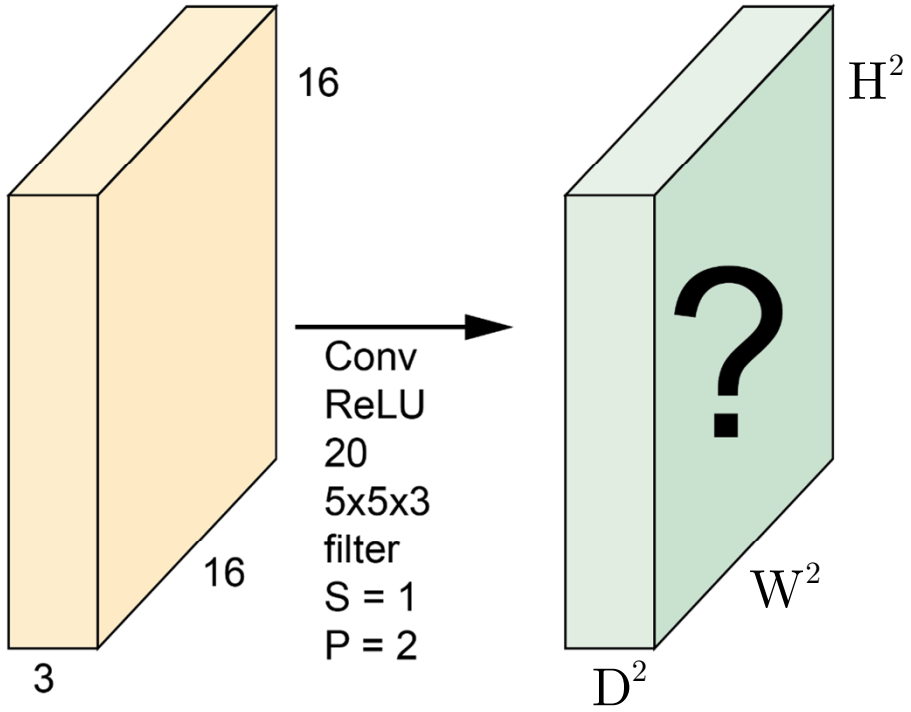
Padding

Without padding, volume shrinks with every convolution layer. This can cause problems in very deep neuronal networks. Use padding!



Convolution Layer

Example



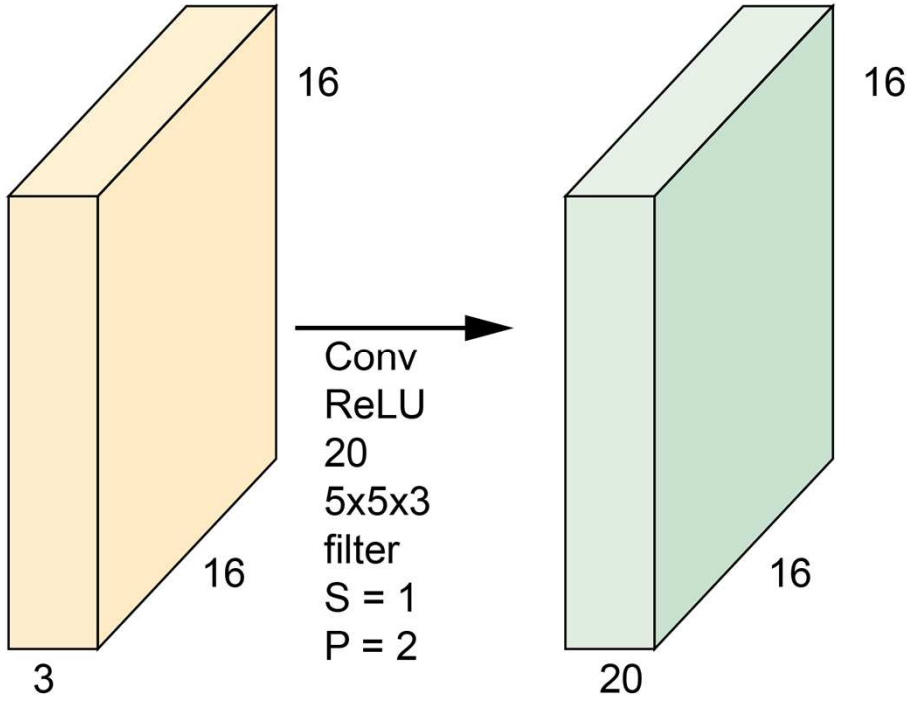
$$W^2 = \frac{W^1 + 2P - F}{S} + 1$$

$$H^2 = \frac{H^1 + 2P - F}{S} + 1$$

$$D^2 = N$$

Convolution Layer

Example



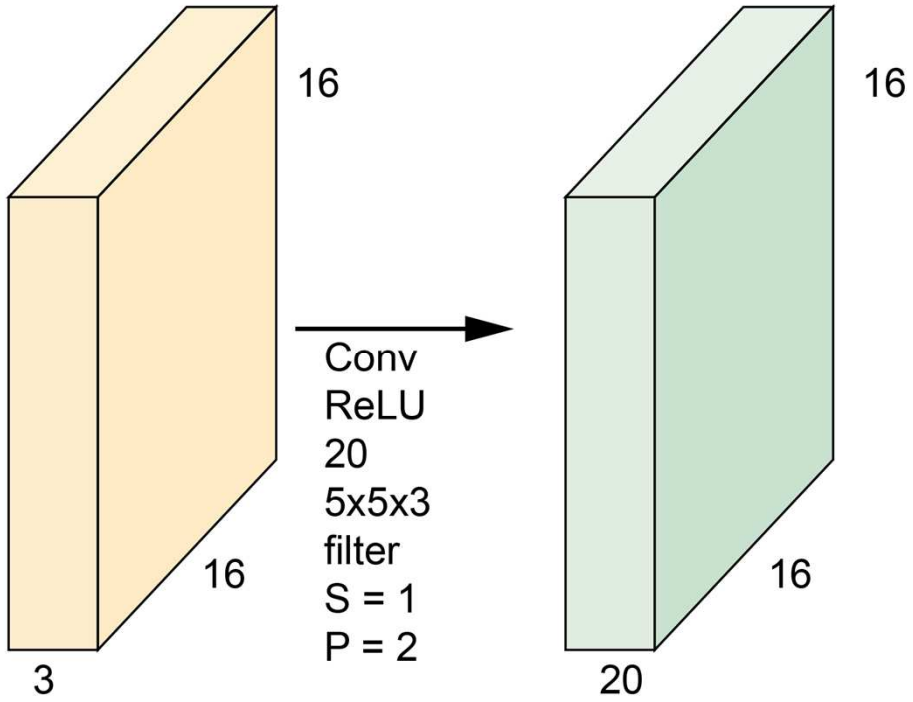
$$W^2 = \frac{16 + 2 \cdot 2 - 5}{1} + 1 = 16$$

$$H^2 = \frac{16 + 2 \cdot 2 - 5}{1} + 1 = 16$$

$$D^2 = 20$$

Convolution Layer

Example



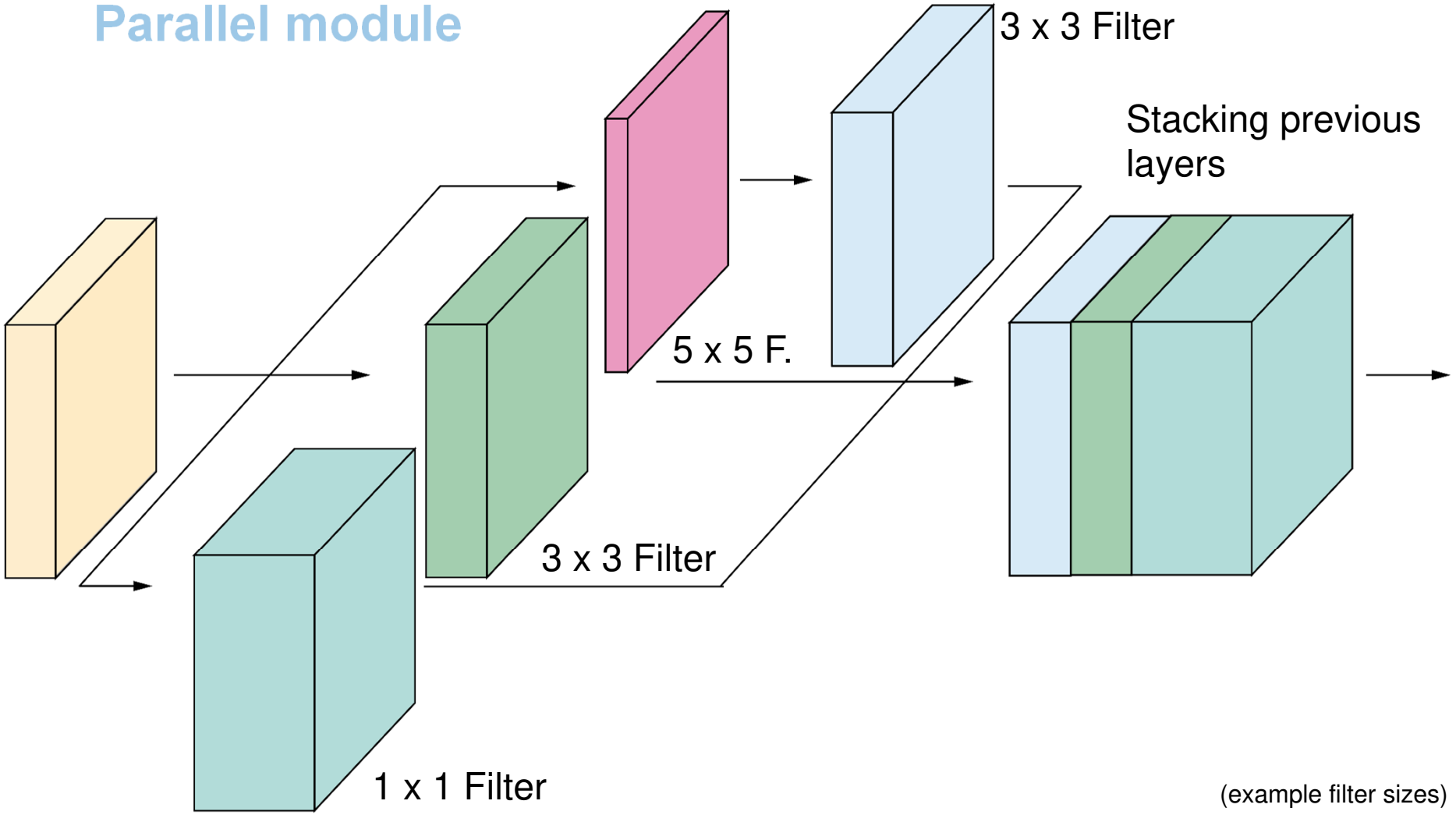
Number of parameters in this layer?

$5 \times 5 \times 3 + 1 = 76$ Paramter per filter

$20 \cdot 76 = 1520$ Paramter

Convolution Layer

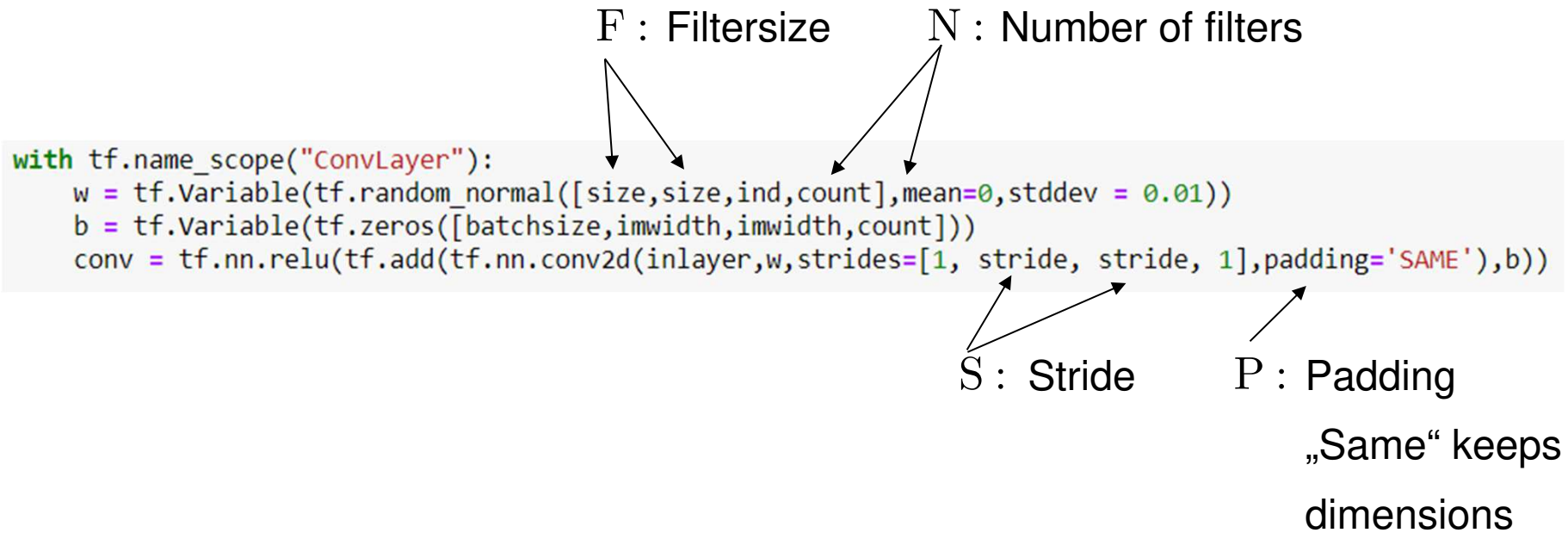
Parallel module



(example filter sizes)

Convolution Layer

Code Example

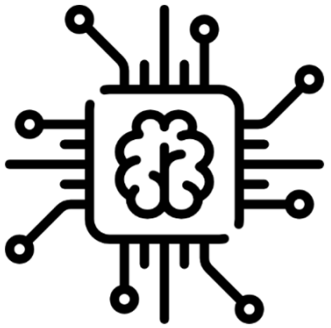


Deep Neural Networks

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

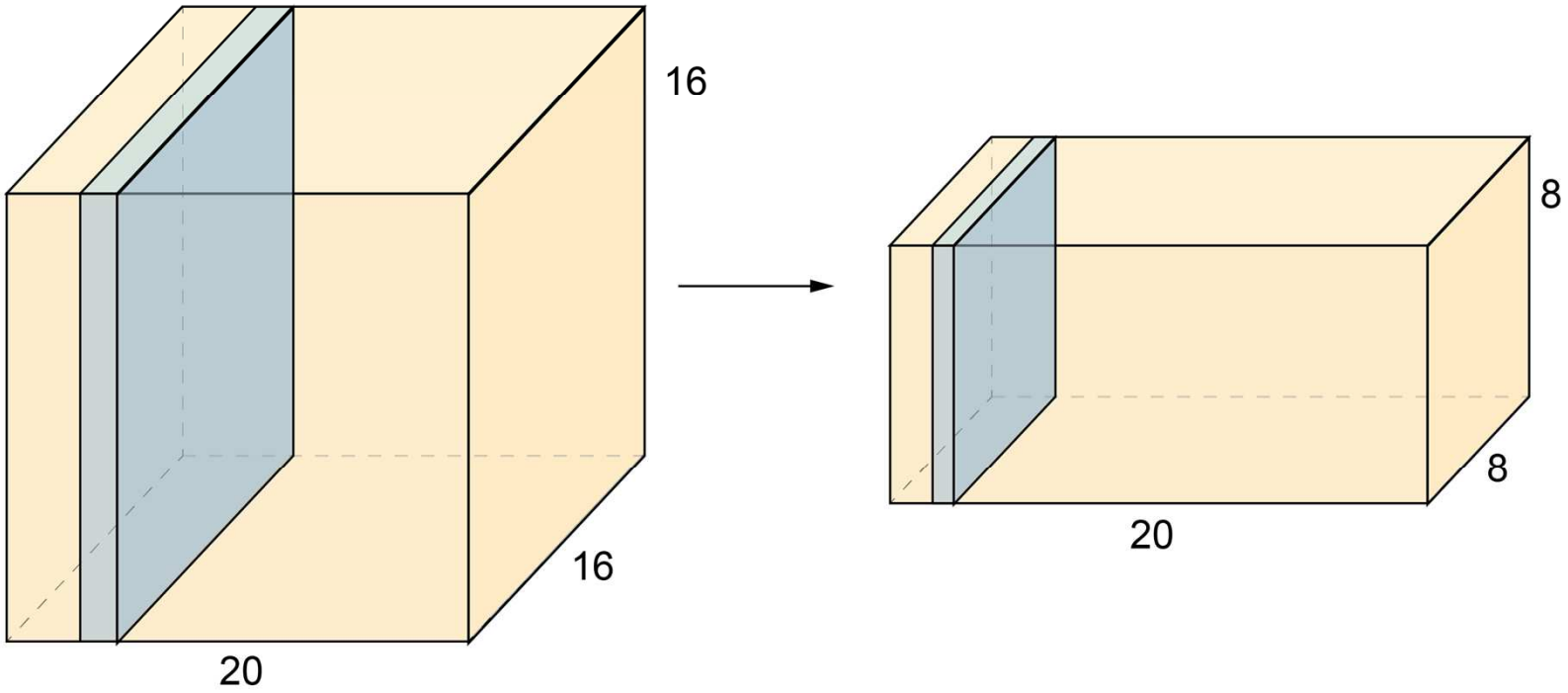
Agenda

1. Chapter: Convolutional Neural Networks
    - 1.1 Motivation
    - 1.2 Introduction
    - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
    - 2.1 Optimizer
    - 2.2 Data Formatting



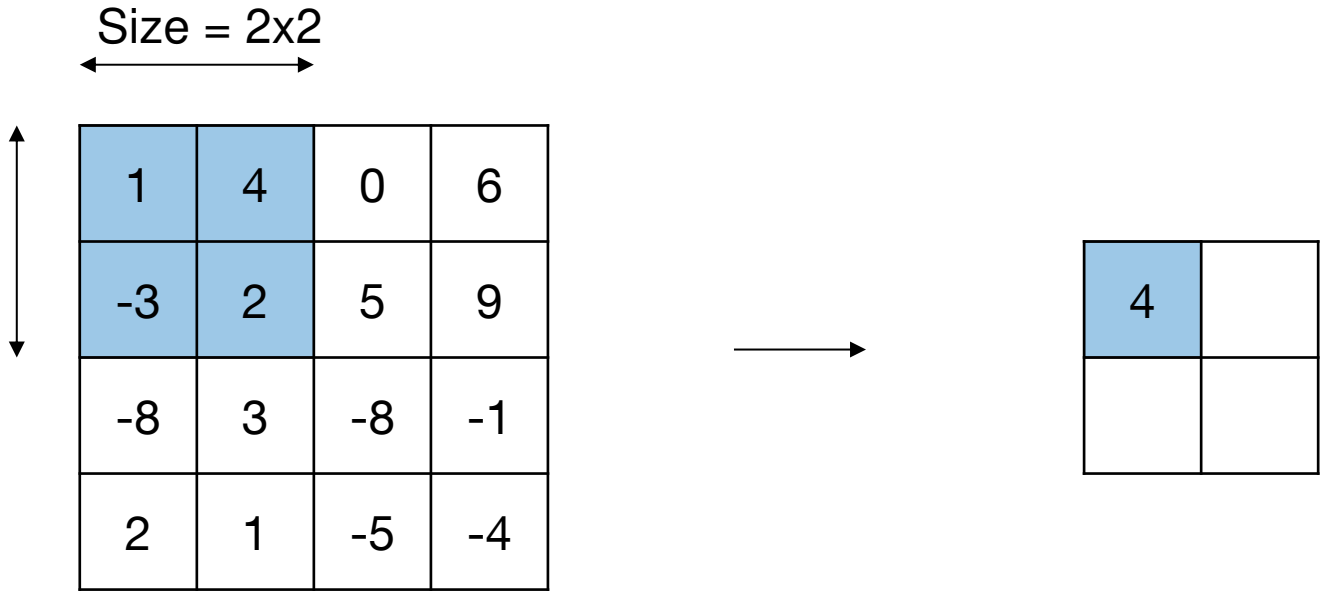
Pooling

- reduces the spacial output dimensions
- operates over each activation map idependently



Pooling

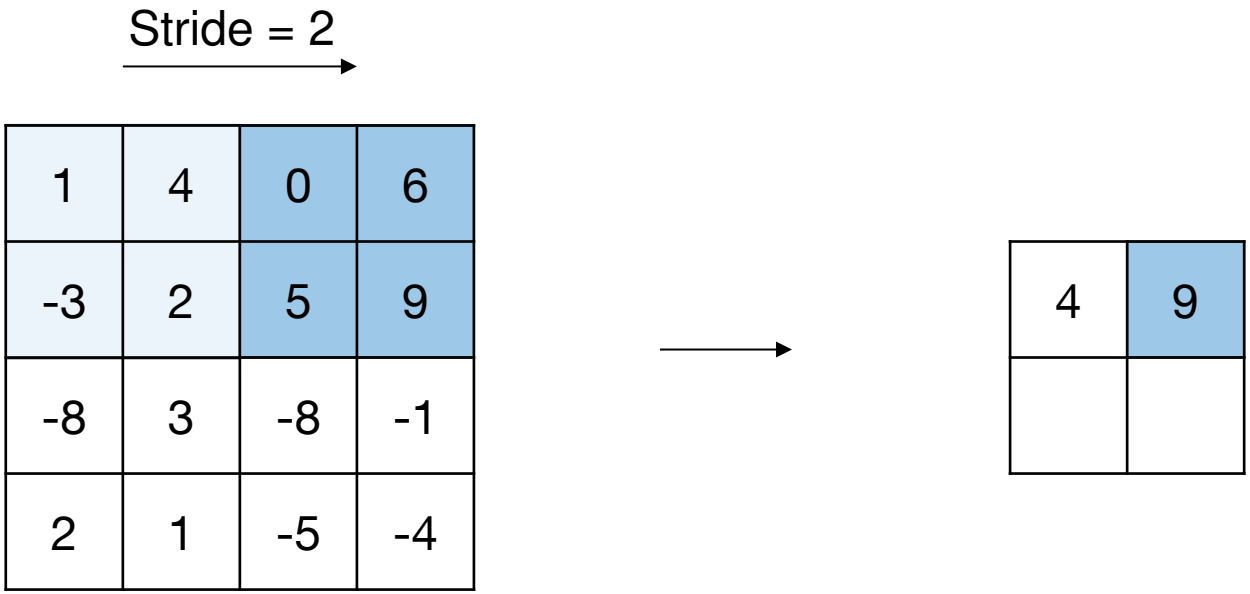
Max Pool



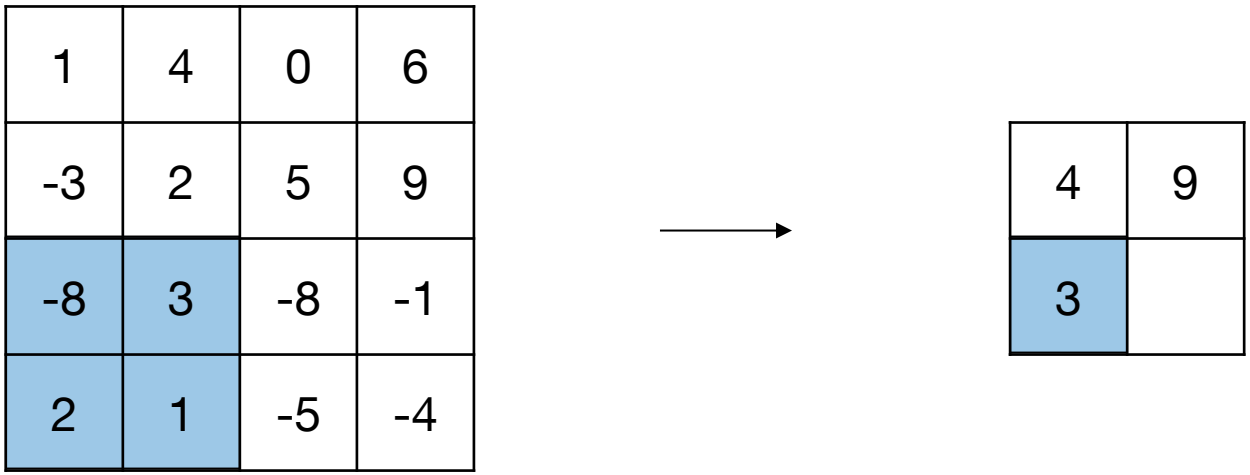
$$L^2_{i,j} = \max(L^1_{i,j}, L^1_{i-1,j}, L^1_{i,j-1}, L^1_{i-1,j-1})$$

Pooling

Max Pool

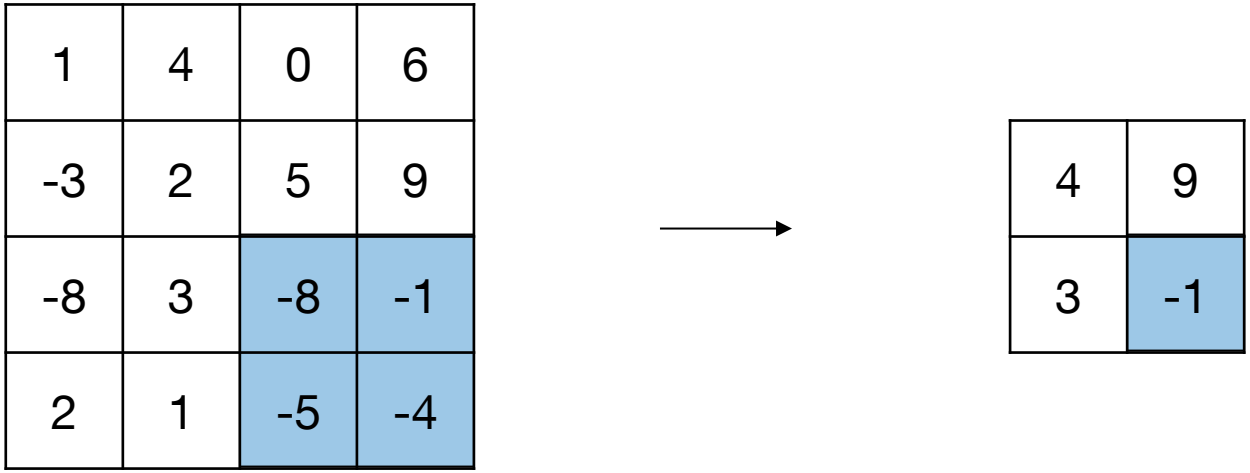


Pooling
Max Pool



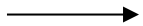
Pooling

Max Pool



Pooling
Max Pool

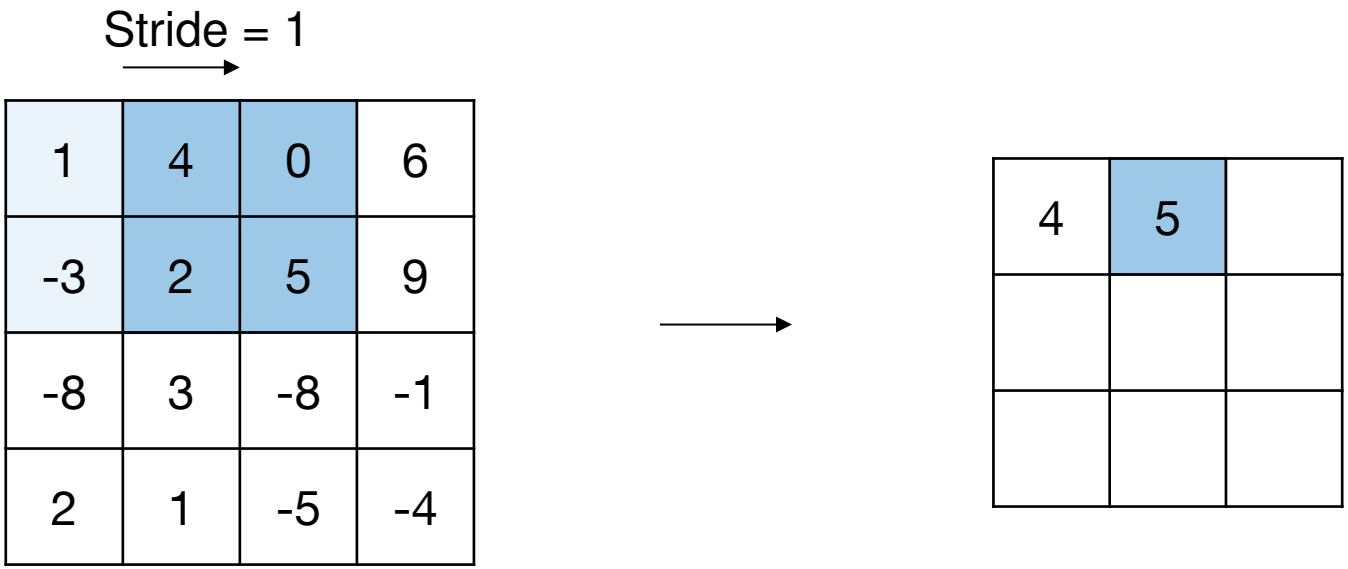
1	4	0	6
-3	2	5	9
-8	3	-8	-1
2	1	-5	-4



4		

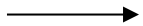
Pooling

Max Pool



Pooling
Max Pool

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

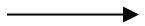


4	5	9
2	5	9
3	3	-1

Pooling

Average Pool

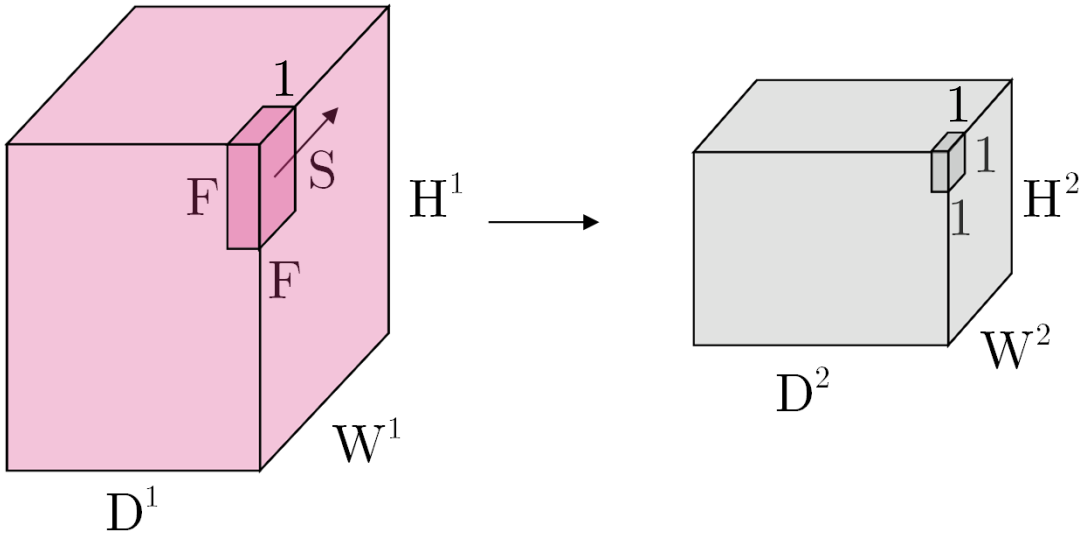
1	4	0	6
-3	2	5	9
-8	3	-8	-1
2	1	-5	-4



1	5
-0.5	-4.5

$$L^2_{i,j} = \frac{1}{4} \sum_{x=0}^{1} \sum_{y=0}^{1} L^1_{2i-x,2j-y}$$

Pooling
Dimensions



$$L^1 \in K^{D^1 \times W^1 \times H^1} \rightarrow L^2 \in K^{D^2 \times W^2 \times H^2}$$

New Dimensions:

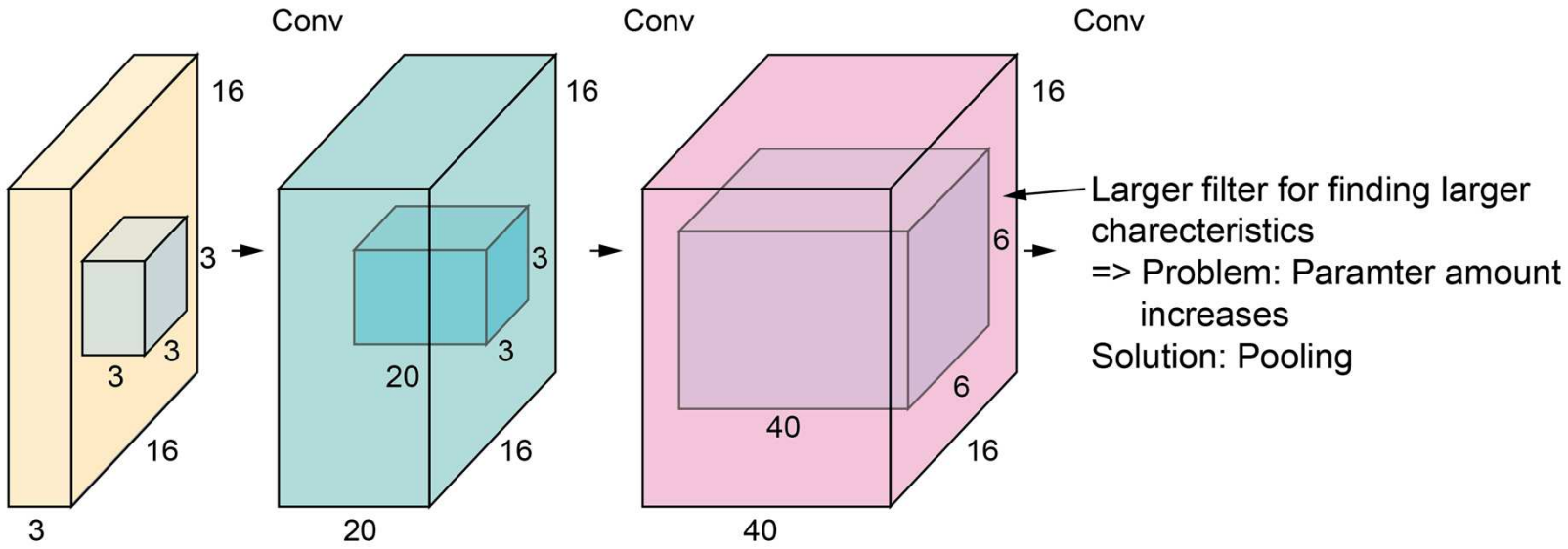
$$W^2 = \frac{W^1 - F}{S} + 1$$

$$H^2 = \frac{H^1 - F}{S} + 1$$

$$D^2 = D^1$$

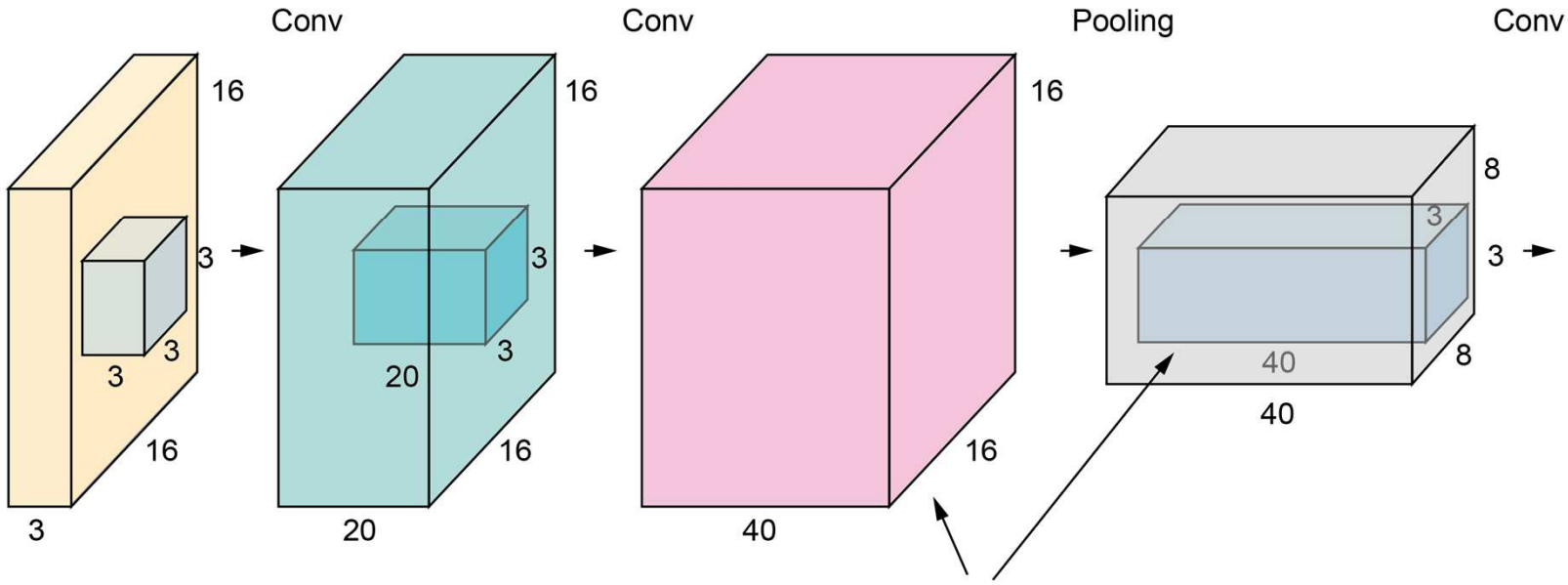
Pooling

With Convolution layer



Pooling

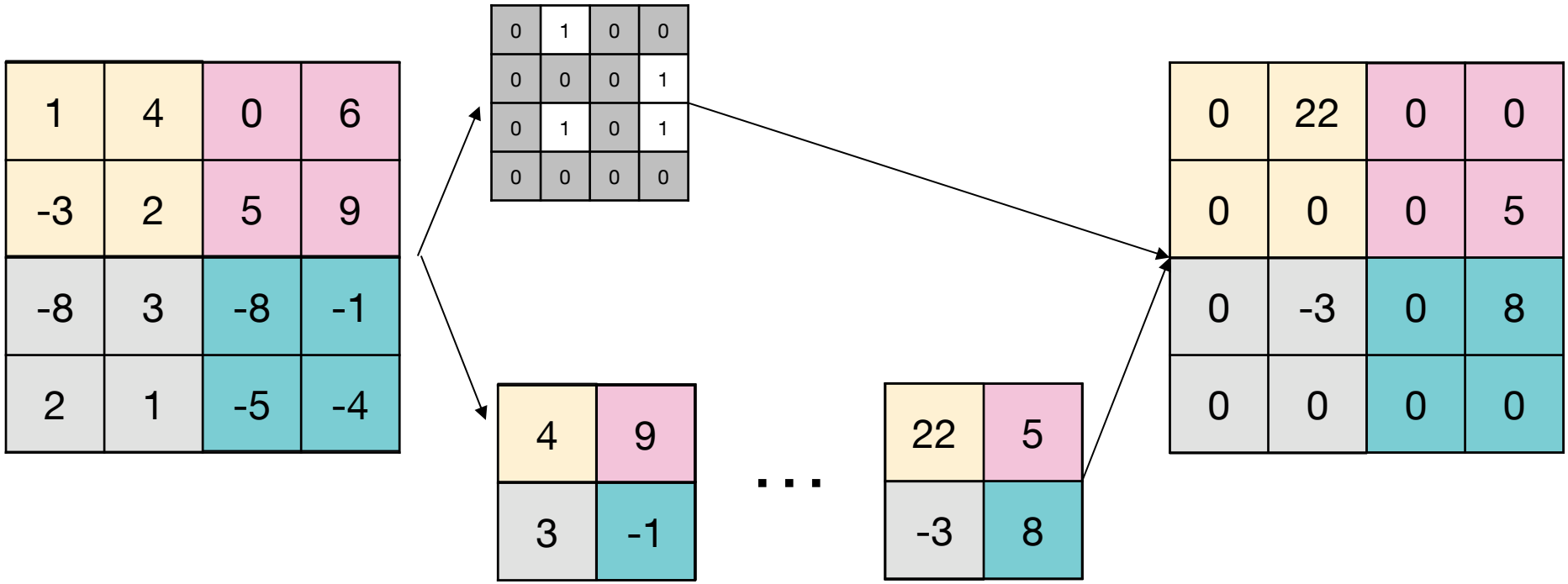
With Convolution layer



With the use of Pooling layers larger charisitics can be found with smaller filters. Reducing the parameter amount in this example to 25 %

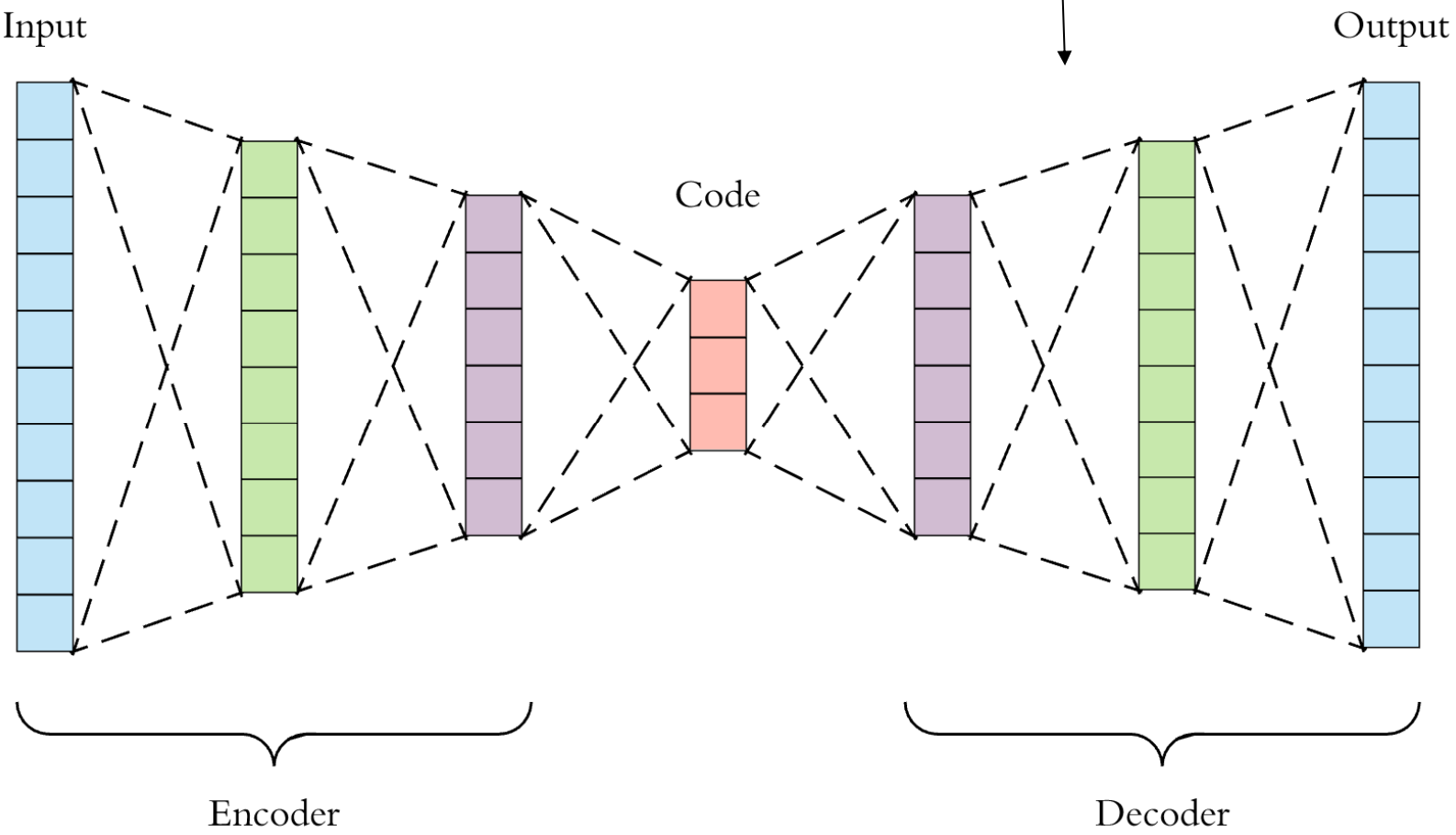
Unpooling

Max Unpool



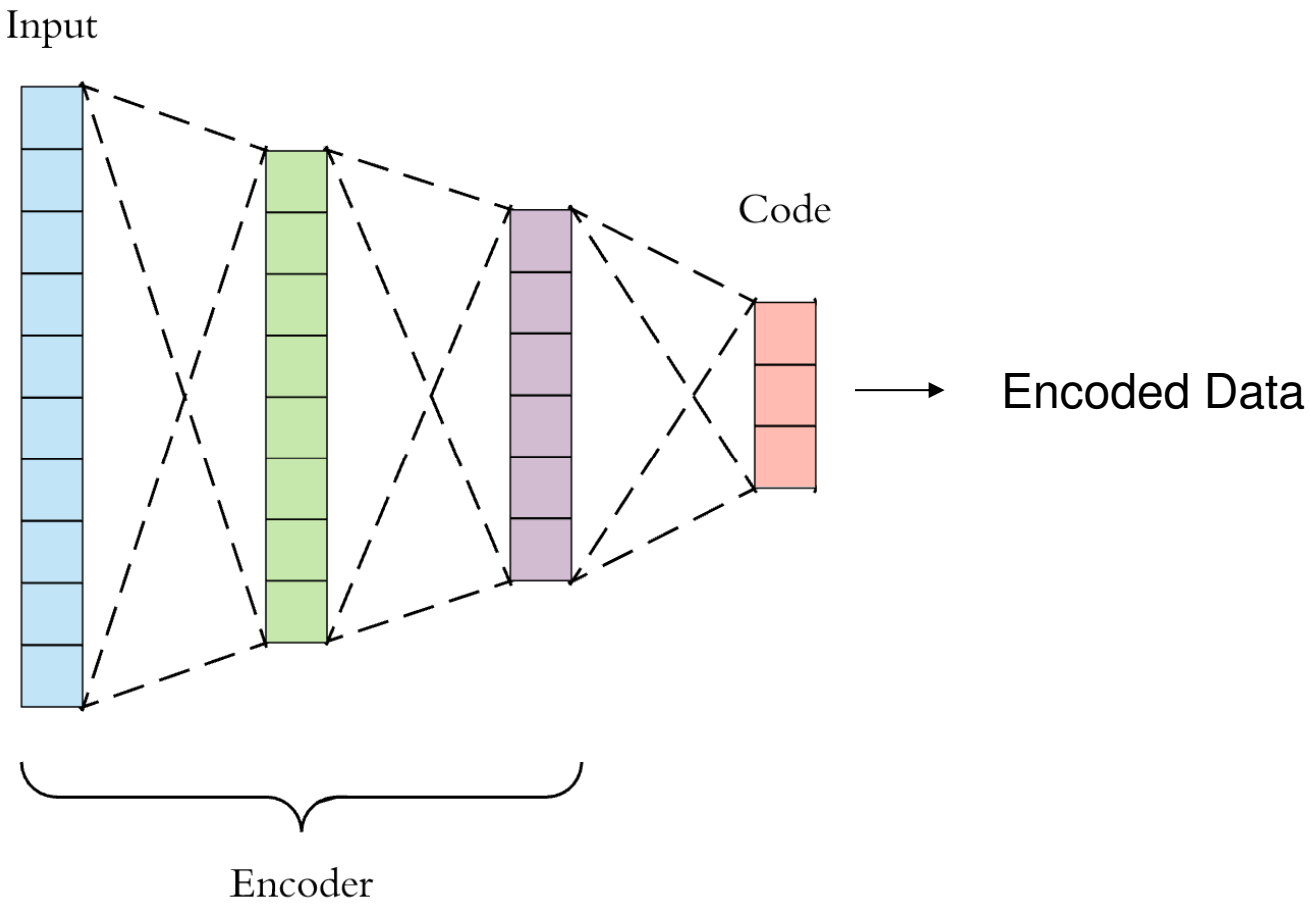
Dimension Reduction

Fully Connected - Autoencoder



Dimension Reduction

Fully Connected - Autoencoder

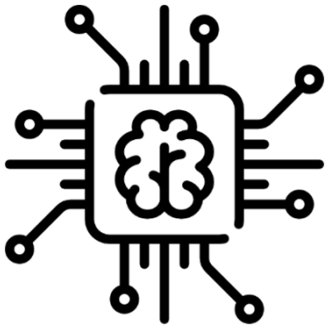


Deep Neural Networks

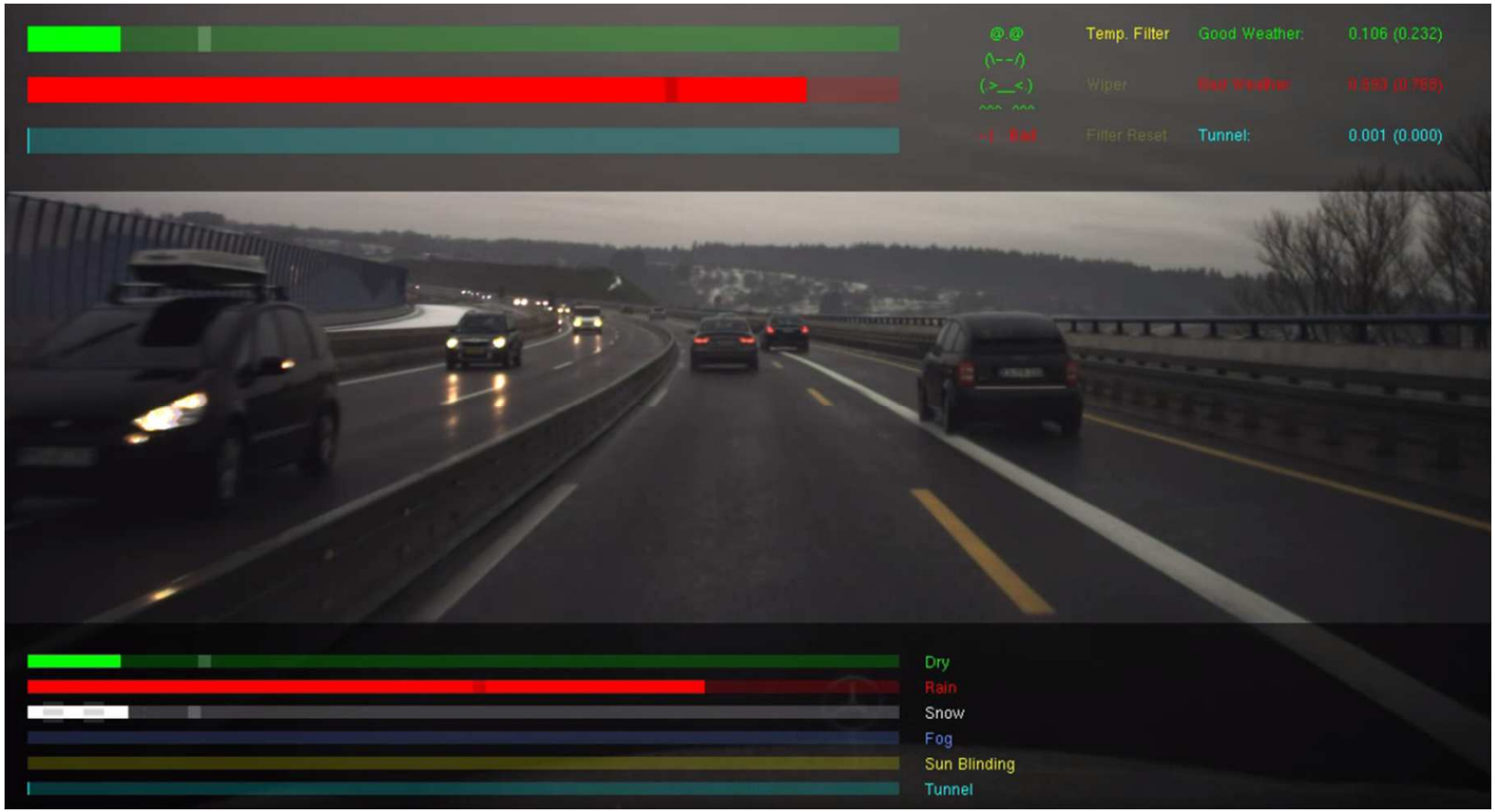
Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

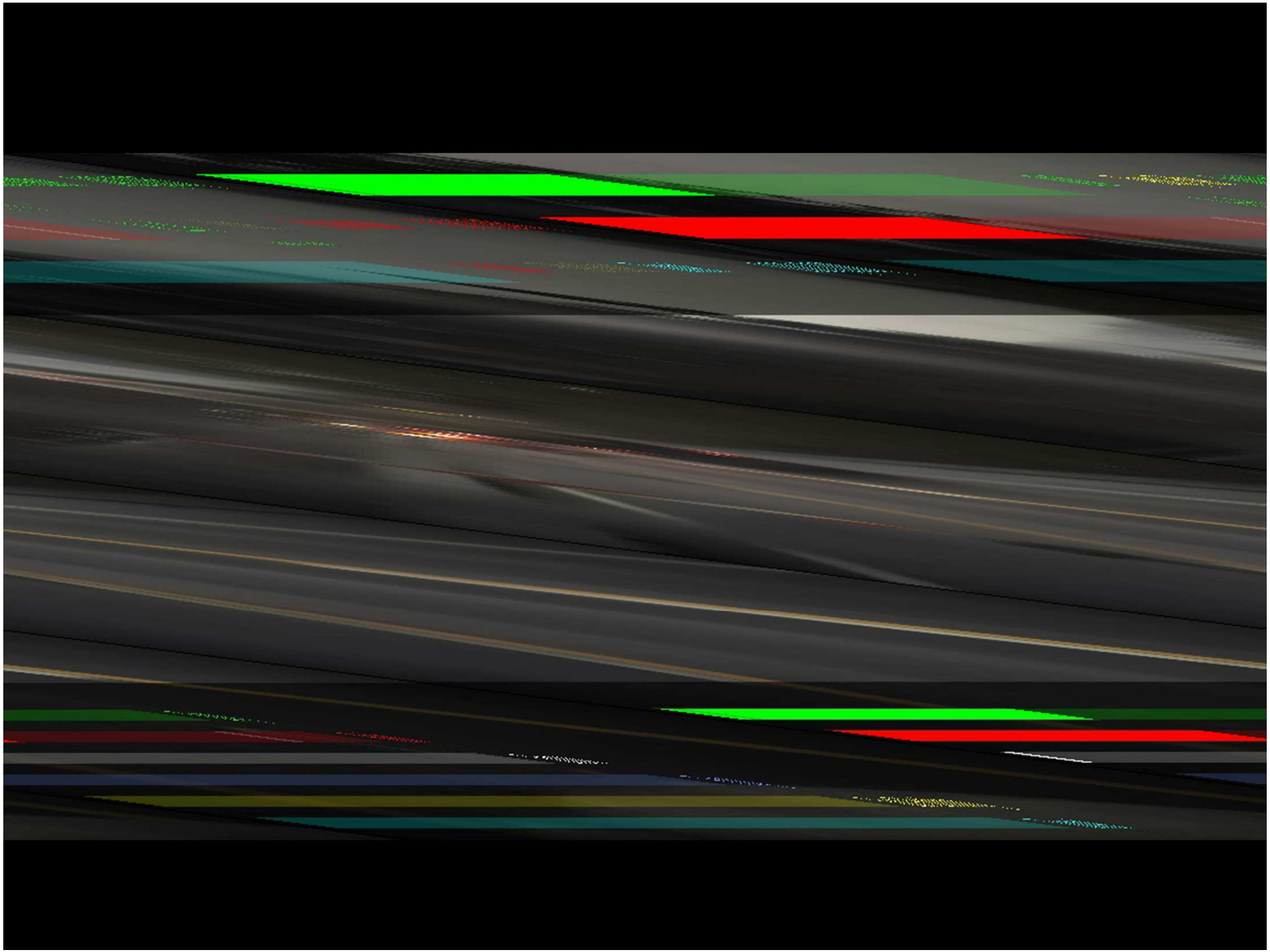
Agenda

1. Chapter: Convolutional Neural Networks
    - 1.1 Motivation
    - 1.2 Introduction
    - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
    - 2.1 Optimizer
    - 2.2 Data Formatting

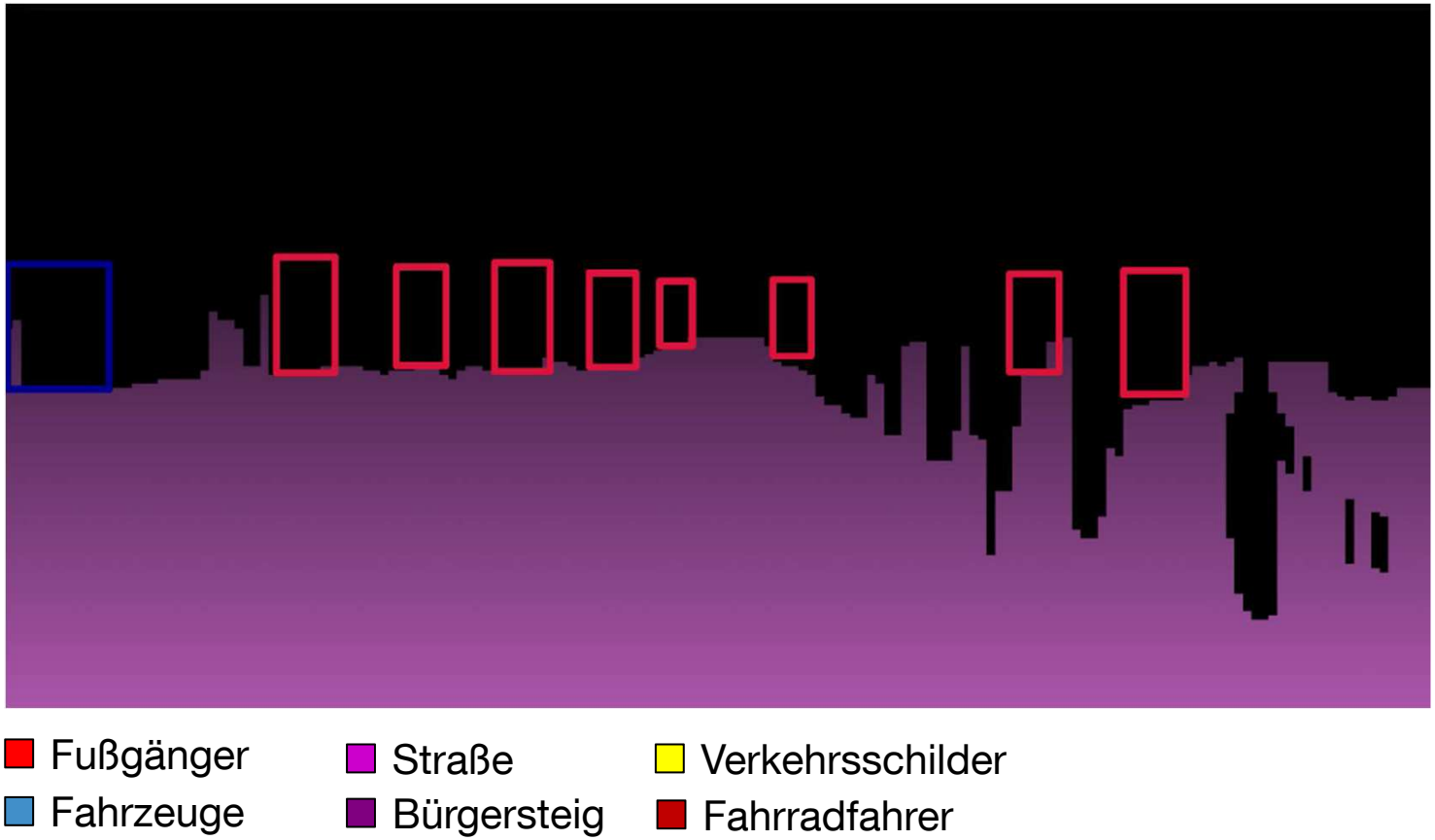


Daimler: Weather Prediction Dr. Uwe Franke et al.





Bertha Benz 1888 -> 2013 autonom



Bertha Benz Fahrt

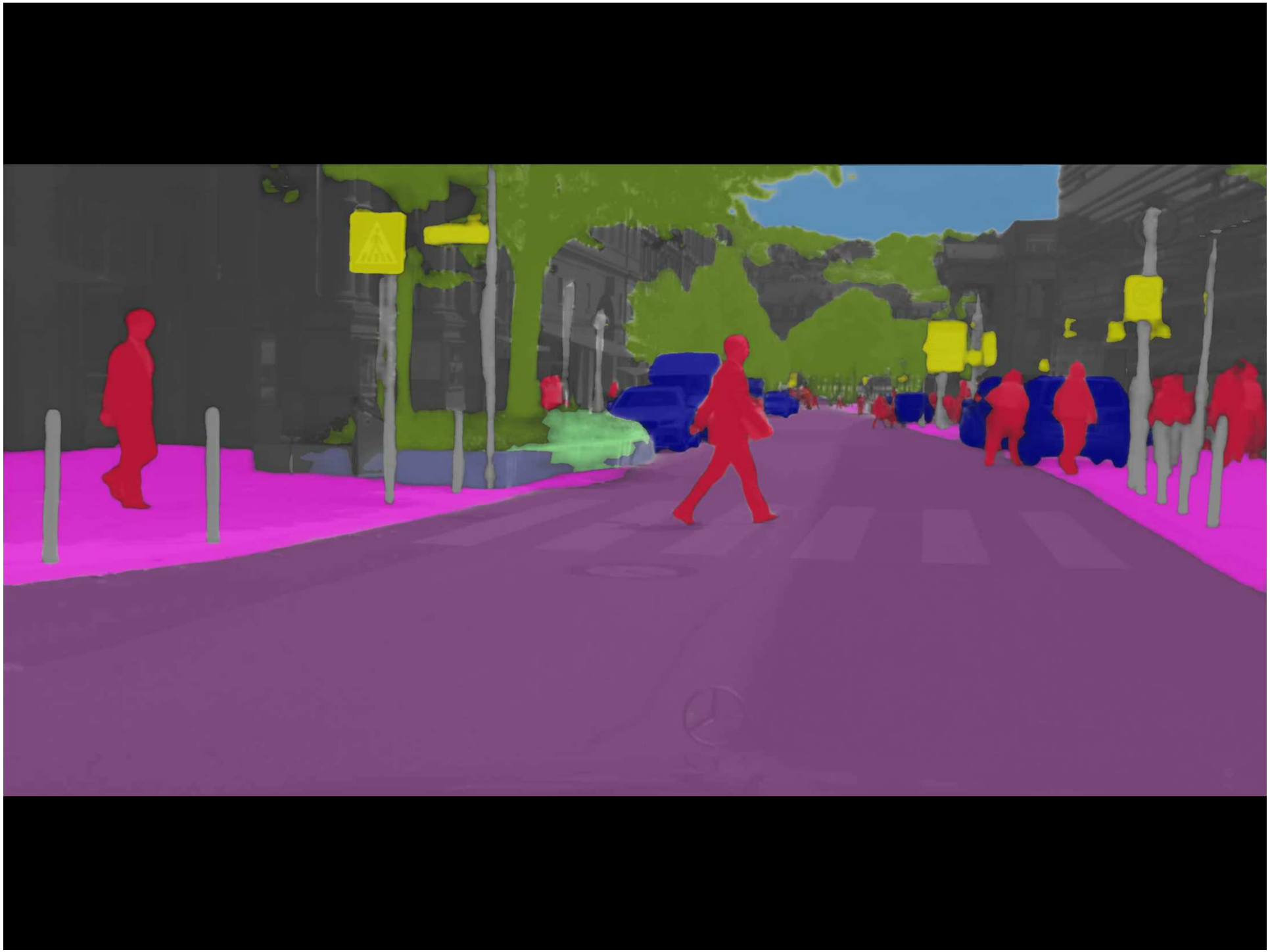
Beispielszene

Beispielszene

Daimler: Cityscapes Dataset

x

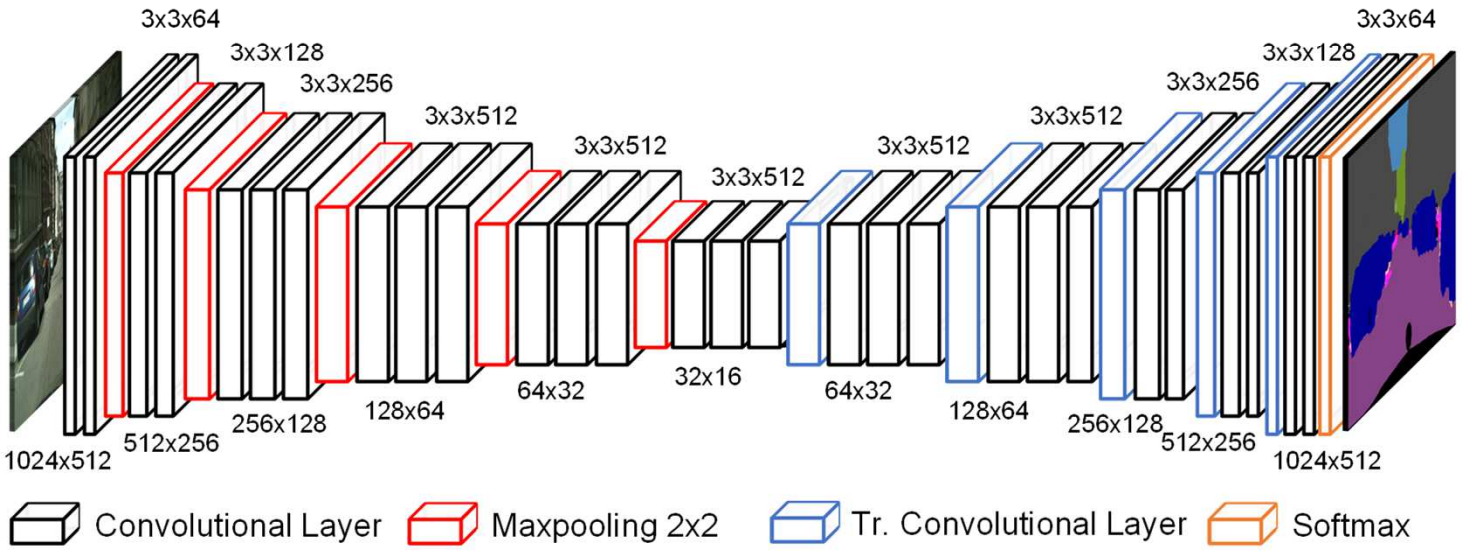
$\hat{y}$



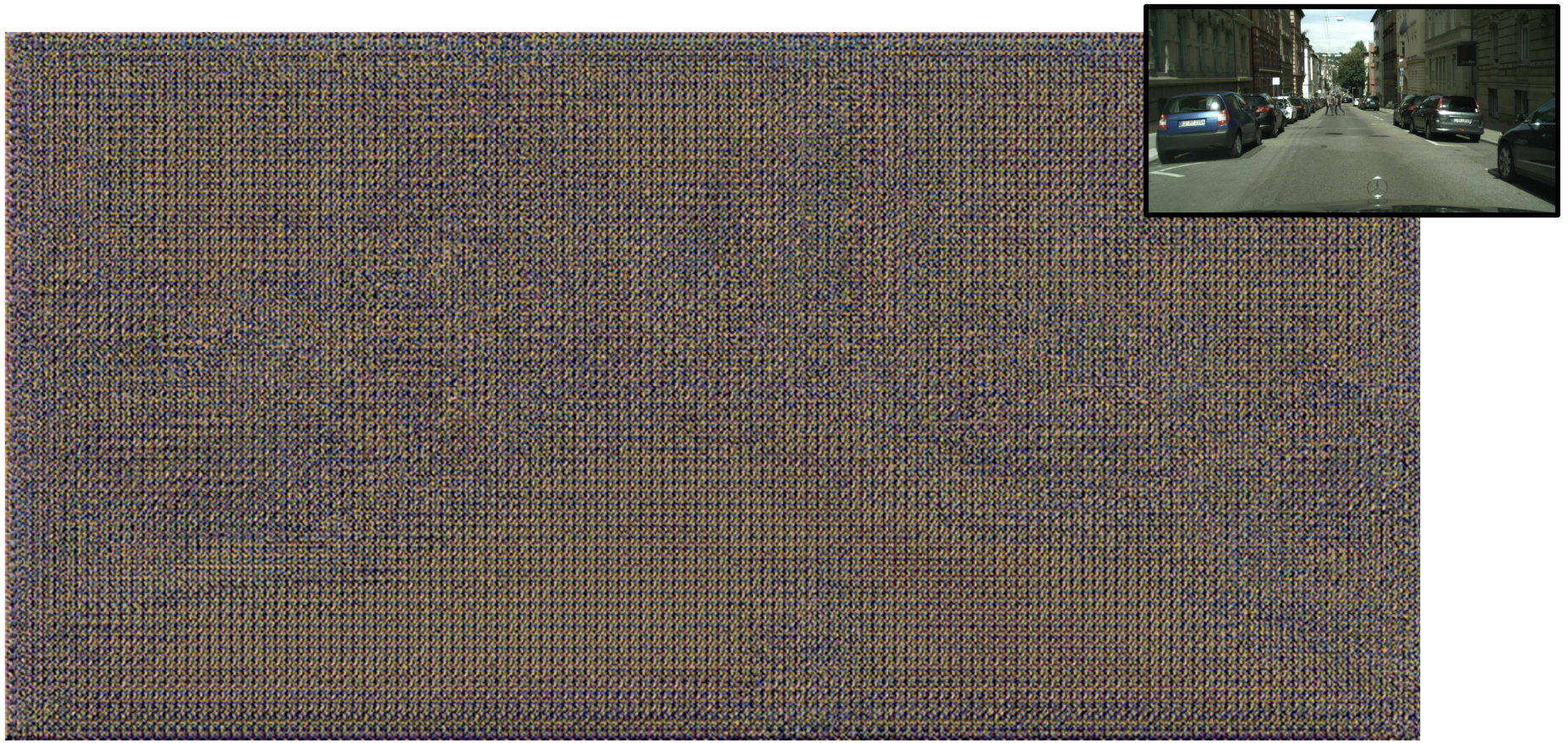
Cityscapes FCN: Learning Process

CNN-Struktur

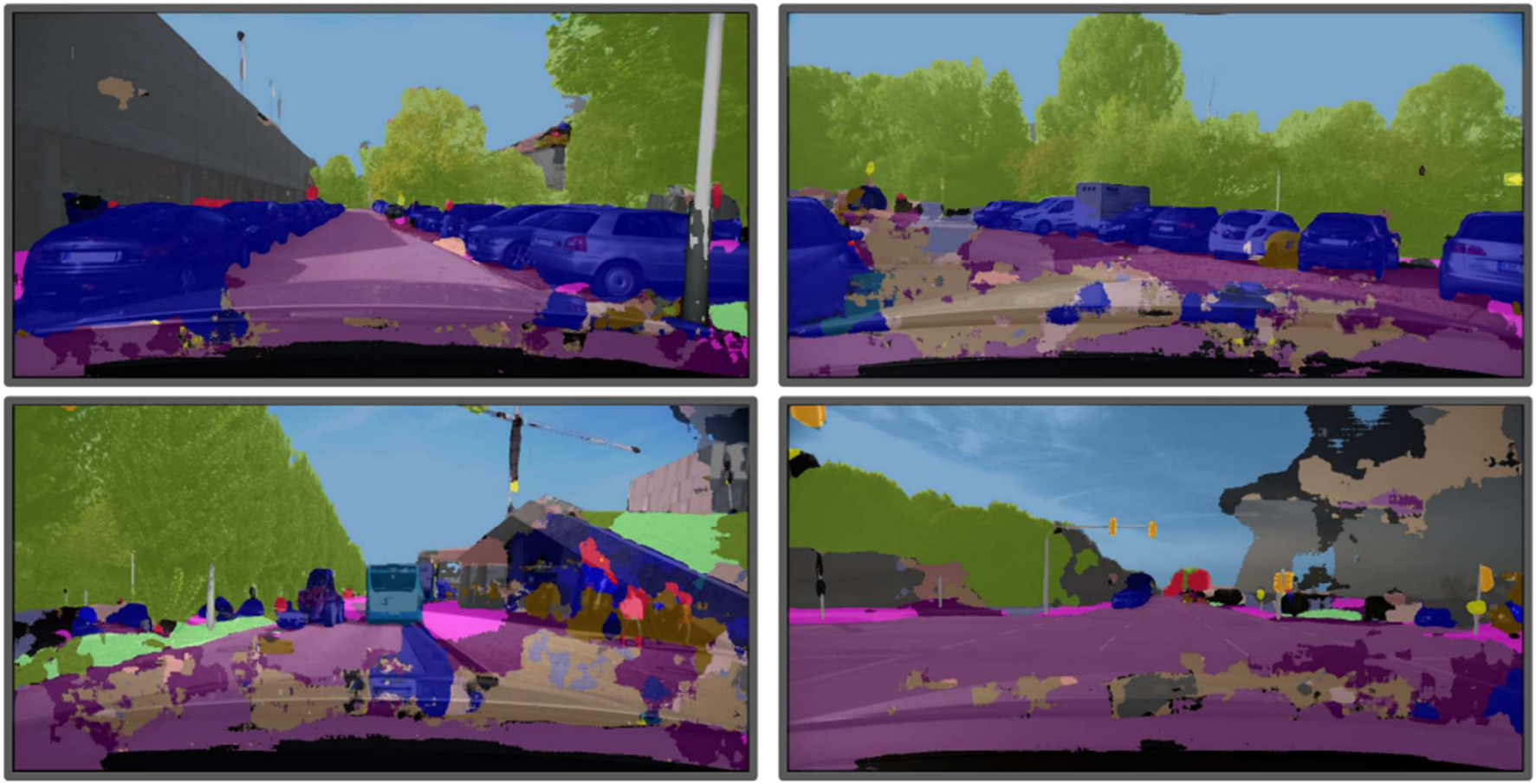
VGG16 Mirrored like code (VGG) (feature_drop_before_cvtr)



Cityscapes CNN: Learning Process

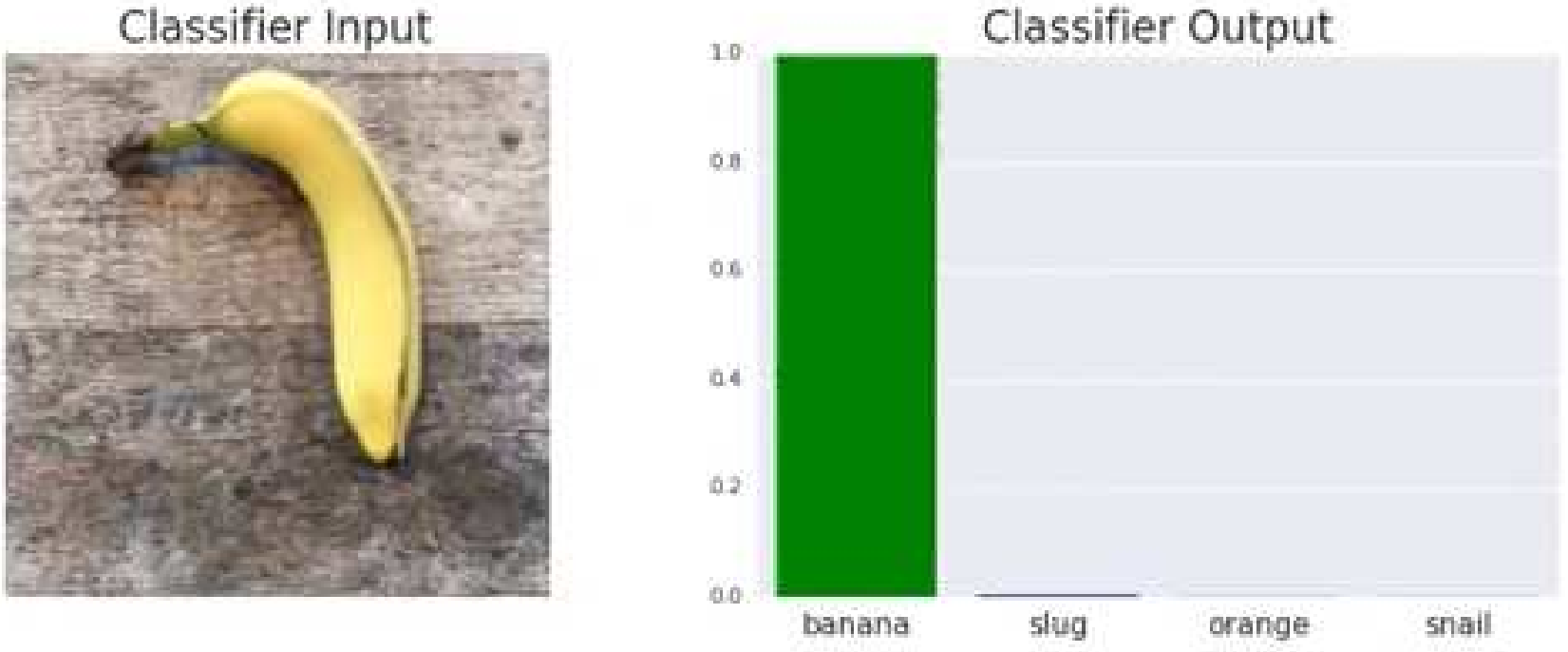


Pylon Camera Results

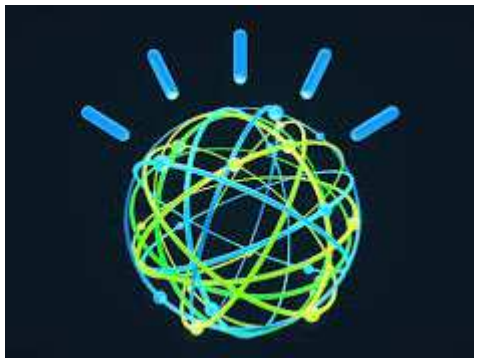


Galaxy S7 Results

Video: Adversial Patch



Mensch vs. Maschine



	Mensch	IBM Watson, 2012
Geschwindigkeit	10.000.000 Mia Op/s	80.000 Mia. Op/s
Energie	20 Watt	200 Kilo Watt
Effizienz	500x10^6 Mega Flops/Watt	400 Mega Flops/Watt

=> Mehr als 50% des Hirns beschäftigt sich mit Sehen

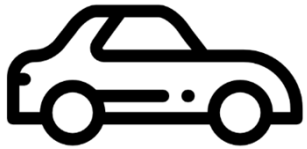
Source: Yu Wang, Tsinghua University, Feb 2016; https://www.quora.com/How-much-of-the-brain-is-involved-with-vision

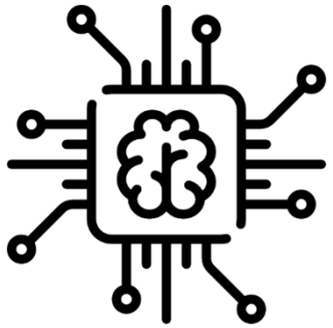
Deep Neural Networks

Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

Agenda

1. Chapter: Convolutional Neural Networks
    - 1.1 Motivation
    - 1.2 Introduction
    - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
    - 2.1 Optimizer
    - 2.2 Data Formatting





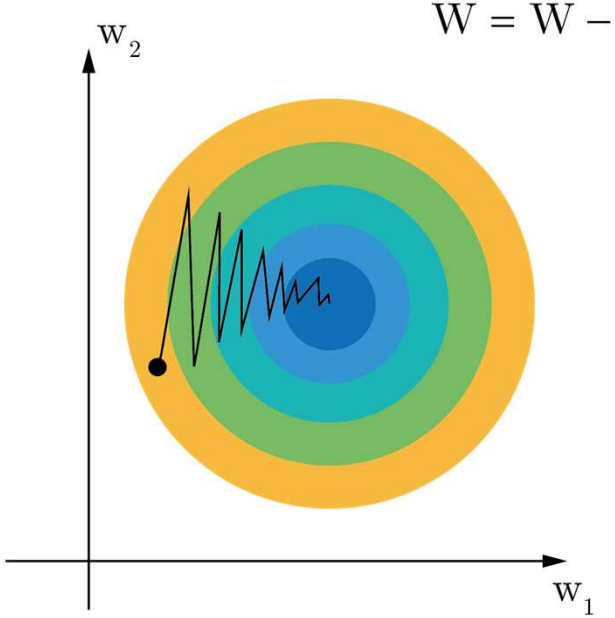
Optimizer

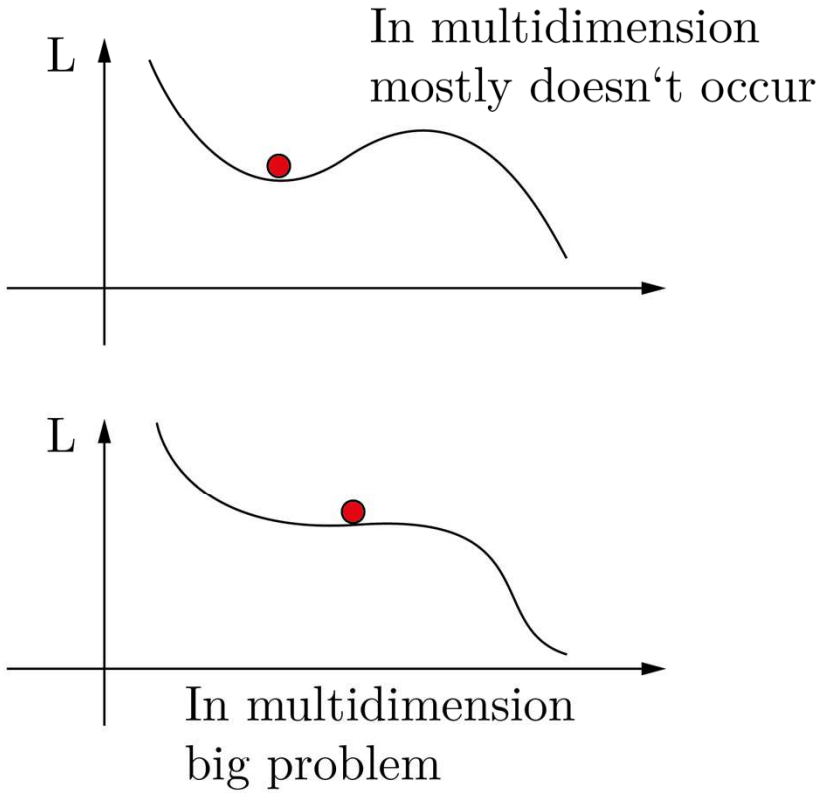
Stochastic Gradient Decent

Problems with SGD:

- Loss might change quickly in one direction, but slow in the other!
- local minima and sattle points

$$W = W - \alpha \nabla_W L$$



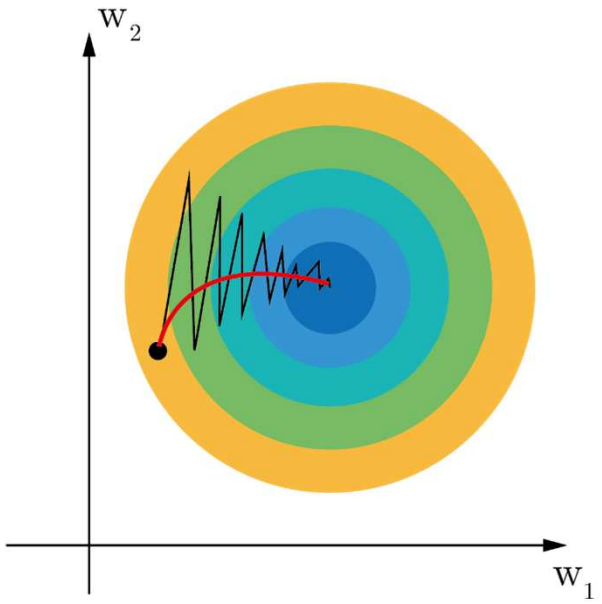


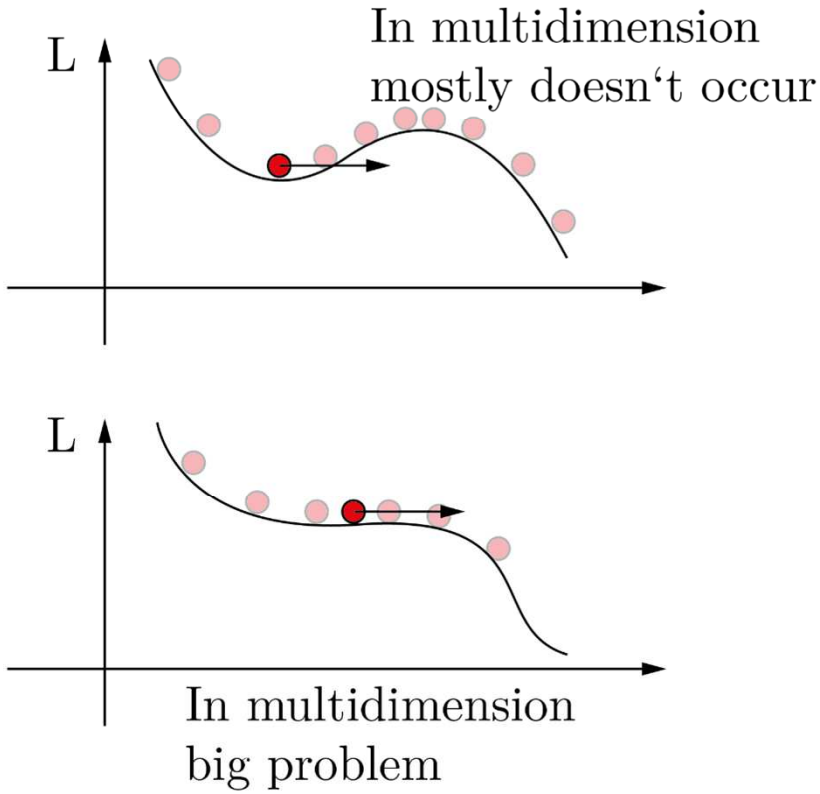
Optimizer

Stochastic Gradient Decent + Momentum

$$W_{t+1} = W_t - \alpha v_t \qquad v : \text{velocity}$$

$$v_{t+1} = \rho v_t + \nabla_W L_t \qquad \rho : \text{friction}$$

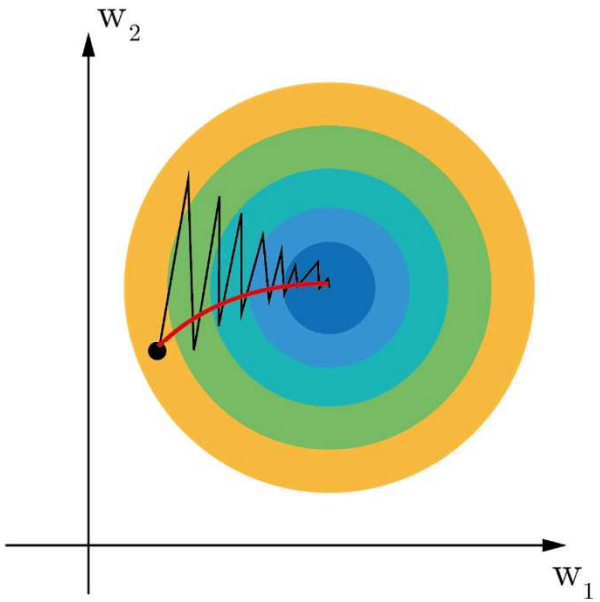




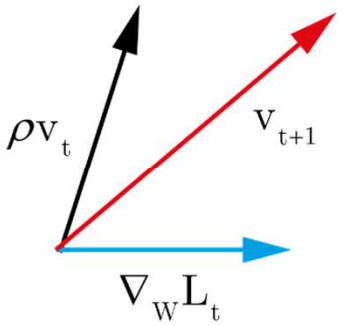
Optimizer

Nesterov+ Momentum

$$W_{t+1} = W_t - \alpha v_t$$
$$v_{t+1} = \rho v_t + \nabla_W L_t(x + v_t)$$

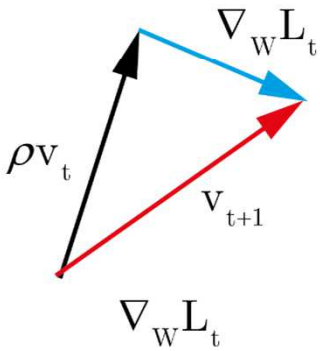


SGD + Momentum



$$v_{t+1} = \rho v_t + \nabla_W L_t(x)$$

Nesterov Momentum



$$v_{t+1} = \rho v_t + \nabla_W L_t(x + v_t)$$

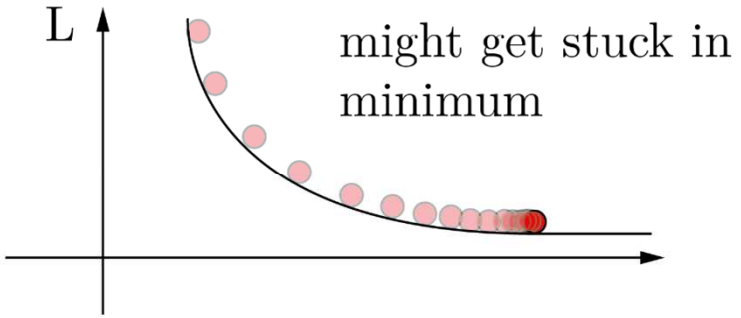
Optimizer

AdaGrad and RMSProp

AdaGrad:

$$\Phi_{t+1} = \Phi_t + \nabla_W L_t * \nabla_W L_t$$

$$W_{t+1} = W_t - \alpha \frac{\nabla_W L_t}{\sqrt{\Phi_t + \varepsilon}}$$

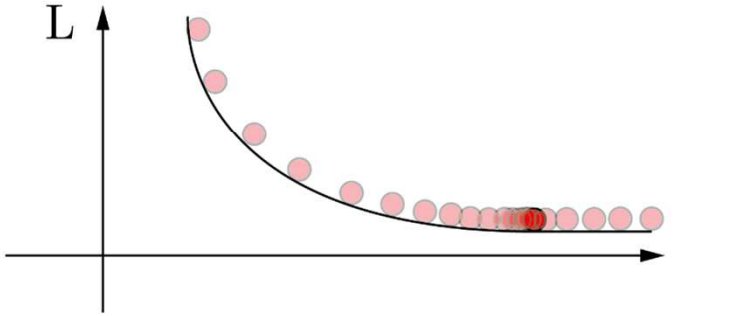


RMSProp:

$$\Phi_{t+1} = \delta\, \Phi_t + (1-\delta)\nabla_W L_t * \nabla_W L_t$$

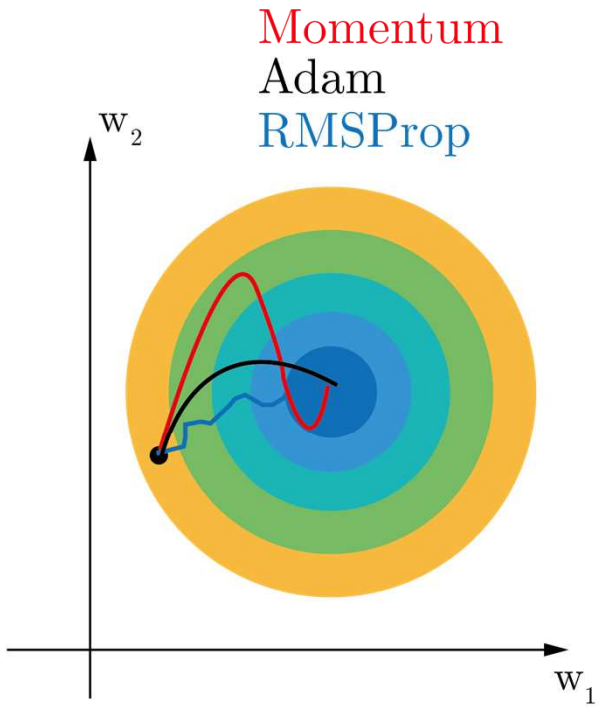
$$W_{t+1} = W_t - \alpha \frac{\nabla_W L_t}{\sqrt{\Phi_t + \varepsilon}}$$

δ : decay rate



Optimizer

Adam



Combinig RMSProp and Velocity:

$$\Upsilon_{t+1} = \delta_1 \Upsilon_t + (1-\delta_1)\nabla_W L$$
$$\Phi_{t+1} = \delta_2 \Phi_t + (1-\delta_2)\nabla_W L_t * \nabla_W L_t$$
$$W_{t+1} = W_t - \alpha \frac{\Upsilon_t}{\sqrt{\Phi_t + \varepsilon}}$$

Problem: What happens at the beginning?

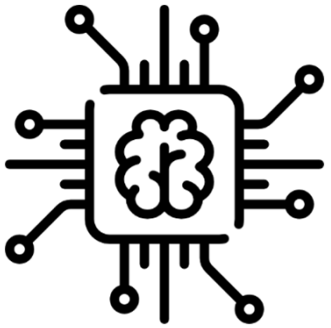
$$\Upsilon_{t+1} = \delta_1 \Upsilon_t + (1-\delta_1)\nabla_W L$$
$$\Phi_{t+1} = \delta_2 \Phi_t + (1-\delta_2)\nabla_W L_t * \nabla_W L_t$$
$$\hat{\Upsilon}_{t+1} = \frac{\Upsilon_{t+1}}{1-\delta_1^t}$$
$$\hat{\Phi}_{t+1} = \frac{\Phi_{t+1}}{1-\delta_1^t}$$
$$W_{t+1} = W_t - \alpha \frac{\hat{\Upsilon}_t}{\sqrt{\hat{\Phi}_t + \varepsilon}}$$

Deep Neural Networks

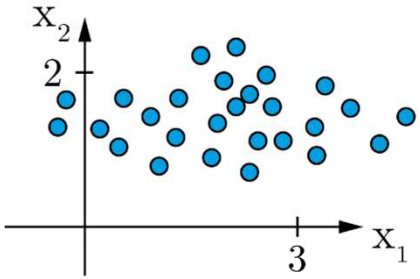
Johannes Betz / Prof. Dr. Markus Lienkamp / Prof. Dr. Boris Lohmann
(Jean-Michael Georg, M. Sc.)

Agenda

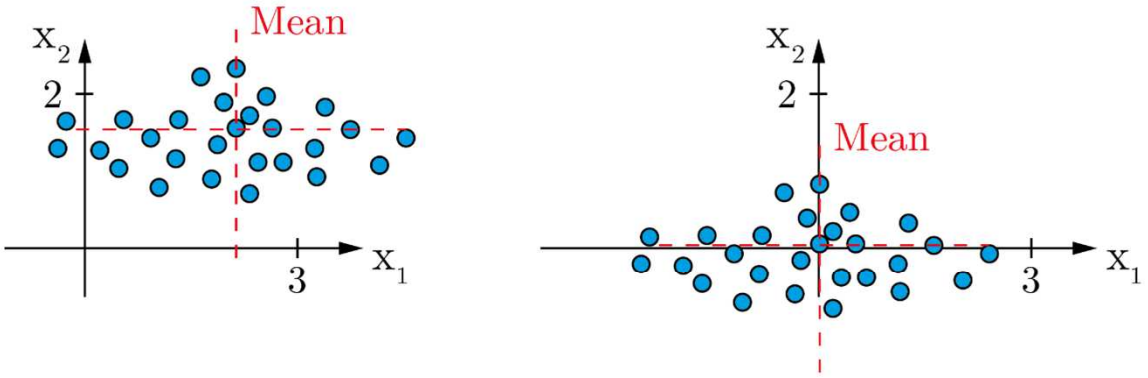
1. Chapter: Convolutional Neural Networks
    - 1.1 Motivation
    - 1.2 Introduction
    - 1.3 Dimensions, Padding, Stride
2. Chapter: Pooling
3. Chapter: CNN in Action
4. Chapter: Advanced Topics
    - 2.1 Optimizer
    - 2.2 Data Formatting



Data Distribution of the Input Data



Data Distribution of the Input Data

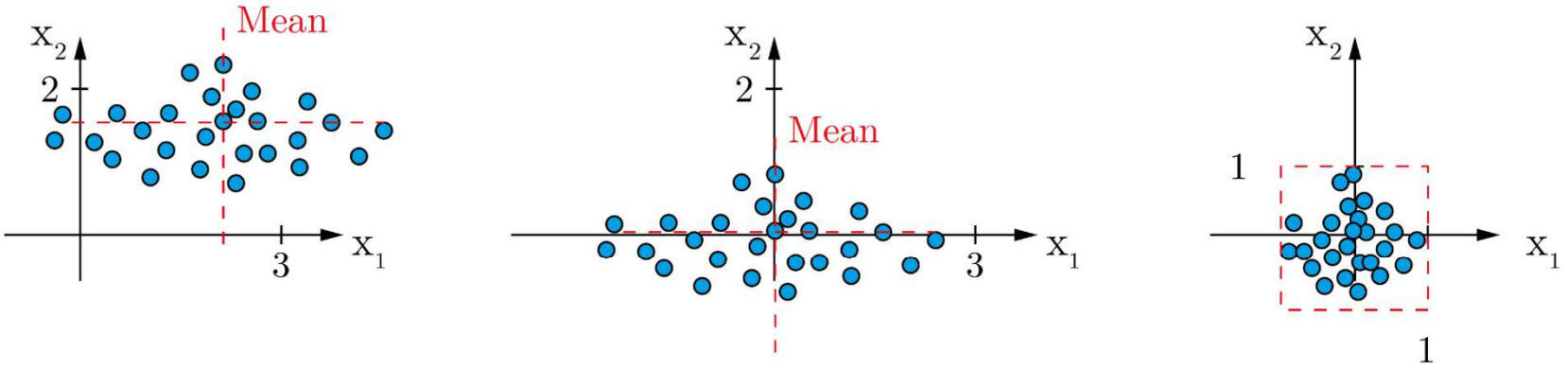


1. Subsract Mean

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$x_i = x_i - \mu$$

Data Distribution of the Input Data



1. Subsract Mean

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$x_i = x_i - \mu$$

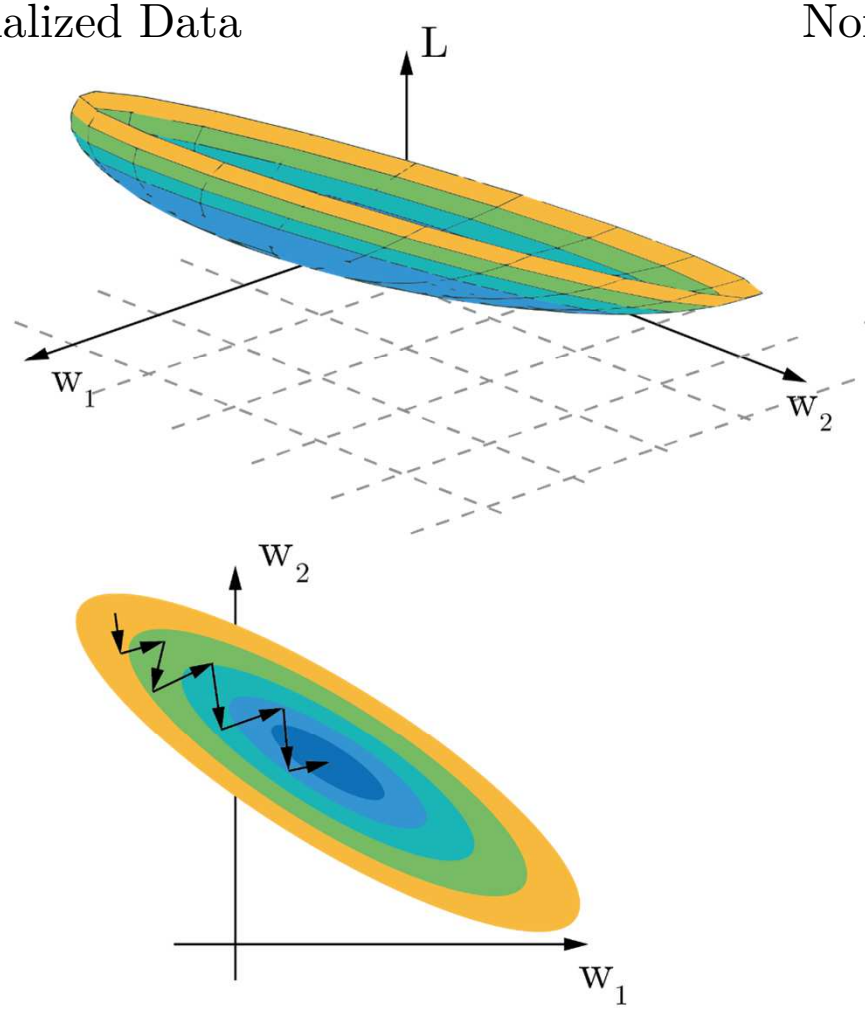
2. Normalize Variance

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} x_i^2$$

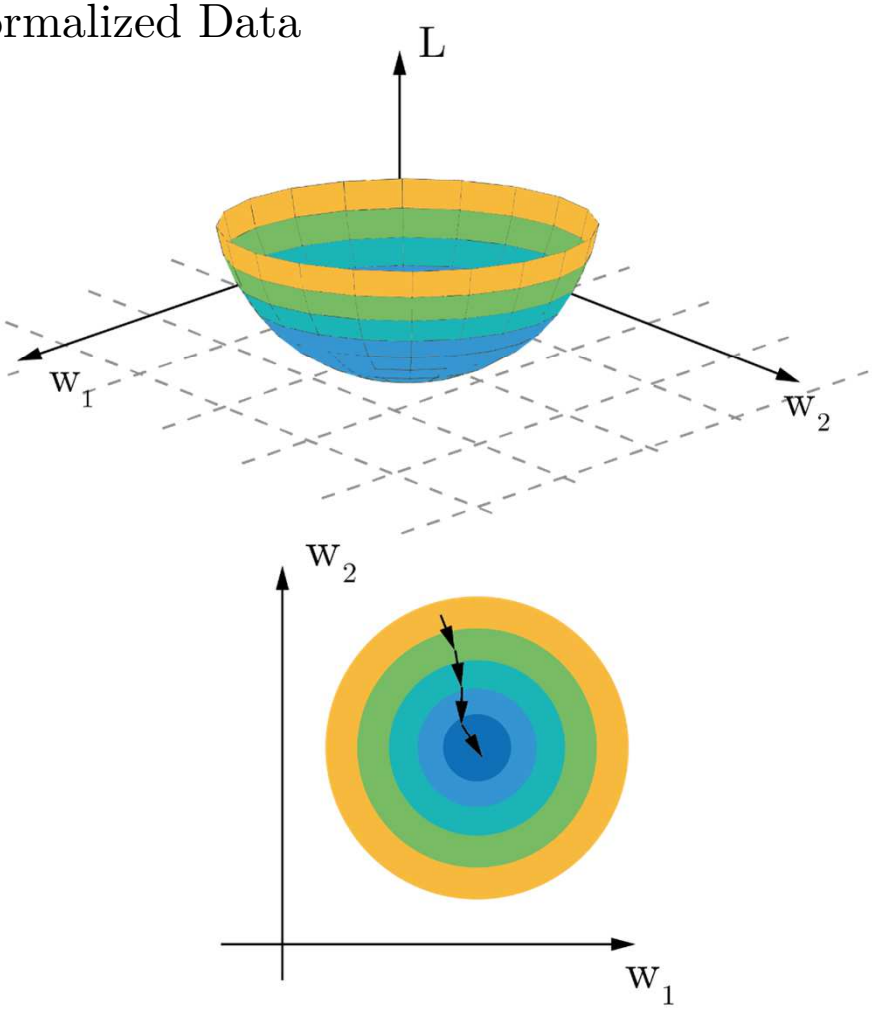
$$x_i = \frac{x_i}{\sigma^2}$$

Data Distribution of the Input Data

Unormalized Data



Normalized Data



Excursion

Regularization

Until now:

$$\Theta \equiv \{W^1, b^1, ..., W^L, b^l\}$$

Training step :

$$\Theta = \Theta - \alpha \Delta \Theta$$

$$\Delta \Theta = \vec{\nabla}_{\Theta} L(x, y)$$

Adding regularization term:

$$\Delta \Theta = \vec{\nabla}_{\Theta} L(x, y) + \lambda \vec{\nabla}_{\Theta} \Omega(\Theta)$$

Why?

- penalize complicated neural networks
- method to stop specialized neural networks and train more generalized networks
- works well against overfitting

L1 regularization :

$$\Omega(\Theta) = \sum_{l} \sum_{i} \sum_{j} | w^l_{i,j} |$$

$$\vec{\nabla}_{\Theta^l} \Omega(\Theta) = \text{sign}(W^l)$$

L2 regularization :

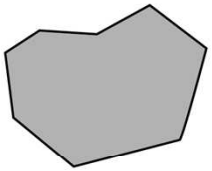
$$\Omega(\Theta) = \sum_{l} \sum_{i} \sum_{j} (w^l_{i,j})^2$$

$$\vec{\nabla}_{\Theta^l} \Omega(\Theta) = 2W^l$$

Regularization

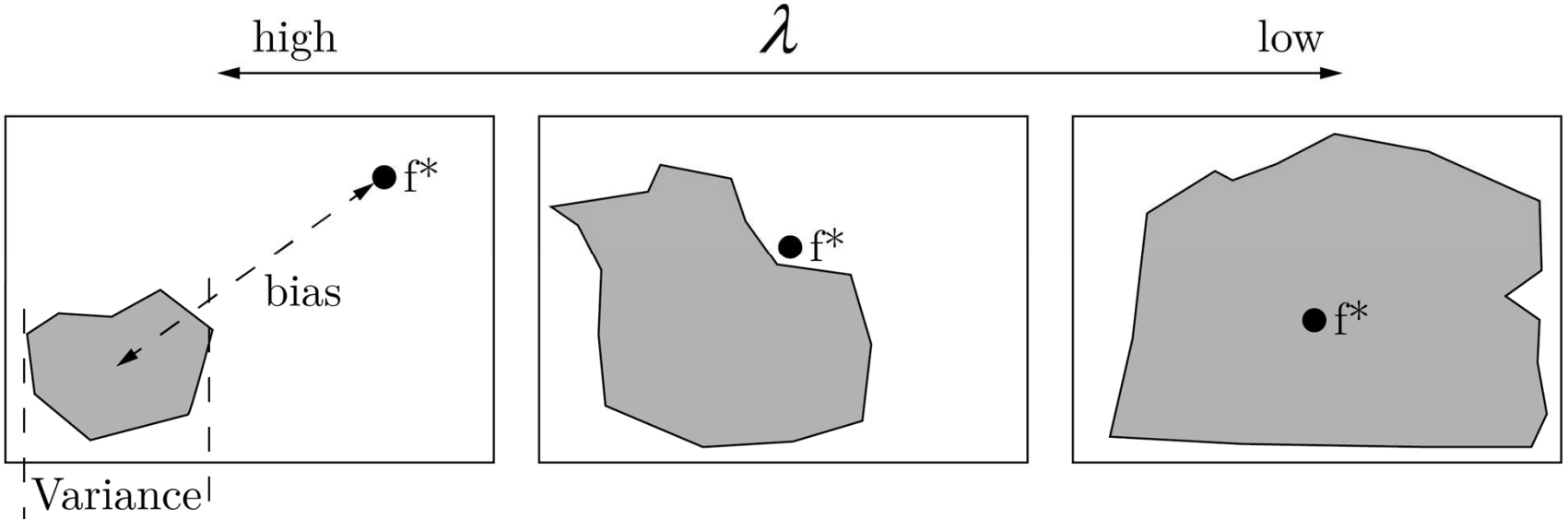
Bias and Variance of Neural Networks:

- f* : the real function

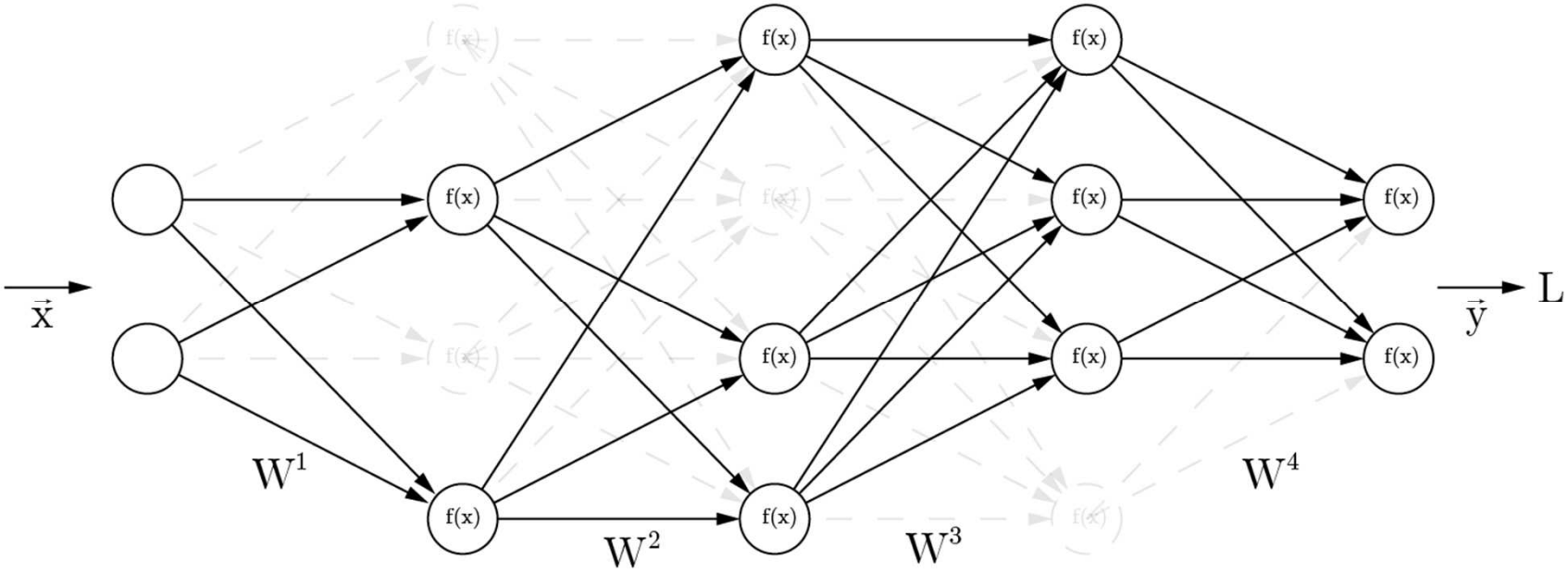


:possible Neural network function

$$\Delta\Theta = \vec{\nabla}_{\Theta} L(x, y) + \lambda \vec{\nabla}_{\Theta} \Omega(\Theta)$$



Regularization: Dropout



Kommentarfolie

*Dies ist eine Kommentarfolie.

Wir nutzen diese Folien, um zusätzeliche Inhalte zu erläutern (Beispielsweise größere Rechnungen, Zusatztexte in langer Prosa etc.)

Diese Folie wird NICHT in der Vorlesung gezeigt, sondern auskommentiert

Diese Folie kommt in das Skript für die Studenten und ist Zusatzinformation.*