Lehrstuhl für Fahrzeugtechnik
School of Engineering and Design
Technische Universität München

Fahrerassistenzsysteme im Kraftfahrzeug

Prof. Dr.-Ing. Markus Lienkamp

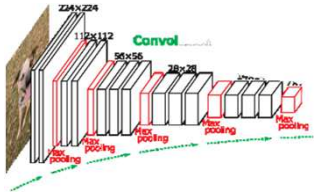
Vorlesungsübersicht

01 Einführung 28.04.2022 – Prof. Lienkamp	**01 Einführung** 28.04.2022 – Prof. Lienkamp	**01 Übung Einführung** 28.04.2022 – Hoffmann
02 Sensorik / Wahrnehmung I 05.05.2022 – Prof. Lienkamp	**02 Sensorik / Wahrnehmung I** 05.05.2022 – Prof. Lienkamp	**02 Sensorik / Wahrnehmung I** 05.05.2022 – Prof. Lienkamp
03 Sensorik / Wahrnehmung II 12.05.2022 – Dr.-Ing. Diermeyer	**03 Sensorik / Wahrnehmung II** 12.05.2022 – Dr.-Ing. Diermeyer	**03 Übung Sensorik / Wahrnehmung II** 12.05.2022 – Schimpe
04 Sensorik / Wahrnehmung III 19.05.2022 – Schimpe	**04 Sensorik / Wahrnehmung III** 19.05.2022 – Schimpe	**04 Übung Sensorik / Wahrnehmung III** 19.05.2022 – Schimpe
05 Funktionslogik / Regelung 02.06.2022 – Dr.-Ing. Winkler	**05 Funktionslogik / Regelung** 02.06.2022 – Dr.-Ing. Winkler	**05 Funktionslogik / Regelung** 02.06.2022 – Dr.-Ing. Winkler
06 Übung Funktionslogik / Regelung 09.06.2022 – Dr.-Ing. Winkler	**06 Funktionale Systemarchitektur** 09.06.2022 – Prof. Lienkamp	**06 Aktorik** 09.06.2022 – Prof. Lienkamp
07 Deep Learning 23.06.2022 – Majstorovic	**07 Deep Learning** 23.06.2022 – Majstorovic	**07 Übung Deep Learning** 23.06.2022 – Majstorovic
08 MMI 30.06.2022 – Prof. Bengler	**08 MMI** 30.06.2022 – Prof. Bengler	**08 MMI Übung** 30.06.2022 – Prof. Bengler
09 Controllability 07.07.2022 – Prof. Bengler	**09 Controllability** 07.07.2022 – Prof. Bengler	**09 Übung Controllability** 07.07.2022 – Winkle
10 Entwicklungsprozess 14.07.2022 – Dr.-Ing. Diermeyer	**10 Entwicklungsprozess** 14.07.2022 – Dr.-Ing. Diermeyer	**10 Übung Entwicklungsprozess** 14.07.2022 – Hoffmann
11 Analyse und Bewertung FAS 21.07.2022 – Dr.-Ing. Feig	**11 Analyse und Bewertung FAS** 21.07.2022 – Dr.-Ing. Feig	**11 Übung Analyse und Bewertung FAS** 21.07.2022 – Dr.-Ing. Feig
12 Aktuelle und künftige Systeme 28.07.2022 – Prof. Lienkamp	**12 Aktuelle und künftige Systeme** 28.07.2022 – Prof. Lienkamp	**12 Aktuelle und künftige Systeme** 28.07.2022 – Prof. Lienkamp

Leitfragen
07 – Deep Learning

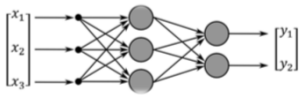
- Wie ist ein einzelnes künstliches Neuron aufgebaut und wie funktioniert es?

  - 单个人工神经元是如何构建的?



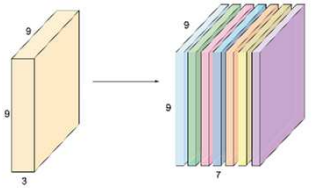
- Wie funktionieren künstliche neuronale Netze grundsätzlich?

  - 人工神经网络的原理是什么?



- Welche Funktionsweise haben die unterschiedlichen Layer-Arten?

  - 不同类型的层是如何工作的?



- Wie funktionieren Convolutional Neural Networks (CNN) im Detail, und warum sind so effektiv bei der Verarbeitung von Bilddaten?

  - 卷积神经网络 (CNN) 的详细工作原理是什么?

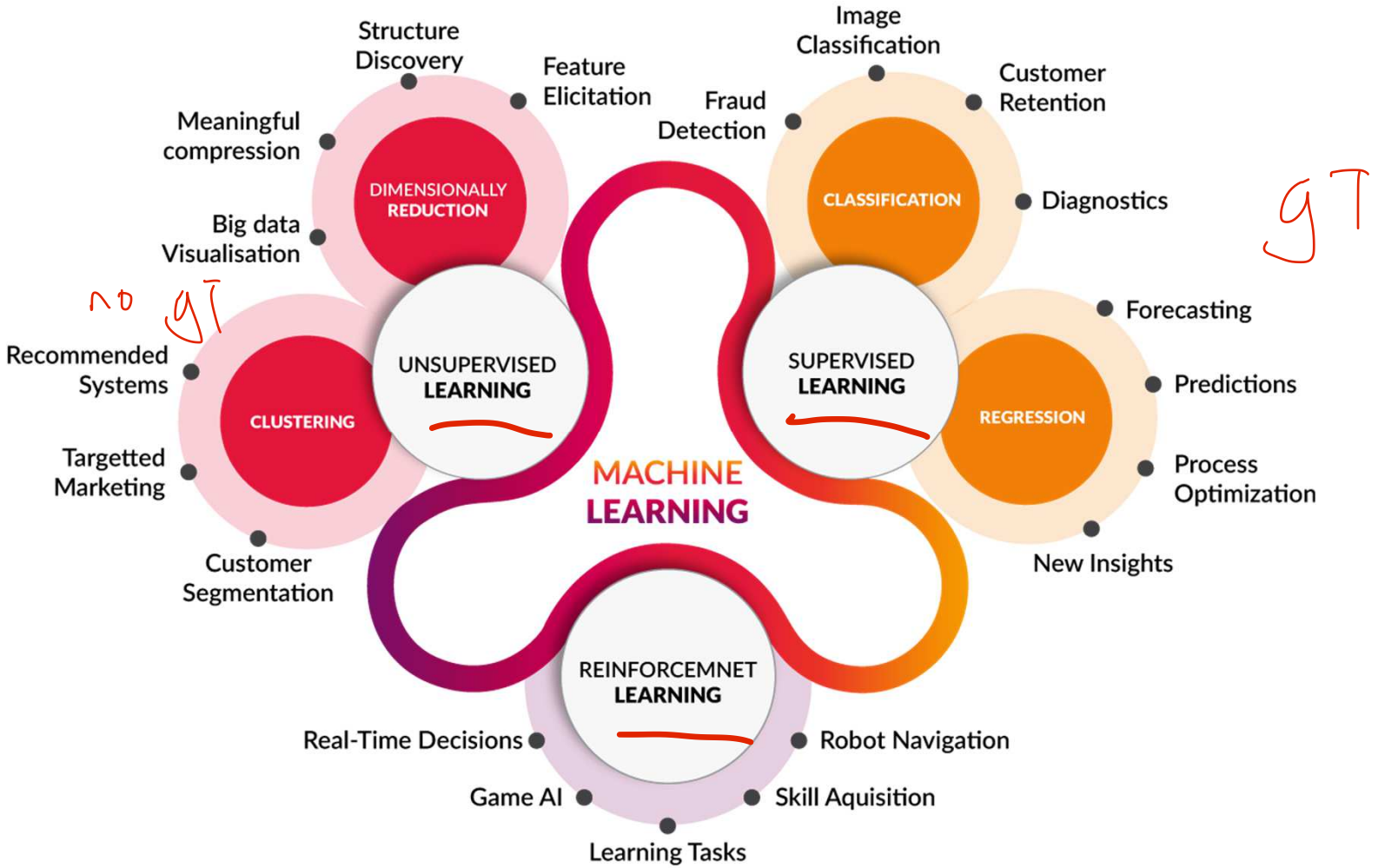
Deep Learning
Domagoj Majstorovic, M.Sc.

Agenda

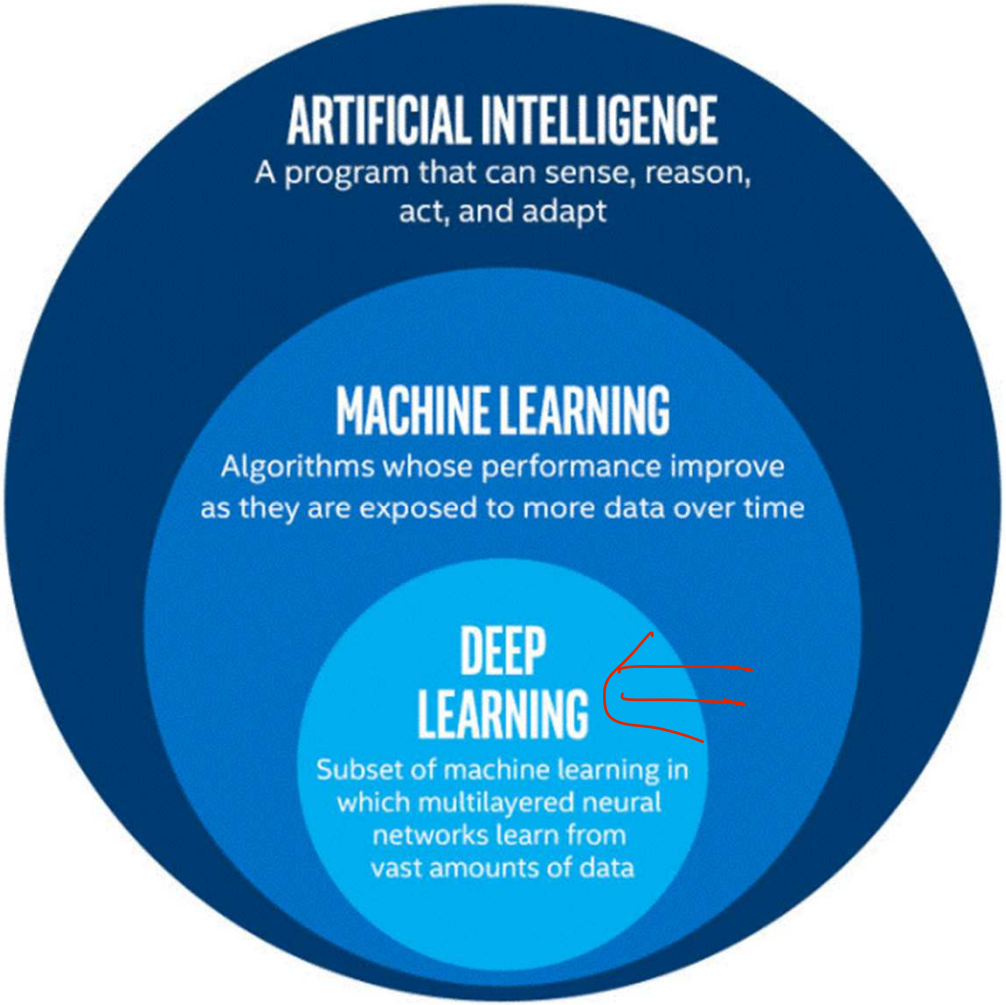
7 Deep Learning / Neuronale Netze

- 7.1 Grundlagen und Einleitung
- 7.2 Netzdefinition
- 7.3 Netztraining
- 7.4 Ergebnisse
- 7.5 Herausforderungen

Machine Learning - Übersicht

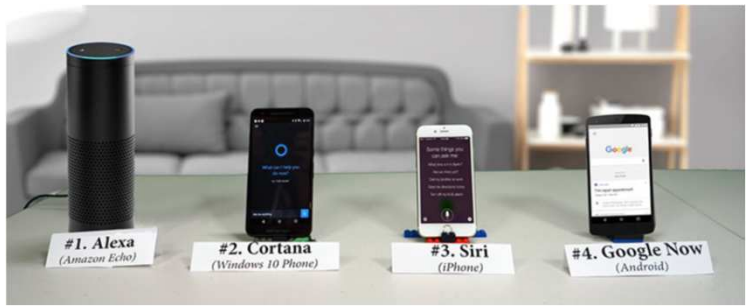


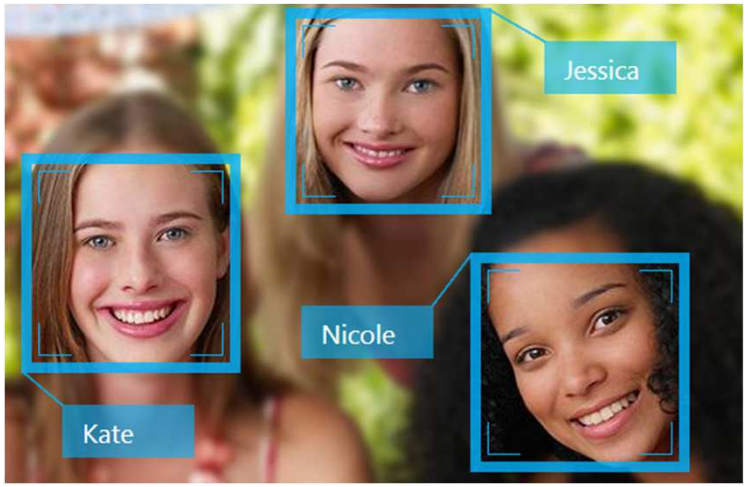
Machine Learning - Übersicht

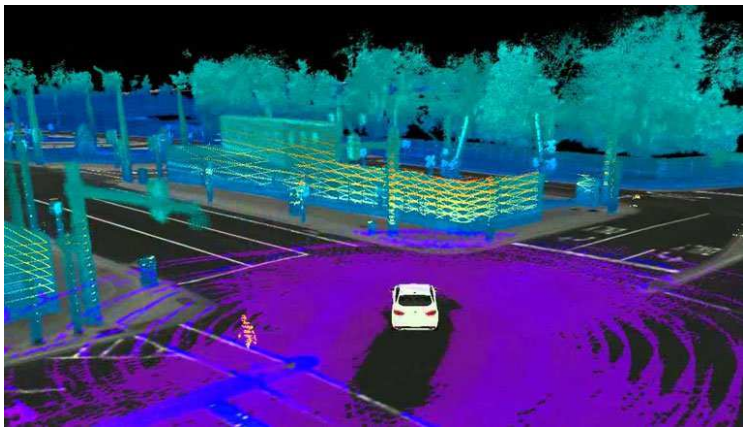


https://datasolut.com/wp-content/uploads/2019/11/1TiORvHgrJPme_IEiX3oIVA.png

Deep Learning - Übersicht

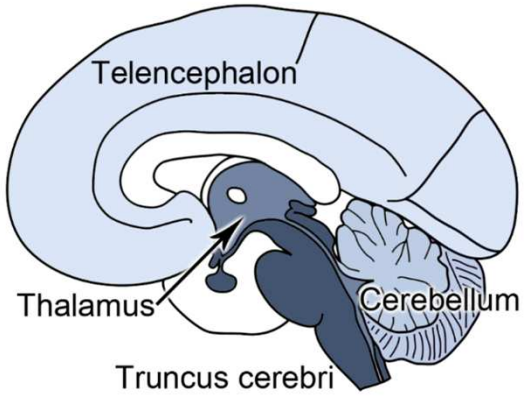


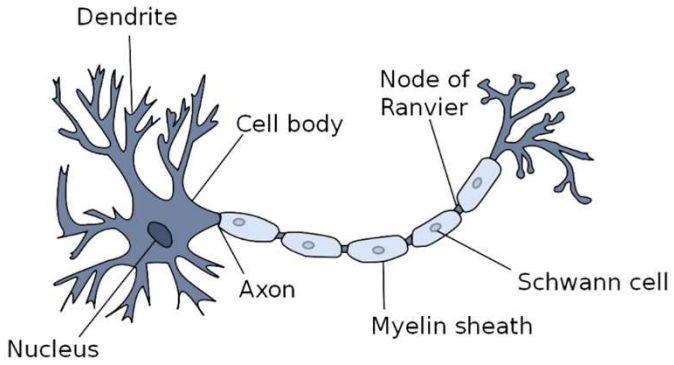


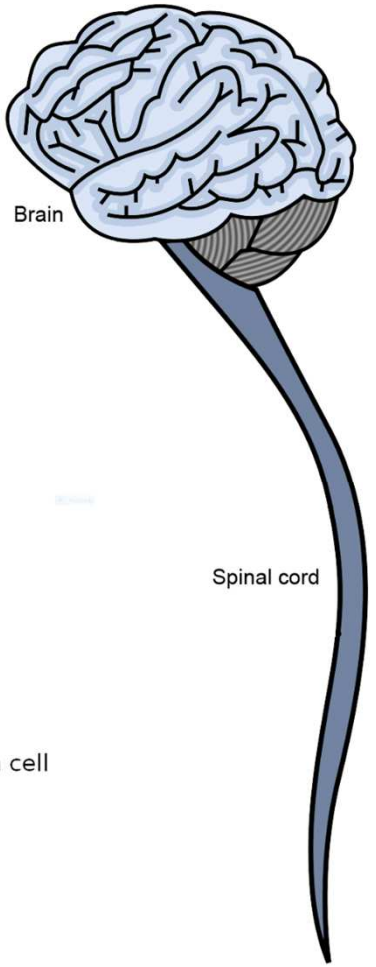


https://medium.com/the-21st-century/face-detection-using-python-a04b08118ae6

Grundlagen


Telencephalon
Thalamus
Cerebellum
Truncus cerebri

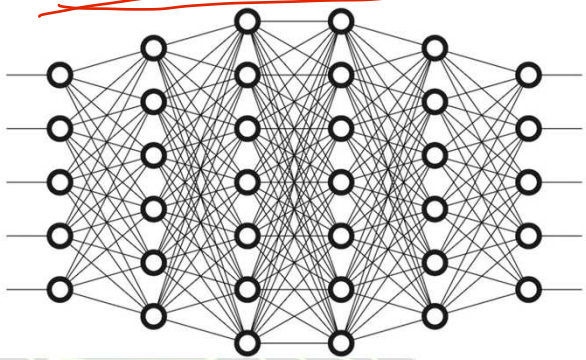

Dendrite
Cell body
Node of
Ranvier
Axon
Schwann cell
Myelin sheath
Nucleus


Brain
Spinal cord

Grundlagen

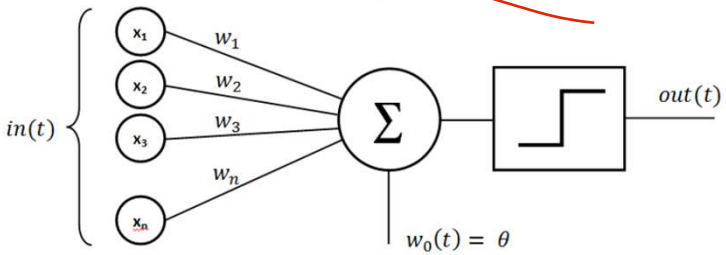
Menschliches Gehirn

Künstliches Neuronales Netz



Neuron

Perceptron/Neuron



Deep Learning – Übersicht

1942 Beschreibung erster neuronaler Netze

Erste neuronale Lernregeln, Hebsche Regel **1949**

$$\Delta w_{ij} = \eta \cdot a_i \cdot o_j$$

1957 Erster erfolgreicher Neurocomputer 20 x 20 px

Weitere theoretische Arbeiten, bis **1969**

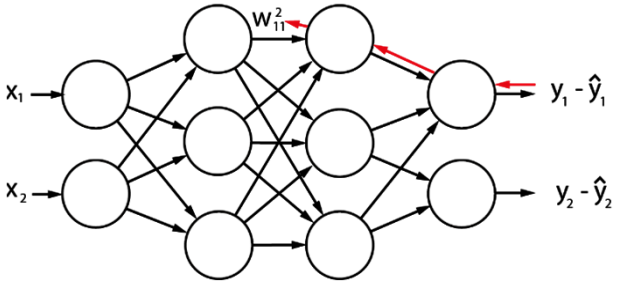
1974 Theoretische Entwicklung des Backpropagation Verfahrens

Lösung des „traveling salesman“ Problem mit selbstlernenden Netzen **1985**

1986 weitere Publikation zur Backpropagation

Erweiterung des Backpropagation Verfahrens für größere neuronale Netze **2006**

2007 Einsatz von GPU zur Berechnung von NN + Verfügbarkeit einer großen Anzahl benannter Trainingsdaten



Deep Learning – Übersicht

Projekt „ALVINN“, 1989.

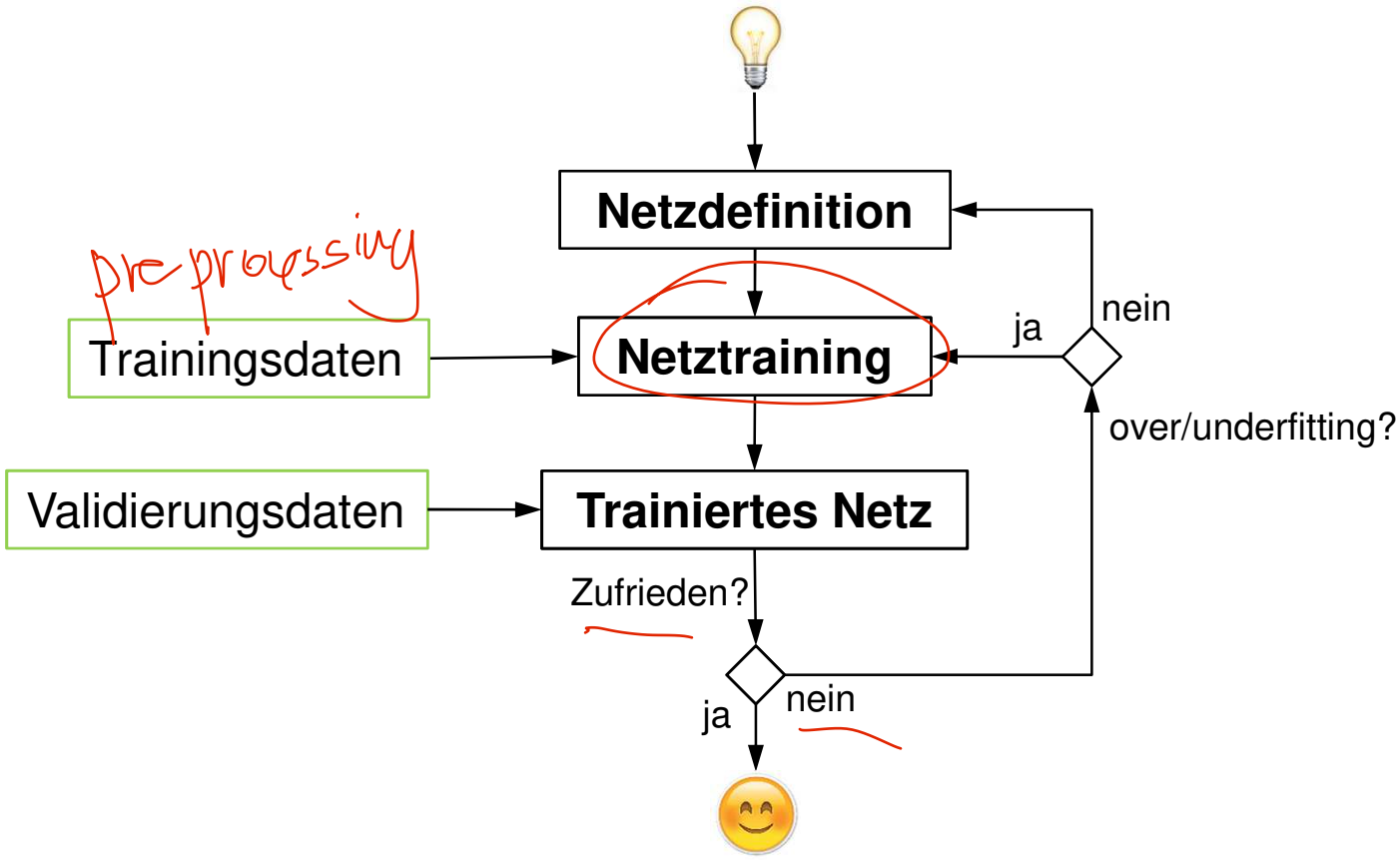
https://proceedings.neurips.cc/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf

Deep Learning – Übersicht

Tesla, FSD Beta 10.10.2, Feb 2022.

https://www.youtube.com/watch?v=xLDWsJ2ReBg

Weg zum neuronalen Netz



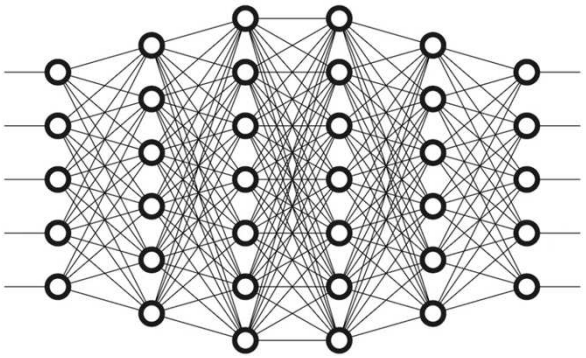
Deep Learning
Domagoj Majstorovic, M.Sc.

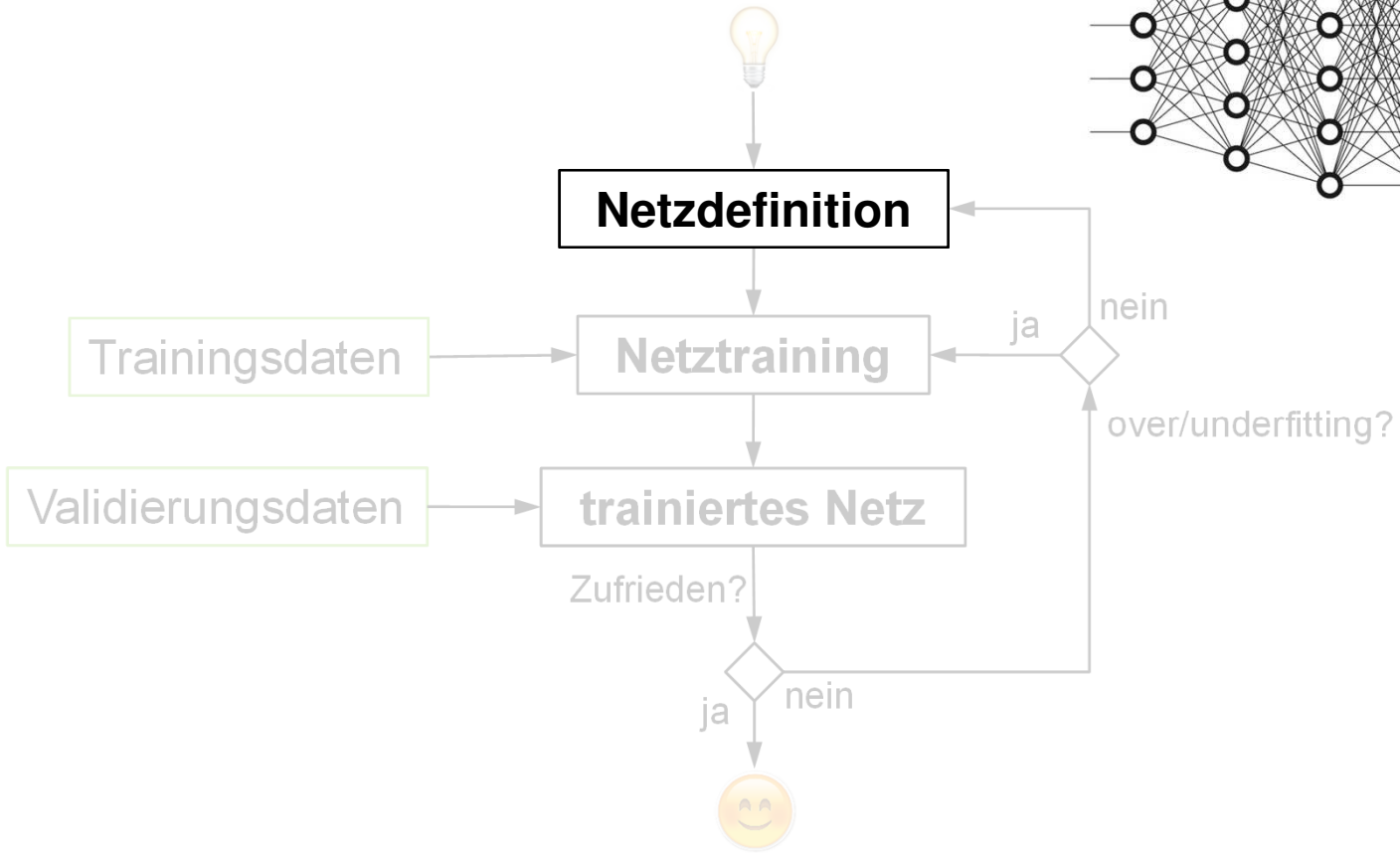
Agenda

7 Deep Learning / Neuronale Netze

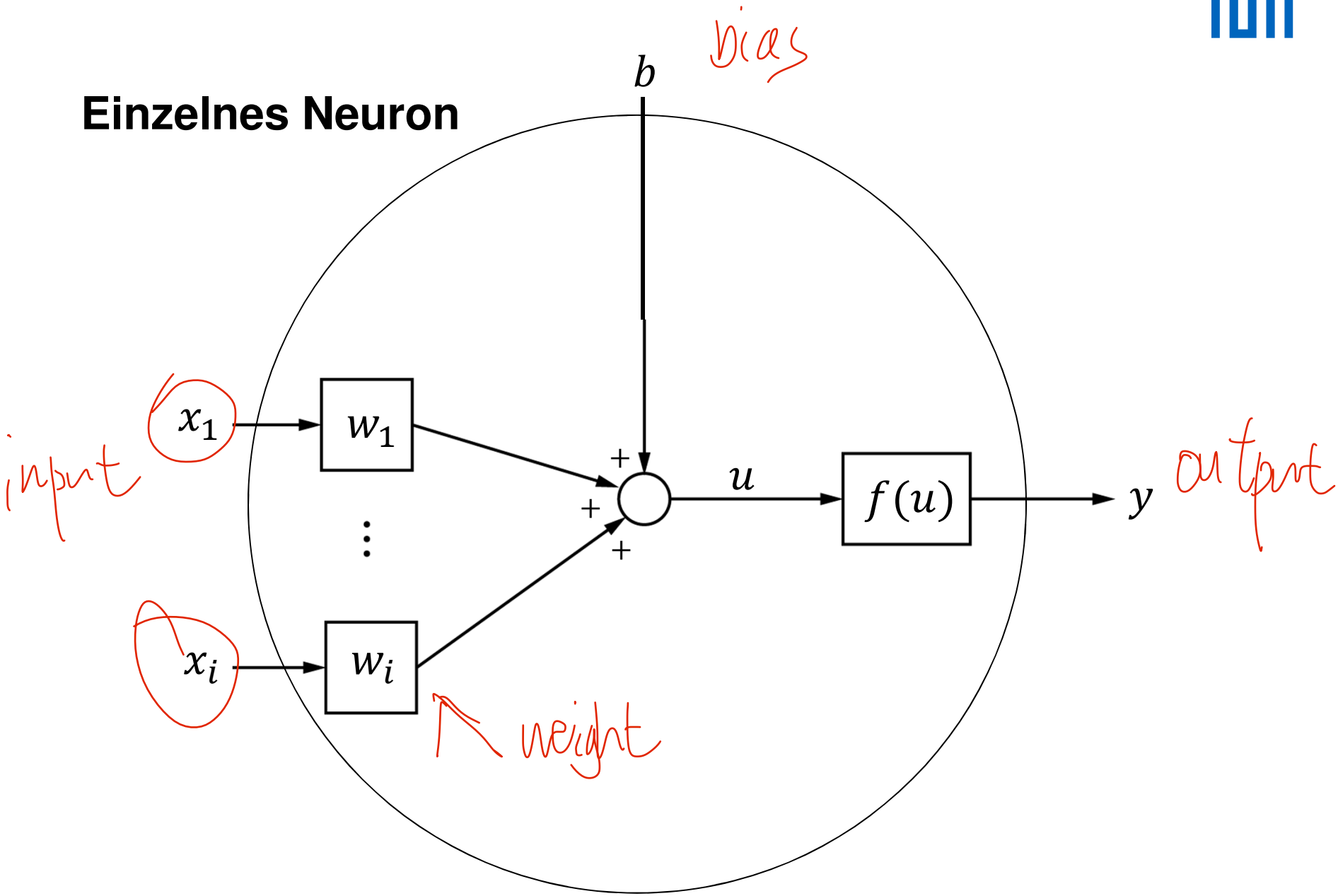
- 7.1 Grundlagen und Einleitung
- 7.2 Netzdefinition
- 7.3 Netztraining
- 7.4 Ergebnisse
- 7.5 Herausforderungen

Weg zum neuronalen Netz





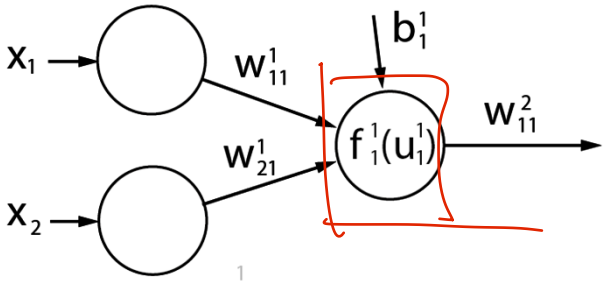
Einzelnes Neuron



Aktivierungsfunktion

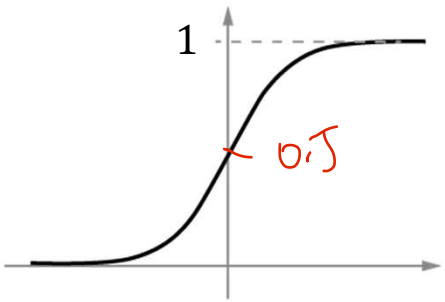
Aktivierung:

$$u_1 = x_1 w_{11}^1 + x_2 w_{21}^1 + b_1^1$$



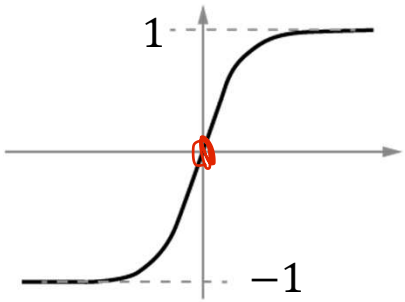
Aktivierungsfunktion $f(u)$ z.B.:

Sigmoid



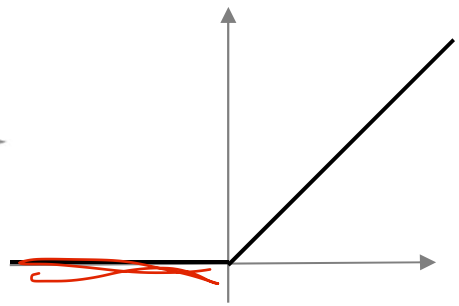
$$f(u) = \frac{1}{1 + e^{-u}}$$

TanH



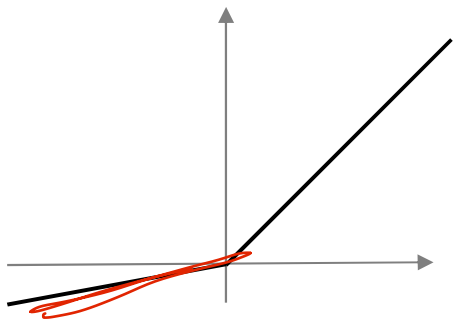
$$f(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$

ReLU



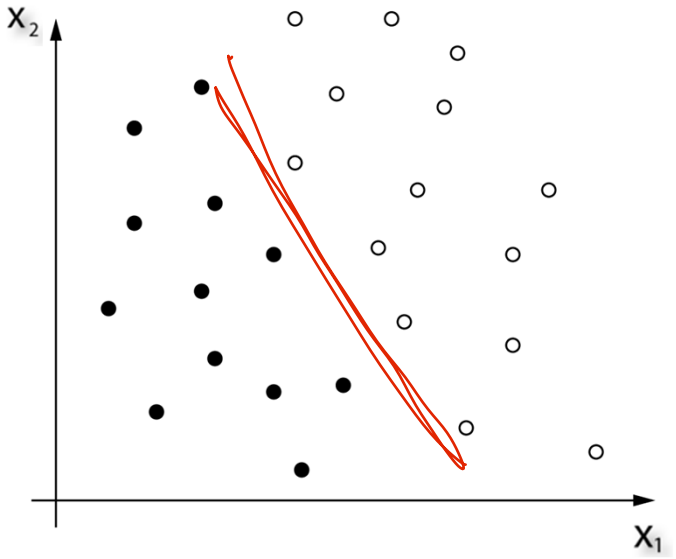
$$f(u) = \max(0, u)$$

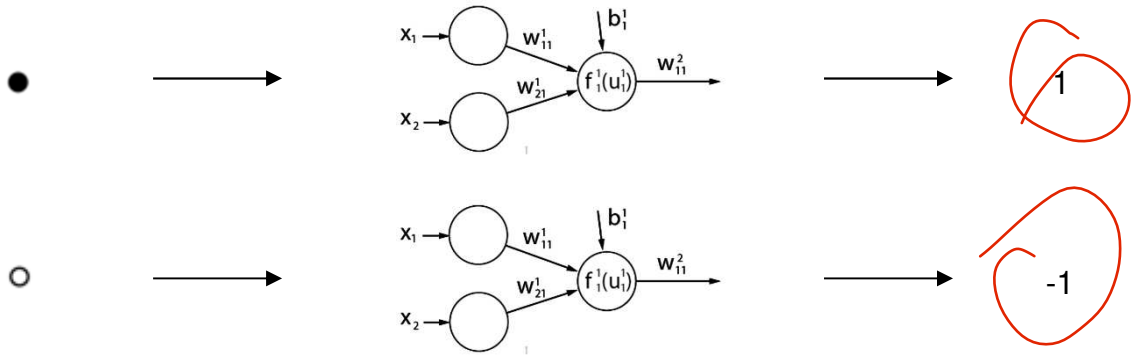
Leaky ReLU



$$f(u) = \max(0.01 * u, u)$$

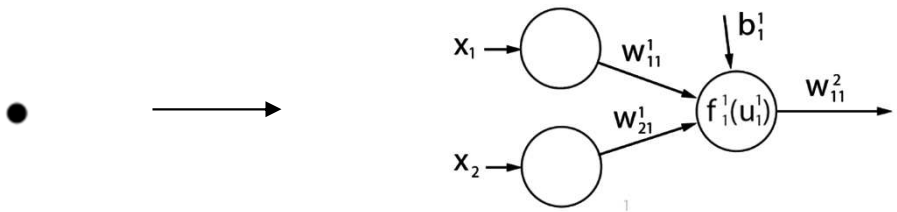
Single Neuron Classifier



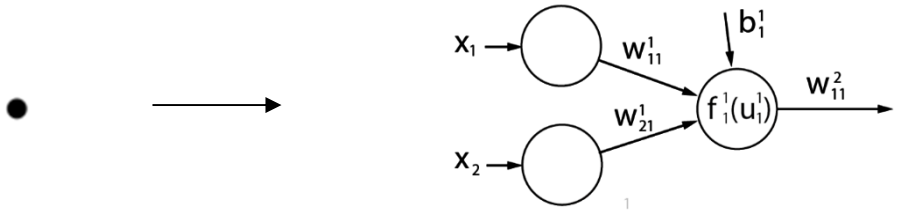


Wie werden die Gewichte bestimmt?

Ziel:



Im Training:

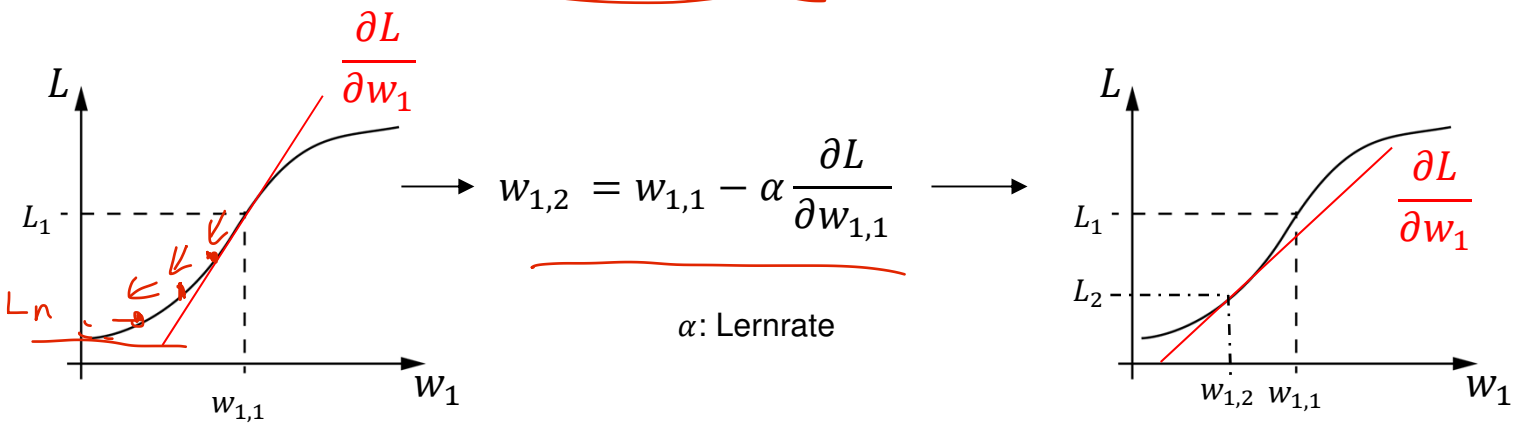


Fehler z.B.: $L = 0.9^2$

Wie müssen w_1, w_2, b geändert werden damit L minimal wird?

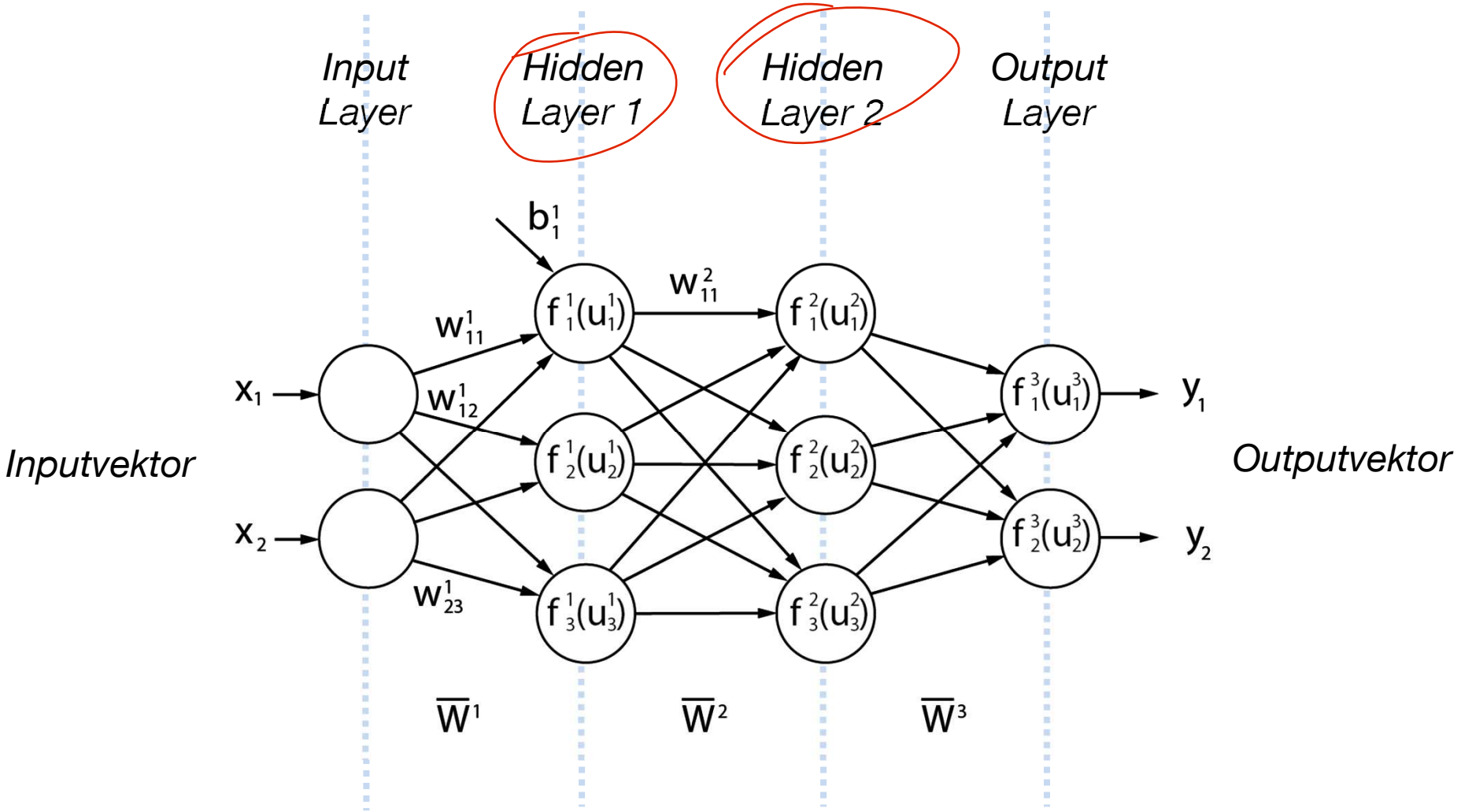
Wie werden die Gewichte bestimmt?

Lösung: lokale Ableitung / Gradient



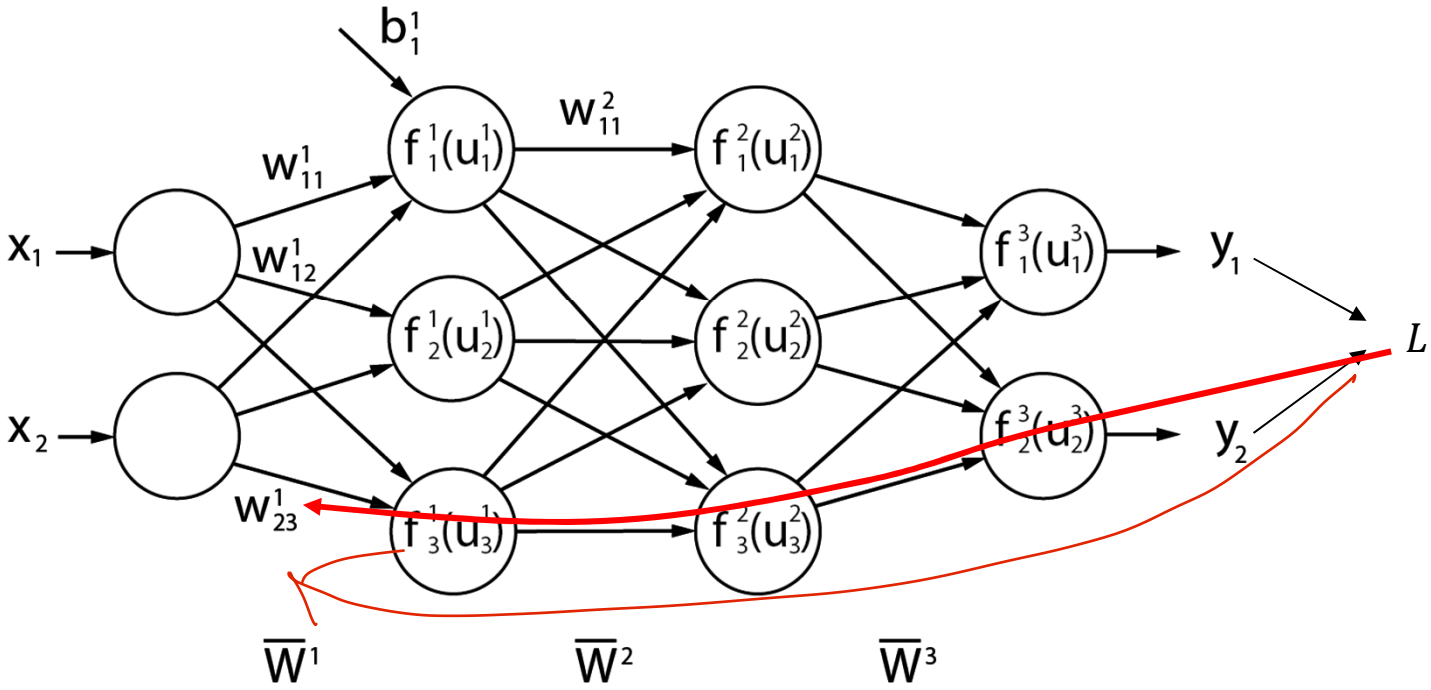
$$w_{1,3} = \cdots$$

Netzaufbau/Netzdefinition



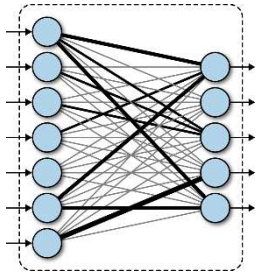
Backpropagation

… ist ein einfaches Verfahren zur Berechnung der lokalen Ableitungen in einem Netzwerk, mit dem Ziel die Änderungen der Gewichte so zu berechnen, dass der Fehler minimal wird.

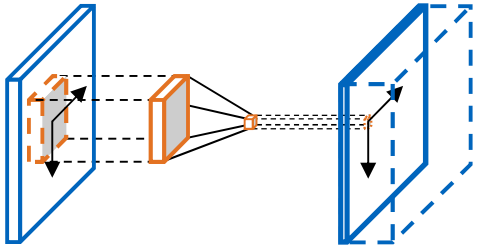


Netzwerkarten

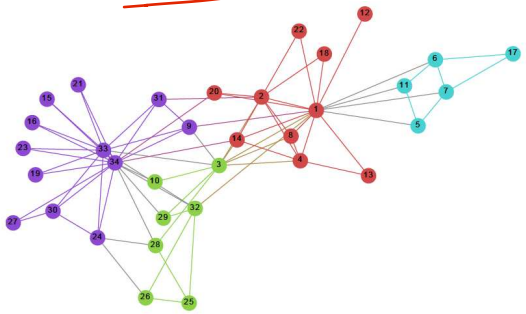
Fully Connected Network



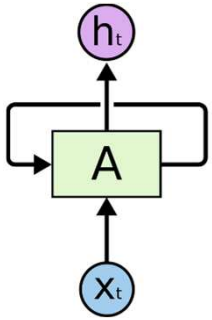
Convolutional Neural Network



Graph Neural Network



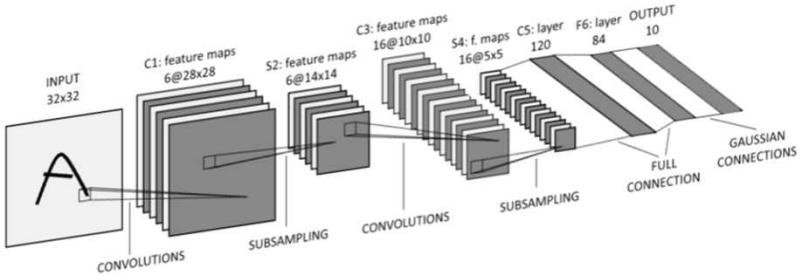
Recurrent Neural Network



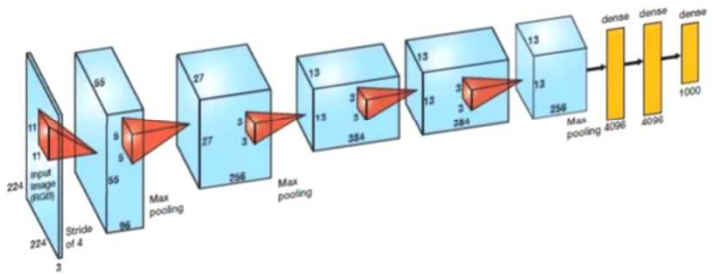
Netzwerkarchitekturen

CNN

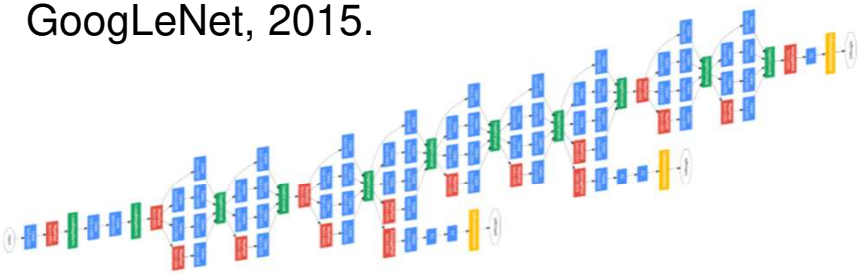
LeNet, 1989.



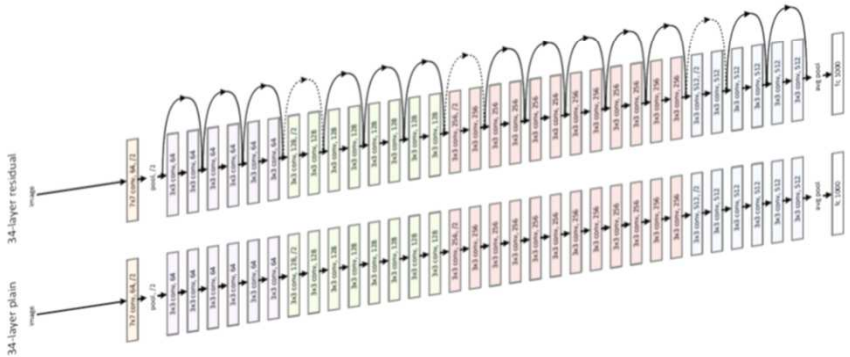
AlexNet, 2012.



GoogLeNet, 2015.

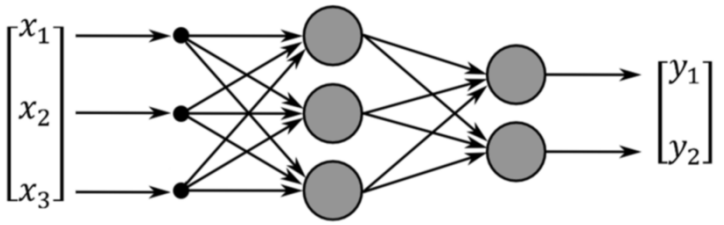


ResNet, 2015.

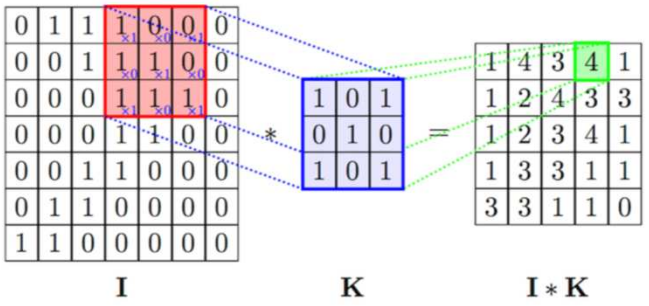


Layerarten (MLP & CNN)

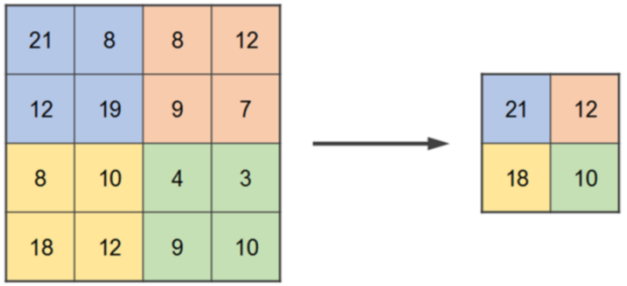
Fully Connected Layer



Convolutional Layer

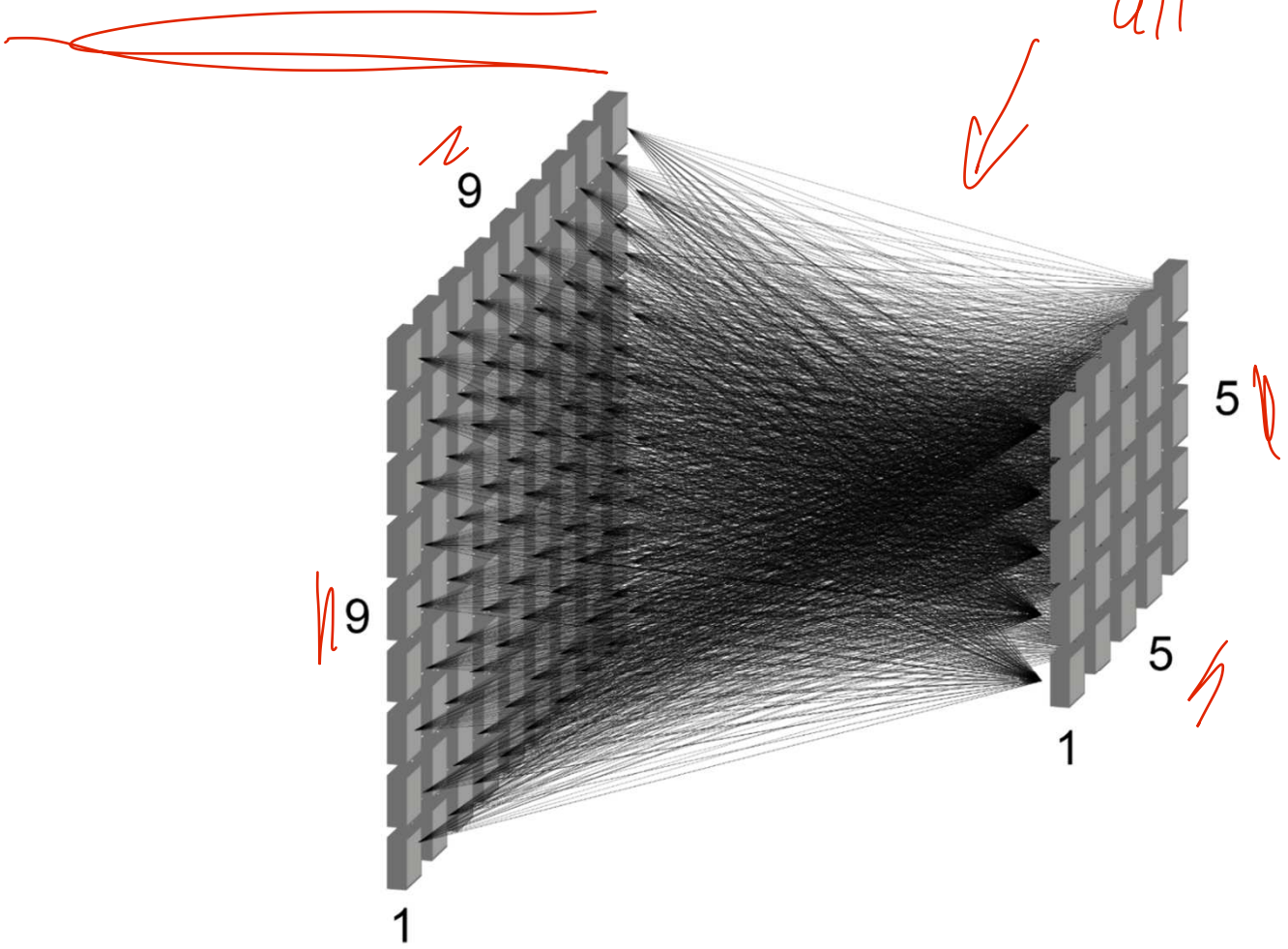


Pooling Layer

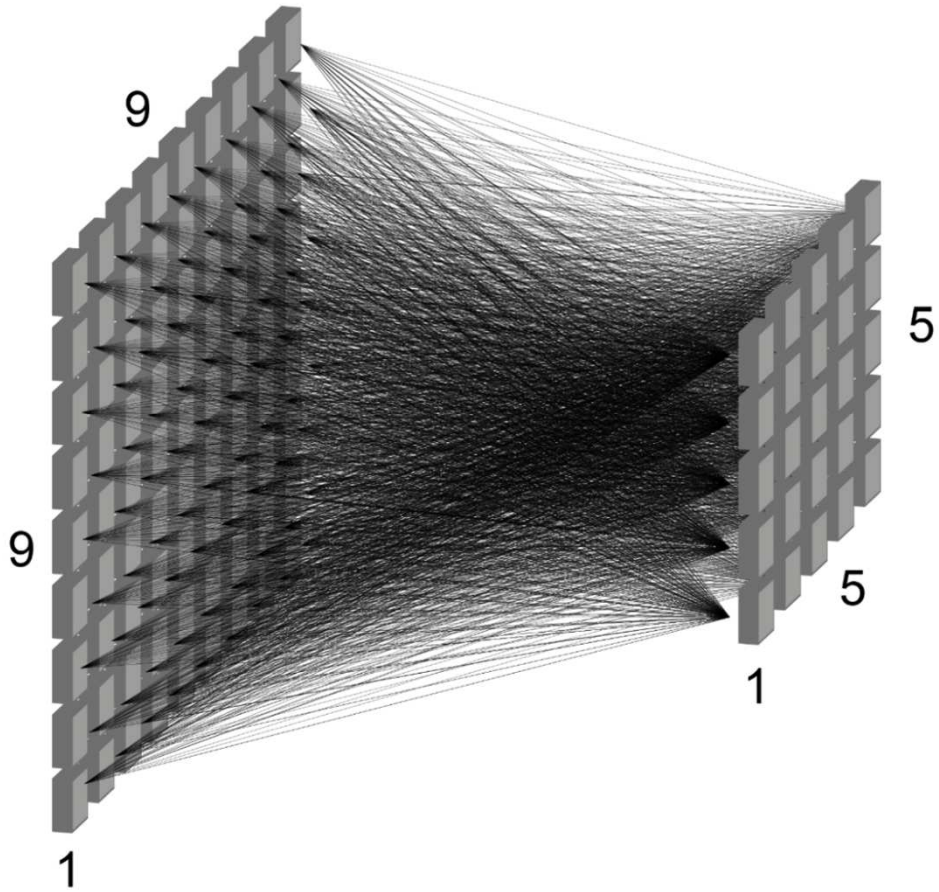


Wie sehen Computer Bilder

Fully Connected Layer



Fully Connected Layer



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

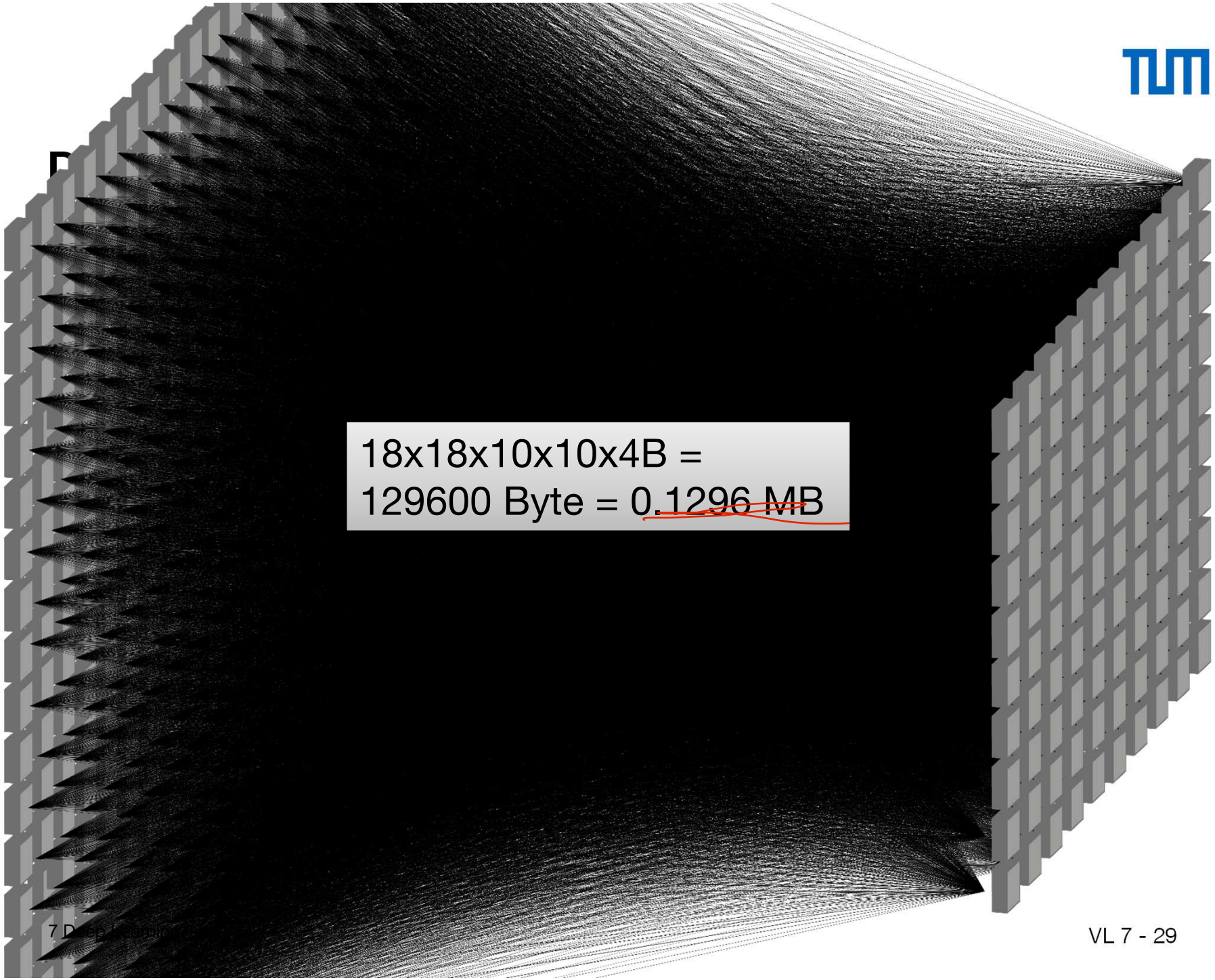
$$\vec{y} \in K^{1\times 25}$$

$$W \in K^{81\times 25}$$

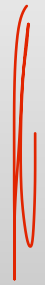
$$W = \begin{pmatrix} w_{11} & \cdots & w_{1,25} \\ \vdots & \ddots & \vdots \\ w_{81,1} & \cdots & w_{81,25} \end{pmatrix}$$

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

2025x4B =
8100 Byte = 8 kB

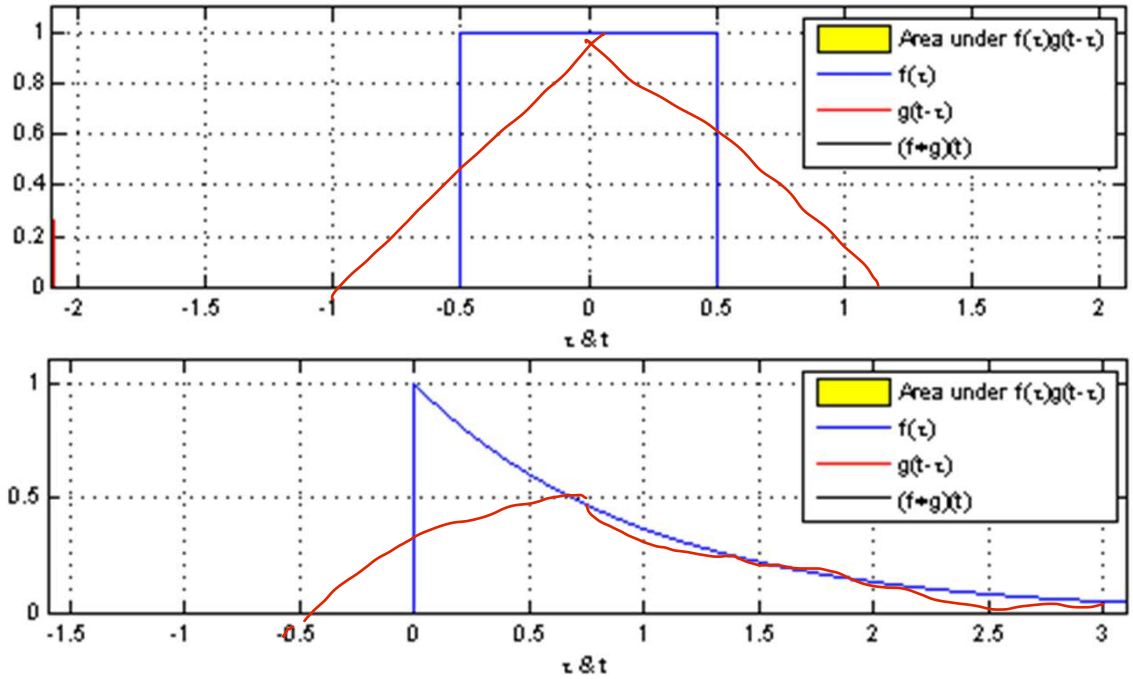

18x18x10x10x4B =
129600 Byte = 0.1296 MB

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



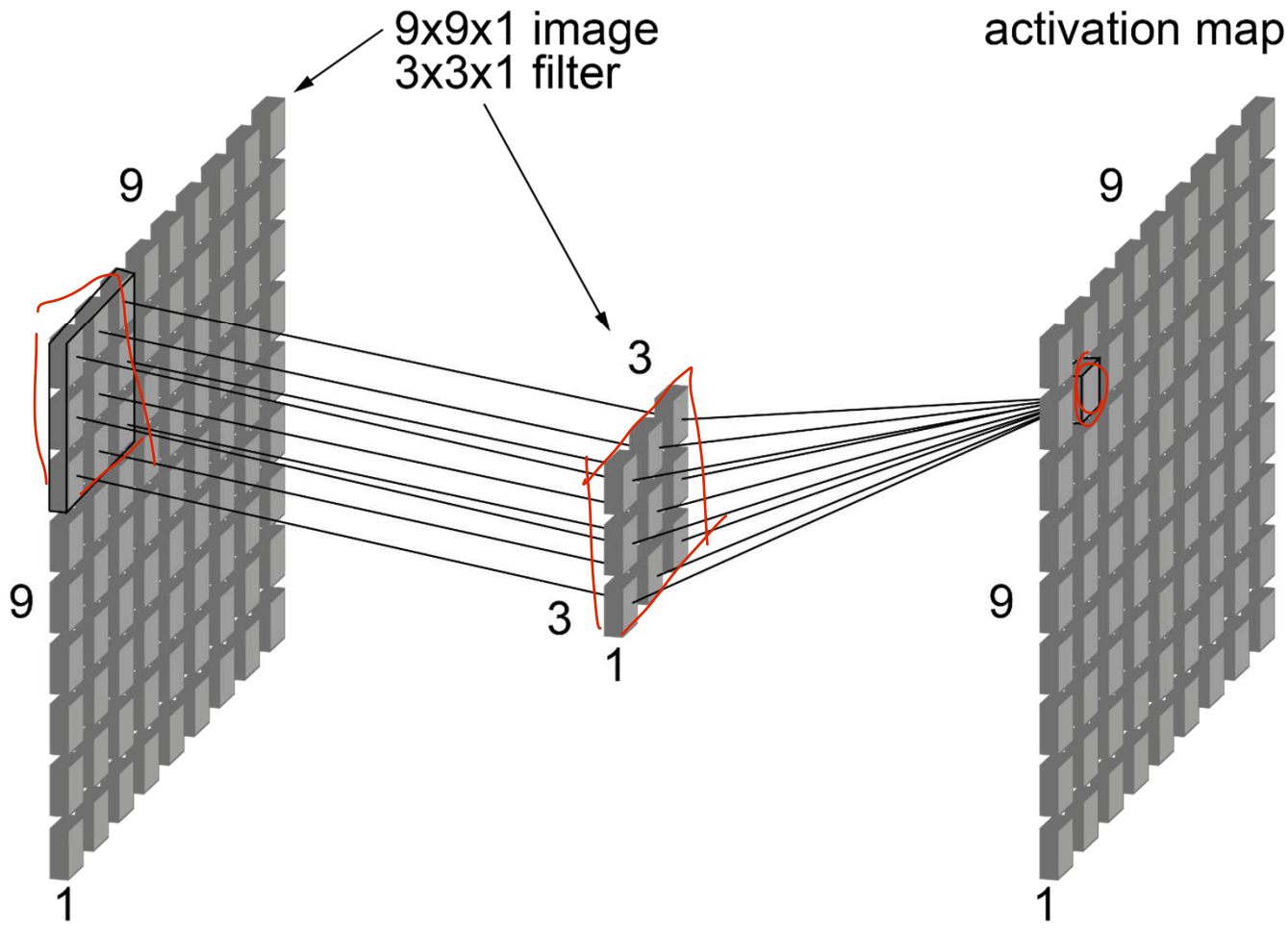
Convolution Layer
in der Mathematik und Regelungstechnik

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

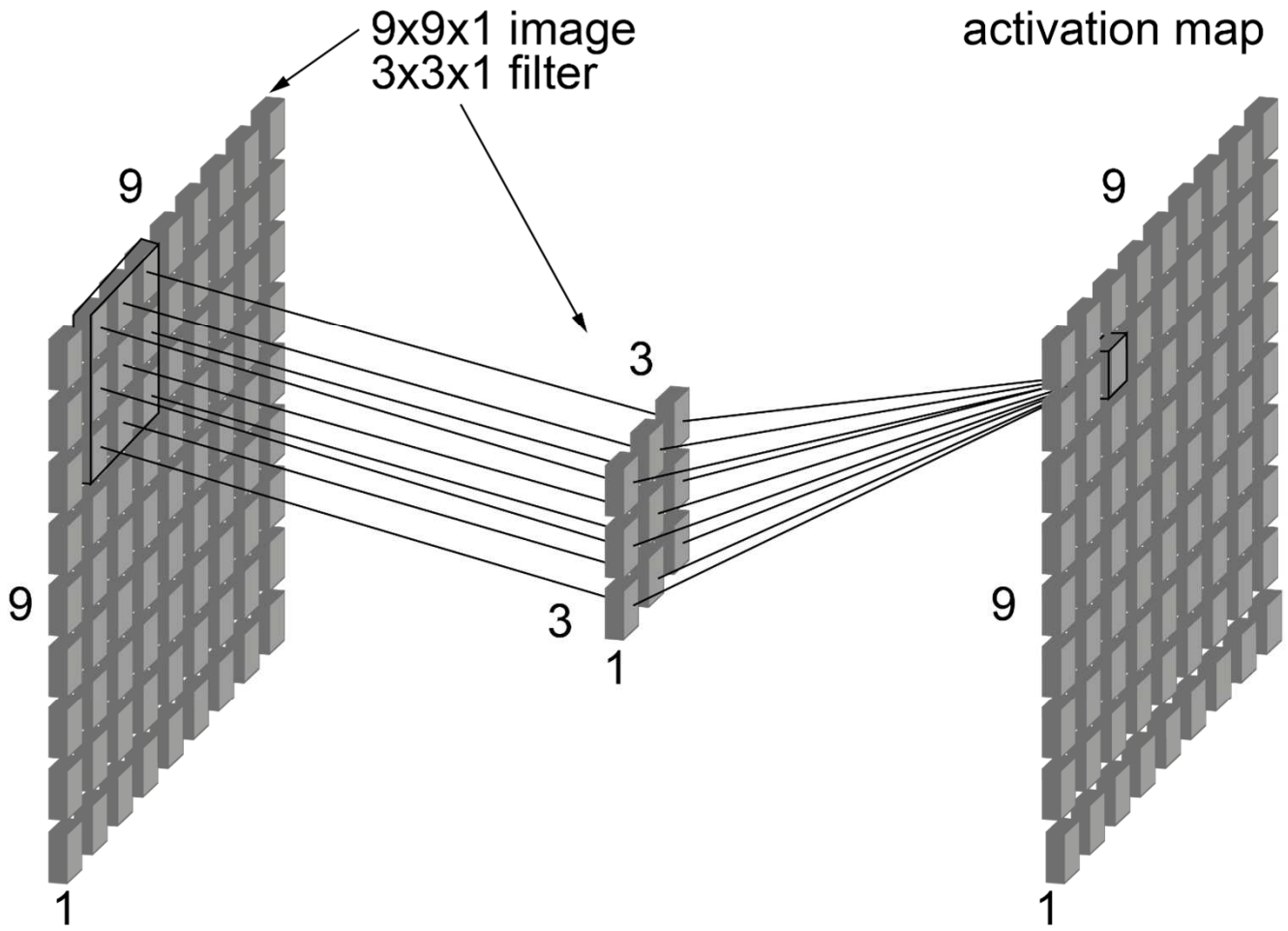


$$f(x,y) * g(x,y) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f(n_1,n_2) \cdot g(x - n_1, y - n_2)$$

Convolution Layer

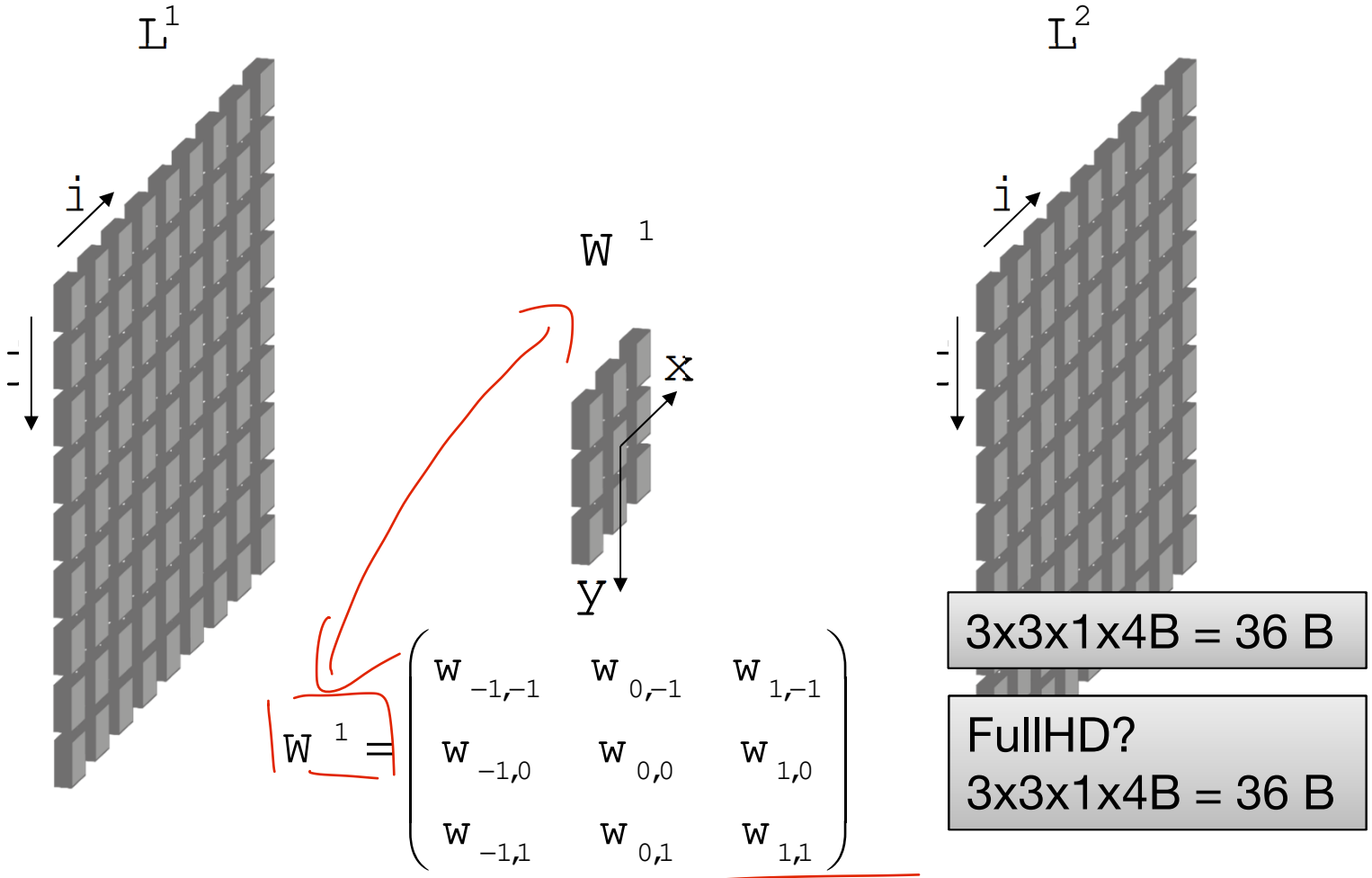


Convolution Layer

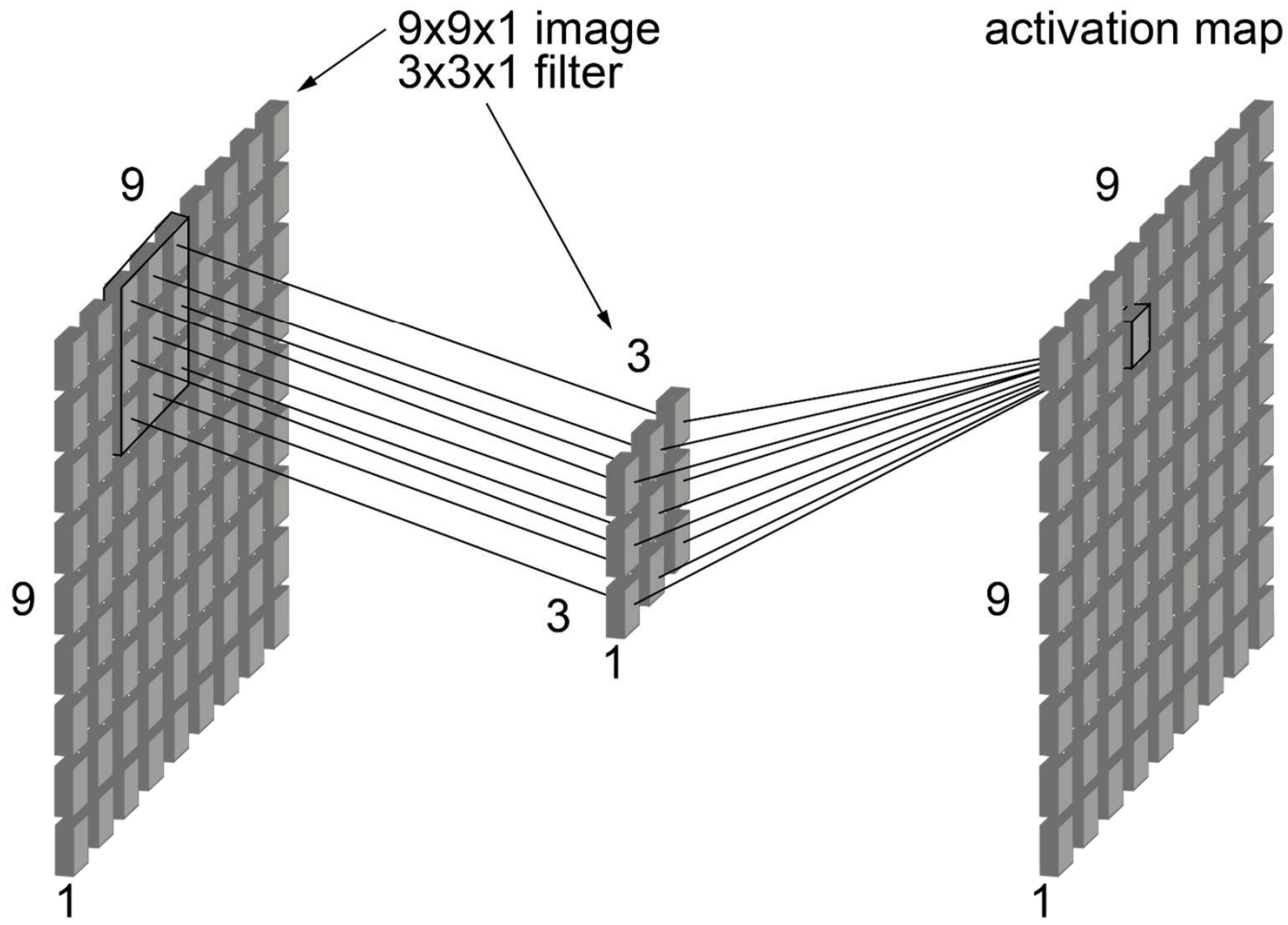


Convolution Layer

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

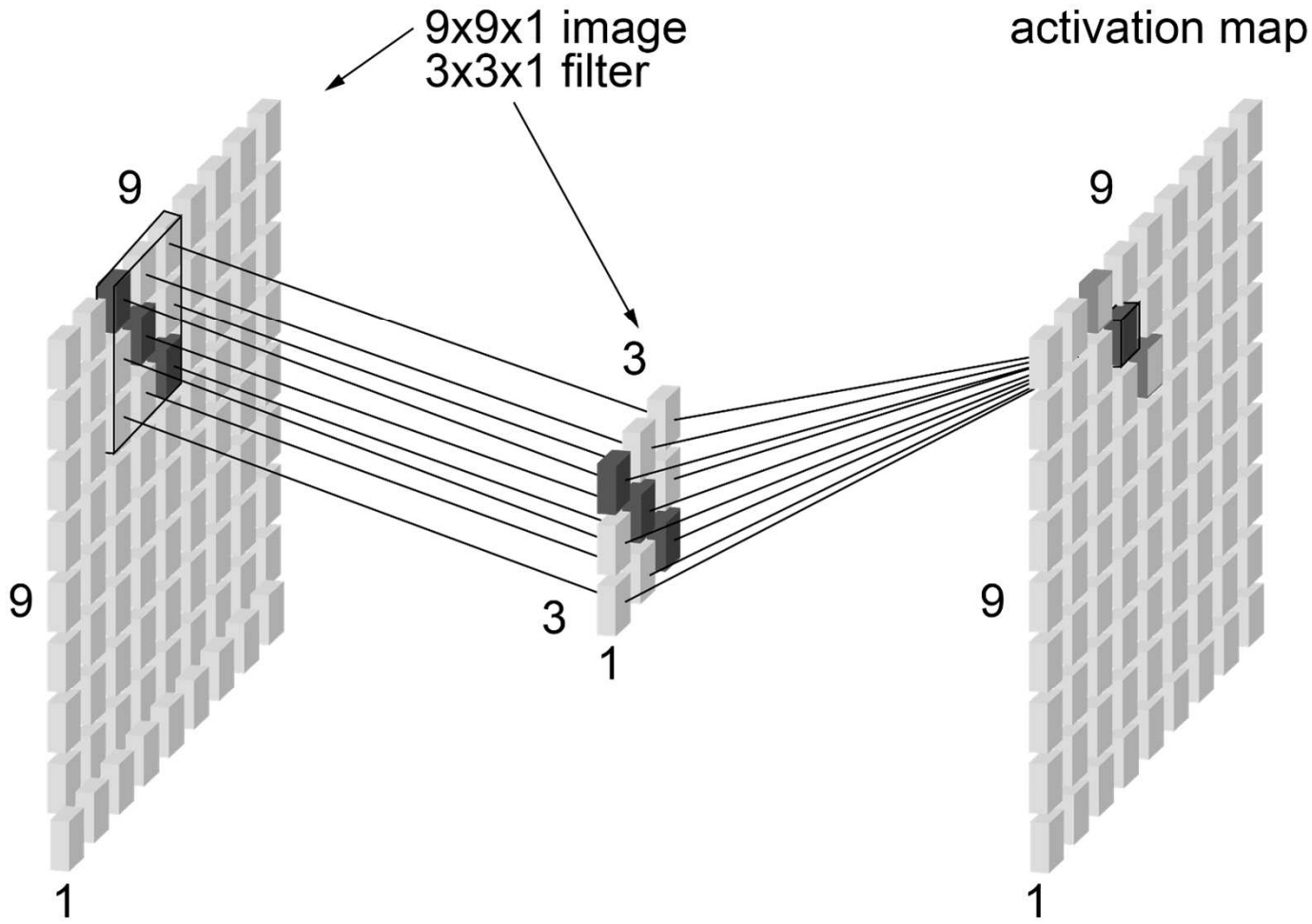


Convolution Layer

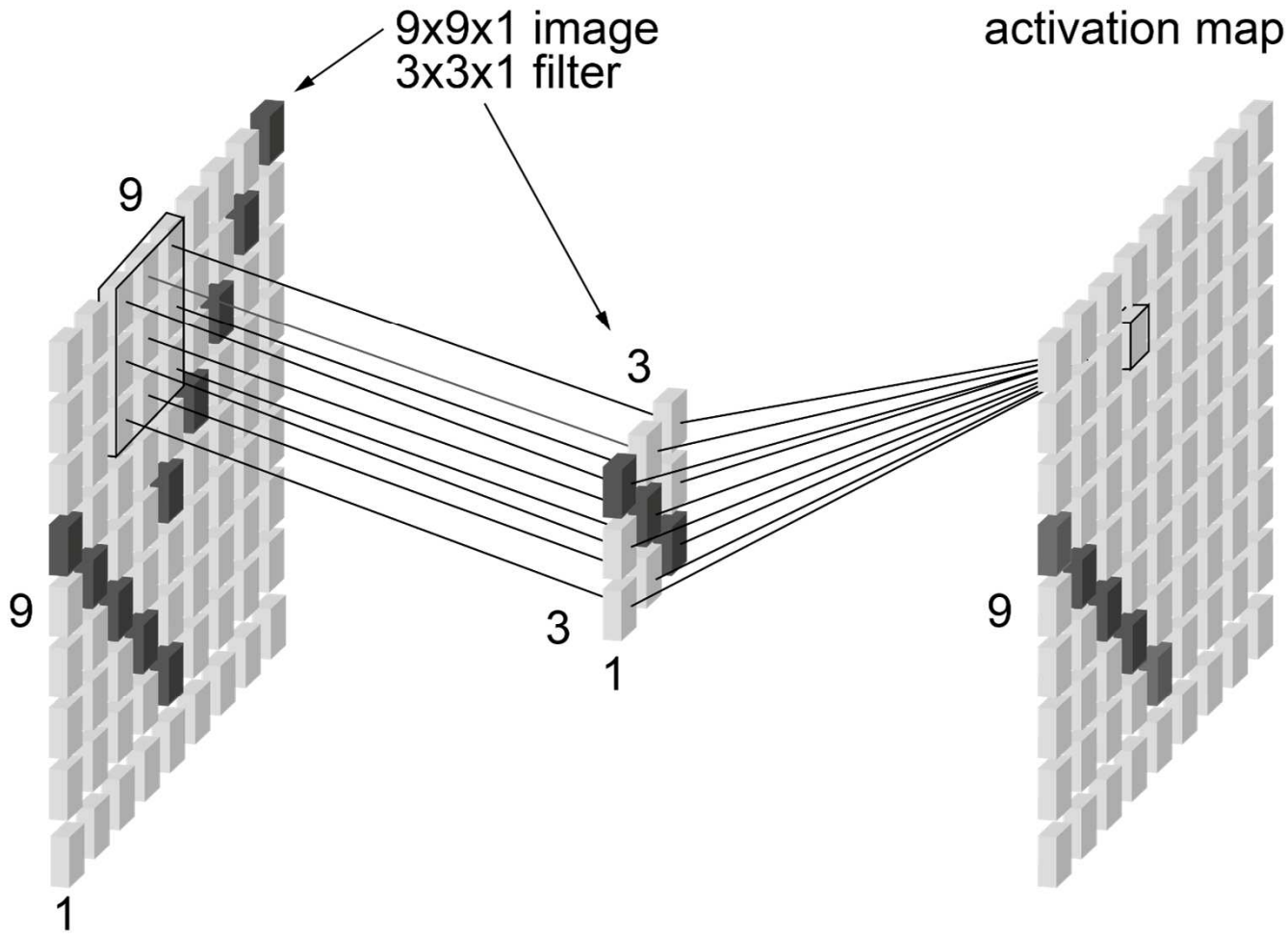


Convolution Layer

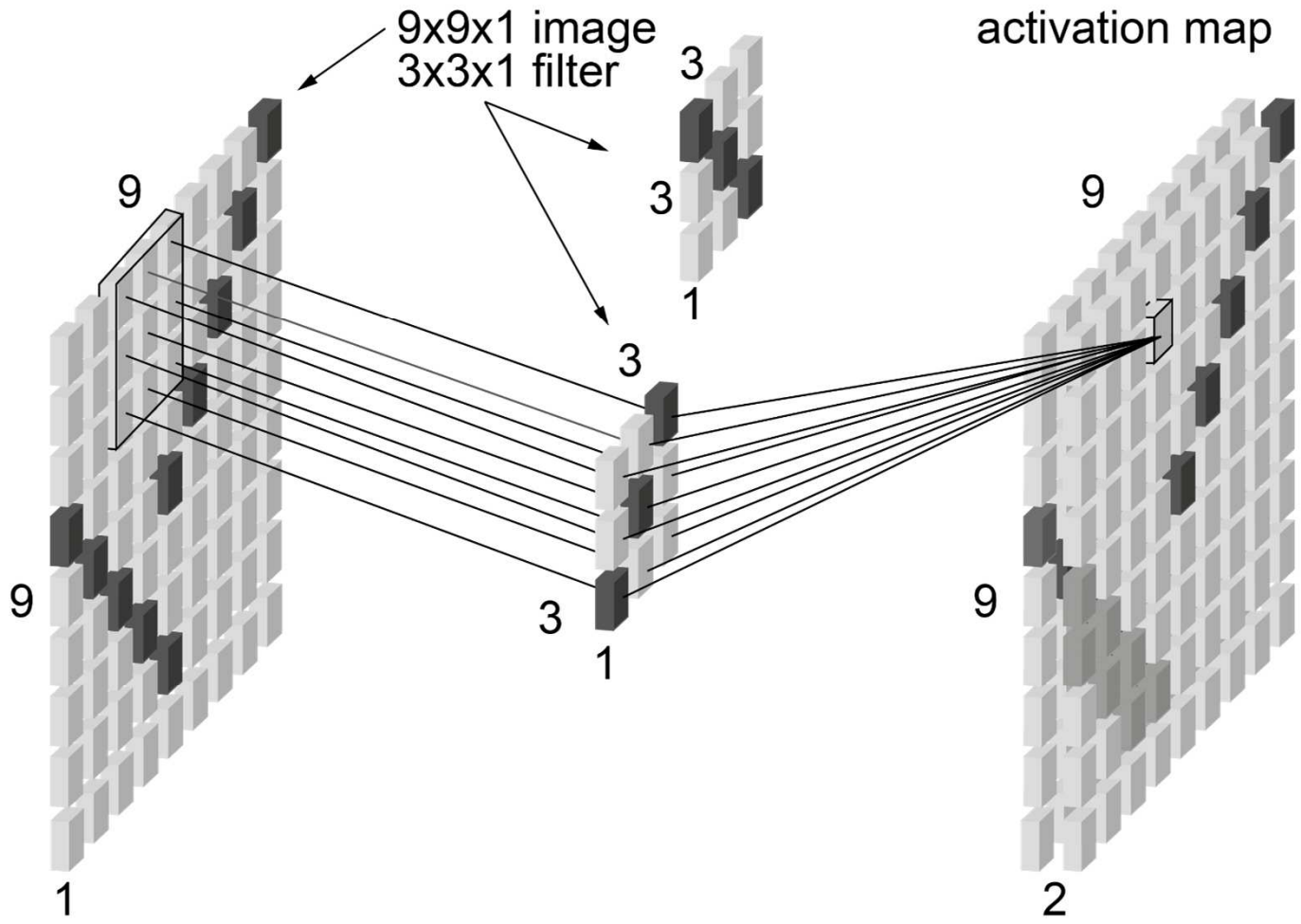
$$L^2_{i,j} = f\left(\sum_{x=-1}^{1}\sum_{y=-1}^{1} L^1_{x+i,y+j} \cdot w_{x,y} + b_{i,j}\right)$$



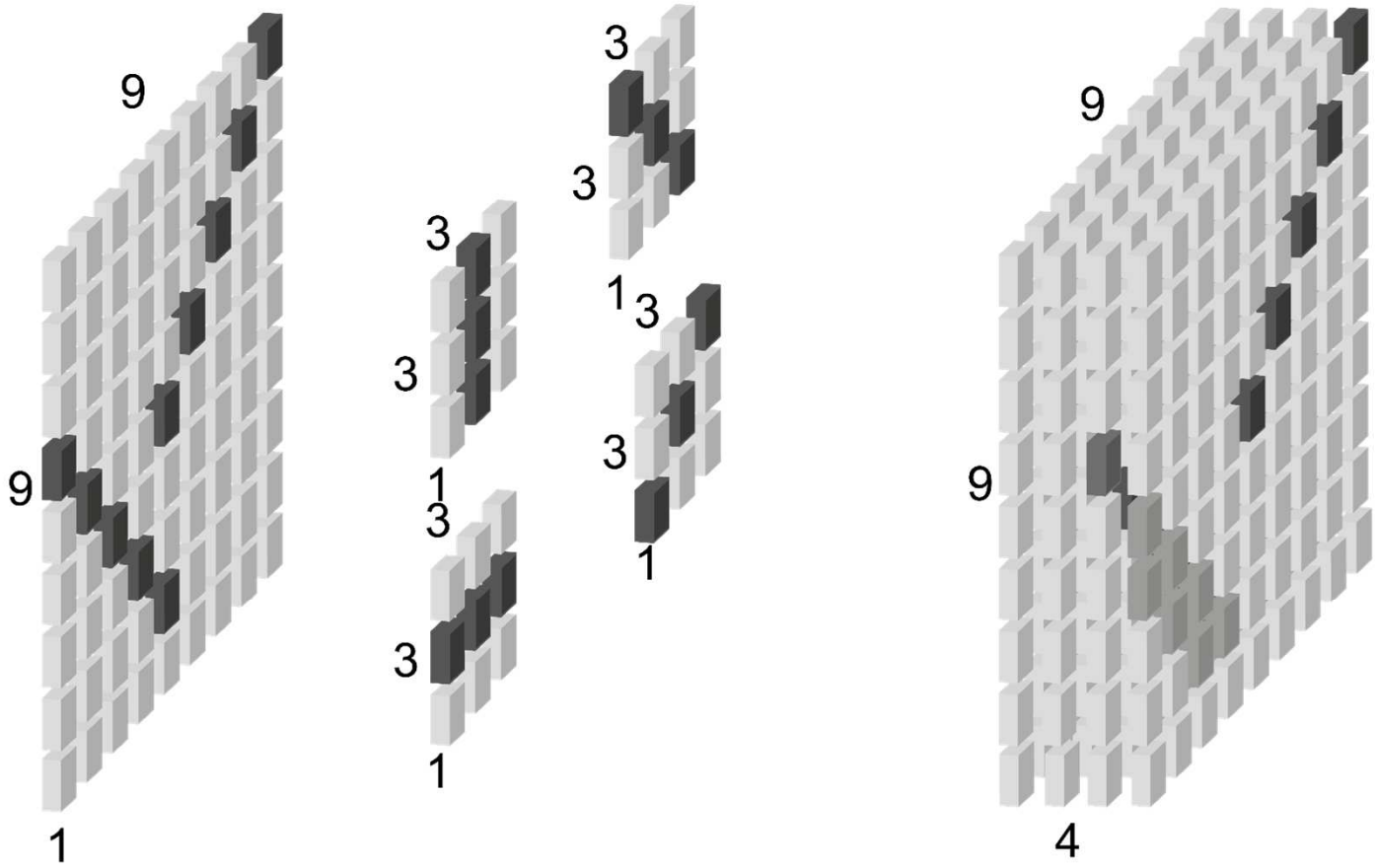
Convolution Layer



Convolution Layer

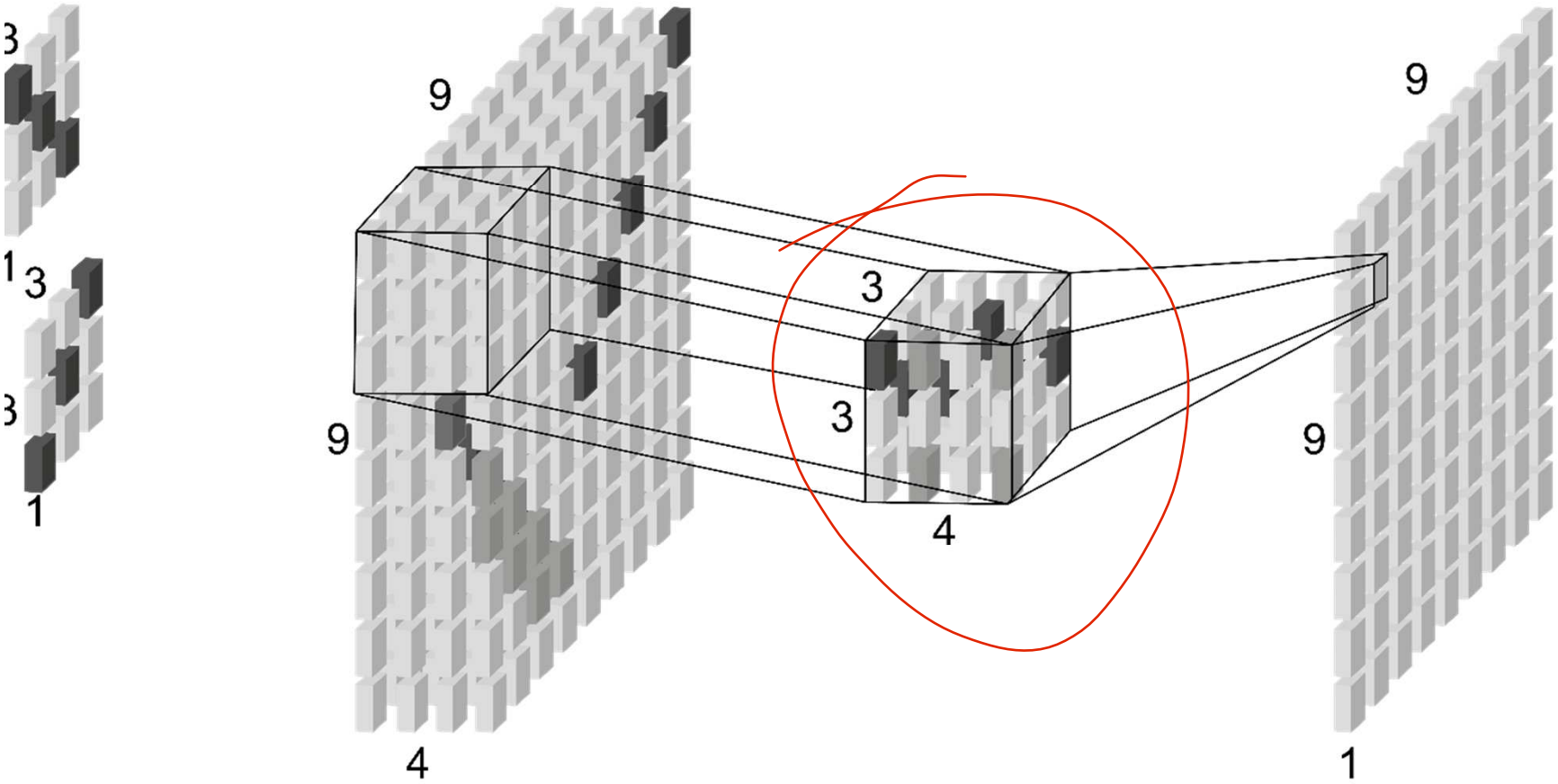


Convolution Layer



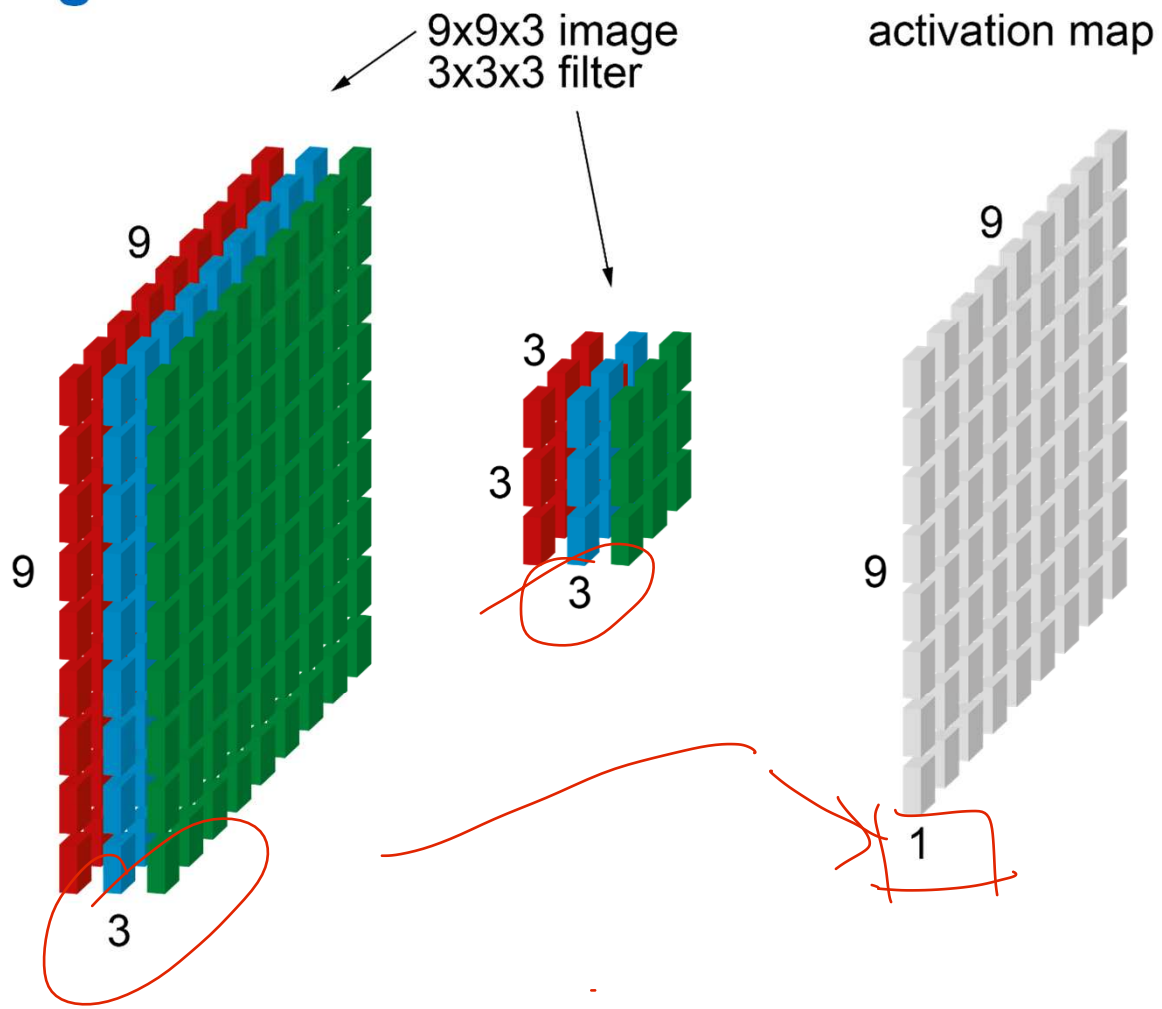
Convolution Layer

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

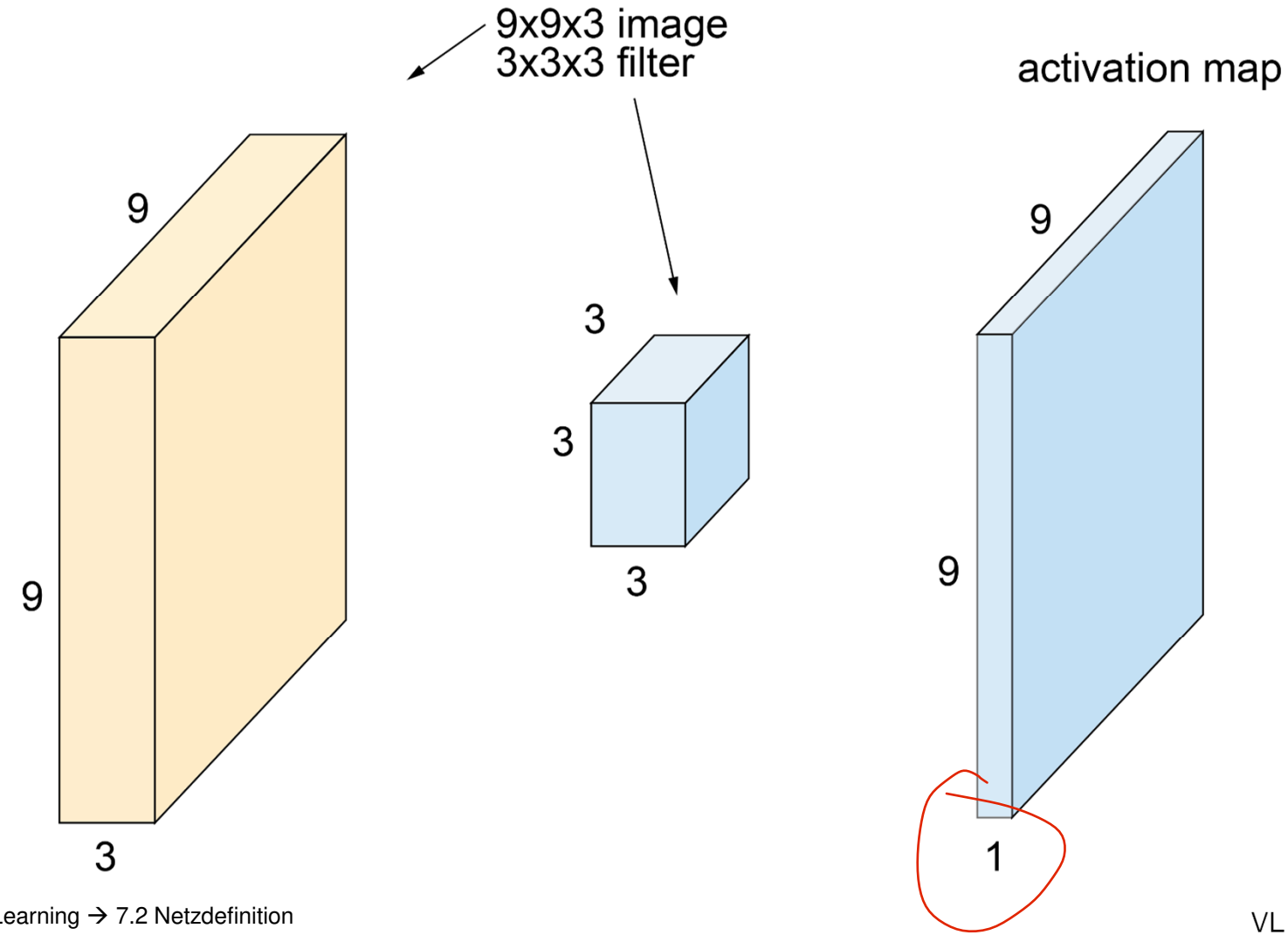


Convolution Layer

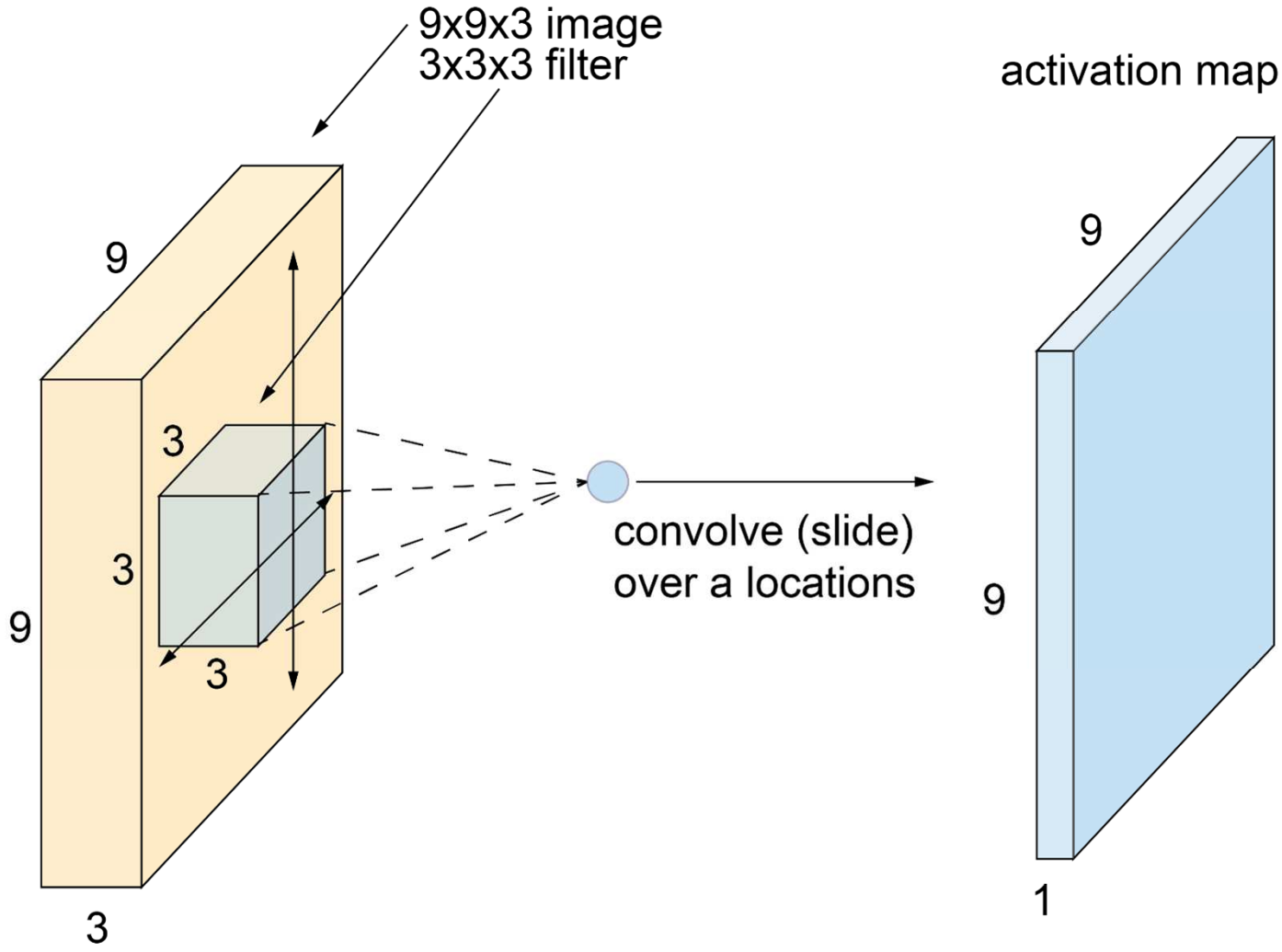
RGB Image



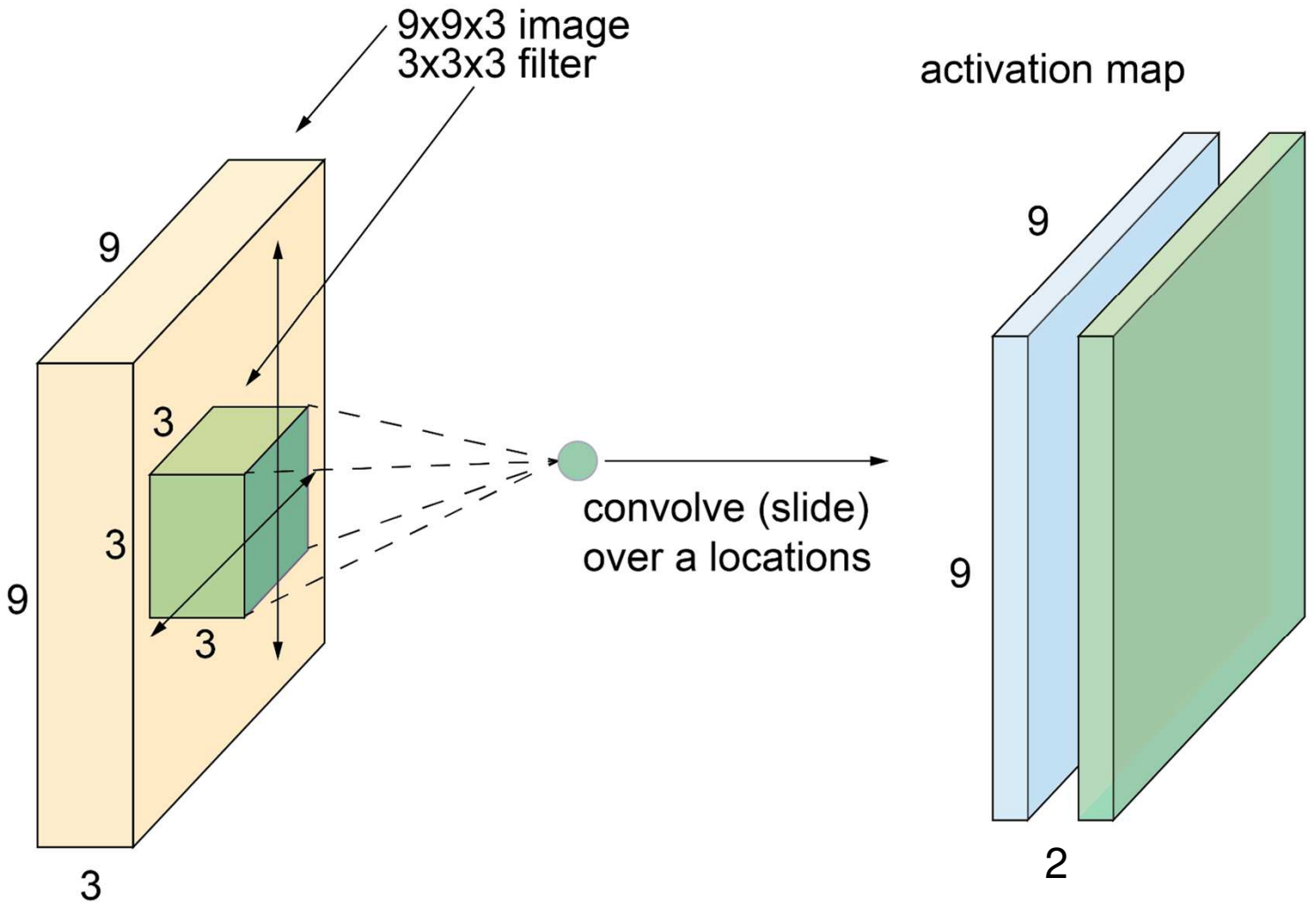
Convolution Layer



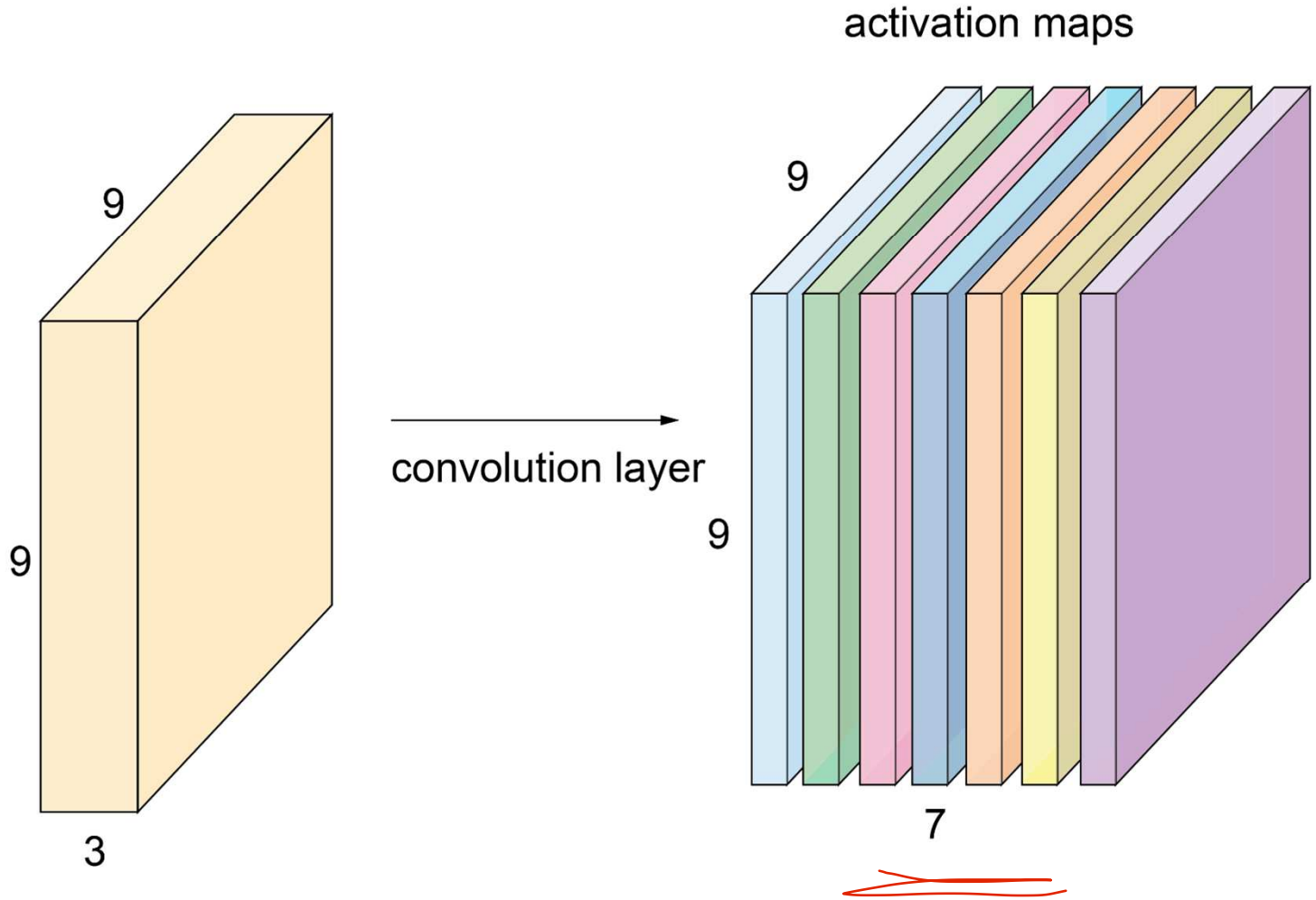
Convolution Layer



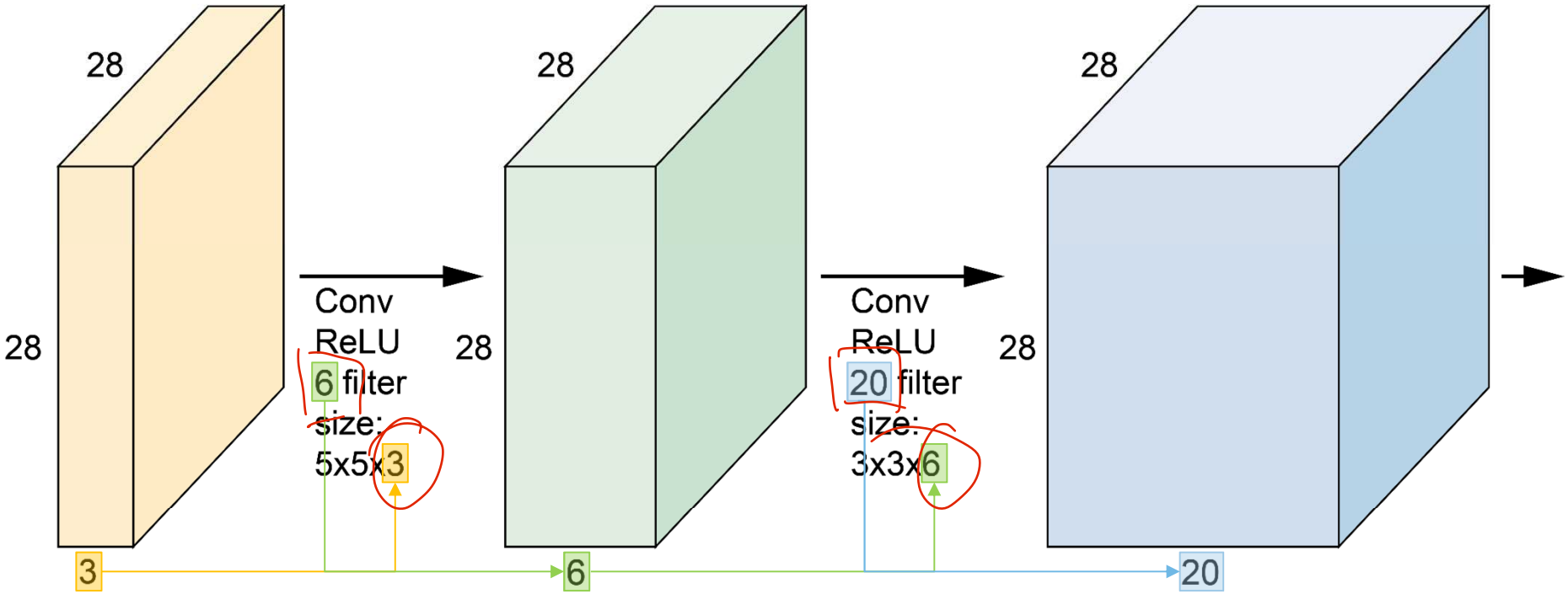
Convolution Layer



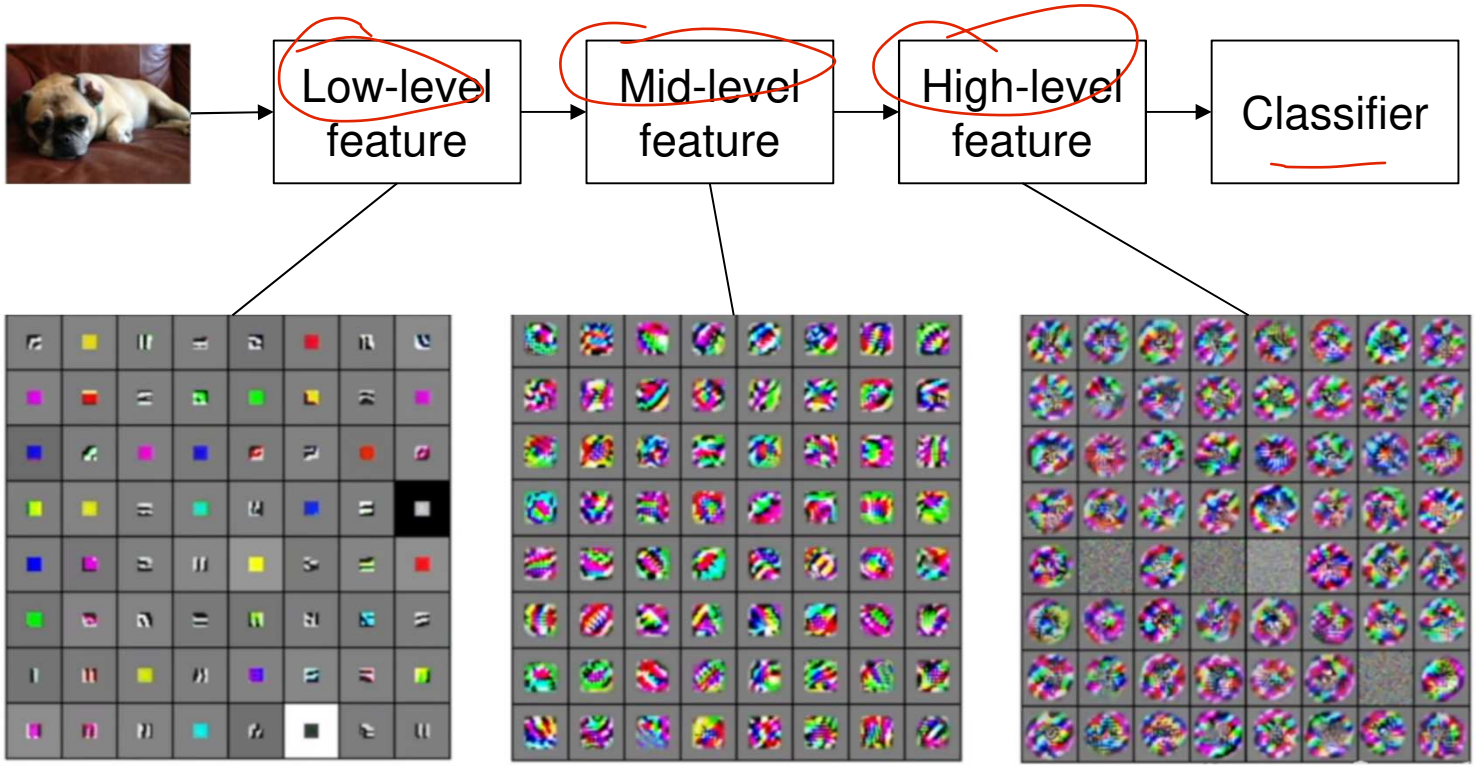
Convolution Layer



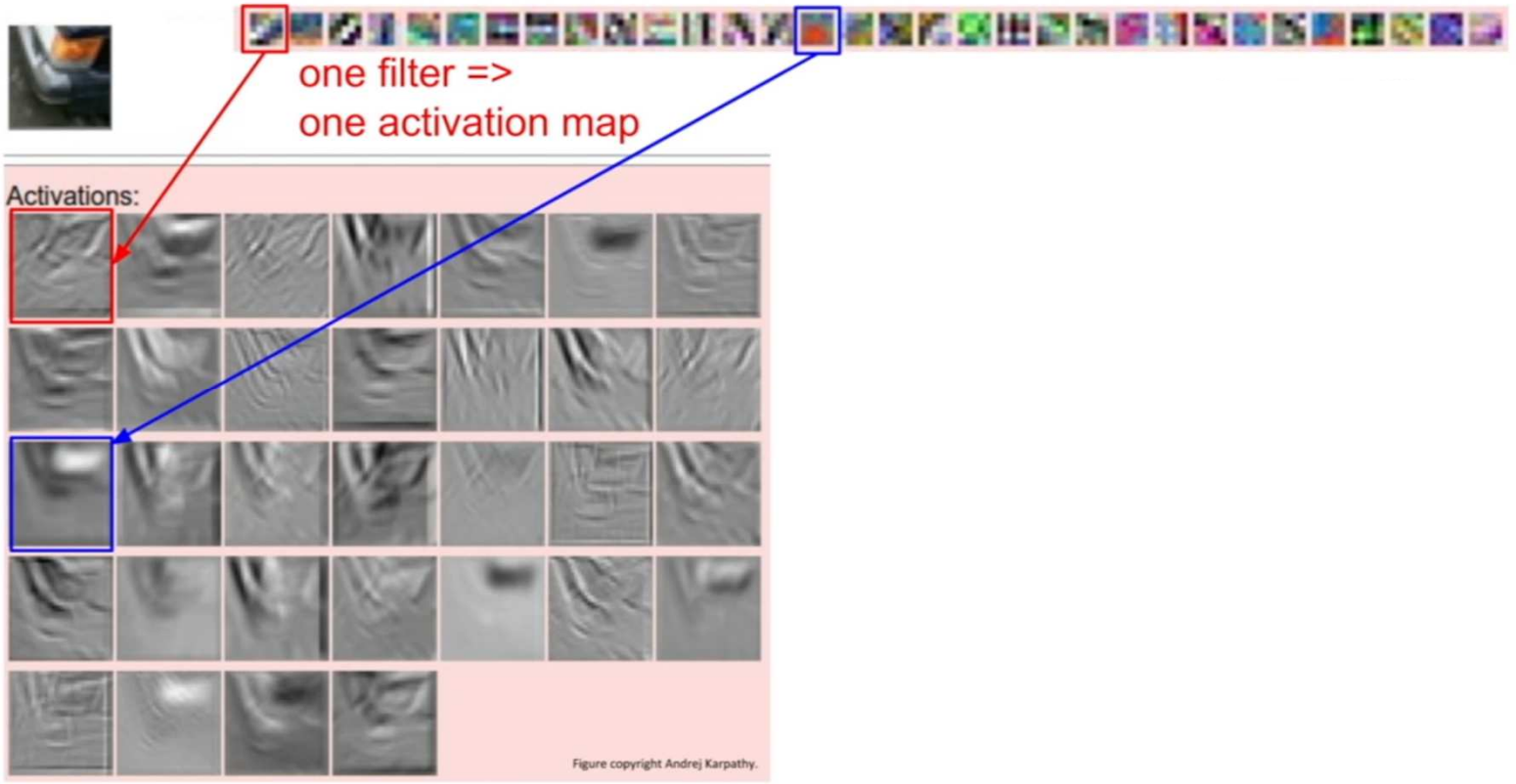
Convolution Layer



Convolution Layer
Filter Visualisation

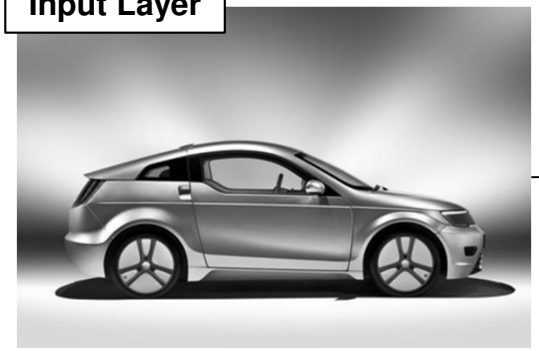


Convolution Layer
Activation Map Visualisation

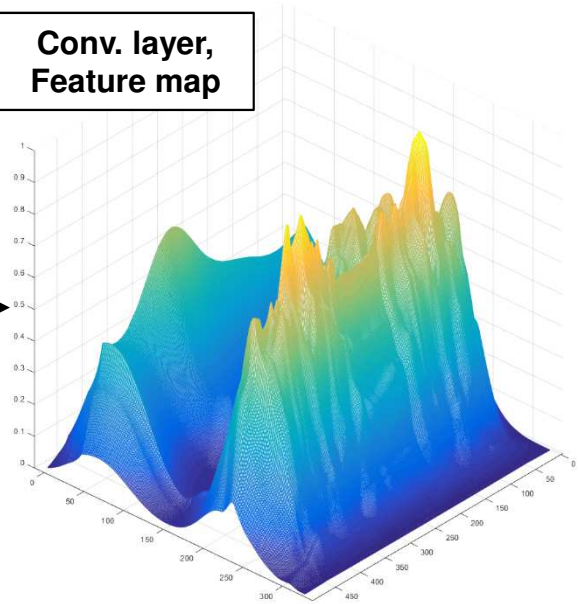


Layerarten: Convolution / Filter

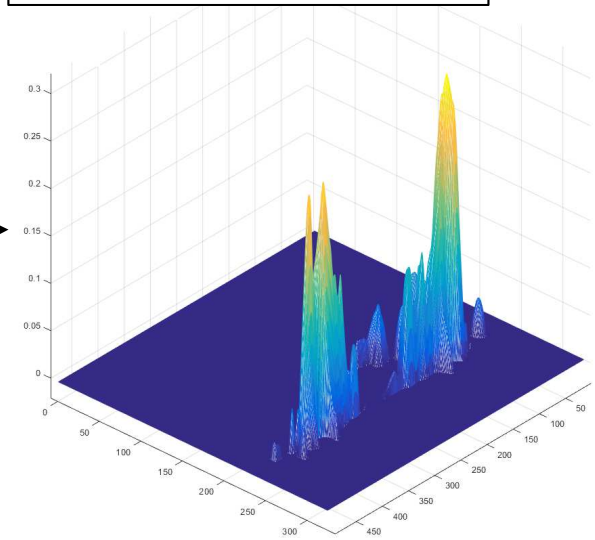
Input Layer



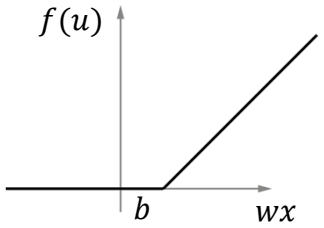
Conv. layer, Feature map



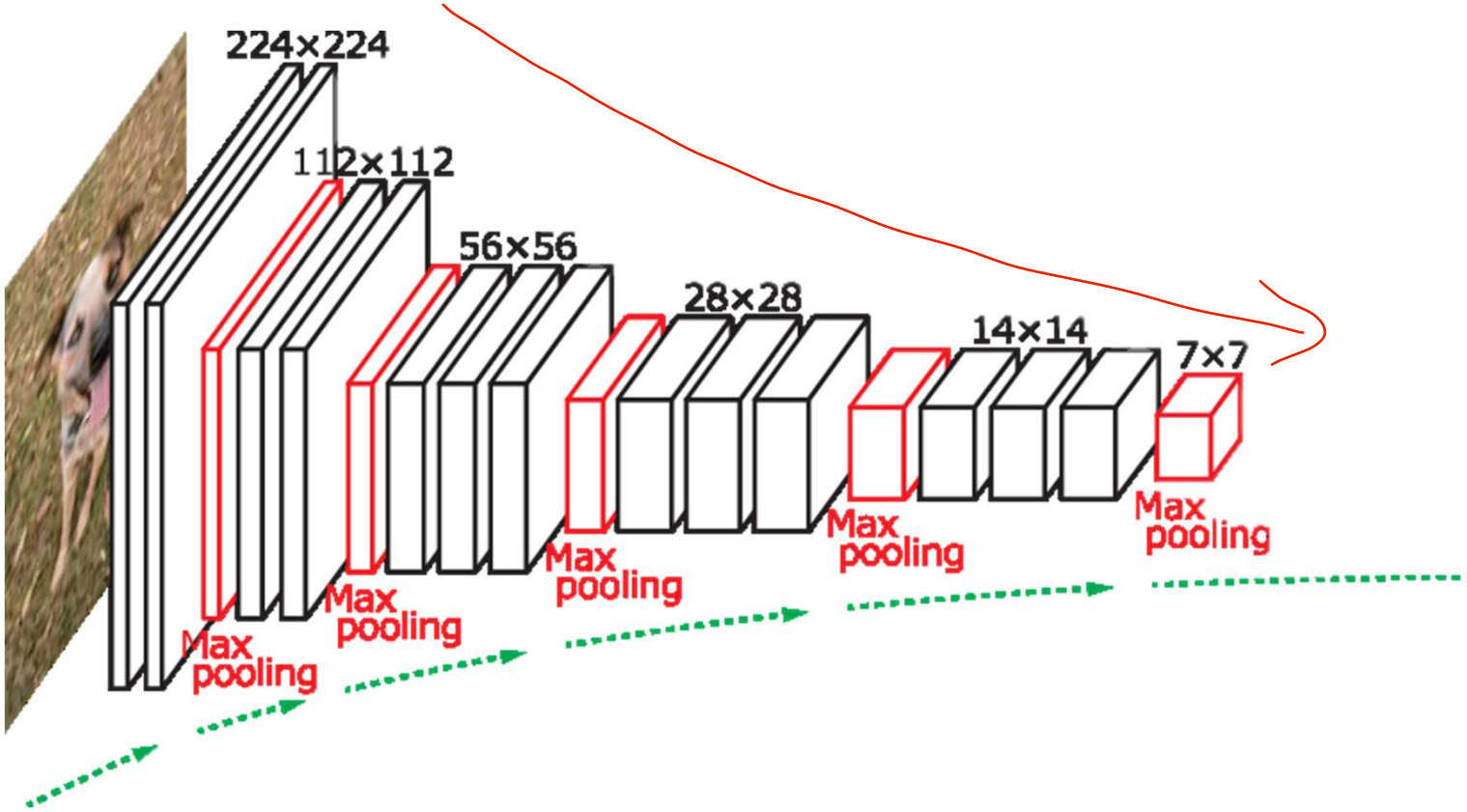
Relu Aktivierungfuntkion b=-0.7



Feature 1

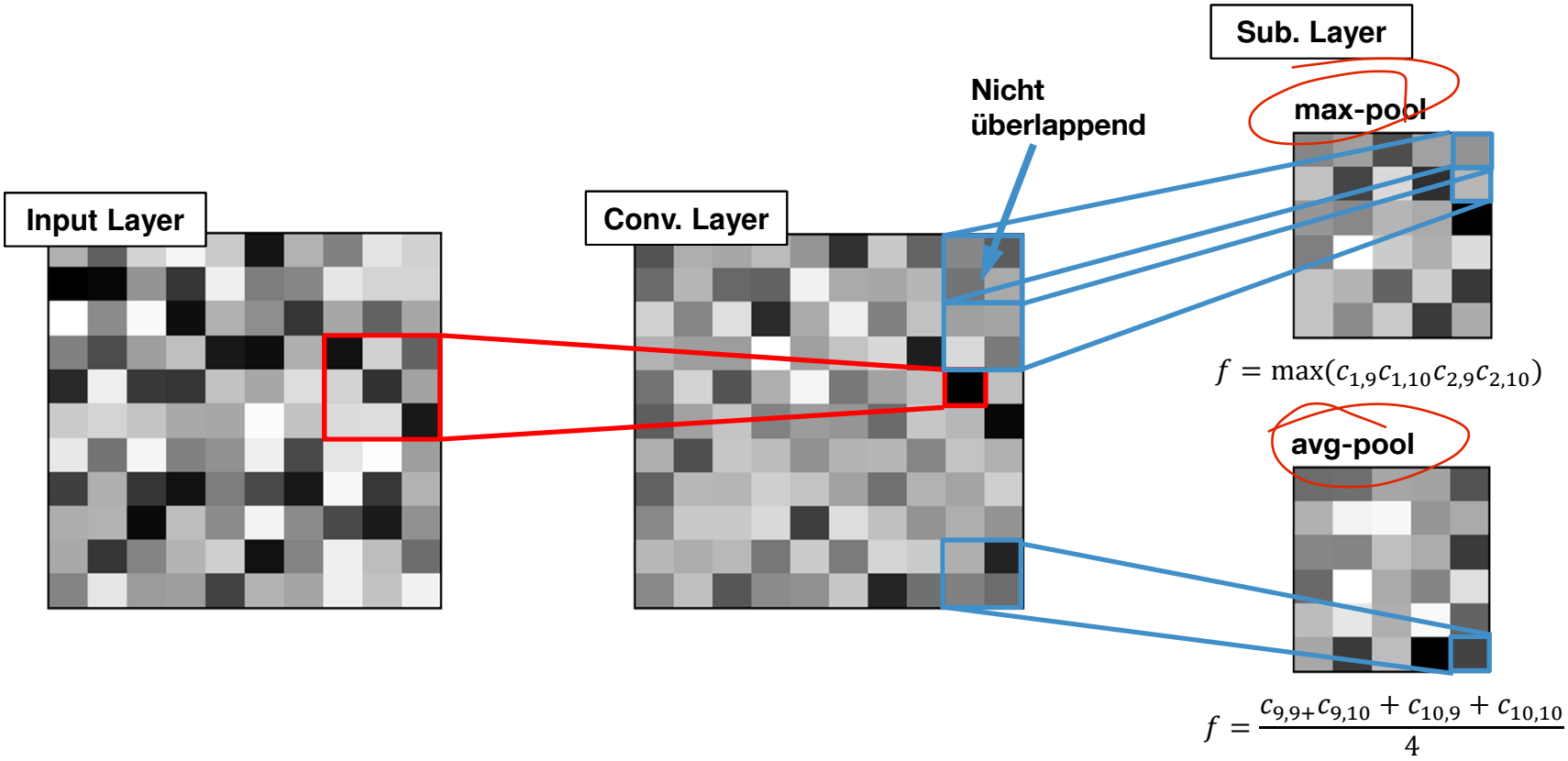


Netzbeispiel – Visual Geometry Group Net (Oxford)

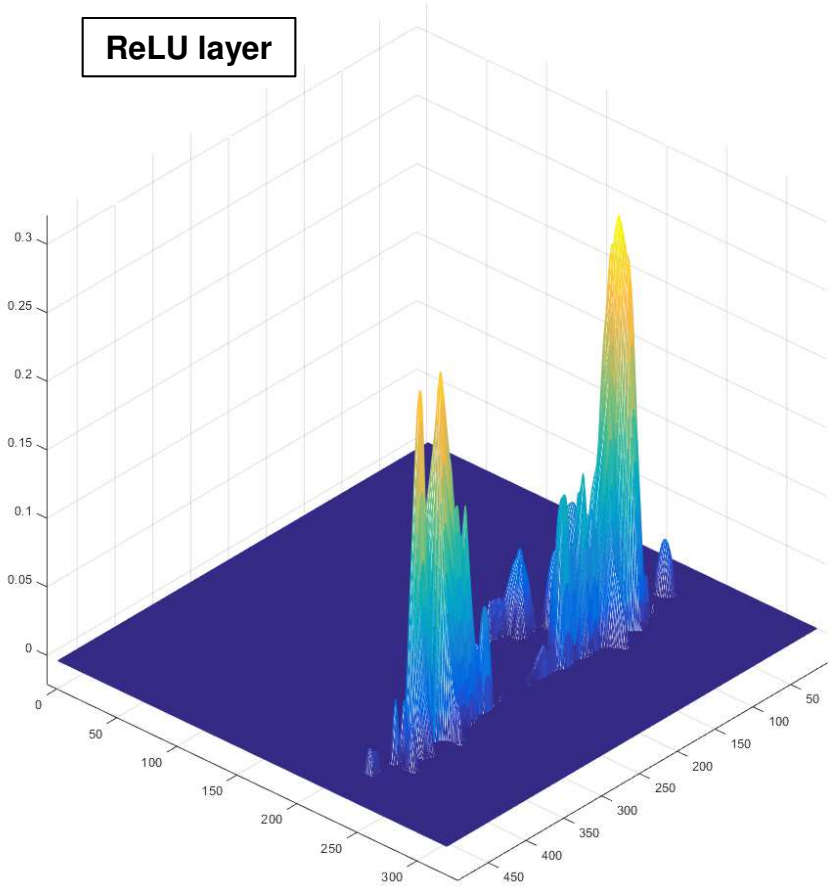


Modeldefinition: 20kb
Trainiertes Modell: 961Mb

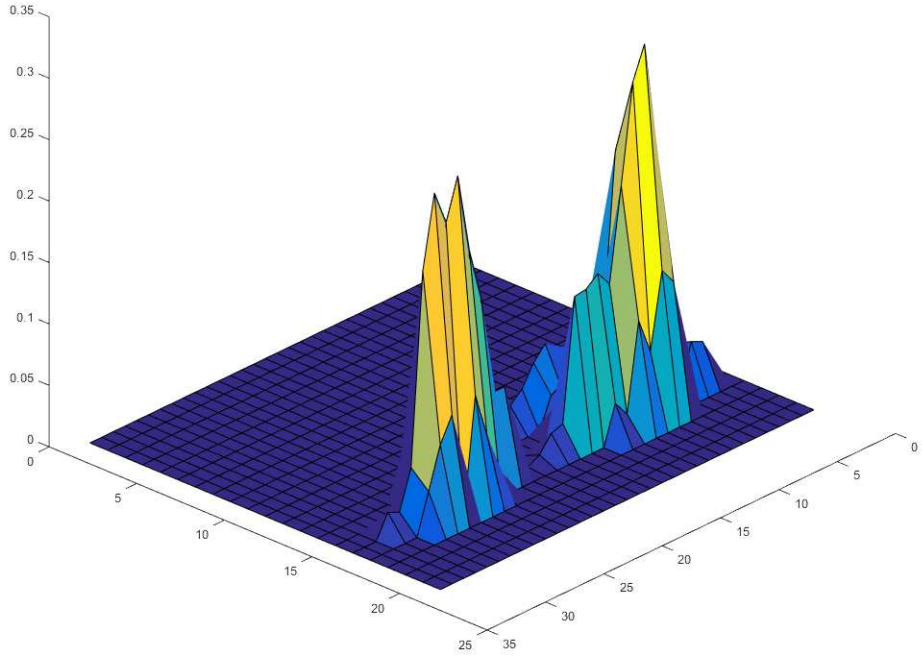
Layerarten: Pooling / subsampling



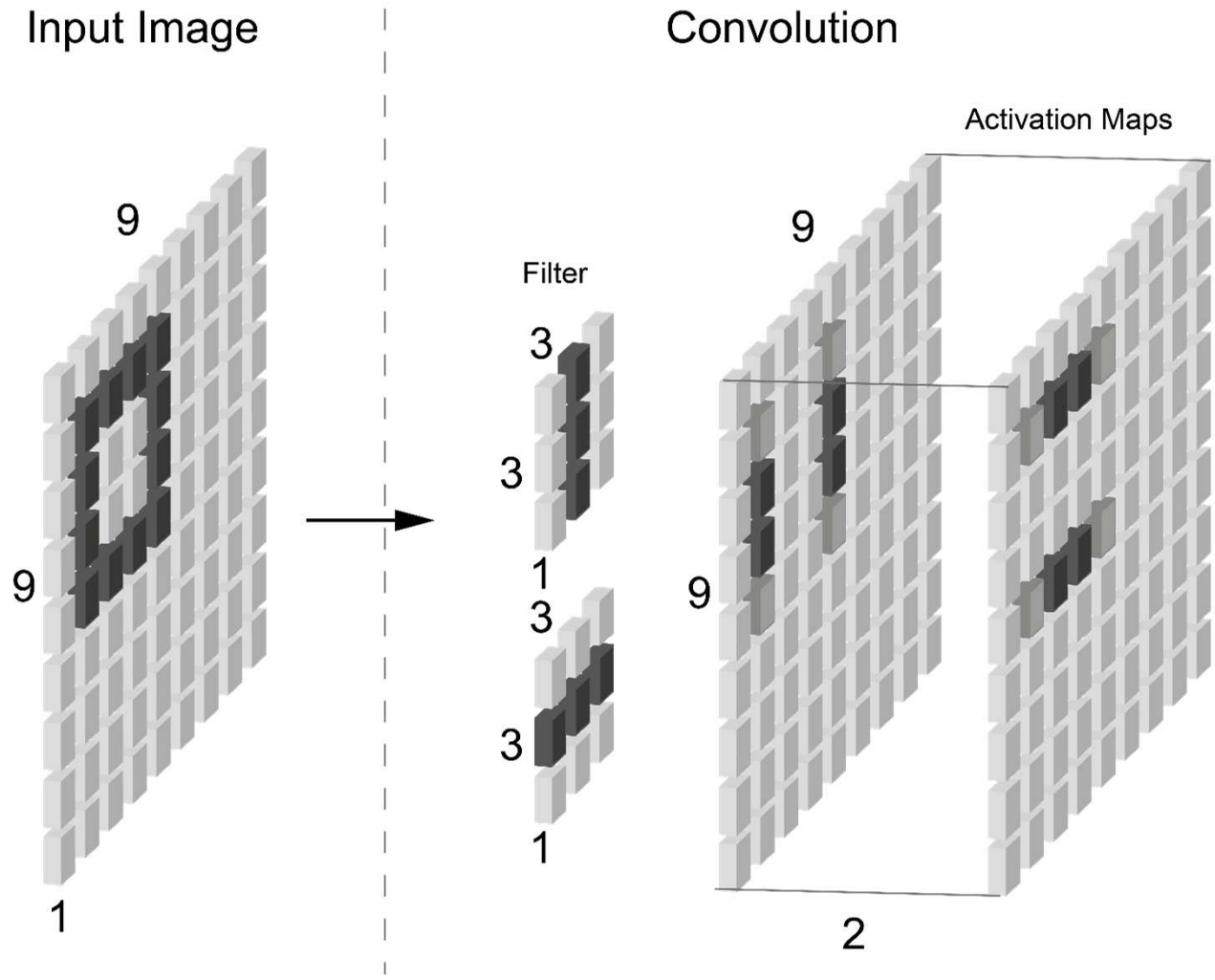
ReLU layer



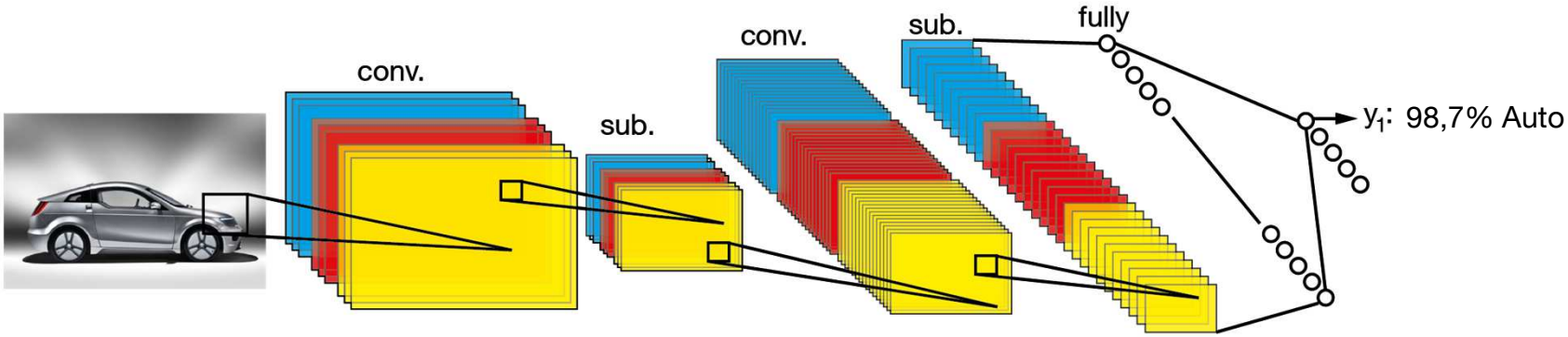
Sub. Layer

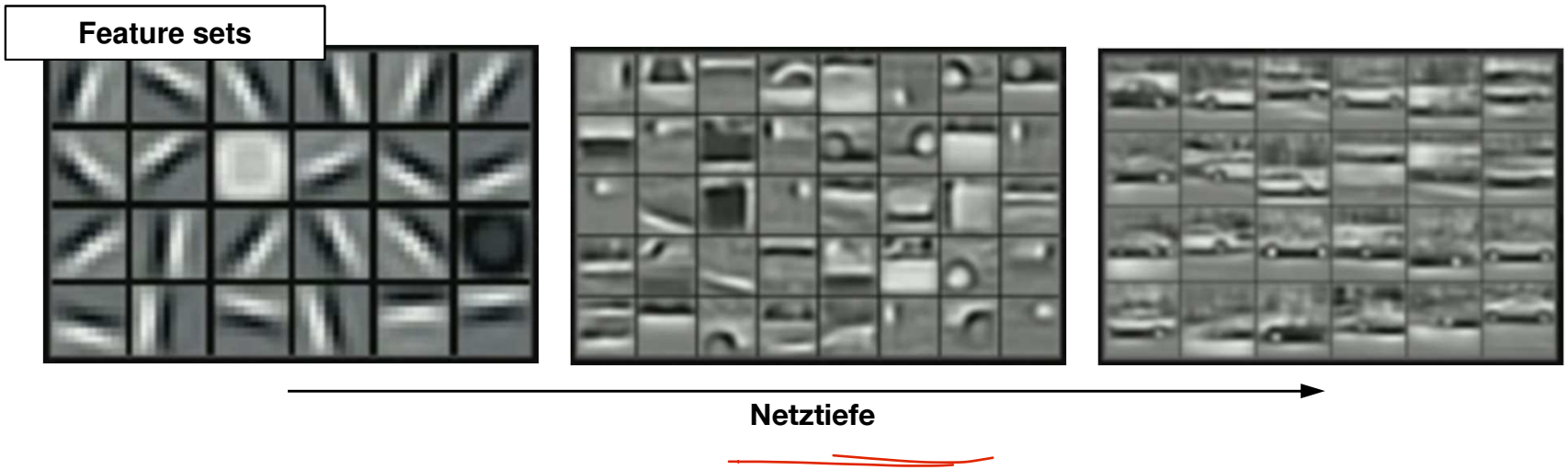


Convolution + Pooling



Gesamtnetz





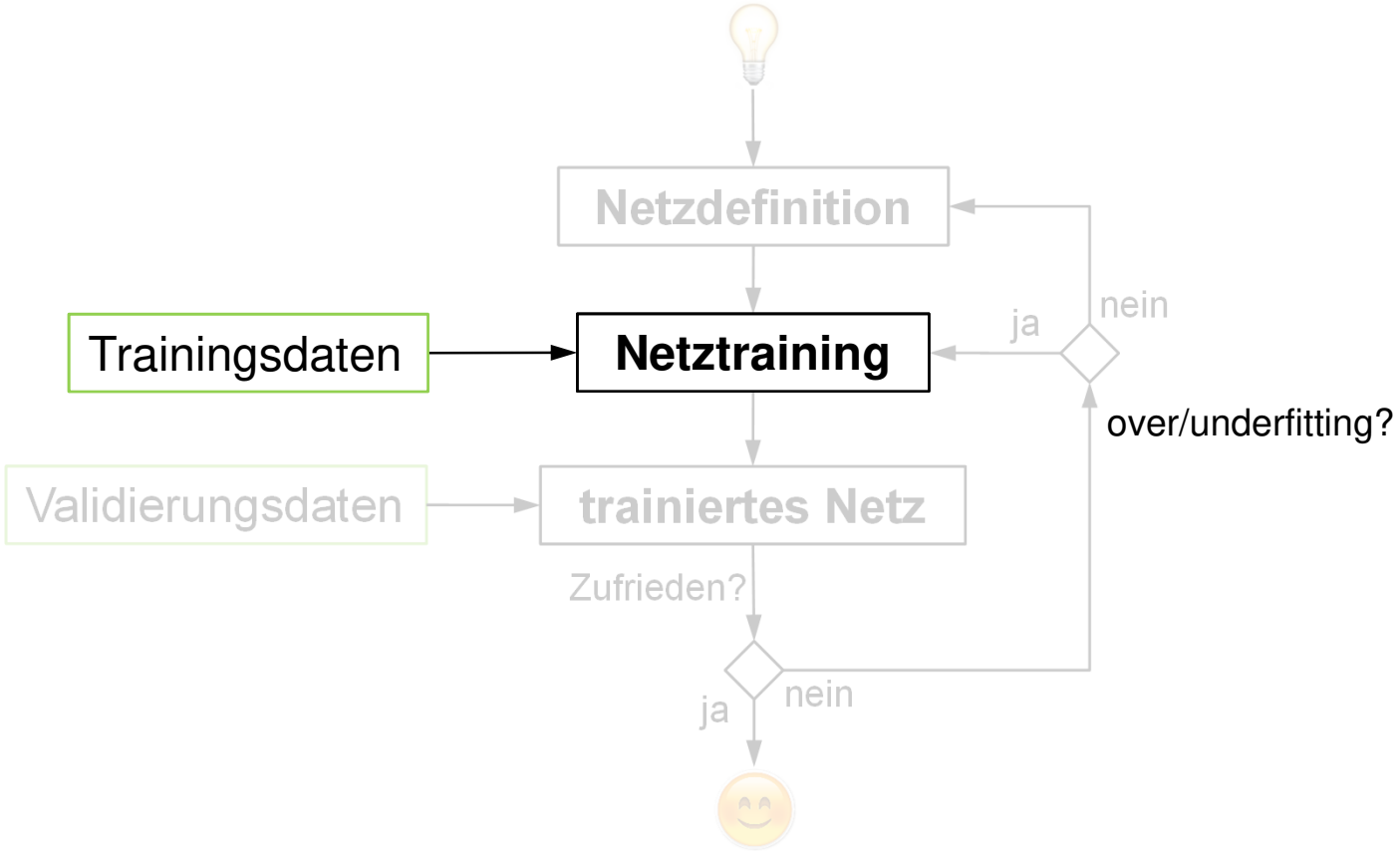
Deep Learning
Domagoj Majstorovic, M.Sc.

Agenda

7 Deep Learning / Neuronale Netze

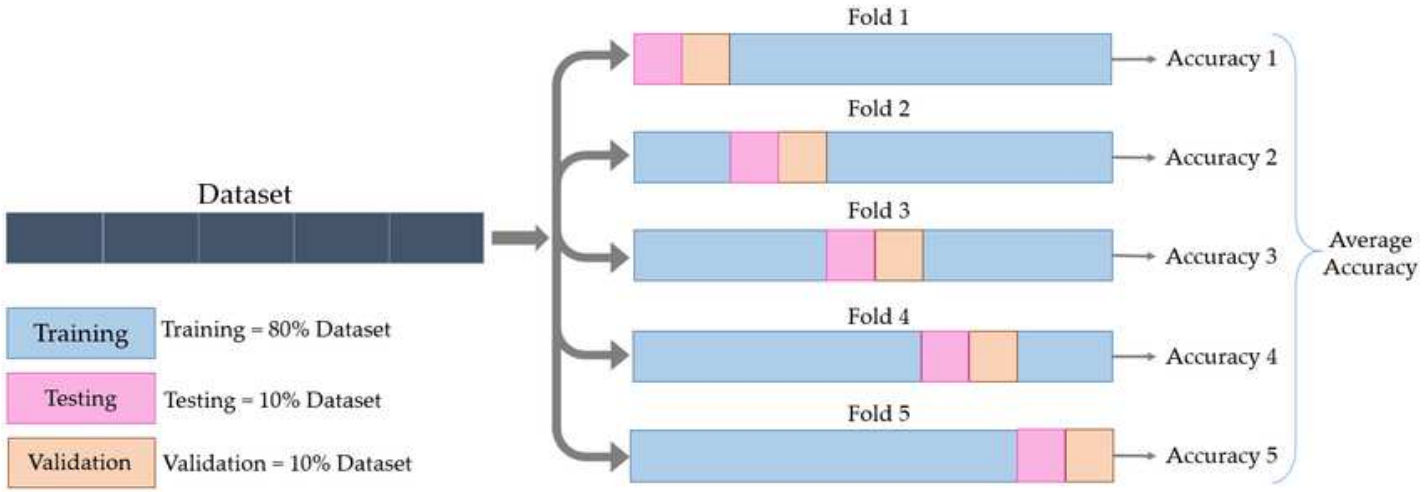
- 7.1 Grundlagen und Einleitung
- 7.2 Netzdefinition
- 7.3 Netztraining
- 7.4 Ergebnisse
- 7.5 Herausforderungen

Weg zum neuronalen Netz



Trainingsdaten

1. **Datenaufnahme**
2. **Datenvorbereitung**
   - Daten aufteilen, labeln, augmentieren, laden, ...



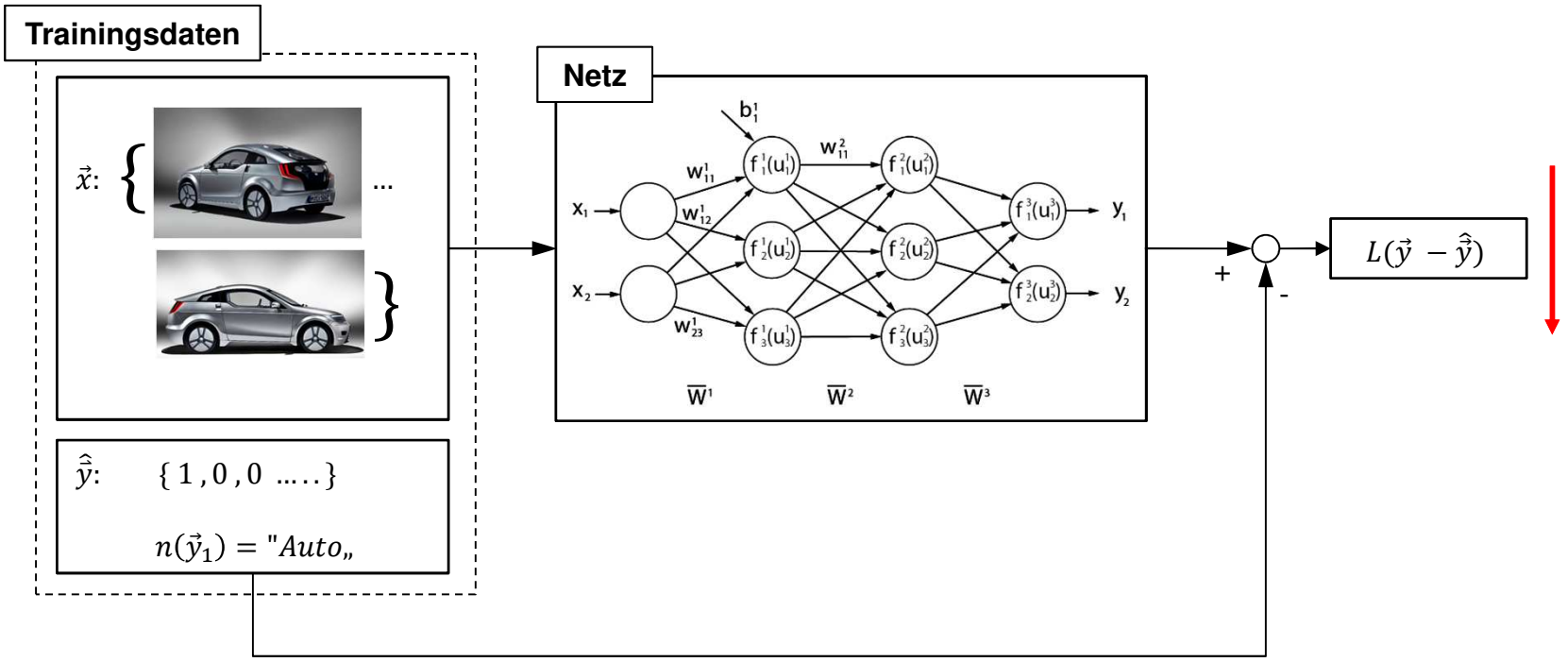
https://www.researchgate.net/publication/338676025_Smart_Tactile_Sensing_Systems_Based_on_Embedded_CNN_Implementations

Trainingsdaten

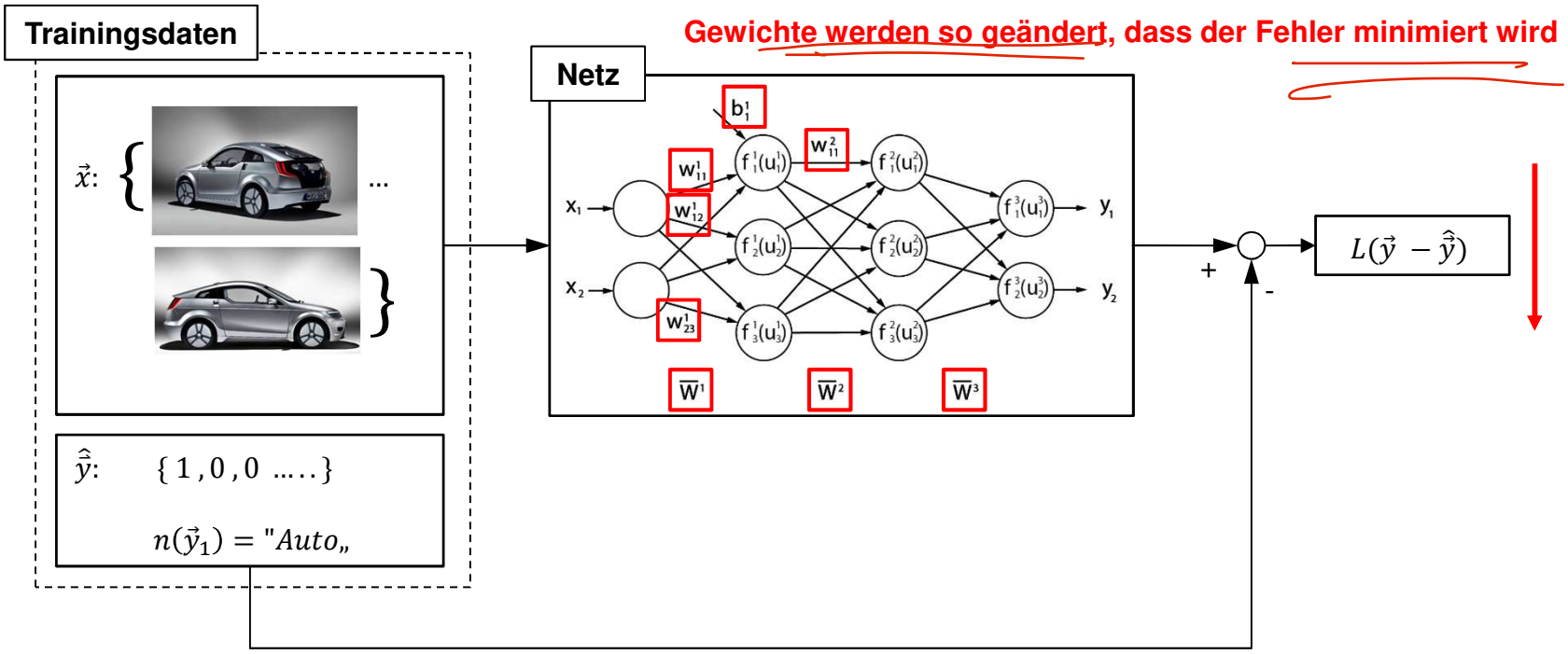
	Top-1 Accuracy	Top-5 Accuracy
Baseline	48.13 ± 0.42%	64.50 ± 0.65%
Flipping	49.73 ± 1.13%	67.36 ± 1.38%
Rotating	50.80 ± 0.63%	69.41 ± 0.48%
Cropping	**61.95 ± 1.01%**	**79.10 ± 0.80%**
Color Jittering	**49.57 ± 0.53%**	67.18 ± 0.42%
Edge Enhancement	49.29 ± 1.16%	66.49 ± 0.84%
Fancy PCA	49.41 ± 0.84%	**67.54 ± 1.01%**

https://www.researchgate.net/publication/338676025_Smart_Tactile_Sensing_Systems_Based_on_Embedded_CNN_Implementations

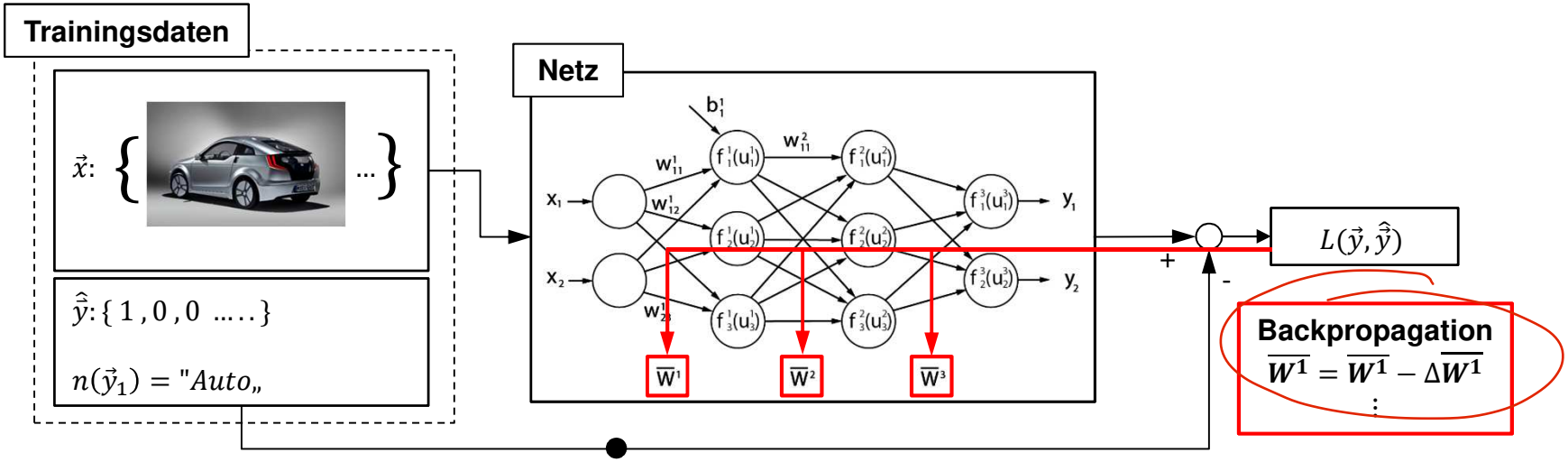
Netztraining

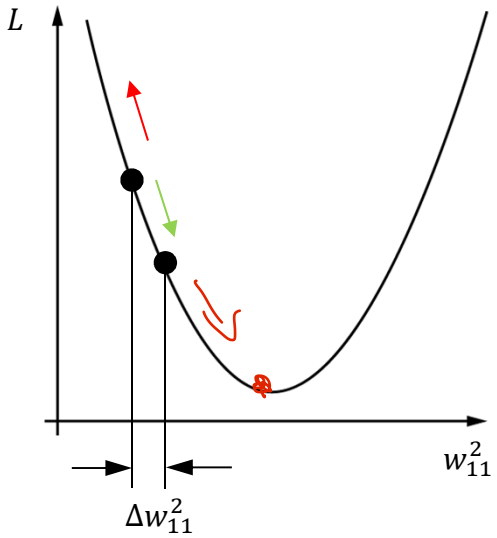


Netztraining

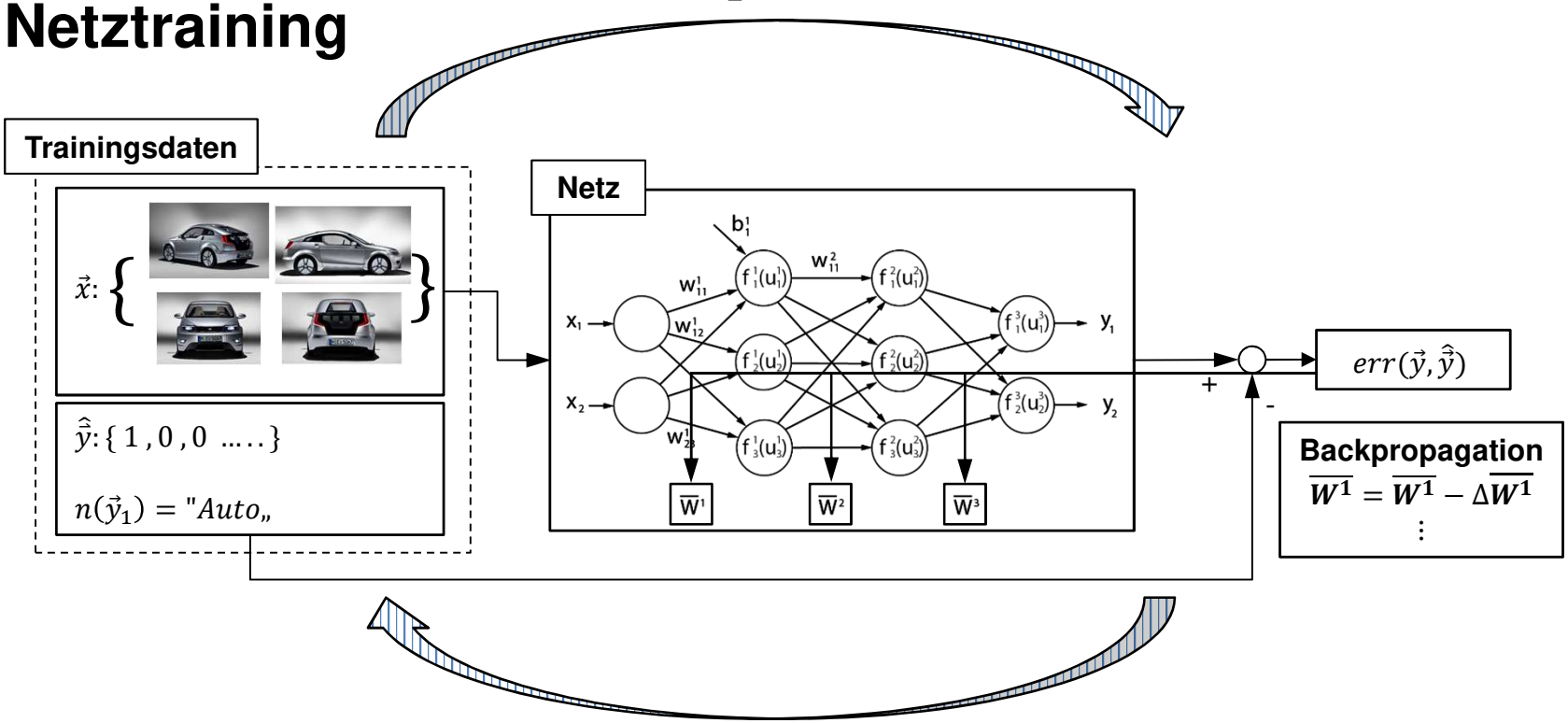


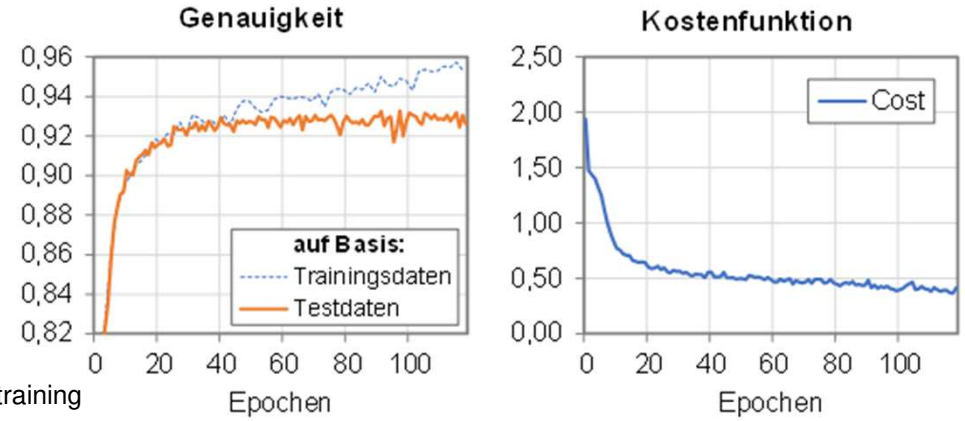
Netztraining





Netztraining

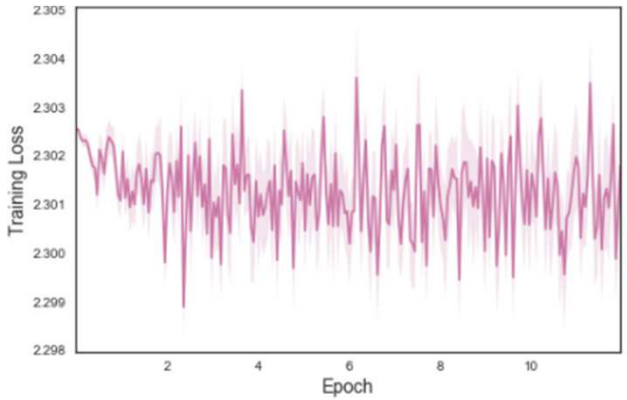




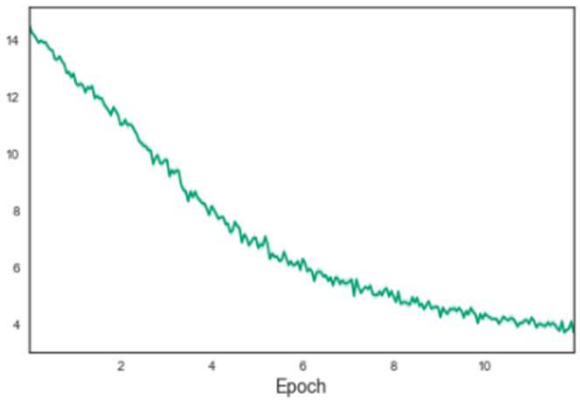
Initialisierung der Gewichte

$$W_{11} = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix}$$

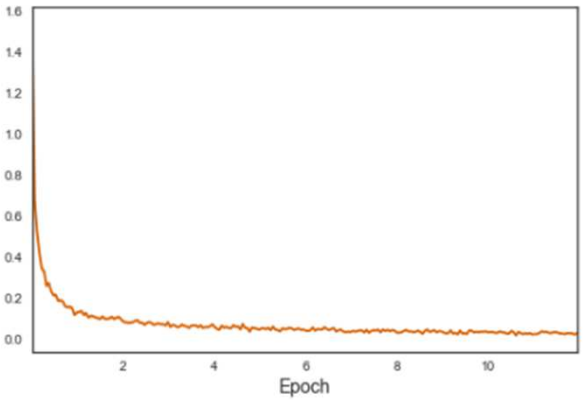
Initialisierung als 0



Normalverteilung (0, $\sigma = 0.4$)



Normalverteilung (0, $\sigma \sim \sqrt{\frac{2}{n_i}}$)



n_i: Anzahl Input Neuronen der Schicht

z.B.:

$$W_{11} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

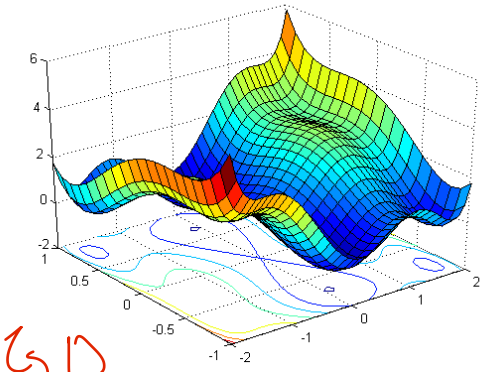
$$W_{11} = \begin{pmatrix} 0.01 & -0.50 & 0.14 \\ -0.10 & 0.11 & -0.06 \\ -0.33 & 0.47 & 0.32 \end{pmatrix}$$

$$W_{11} = \begin{pmatrix} 1.20 & -1.50 & 1.4 \\ -1.2 & 0.97 & -0.6 \\ 0.63 & -0.47 & 0.52 \end{pmatrix}$$

$n_i = 2 \rightarrow \sigma = 1$

https://intoli.com/blog/neural-network-initialization/

Optimierungs-Verfahren

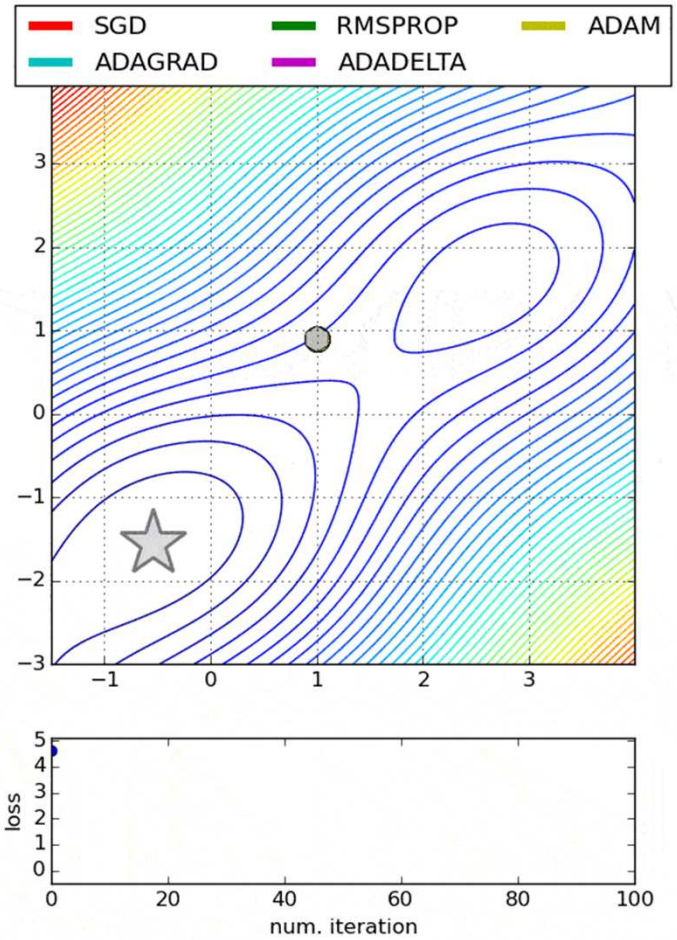


	Batch-Gradientenabstieg	Stochastischer Gradientenabstieg	Mini-Batch-Gradientenabstieg
Beschreibung	Alle Trainingsdaten werden für einen Optimierungsschritt verwendet.	Nur ein Datenpunkt wird für einen Optimierungsschritt verwendet.	Ein kleiner Teil der Traingsdaten wird für einen Optimierungsschritt verwendet.
Vorteile	• Effiziente Rechenaufwand • Stabile Konvergenz	• Schnelleres Lernen • Unmittelbare Einblicke in die Leistung	• Effiziente Berechnung • Stabile Konvergenz • Schnelleres Lernen
Nachteile	• Langsameres Lernen • Lokale Minima und Sattelpunkte	• Hohe Rechenleistung erforderlich • Verrauschte Gradienten	• Neue Hyperparameter

https://artemoppermann.com/de/variationen-des-gradientenabstiegs-im-backpropagation-algorithmus/

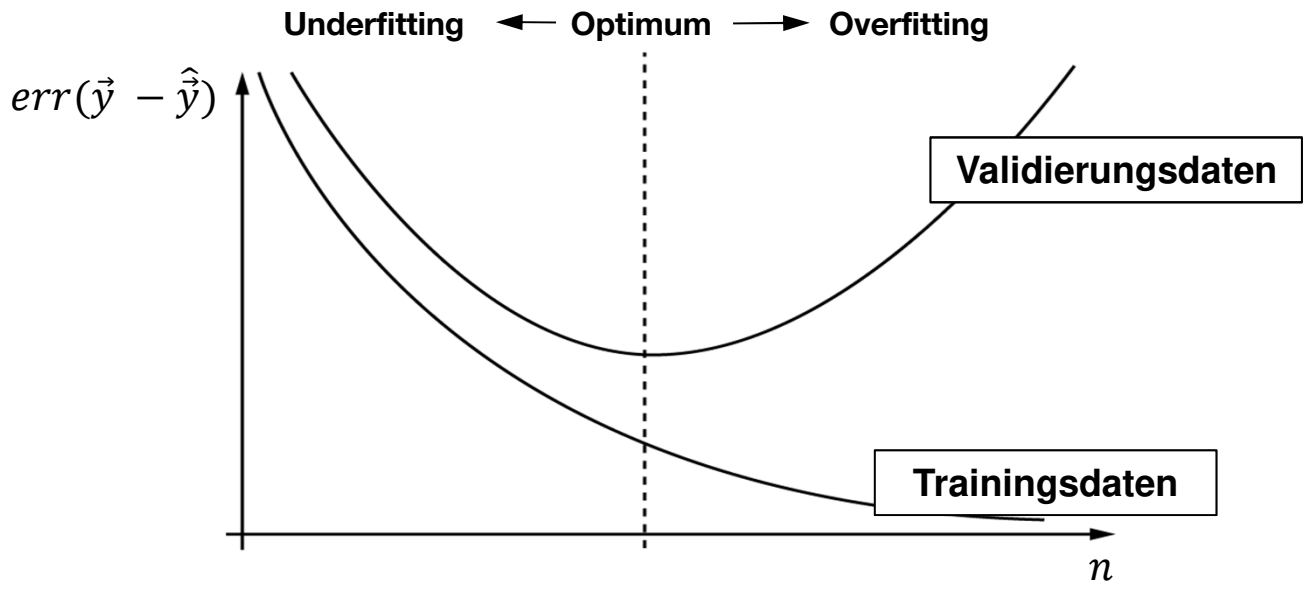
Optimierungs-Algorithmen

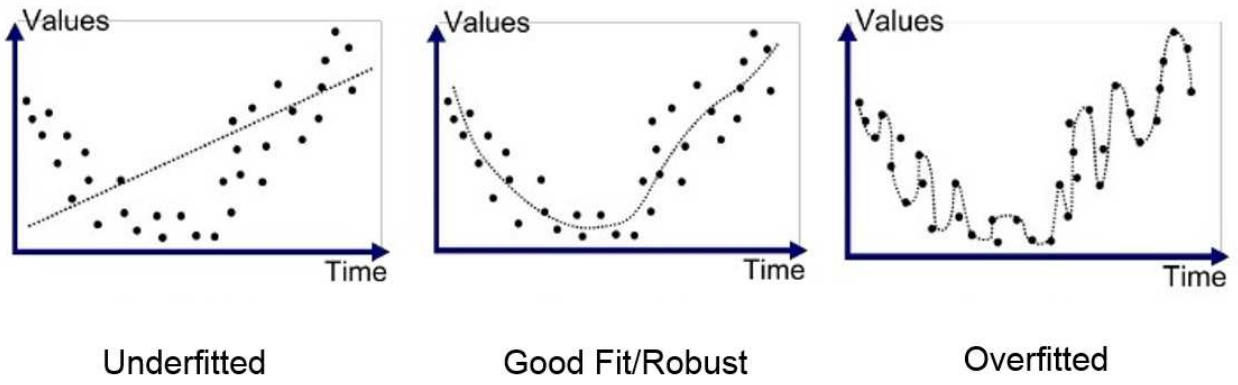
- Stochastic Gradient Decent
- Root Mean Squared Propagation
- Adaptive moment estimation (Adam)
- Adaptive Gradient Decent
- Adadelta
- ...



https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6

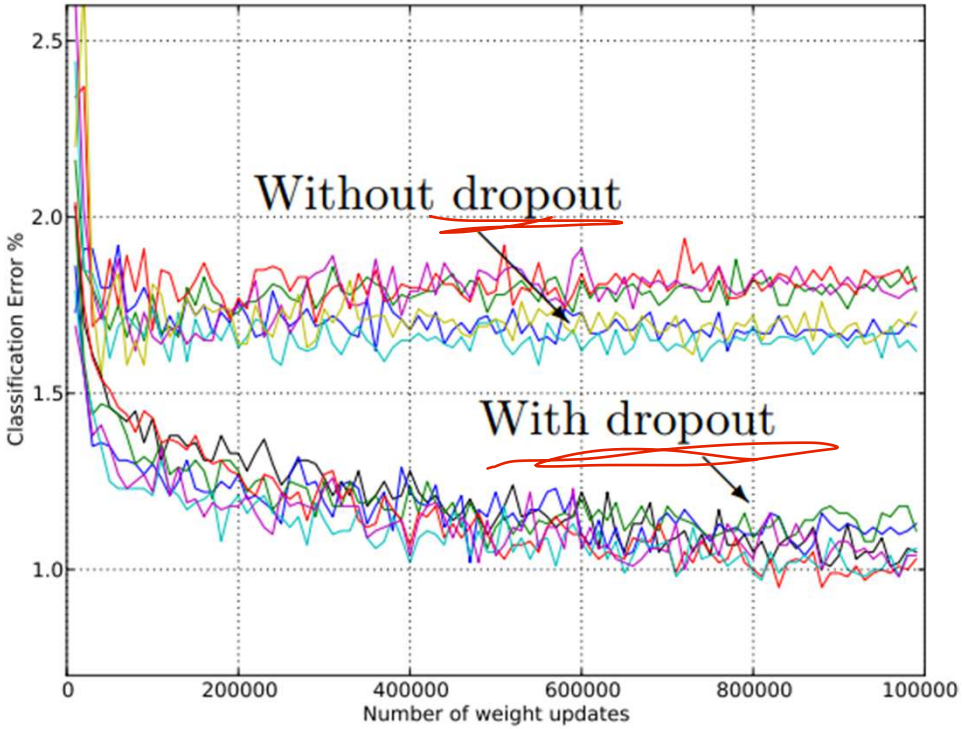
Netztraining

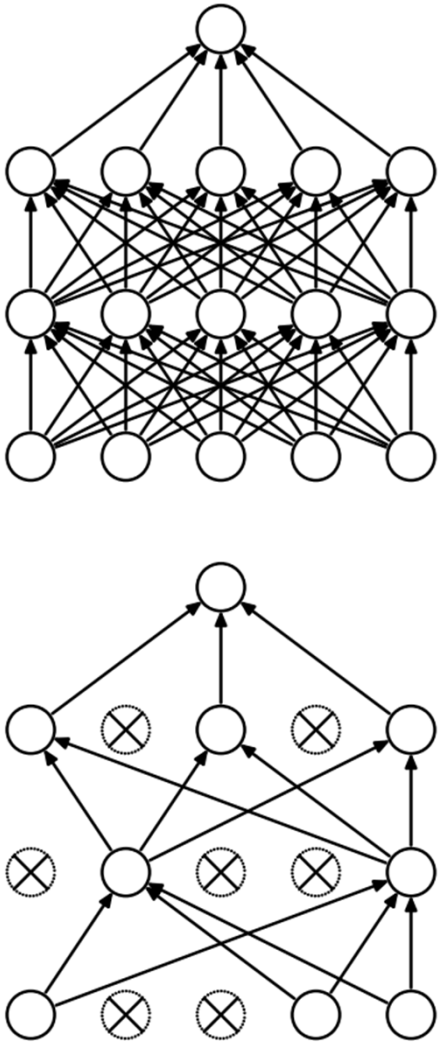




Netztraining

Overfitting? → **Dropout, Regularization, ...**



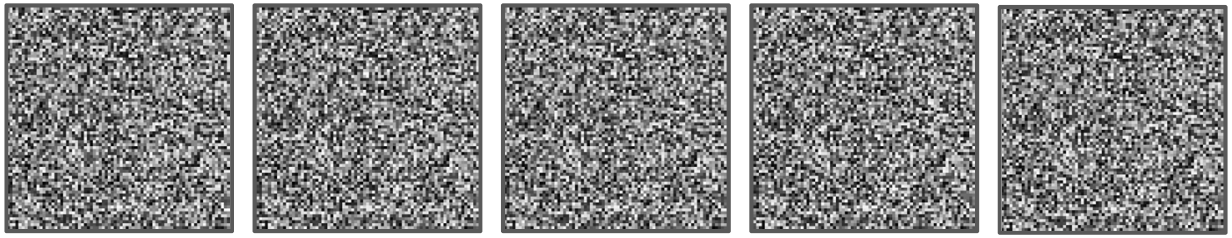


Fully vs. Convolution

Input

Fully Connected Netz (4-Layer)

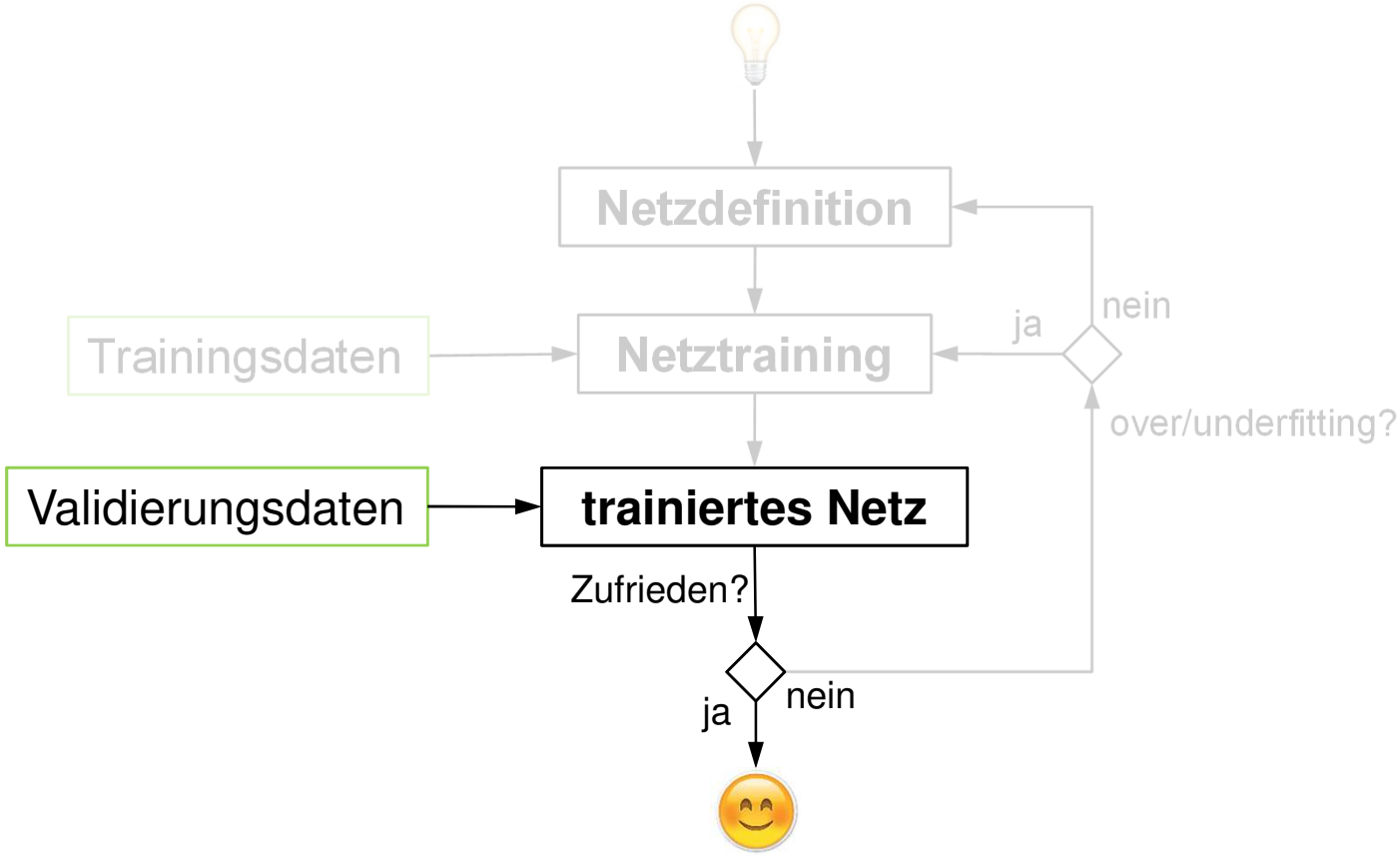
93,3 %



Convolutional Netz (4-Layer)

99,2 %

Weg zum neuronalen Netz

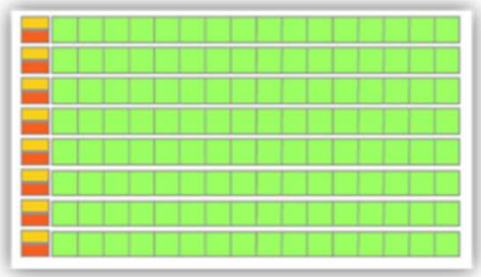


Hardware

Central **P**rocessing **U**nit
(CPU)

- Geringe Berechnungsdichte
- Komplexe Logik Steuerung
- Großer Cache
- Geringe Latenztolleranzen

Graphics **P**rocessing **U**nit
(GPU)

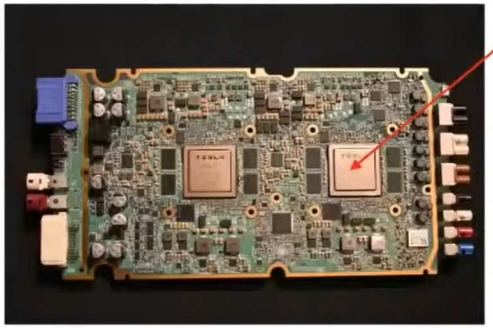


- Hohe Berechnungsdichte
- Viele Berechnungen pro Speicheraufruf
- Optimiert für Parallele Rechnungen
- Hohe Latenztolleranz

Hardware

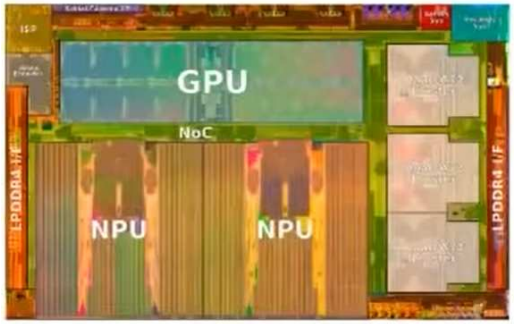
Tesla FSD Computer

FSD computer

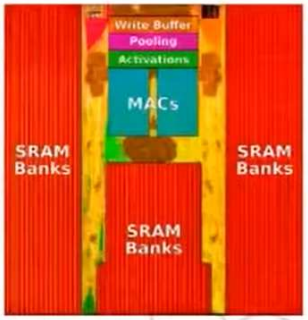


1 FSD Chip:
12 CPUs @ 2.2 GHz
GPU (600 GFLOPS)
2X NPU (36.86 TOPS / NPU)
36W

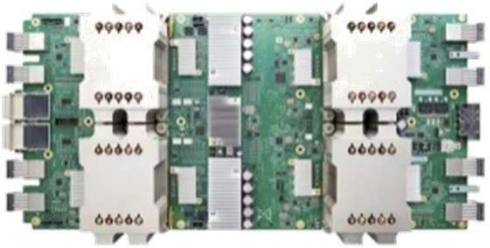
FSD chip



NPU

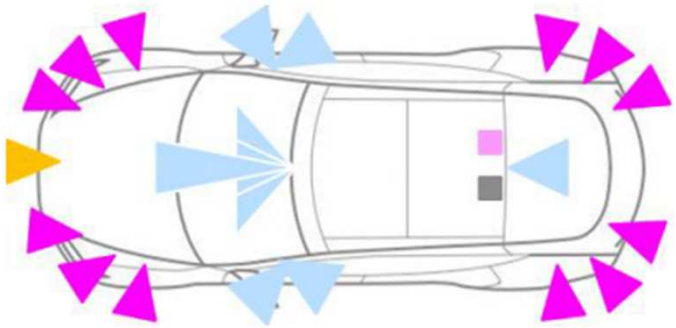


Tensor **P**rocessing **U**nit (TPU) (Google)

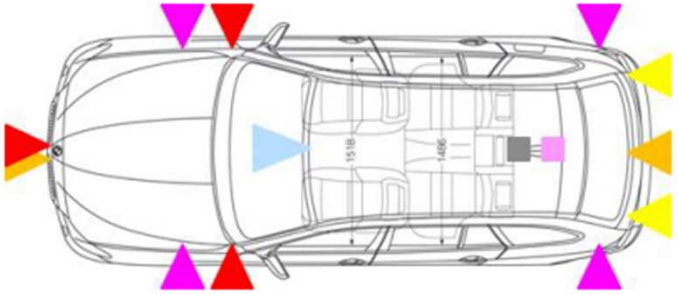


Sensorkonzepte

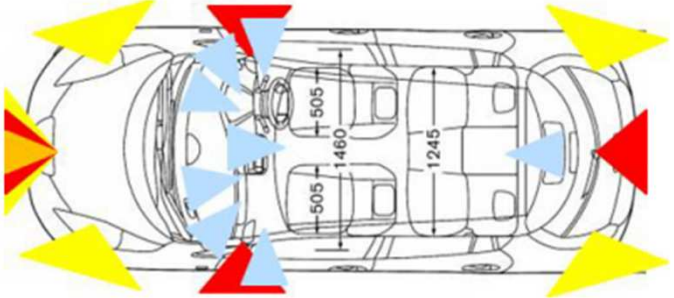
Tesla
Model S
Production
Vehicle
[19]



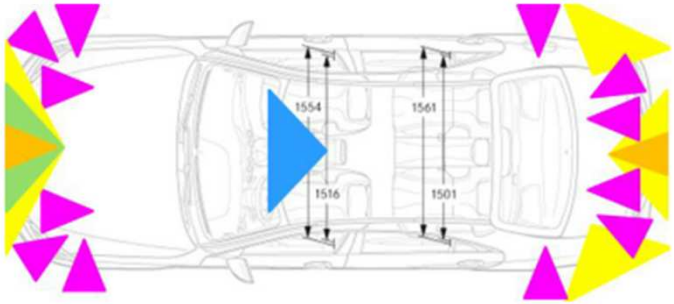
BMW
536i
Concept
Vehicle
[20]



Nissan
Leaf Pro-Pilot
Concept
Vehicle
[21]



Mercedes
S Class
Production Car
[22]

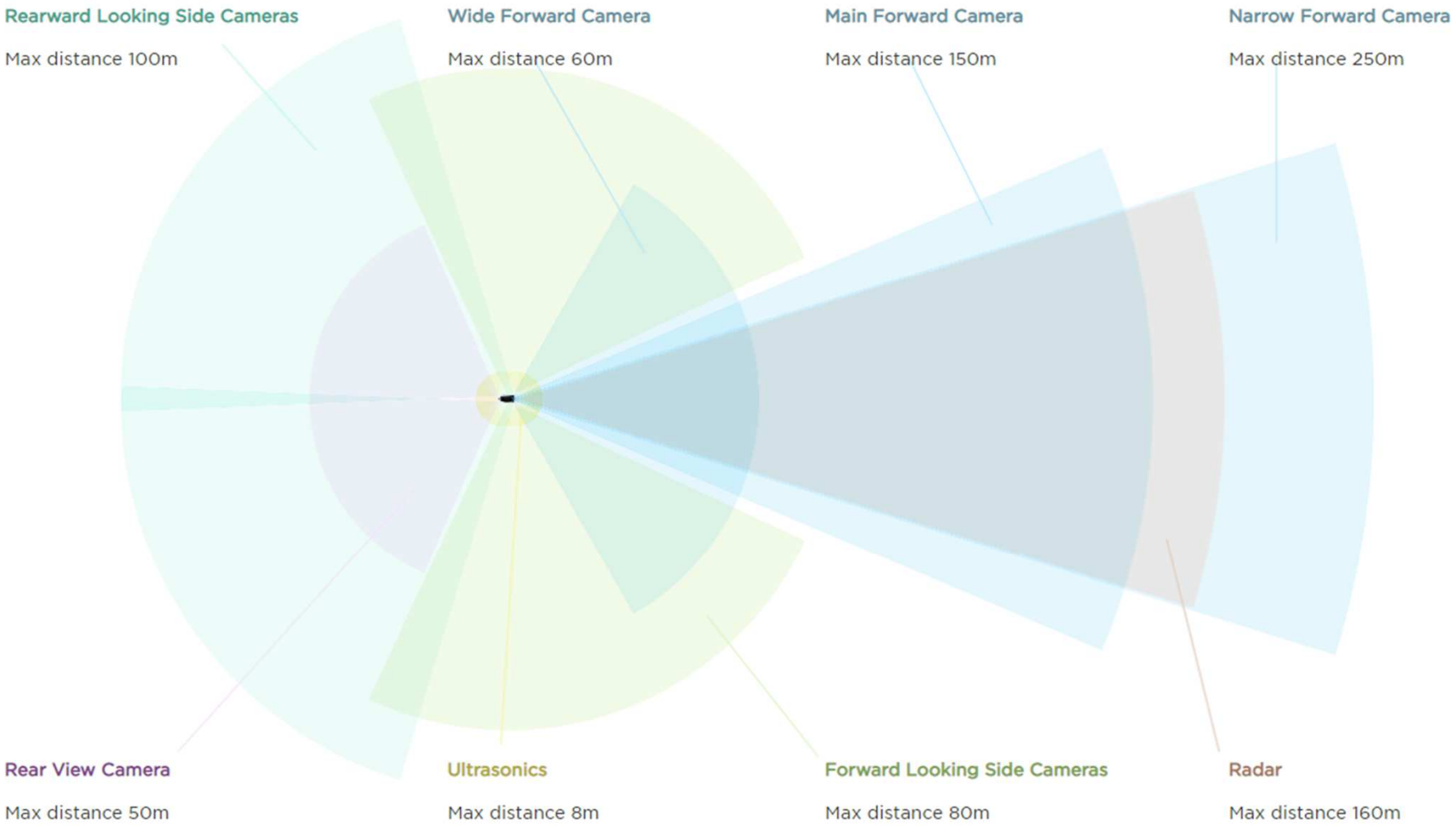


Radar - Short Range	Radar - Long Range	GPS	Lidar / Laser
Camera - Mono	Camera – Stereo	V2X – Sensor	
Ultrasonic	Infrared	Maps	

[1] D. Will, P. Gronert, S. von Bargen, F. B. Levrin, and G. Larini, “Report on the state of the art of connected and automated driving in Europe (Final),” Aachen, Germany, 2017.

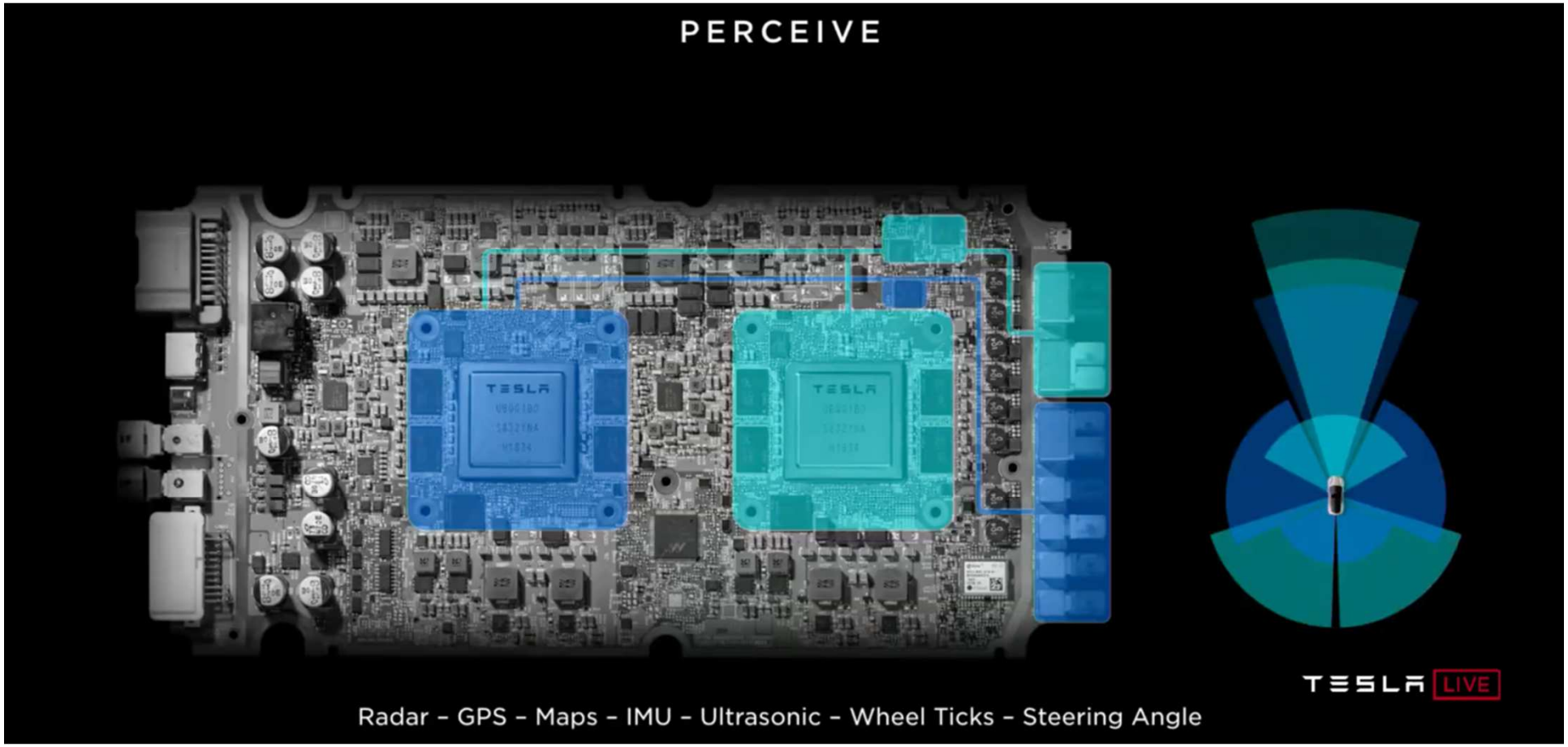
Kameras in Autonomen Fahrzeugen

Tesla Model 3 - Sensoren



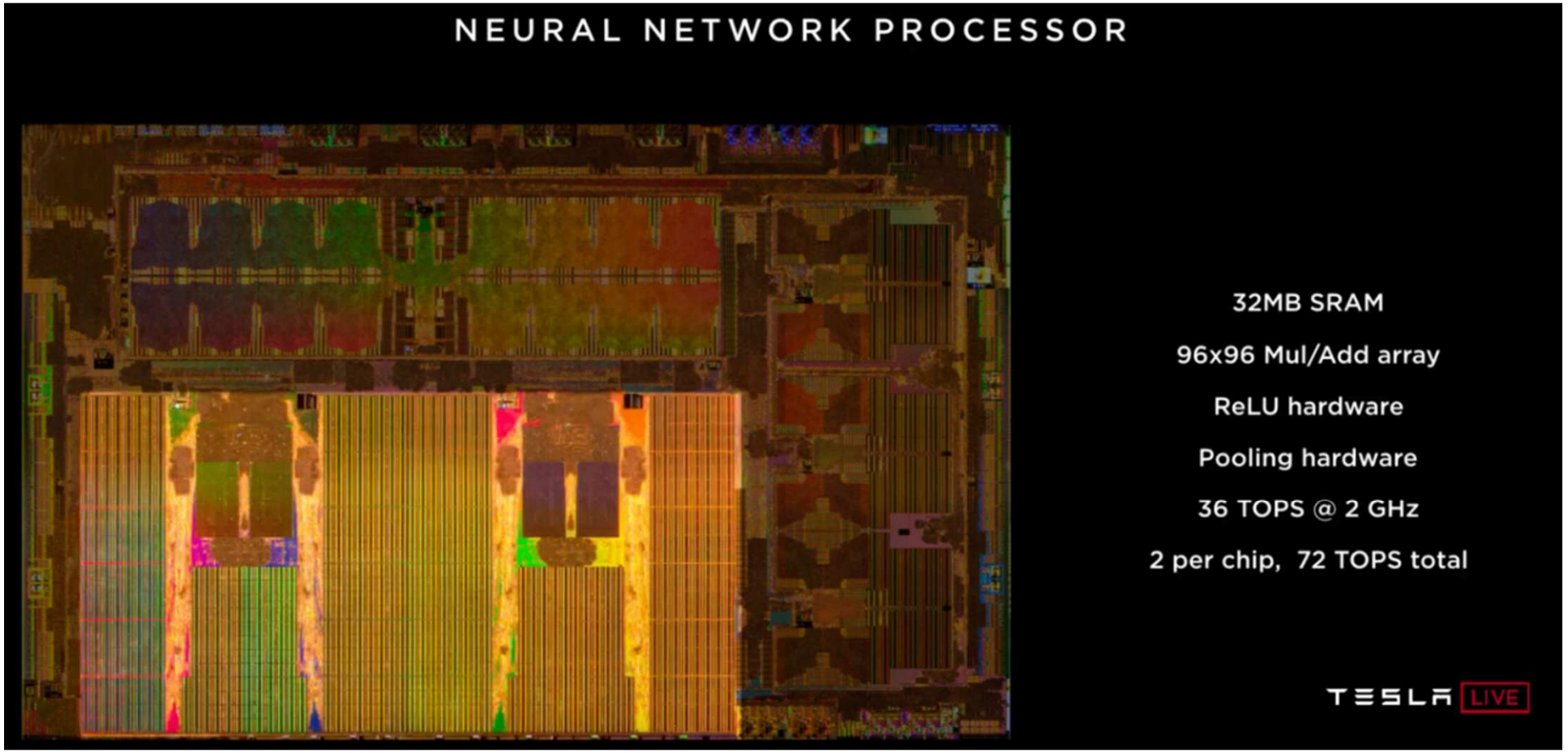
https://www.tesla.com/autopilot

Tesla Autonomy Day
22.04.2019



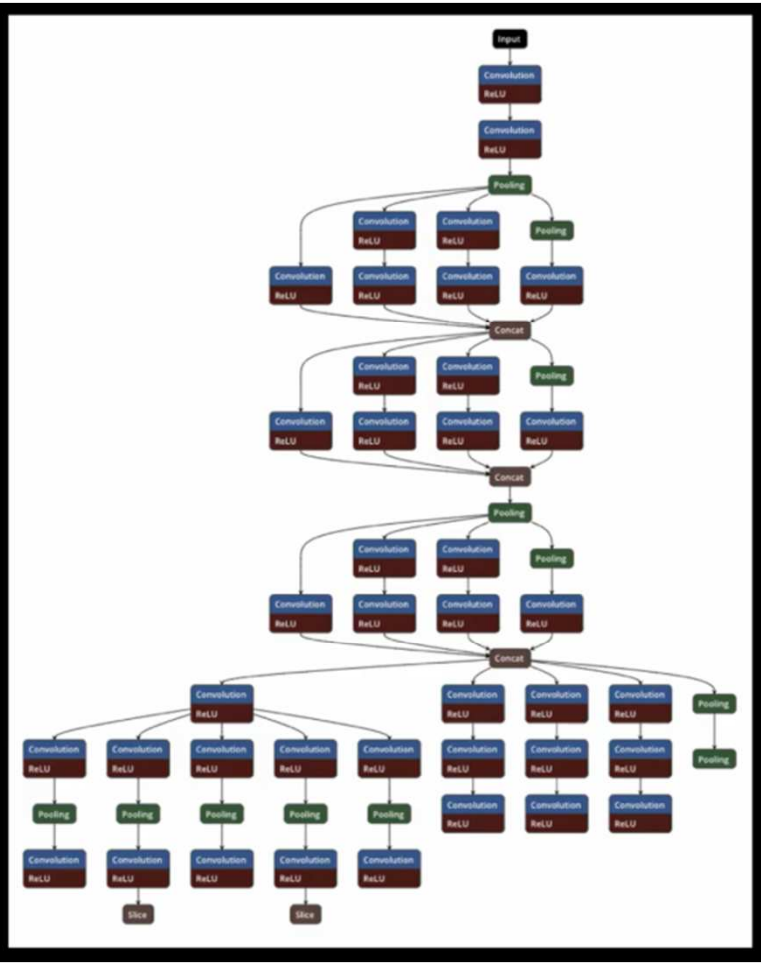
https://theteslashow.com/tesla-autonomy-investor-day/

Tesla Autonomy Day
NN Chip



https://theteslashow.com/tesla-autonomy-investor-day/

Tesla Autonomy Day
NN Architektur



Operation	MOPS	%
Convolution	34275	98.1
Deconvolution	576	1.6
ReLU	123	0.1
Pooling	13	0.2

99.7% of operations are multiply add

TESLA LIVE

https://theteslashow.com/tesla-autonomy-investor-day/

Deep Learning
Domagoj Majstorovic, M.Sc.

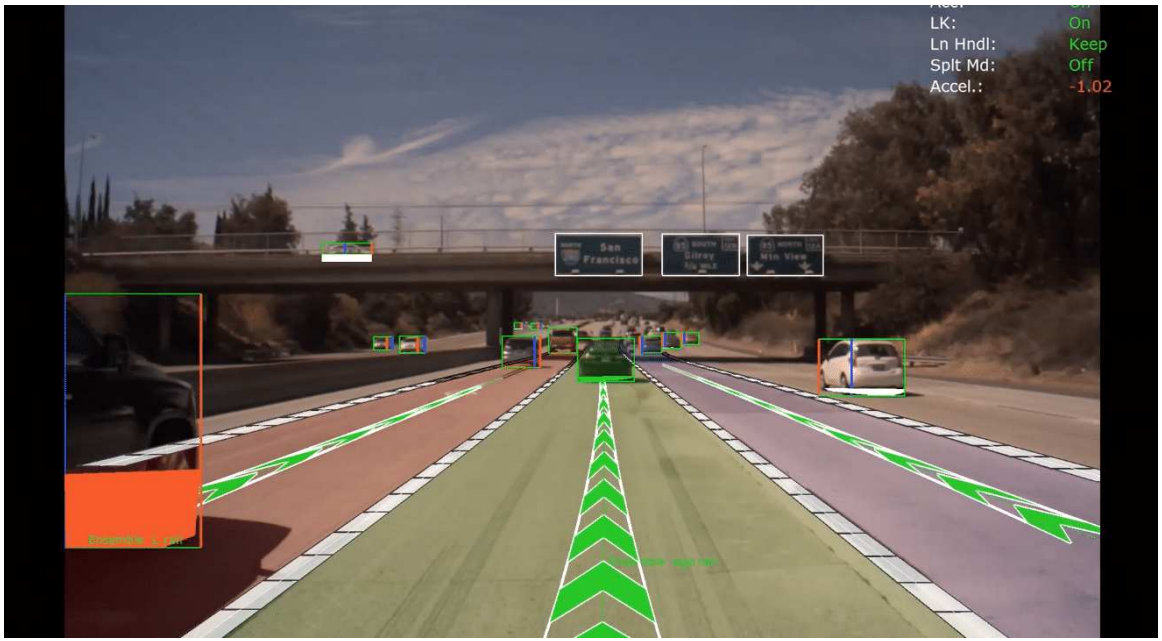
Agenda

7 Deep Learning / Neuronale Netze

- 7.1 Grundlagen und Einleitung
- 7.2 Netzdefinition
- 7.3 Netztraining
- 7.4 Ergebnisse
- 7.5 Herausforderungen

Nvidia

- OpenRoadNet
- PathNet
- LaneNet
- MapNet
- DriveNet
- LightNet
- SignNet
- WaitNet
- ClearSightNet
- ParkNet



https://blogs.nvidia.com/blog/2019/05/07/self-driving-cars-make-decisions/

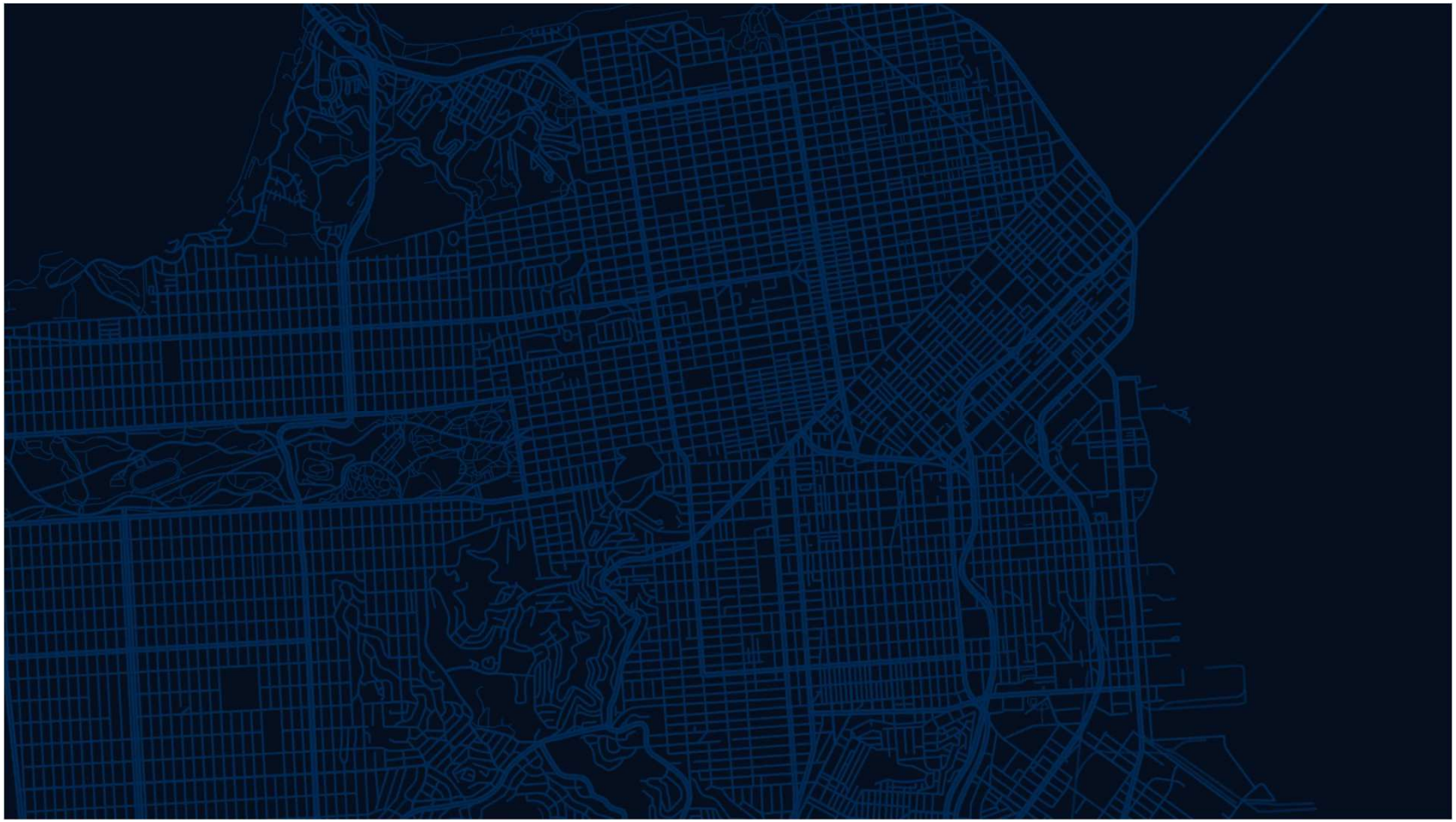
Nvidia

https://blogs.nvidia.com/blog/2019/05/07/self-driving-cars-make-decisions/

Tesla (Video)

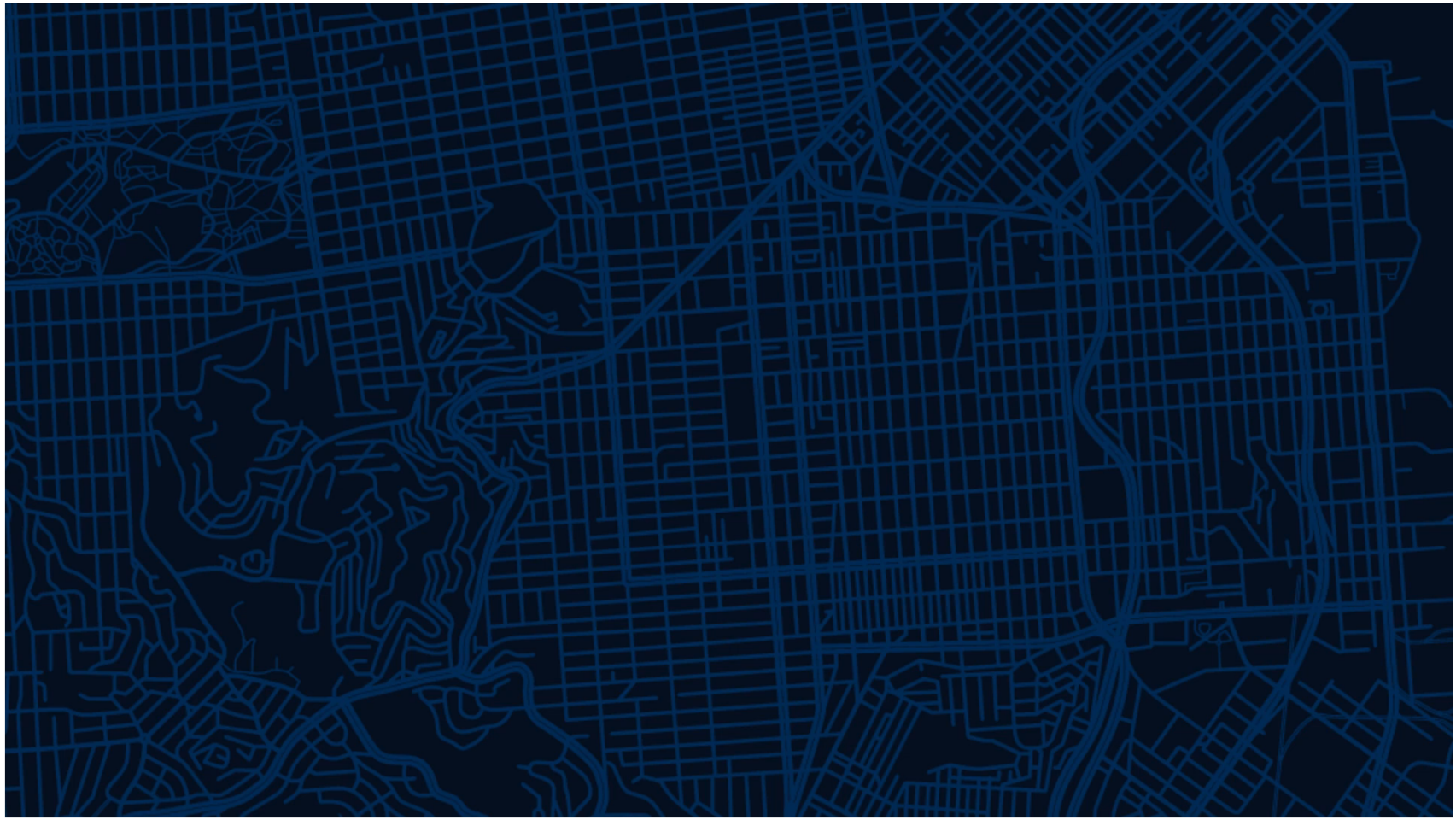
https://www.youtube.com/watch?v=xLDWsJ2ReBg

Waymo (Video)



https://www.youtube.com/watch?v=7pEwbS2X-2E

Waymo (Video)



https://www.youtube.com/watch?v=11BrxFe3iWE

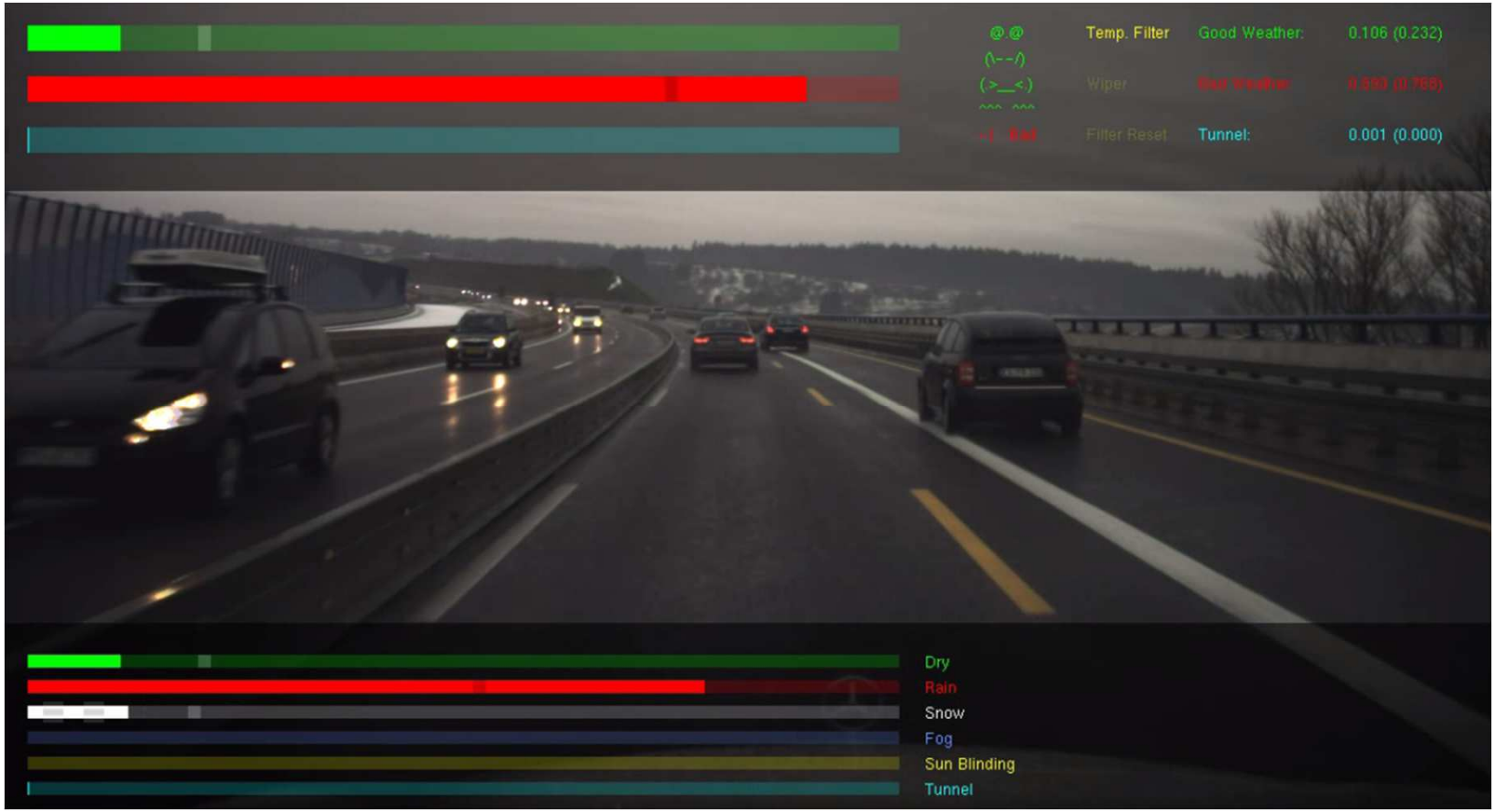
Cruise (Video)

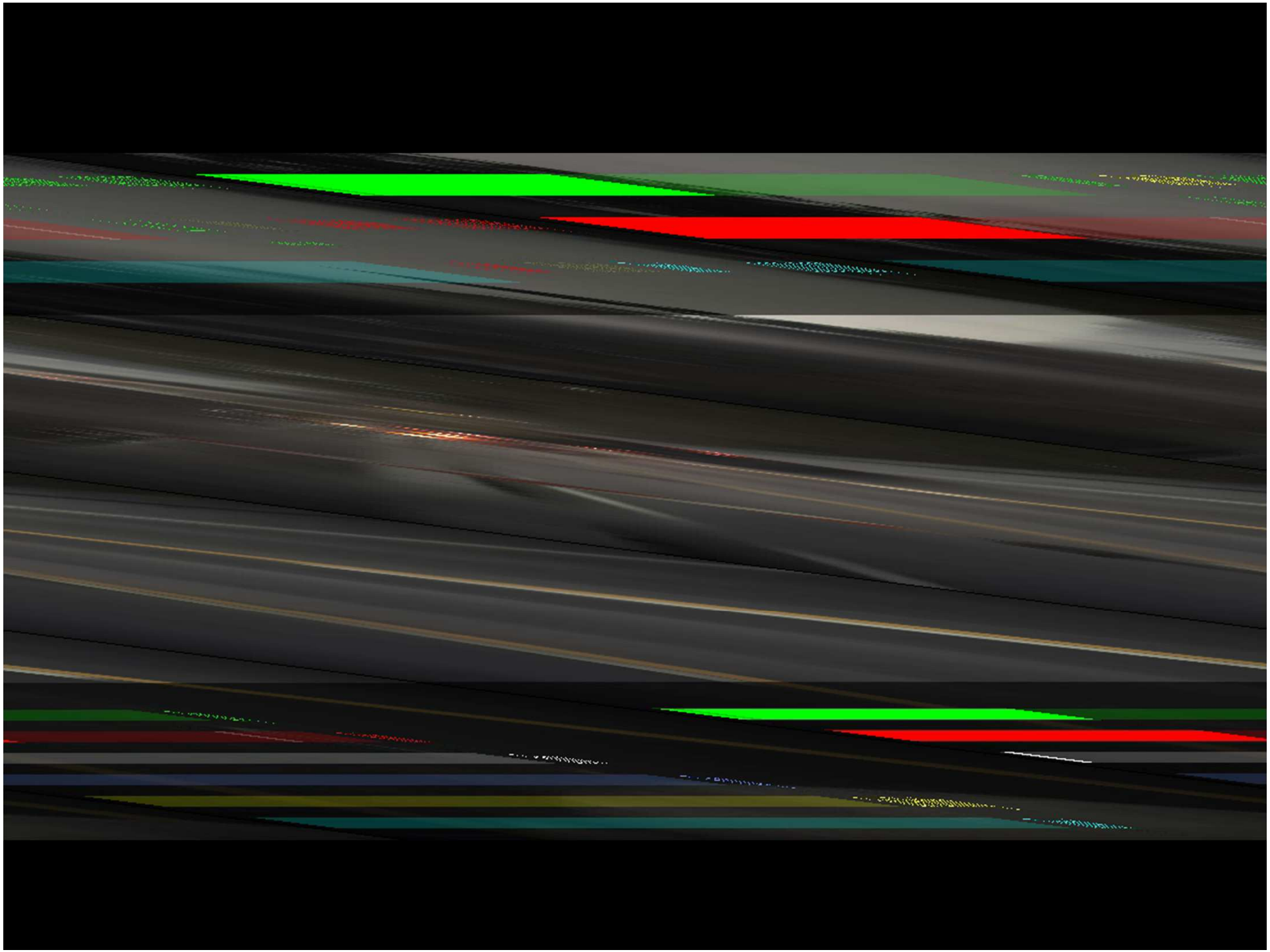
https://www.youtube.com/watch?v=x5g_3B8ya4k

Zoox (Video)

https://www.youtube.com/watch?v=rZ6hH2Frrf4

Daimler (Video)





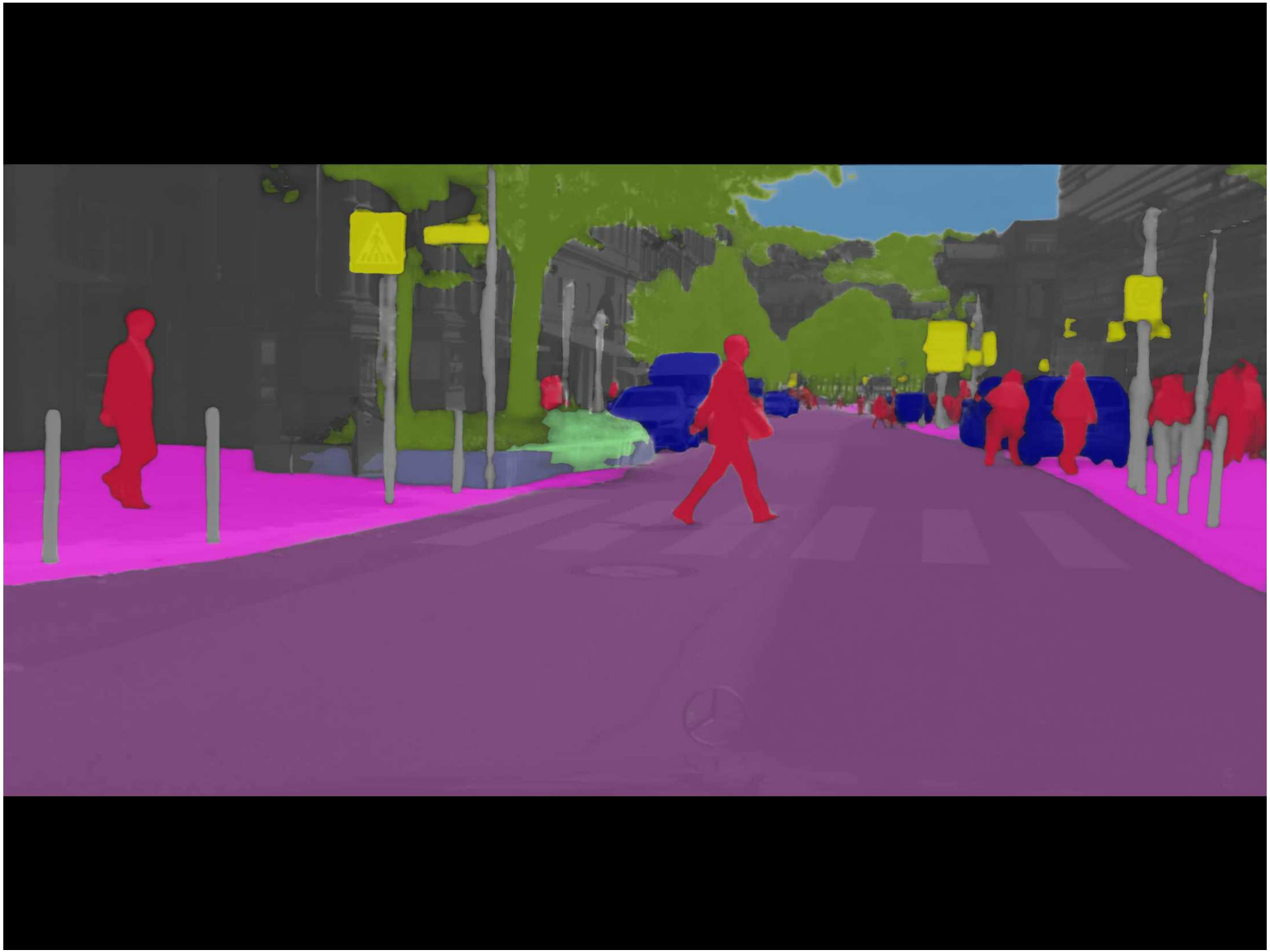
Beispielszene

Beispielszene

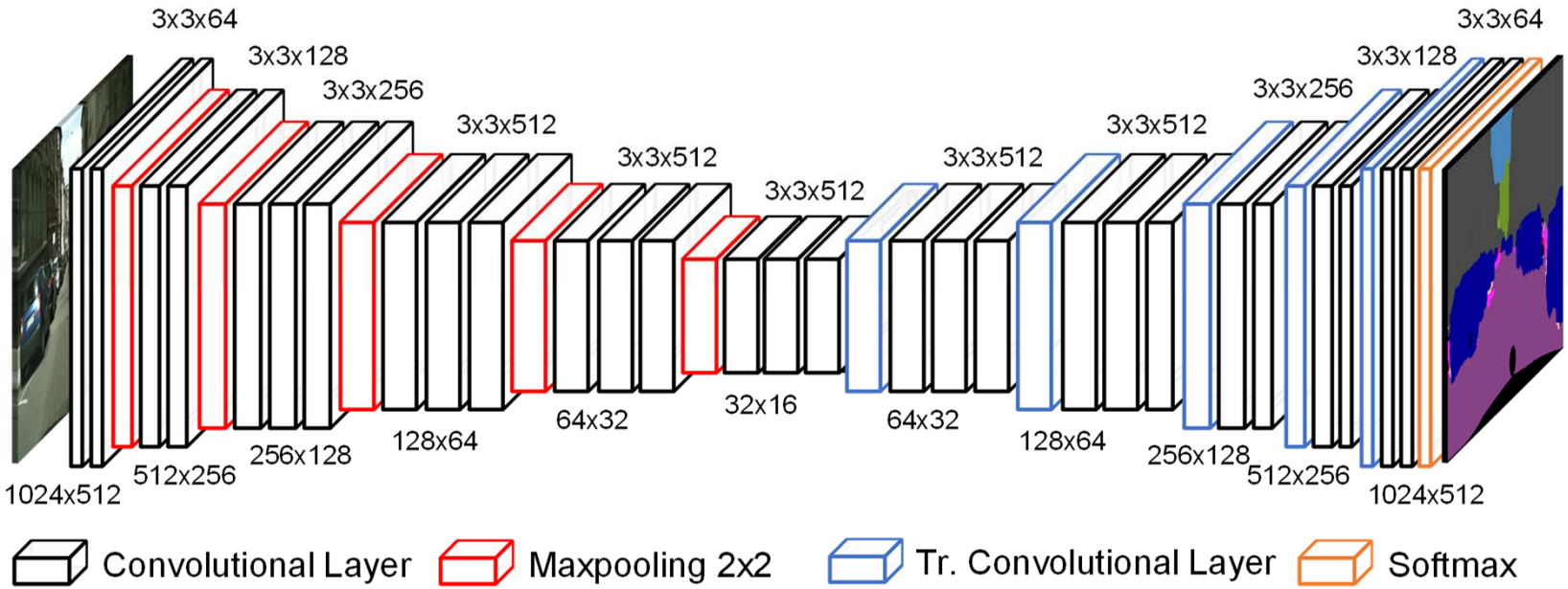
Daimler: Cityscapes Dataset

x

$\hat{y}$

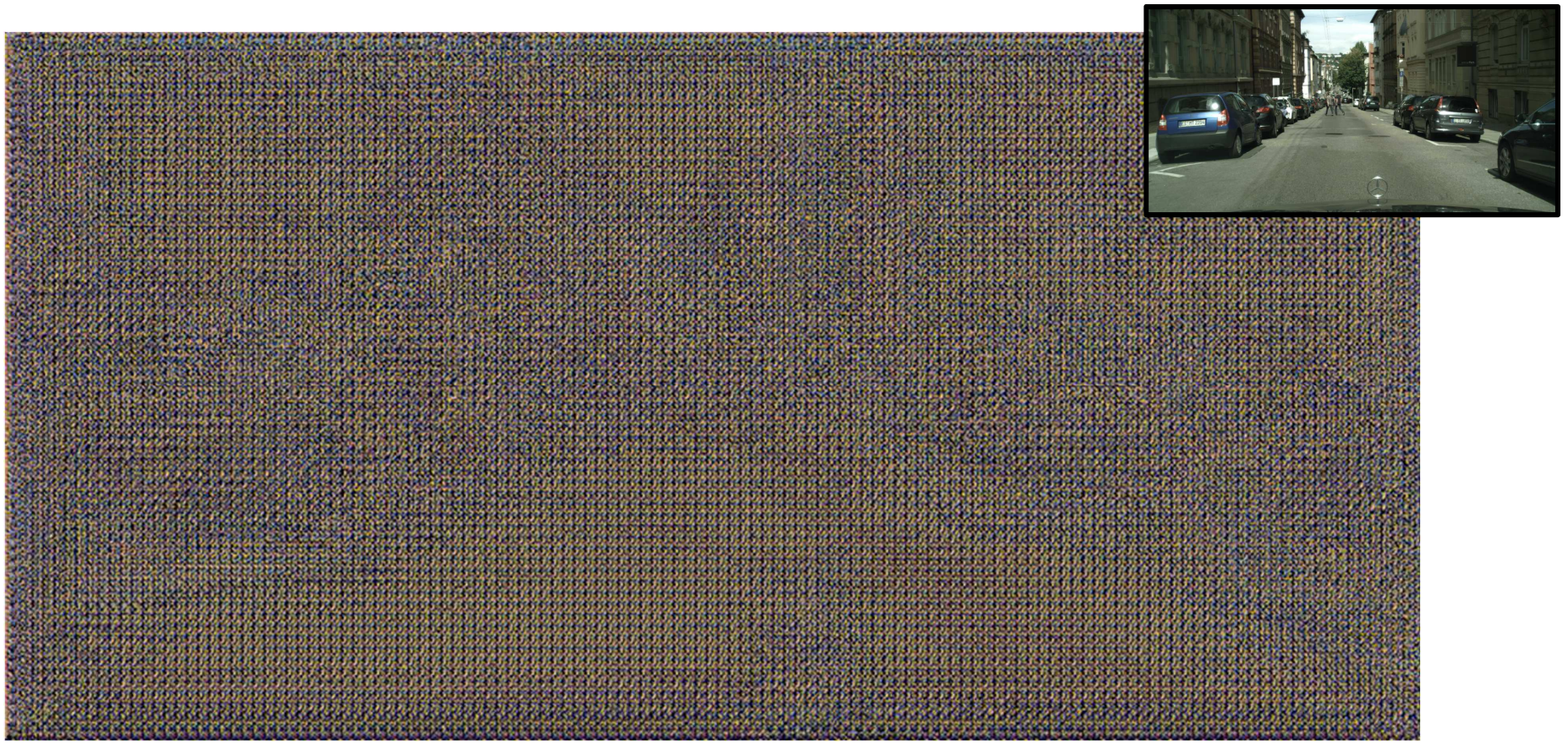


CNN-Struktur



Cityscapes Learning Process – Fully Connected Network

Cityscapes Learning Process – Convolutional Neural Network



Reich mit Convnets

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.

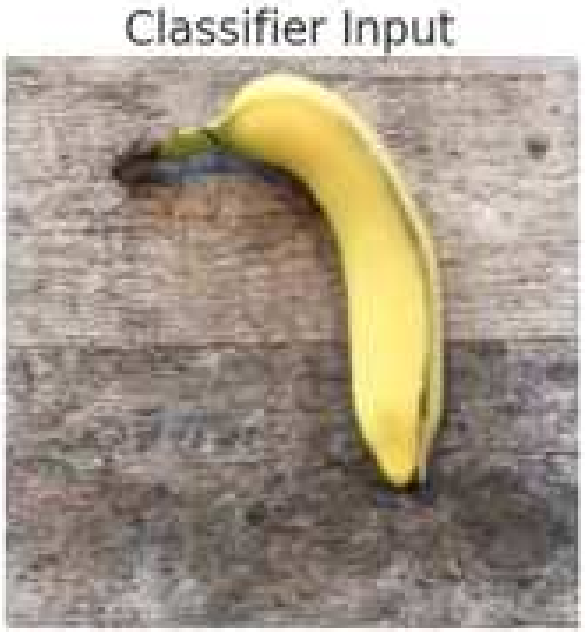
Deep Learning
Domagoj Majstorovic, M.Sc.

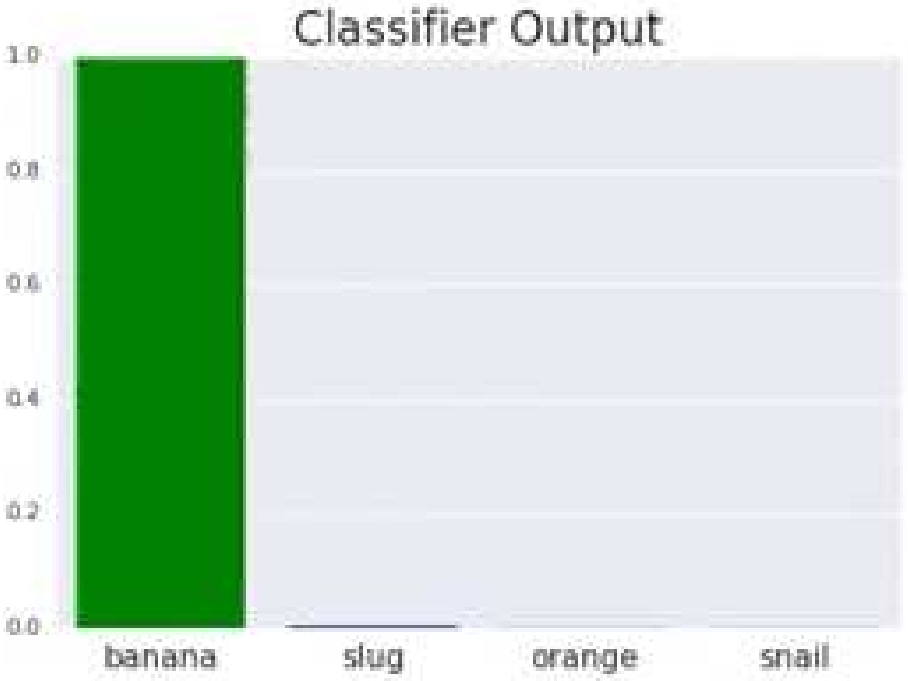
Agenda

7 Deep Learning / Neuronale Netze

- 7.1 Grundlagen und Einleitung
- 7.2 Netzdefinition
- 7.3 Netztraining
- 7.4 Ergebnisse
- 7.5 Herausforderungen
- 7.6 Demo

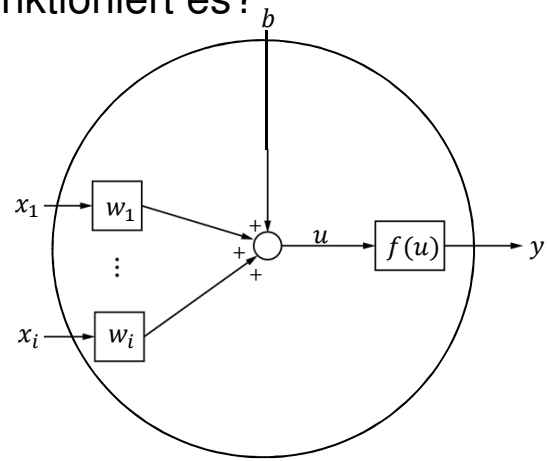
Video: Segmentation Time


Classifier Input

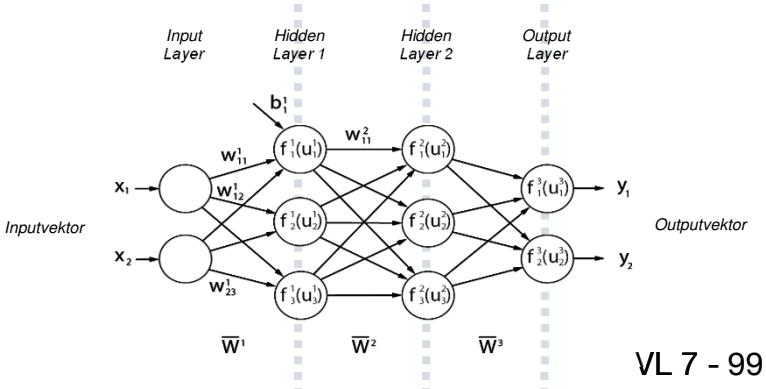

Classifier Output
1.0
0.8
0.6
0.4
0.2
0.0
banana
slug
orange
snail

Zusammenfassung der Leitfragen

- Wie ist ein einzelnes Neuron aufgebaut und wie funktioniert es?
  - Verschiedene Aktivierungsfunktionen
  - Bestimmung der Gewichte über lokale Ableitungen (Gradienten)



- Wie funktionieren künstliche neuronale Netze grundsätzlich?
  - Vernetzung von Neuronen
  - Verschiedene Layerarten



Zusammenfassung der Leitfragen

- Welche Funktionsweise haben die unterschiedlichen Layer-Arten?
  - Fully Connected Layer
  - Convolutional Layer
  - Pooling Layer

- Wie funktionieren Convolutional Neural Networks im Detail, und warum sind sie so effektiv bei der Verarbeitung von Bilddaten?
  - Komprimierung der Eingangsdaten → Verkürzung der Rechenzeit im Gegensatz zu Fully Connected Netzen
  - Verschiedene Filter, die auf unterschiedliche Features ausgerichtet sind → Erkennung von unterschiedlichen Merkmalen wie Straßenschildern, Linien, Menschen, …