

Vorlesung 1: Einführung in künstliche Neuronale Netze

BA-INF 153: Einführung in Deep Learning für Visual Computing

Prof. Dr. Reinhard Klein

Nils Wandel

Informatik II, Universität Bonn

Einführung in Deep Learning für Visual Computing

Wichtige Anwendungen von Deep Learning in Visual Computing



Figure: Computer Vision: z.B. Autonomes Fahren oder Tumorerkennung

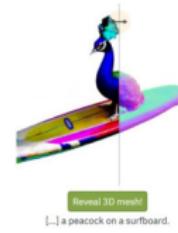


Figure: Computer Grafik: z.B. Generierung von Bildern (Dall-e2), Videos (SORA) oder 3D Modellen (Magic3D) aus Textbeschreibungen

Einführung in Deep Learning für Visual Computing

Vorlesungsziele - Warum sollte ich diese Vorlesung besuchen?

- Erste praktische Erfahrung mit Deep Learning im Bereich von Visual Computing (mit Pytorch)
- Vermittlung von soliden mathematischen Grundlagen für Deep Learning
- Heranführung an neue Entwicklungen auf dem Gebiet von Visual Computing

Einführung in Deep Learning für Visual Computing

Vorlesungsinhalte

- Einführung in künstliche Neuronale Netze
- Wahrscheinlichkeitstheorie
- Gradient Based Optimization
- Regularisierung
- Convolutional Neural Networks (CNNs)
- Autoencoders
- Generative Modelle
 - Autoregressive Models (AR)
 - Variational Autoencoder (VAE)
 - Generative Adversarial Networks (GAN)
 - Flow-based Models
 - Denoising Diffusion Probabilistic Models (DDPM)
- Recurrent Neural Networks (RNNs)
- Transformers

Einführung in Deep Learning für Visual Computing

Voraussetzungen

Grundlegende Kenntnisse in:

- Lineare Algebra
- Analysis
- Angewandte Mathematik: Numerik oder Stochastik (wir empfehlen beide Veranstaltungen zu diesen Themen zu hören)
- Programmieren (z.B. Python oder Matlab oder C++)

Organisation

Themen für heute

- Organisation
- Was bedeutet "Visual Computing"?
- Welche Herausforderungen gibt es in Computer Vision und Computer Graphics?
- Was bedeutet "Deep Learning" und "Machine Learning"?
- Einsatzbeispiele von Machine Learning in Computer Vision und Computer Graphics
- Einführung in künstliche neuronale Netze

Organisation

Kontaktpersonen

- Prof. Dr. Reinhard Klein (rk@cs.uni-bonn.de)
- Nils Wandel (wandeln@cs.uni-bonn.de)
- Jan Uwe Müller (muellerj@cs.uni-bonn.de)
- Alina Pollehn (s6aapoll@uni-bonn.de)
- Johannes van de Locht (s6jovand@uni-bonn.de)

Organisation

Vorlesung

Website: <https://cg.cs.uni-bonn.de/course/DLVC-24>

Kreditpunkte: 6 KP

Ort: Hybrid (HS 7 + Zoom [Einladungslink])

Zeitslot: Mittwoch, 14:15-16:00 Uhr

Vorlesungszeit: 10.4. - 17.7.2024

Ausnahmen (keine Vorlesung!):

- Tag der Arbeit: 1.5.2024
- Dies Academicus: 15.5.2024
- Pfingstferien: 20.5.2024 - 24.05.2024

Organisation

Übungen

Übungsgruppenverteilung mit TVS (URL:

<https://puma.cs.uni-bonn.de/index.php>) **TVS-Password: DLVC-24-TVS**

- Mittwoch, 12:15-14:00 Uhr (Alina Pollehn)
- Mittwoch, 16:15-18:00 Uhr (Alina Pollehn)
- Donnerstag, 16:15-18:00 Uhr (Johannes van de Locht)

Beginn: Heute wird das erste Blatt ausgegeben. Bearbeitungszeit: 1 Woche.

⇒ Korrekturzeit: 1 Woche ⇒ erste Übungsstunde: 24.4.2024

Klausurzulassung

- 50% praktische Übungsaufgaben
- 50% theoretische Übungsaufgaben

Vorläufige Klausurtermine

- 02.08.2024
- 13.09.2024

... Achtung, diese Termine können sich noch geringfügig ändern! Genauere Information folgen im Laufe des Semesters.

⇒ meldet euch bitte rechtzeitig auf TVS / Basis / eCampus an...

Organisation

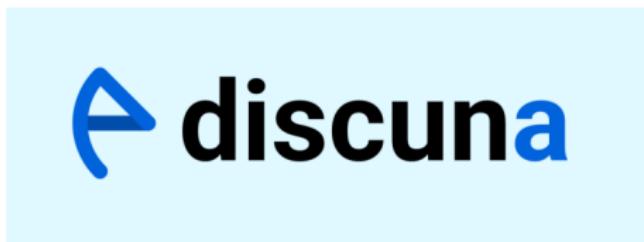
Abgabeformat

Reichen Sie Ihre Lösungen über eCampus in der von TVS zugewiesenen Übungsgruppe ein.

Beachten Sie folgende Bedingungen bzgl. des Formats Ihrer Lösungen:

- ① Abgabe der theoretischen Aufgaben als LaTeX/Word PDF (keine Fotos oder Scans).
- ② Verwenden Sie Python/PyTorch zum Lösen der praktischen Aufgaben.
- ③ Abgabe der praktischen Aufgaben erfolgen als
 - Jupyter Notebook in dem alle Zellen ausgeführt wurden und Ergebnisse gespeichert sind.
 - Python (*.py) Sourcedatei mit einem zusätzlichem PDF in dem alle Ausgaben und Figuren enthalten sind.

Organisation



Discuna

Wenn ihr Fragen zu den Vorlesungsslides oder Übungen habt, dann können wir diese gerne auch auf Discuna (<https://discuna.com/>) diskutieren.

Der Einladungslink zu dieser Vorlesung lautet:

<https://app.discuna.com/invite/dYdphnyaoUoBowxF7dZk>

Part 1

Was bedeutet Visual Computing?
Was bedeutet Deep Learning?

Was bedeutet Visual Computing?



Computer Vision



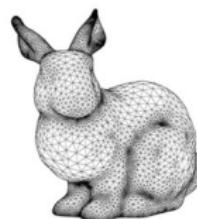
Bild

Pixelraster:
 $h \times w \times 3$

RGB-Werte:
rot, grün, blau



Bildbeschreibung



Computer Grafik



- zum Beispiel:
- Klasse: "Kaninchen"
 - Segmentierung
 - Textbeschreibung
 - Mesh
 - Textur
 - ...

Was bedeutet Visual Computing?



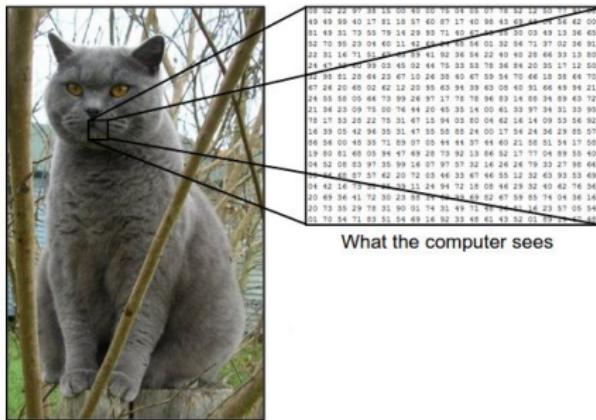
Computer Vision

In Computer Vision möchten wir Algorithmen entwickeln, welche eine intelligente Interpretation von visuellen Szenen ermöglichen (das bedeutet "sinnvolle Informationen aus Bildern / Videos extrahieren").

Computer Graphics

In Computer Graphics möchten wir Algorithmen entwickeln, welche aus Szenenrepräsentationen (z.B. Mesh / Textur / abstrakte Beschreibung / etc) neue Bilder generieren können.

Warum ist Computer Vision ein schweres Problem?



Bilder werden üblicherweise als 3D arrays ($h \times w \times 3$) mit Ganzzahlen zwischen [0,255] repräsentiert. (3 für die RGB Farbkanäle: rot, grün, blau)
Beispiel: $200 \times 100 \times 3$

⇒ Daraus sinnvolle Informationen zu extrahieren ist ein kompliziertes Problem (semantic gap)!

Warum ist Computer Vision ein schweres Problem?

Herausforderungen



Figure: Deformationen



Figure: Intraclass Variabilität

Warum ist Computer Vision ein schweres Problem?

Herausforderungen



Figure: Beleuchtung und Hintergrund

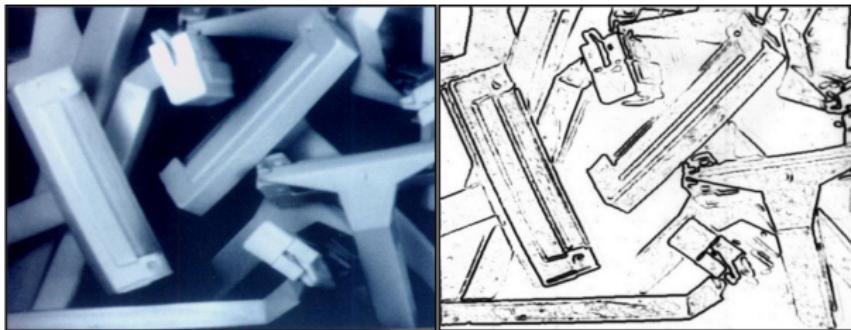


Figure: Verdeckungen

Warum ist Computer Vision ein schweres Problem?

Vorarbeiten

Kantendetektion mit manuell erstellten Edge-Features

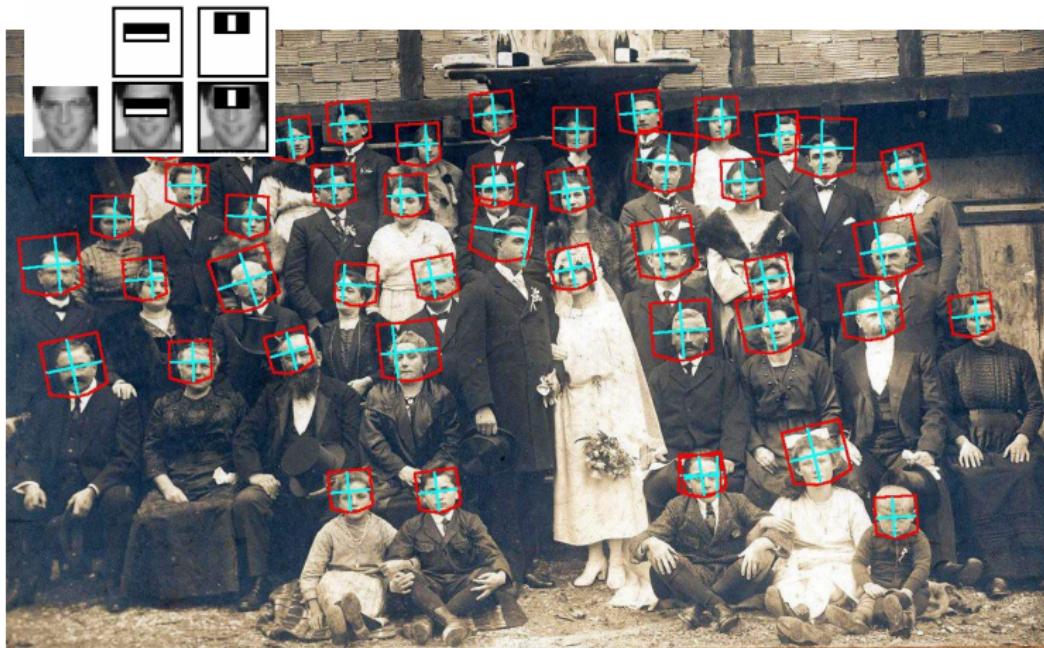


David Lowe, 1987

Warum ist Computer Vision ein schweres Problem?

Vorarbeiten

Gesichterkennung mit manuell erstellten Haar features

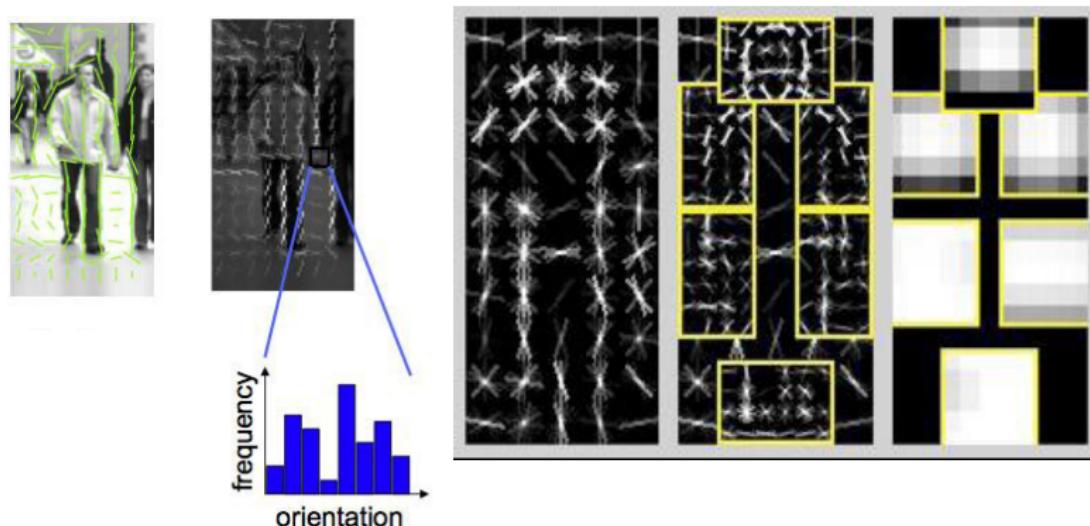


Face Detection, Viola & Jones, 2001

Warum ist Computer Vision ein schweres Problem?

Vorarbeiten

Personenerkennung mit Gradienten Histogrammen



Histogram of Gradients (HoG)
Dalal & Triggs, 2005

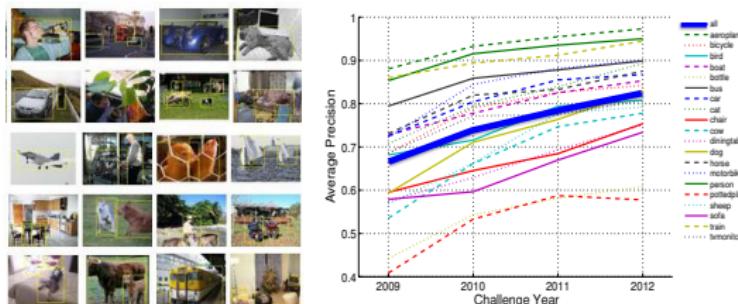
Deformable Part Model
Felzenswalb, McAllester, Ramanan,
2009

Warum ist Computer Vision ein schweres Problem?

Vorarbeiten

Objekterkennung

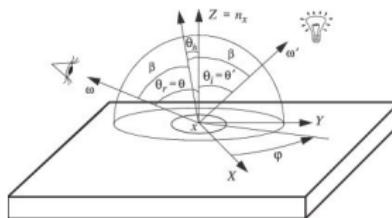
**PASCAL Visual Object Challenge
(20 object categories)**
[Everingham et al. 2006-2012]



- ⇒ Manuelles erstellen von Features ist sehr aufwendig und bei komplexeren Problemen quasi unmöglich.
- ⇒ Können wir solche Features mit Neuronalen Netzen lernen?

Warum ist Computer Graphics ein schweres Problem?

Beispiel: Rendering Equation



$$L(x, \vec{\omega}) = L_e(x, \vec{\omega}) + \int_{\Omega} f_r(x, \vec{\omega}', \vec{\omega}) L(x, \vec{\omega}') (\vec{\omega}' \cdot \vec{n}) d\vec{\omega}'$$

Die Rendering equation wird durch ein aufwändiges Integral beschrieben. Um dieses Integral zu berechnen müssen häufig rechenintensive Methoden wie z.B. Ray- bzw Pathtracing verwendet werden.

⇒ Können wir solche Methoden mit Neuronalen Netzen beschleunigen?

Warum ist Computer Graphics ein schweres Problem?

Beispiel: Superresolution



Figure: Source: Image Super-Resolution via Iterative Refinement [Link].

Ein niedrig aufgelöstes Bild (links) könnte durch eine Vielzahl von hoch aufgelösten Bildern erzeugt werden. Für das inverse Problem der Superresolution gibt es also verschiedene Lösungen und es ist somit schlecht gestellt. Um dennoch eine plausible Lösung zu erhalten müssen wir berücksichtigen, welche der hoch aufgelösten Bilder "realistisch aussehen" bzw wahrscheinlich sind.
⇒ Können wir solche Statistiken mit Neuronalen Netzen lernen?

Warum ist Computer Graphics ein schweres Problem?

Herausforderungen

- Rendering Equation / globale Beleuchtung ist aufwändig zu berechnen
- Abstrakte Szenenbeschreibung in Bild umwandeln
(z.B. "Mann mit Bart", "Frau mit Brille", etc)
- Style transfer
- realistische Interpolation zwischen Szenen
- Rauschen / Unschärfe beseitigen, Superresolution
- Hole filling

⇒ Dies sind häufig schlecht gestellte inverse oder rechenintensive Probleme. Wir müssen deshalb zusätzliche statistische Informationen hinzuziehen, um plausible Lösungen zu erzeugen bzw schneller zu berechnen.
⇒ Dafür kann man Deep Learning Methoden verwenden.

Was bedeutet Deep Learning?

Deep Learning \approx Machine Learning mit Deep Neural Networks

Machine Learning

"A computer program is said to learn form **experience E** with respect to some class of **tasks T** and **performance measure P** , if its performance at tasks in T , as measured by P , improves with experience E ."

"Man sagt, ein Computer Programm lernt von Erfahrung E bezüglich einer Klasse von Aufgaben T und Performanz-Maß P , wenn sich die Performanz auf den Aufgaben in T , gemessen durch P , mit der Erfahrung E verbessert."

(Mitchell, 1997¹)

⇒ Anstatt dem Computer einen detaillierten Algorithmus vorzugeben (z.B. mit manuell erstellten Features), soll der Computer selbst lernen, Aufgaben zu lösen (z.B. an Hand von Beispieldaten).

Deep Neural Networks

Algorithmen, die von biologischen Neuronalen Netzen inspiriert sind.

¹Mitchell, T. M. (1997). Machine Learning. McGraw-Hill, New York.

Part 2

Beispiele von Deep Learning in Computer Vision und Graphics

Objekt-Klassifizierung

MNIST

Das MNIST ("Modified National Institute of Standards and Technology") Datenset enthält Bilder von Ziffern.



Task: Klassifizierung von Bildern ($28 \times 28 \times 1$) in Klassen (Ziffern: 0,1,2,...,9).

Erfahrung: 60000 Beispiel-Bilder mit dazugehörigen Klassen-Labels.

Performanz: Akkurateit der Vorhersage der Ziffern.

Objekt-Klassifizierung

Food-101

The Food-101 Data Set



Figure: Source: https://data.vision.ee.ethz.ch/cvl/datasets_extra/food-101/

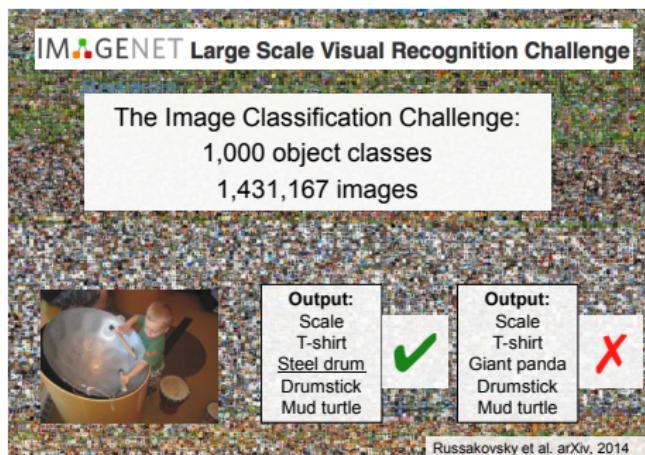
Task: Bilder (Auflösung kann variieren. Maximal: $512 \times 512 \times 3$) in 101 Speisen klassifizieren (z.B. Schokoladenkuchen, Cäsar-Salat, etc)

Erfahrung: 101000 Beispiel-Bilder mit dazugehörigen Klassen-labels

Performanz: Akkurateit der Vorhersage der Speise-Klassen

Objekt-Klassifizierung

ImageNet



Task: Bilder (Auflösung kann variieren. Üblicherweise: $256 \times 256 \times 3$) in 1000 verschiedene Klassen einteilen (z.B. Flugzeug, Parkbank, Goldfisch, etc)

Erfahrung: > 1 Mio Beispiel-Bilder mit dazugehörigen Klassen-labels

Performanz: Akkurateit der Vorhersage der Klassen

Video / Action Klassifizierung



Figure: Video Classification on UCF-101 Action Recognition dataset with Convolutional Neural Networks [link]

Task: Videos (Anzahl Frames \times 170 \times 170 \times 3) in verschiedene Sportarten (z.B. track cycling, ultramarathon, ski touring, etc) unterteilen

Erfahrung: 1 Mio Youtube videos

Performanz: Akkurateit der Vorhersage der verschiedenen Sportarten

Surface Reflectance Modeling

In der Computer Grafik benötigt man häufig die Parameter einer (Spatially Varying) Bidirectional Reflectance Distribution Function ((SV)BRDF), um die Reflektionen von Materialien korrekt rendern zu können.

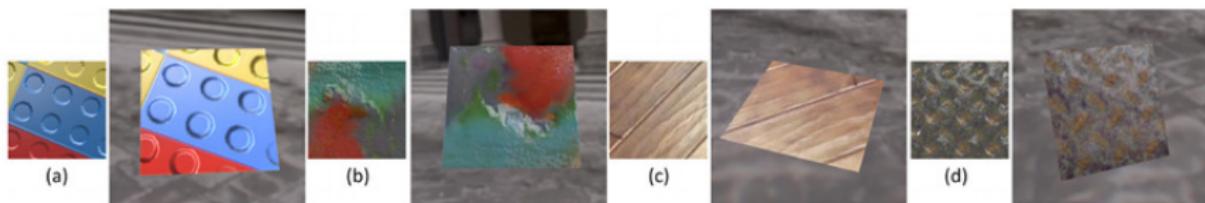


Fig. 1. Physically plausible spatially varying surface appearance estimated using the proposed SA-SVBRDF-net from a single photograph of planar spatially varying plastic (a,b), wood (c) and metal (d) captured unknown natural lighting, and revisualized under a novel lighting condition.

Figure: Modeling Surface Appearance from a Single Photograph using Self-augmented Convolutional Neural Networks [link]

Task: Aus Bildern ($w \times h \times 3$) den specular Albedo, die Rauheit, den spatially varying diffuse Albedo und surface normals ($w \times h \times (3+1+3+3)$ Parameter) schätzen

Erfahrung: Beispieldaten

Performanz: Genauigkeit der Vorhersage von BRDF / SVBRDF Parametern

Bild-Segmentierung

Eine Segmentierung kann als pixelweise Klassifizierung interpretiert werden:



Task: Bilder ($h \times w \times 3$) in Bereiche (z.B. Person, Fahrrad, etc) segmentieren ($h \times w \times$ Anzahl Klassen)

Erfahrung: Beispieldaten

Performanz: Genauigkeit der Vorhersage der verschiedenen Segmente

Voxel-Segmentierung

Auf medizinischen 3D daten (z.B. CT oder MRI) können Organe segmentiert werden:

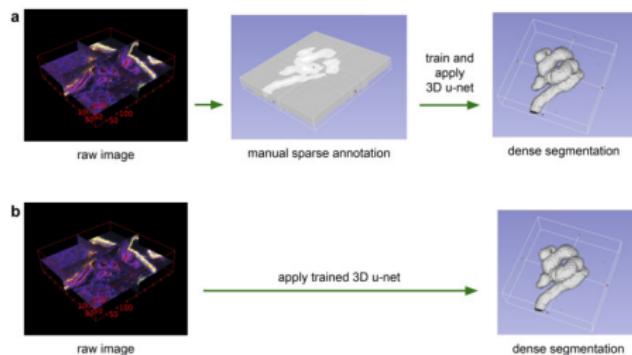


Figure: Source: 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation [Link].

Task: 3D Bilder ($h \times w \times d \times 1$) segmentieren ($h \times w \times d \times$ Anzahl Klassen)

Erfahrung: 3D Scans mit Labels

Performanz: Genauigkeit der Vorhersage der verschiedenen Segmente

Object-Detection

Was ist, wenn mehrere Instanzen der selben Klasse (z.B. mehrere Personen) im Bild zu sehen sind?

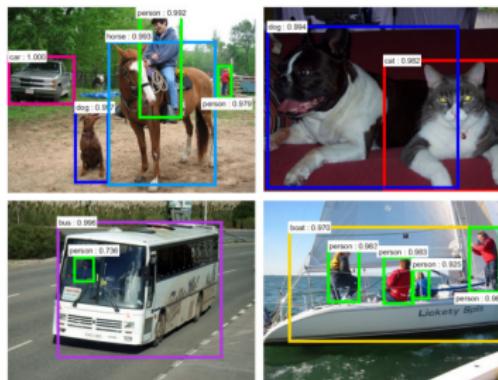


Figure: Source: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks [Link].

Task: in Bildern ($h \times w \times 3$) Bounding Boxes + Klassen-Labels erkennen

Erfahrung: Beispieldaten (z.B. MS COCO)

Performanz: Präzision der Bounding-Boxen für jedes Objekt.

Instanz-Segmentierung

Was ist, wenn wir mehrere Instanzen der selben Klasse (z.B. mehrere Personen) in einem Bild segmentieren möchten?



Figure: Source: Mask R-CNN [Link].

Task: in Bildern ($h \times w \times 3$) Bounding Boxes + Klassen + Segmentierungsmasken erkennen

Erfahrung: Beispieldaten (z.B. MS COCO)

Performanz: Genauigkeit der Segmentierungsmasken für jede Bounding-Box.

Human Pose Estimation

Aus Bildern können menschliche Posen geschätzt werden:

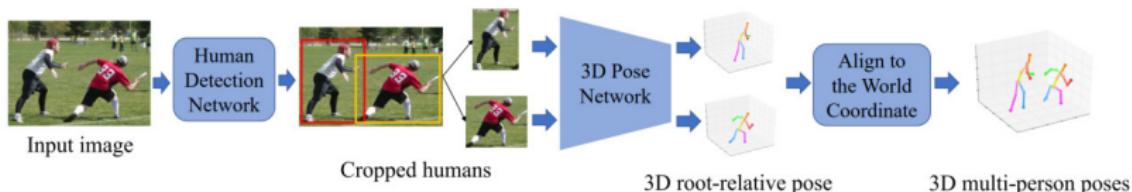


Figure: Source: Deep Learning-Based Human Pose Estimation: A Survey [Link].

Task: in Bounding-Box Bildern ($h \times w \times 3$) Gelenkpositionen (z.B. 18×3 für 18 Gelenke in 3D) erkennen

Erfahrung: Beispieldaten (z.B. MPII Human Pose Dataset)

Performanz: Präzision der Gelenkpositionen für jede Bounding-Box.

Image Captioning

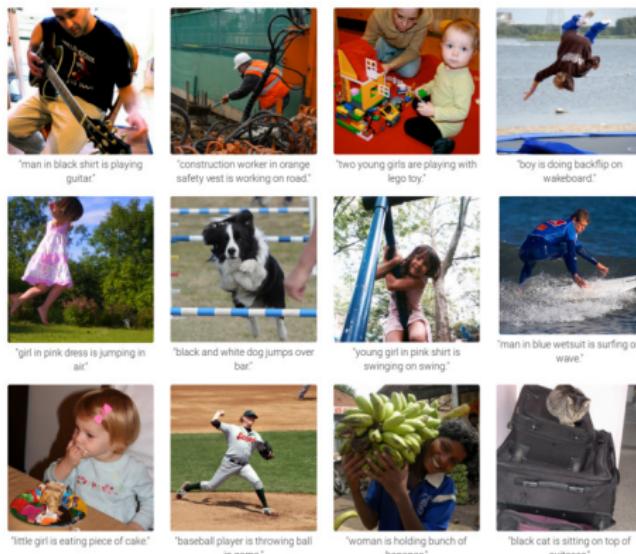


Figure: Deep Visual-Semantic Alignments for Generating Image Descriptions[\[link\]](#)

Task: zu Bildern ($w \times h \times 3$) eine Bildbeschreibung ausgeben

Erfahrung: Beispieldaten (z.B. Flickr30K)

Performanz: Möglichst gute Beschreibung des Bildinhaltes

Image Story Telling



Generated story about image
Model: Romantic Novels

"My bird was making a lot of effort, absorbing her words . I put my hand on the back of her neck , and then he lifted his head to look at her . Of course , I don't know what else to do , so it was only a matter of time .

I could n't help but notice that the girl who had fallen asleep in his arms invited him to play with her . For God's sake, it was as if they were the most precious thing in the world."

Figure: The Neural-Storyteller (2015). [link]

Super-Resolution

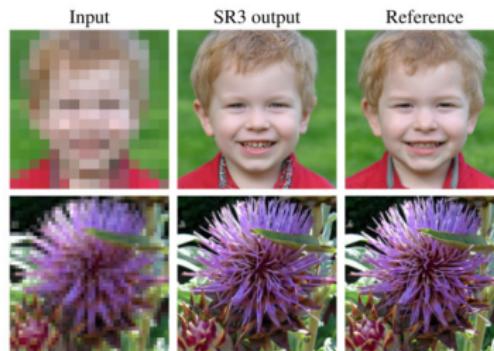


Figure: Image Super-Resolution via Iterative Refinement [Link].

Task: Bilder $(h \times w \times 3)$ in Bilder mit s -fach höherer Auflösung $((s \cdot h) \times (s \cdot w) \times 3)$ umwandeln

Erfahrung: Hochauflöste Beispielbilder

Performanz: Realismus der hoch aufgelösten Bilder

Bildgenerierung mit Generative Adversarial Networks (GANs)

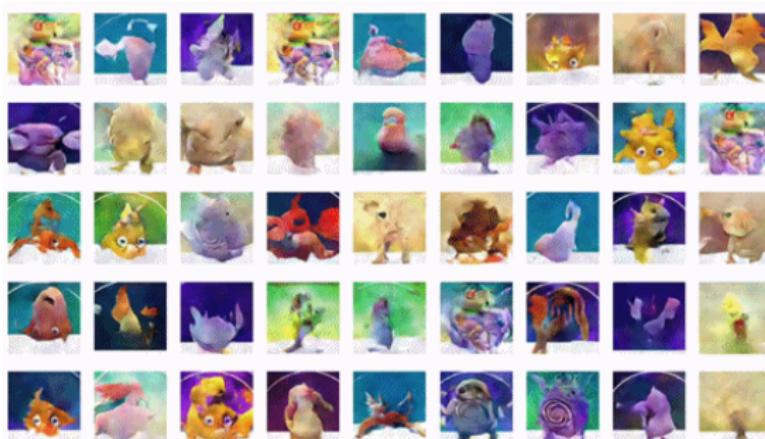


Figure: Pokemon and Neural Networks (2016): generating Pokemon characters [link]

Task: Neue "Pokemon" Bilder ($h \times w \times 3$) erstellen.

Erfahrung: Beispielbilder

Performanz: Realismus und Vielfalt der generierten Pokemon Bilder.

Bildgenerierung mit Neuronalen Netzen

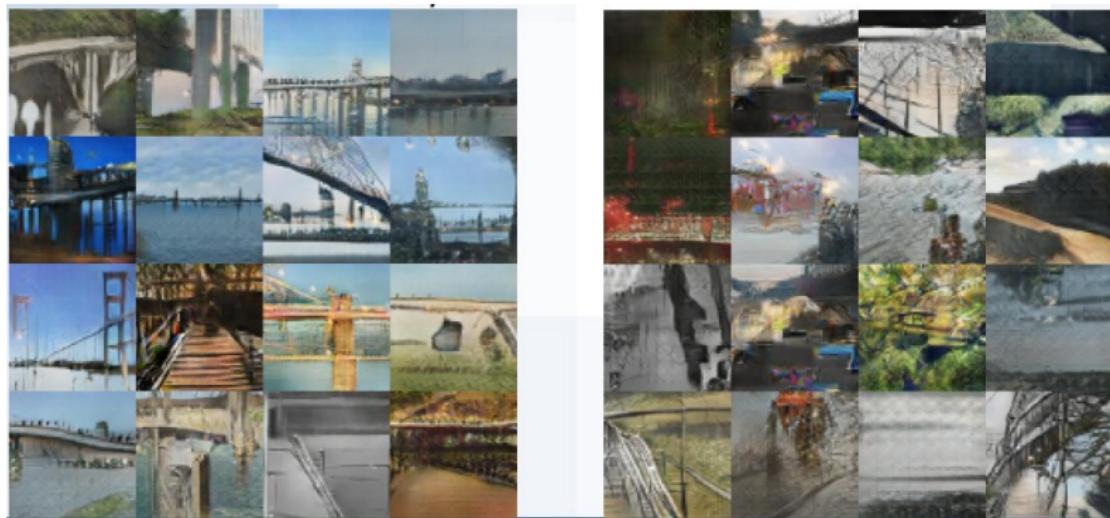


Figure: A DCGAN (Deep Convolutional GAN) trained on the LSUN bridges dataset for 35 epochs. On the left are some of the more believable results, on the right the not so much. (Image: Vahe Hakobyan, Mihail Luchian)

Task: Neue Bilder ($h \times w \times 3$) von Brücken erstellen.

Erfahrung: Beispielbilder

Performanz: Realismus und Vielfalt der generierten Bilder.

Vektorarithmetik im Latenten Vektorraum

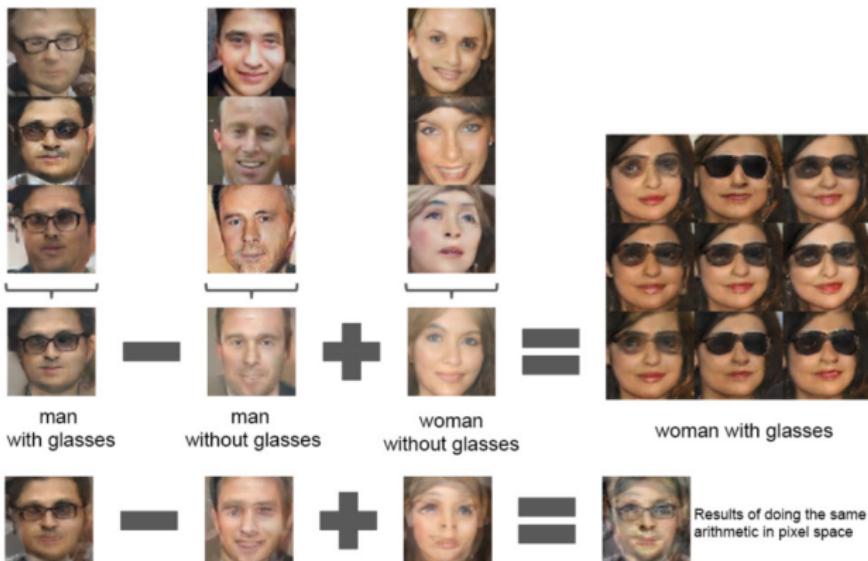


Figure: Unsupervised representation learning with deep convolutional GANs [link]

Im latenten Vektorraum können Bildeigenschaften arithmetisch manipuliert und realistisch interpoliert werden. Im Pixelraum wäre dies nicht möglich.

Artbreeder

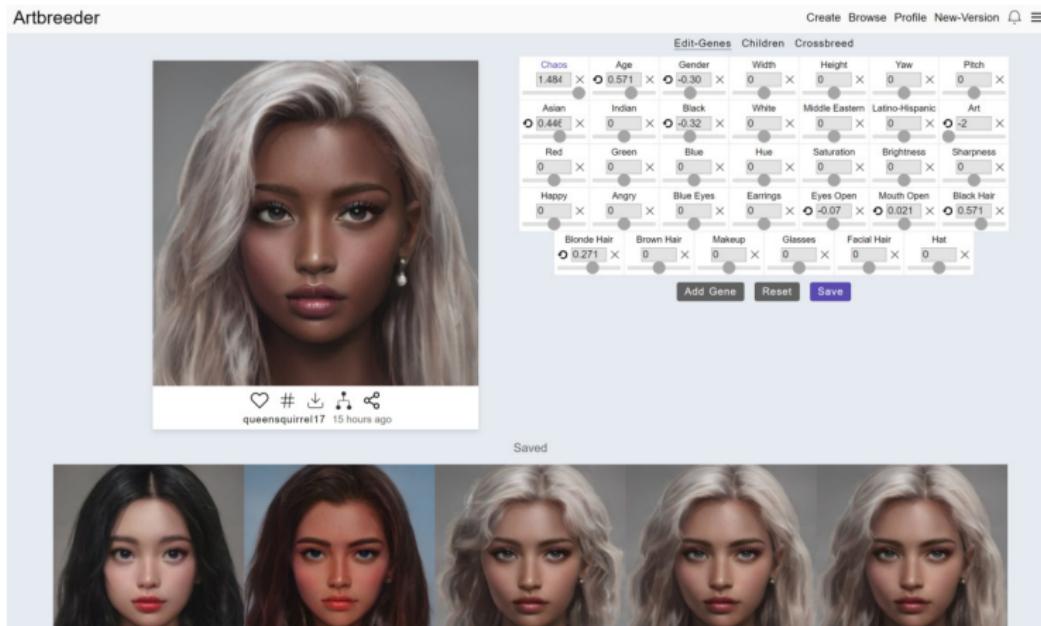


Figure: Bei Artbreeder kannst du eigene Gesichter mit einem GAN erstellen [link]. Die Slider entsprechen Vektoren im Latent space und können gewissen Eigenschaften (wie z.B. Geschlecht, Alter etc) zugeordnet werden.

GauGAN 2

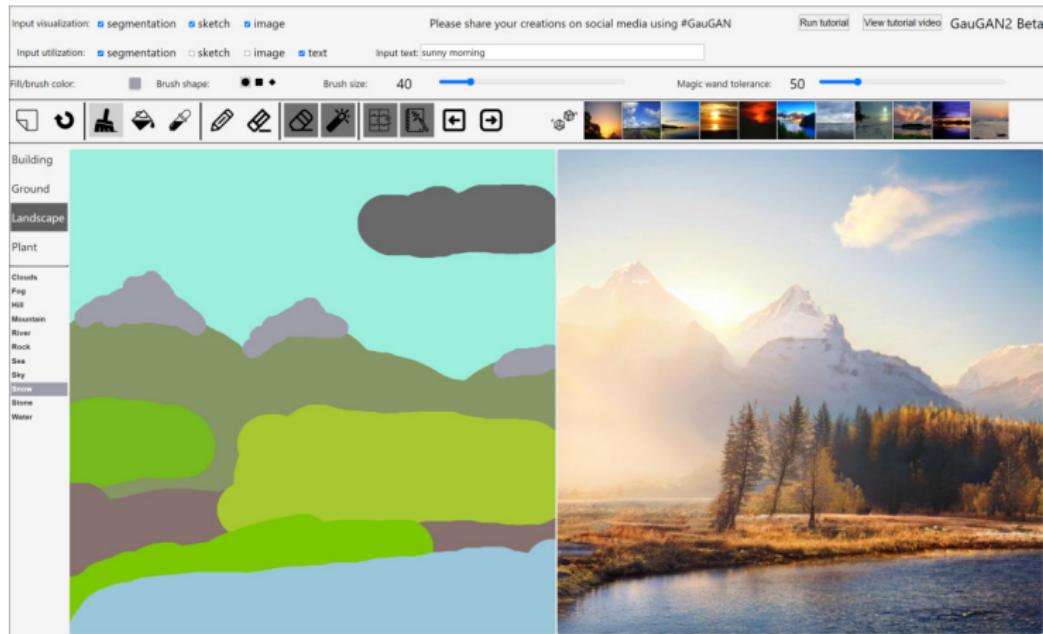


Figure: Mit GauGAN 2 von Nvidia kann man einfach echt wirkende Landschaften erstellen. [Hier] gibt es eine Demo dazu.

Style Transfer mit Neuronalen Netzen



Figure: A Neural Algorithm of Artistic Style [link] : (Photo: Andreas Praefcke).

Task: Content image ($h \times w \times 3$) und Style image ($h \times w \times 3$) in Stilisiertes Bild ($h \times w \times 3$) umwandeln.

Erfahrung: Beispielbilder

Performanz: Realismus und Vielfalt der Pokemon Bilder.

Style Transfer mit Neuronalen Netzen

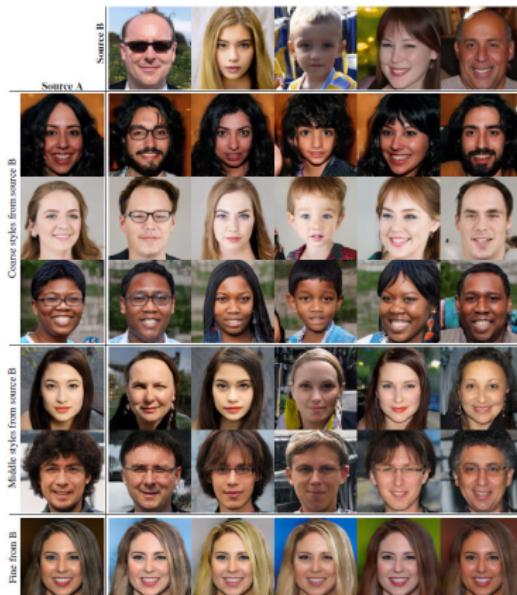


Figure: A Style-Based Generator Architecture for Generative Adversarial Networks [link].

Style Transfer zwischen Bildern von Gesichtern.

Bildgenerierung aus Text-Beschreibung



Figure: "A happy student learning about deep learning and computer graphics. pop art.", erstellt mit Hilfe von [Bing Image Creator]

Mit Hilfe von Tools wie Dall-e2, Midjourney oder Stable Diffusion lassen sich Bilder aus Textbeschreibungen generieren.

Task: Bild ($h \times w \times 3$) aus Textbeschreibung (String) generieren

Erfahrung: Beispielbilder mit Textbeschreibungen

Performanz: Realismus und Texttreue der generierten Bilder.

Playing Games

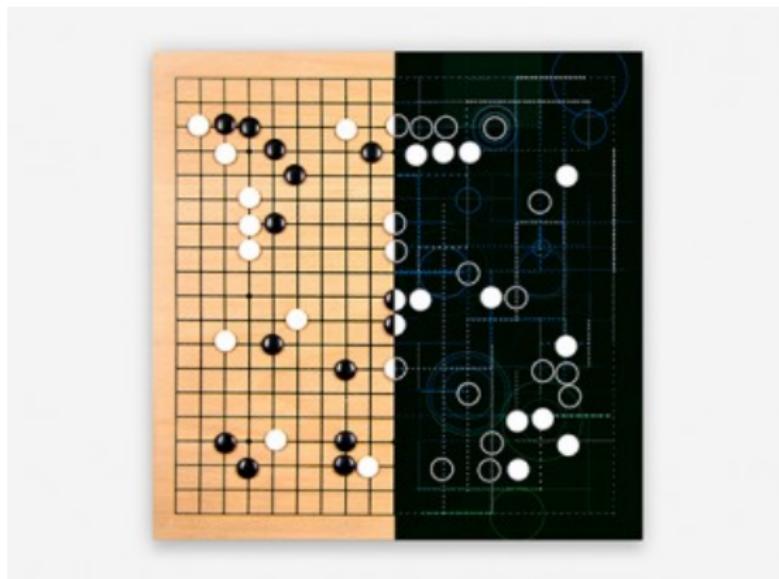


Figure: Google Deep-Mind schlägt den World-Champion in Go ein Jahrzehnt früher als erwartet. [link]

Task: Finde möglichst gute Spielzüge

Erfahrung: Beispiel-Daten von Profi-Spielern und Self-Play

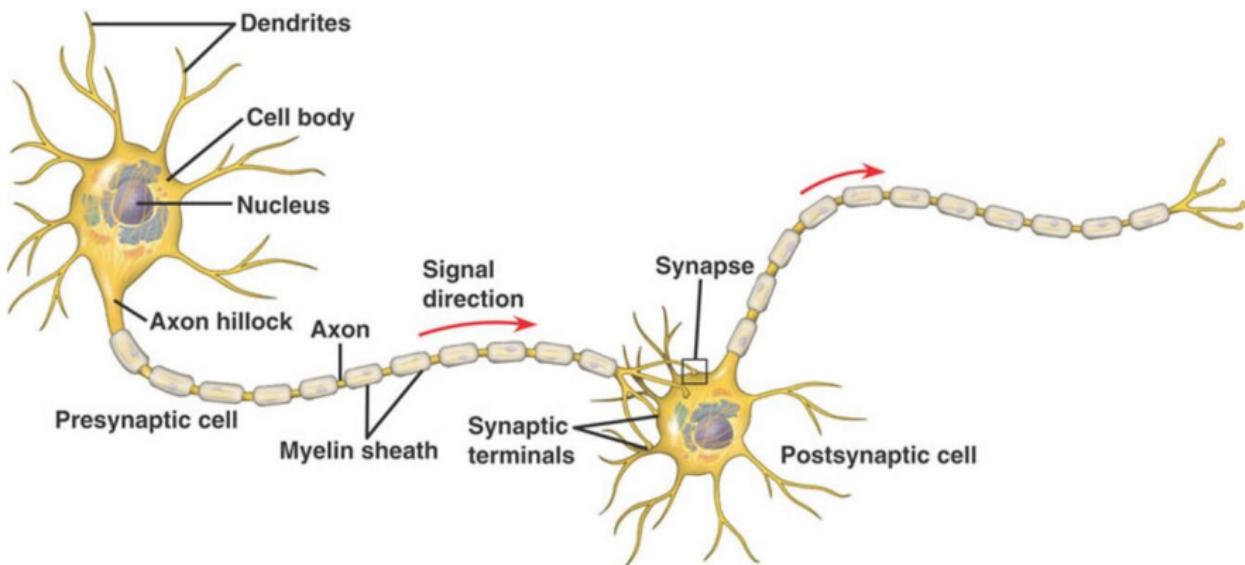
Performanz: Wie oft ein Spiel gewonnen wird

Part 3

Künstliche Neuronale Netze

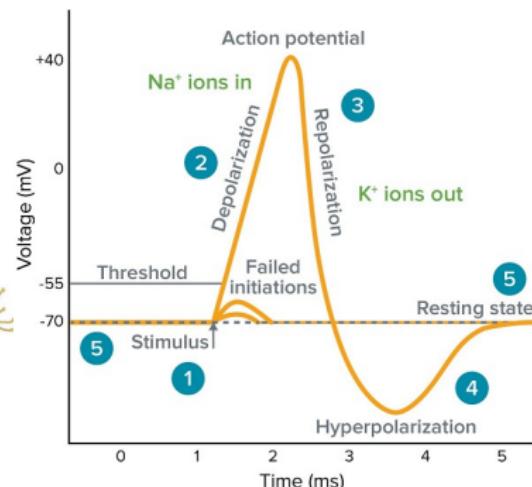
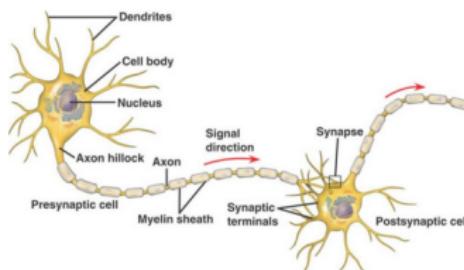
Biologische Inspiration

Signalverarbeitung in Neuronen

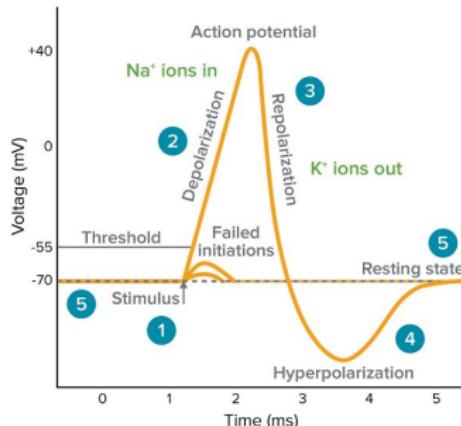
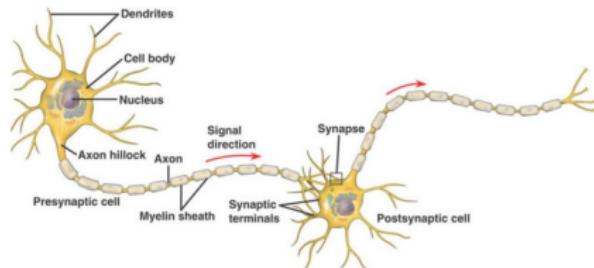


Biologische Inspiration

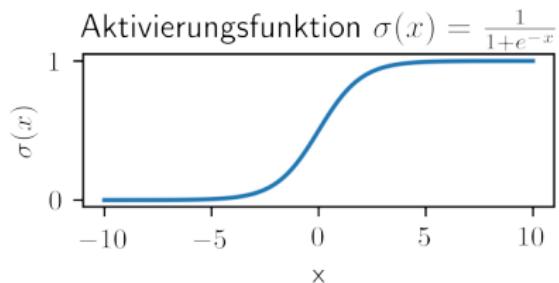
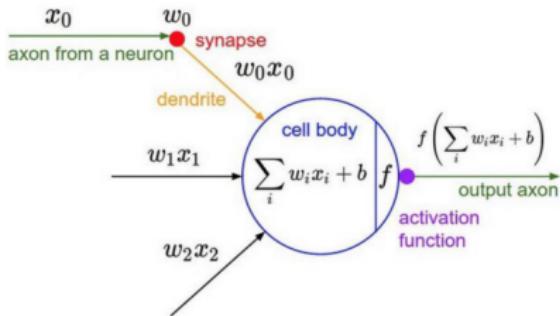
Signalverarbeitung in Neuronen



Künstliche neuronale Netze



Mathematisches Modell und Beispiel für Activation Function (Sigmoid)



Künstliche neuronale Netze

Einfaches Multilayer Perceptron (MLP)

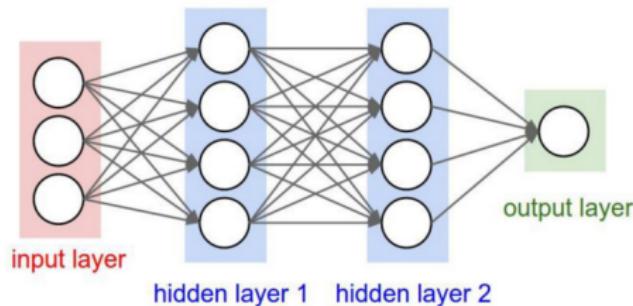
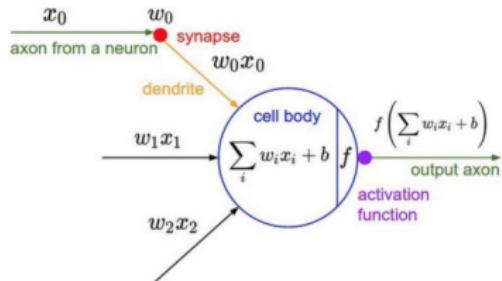
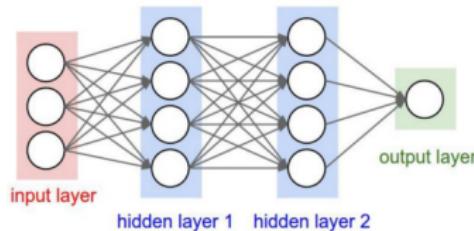
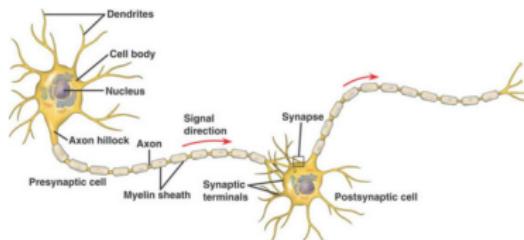


Figure: [Hier] gibt es eine interaktive Web-Demo zu MLPs.

Künstliche neuronale Netze



Vergleich zwischen künstlichen und biologischen neuronalen Netzen

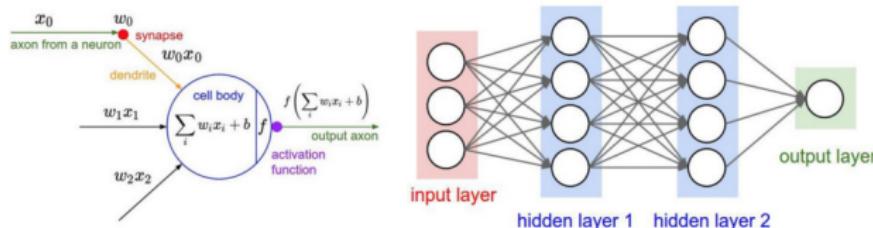
Ähnlichkeiten:

- Synapsen und synaptische Gewichte
- Membranpotential / Aktivierungsfunktion
- Abstraktionsebenen

Unterschiede (künstliches vs biologisches NN):

- Neuronenausgabe (firing rate vs individual spikes)
- Lernprozess (training mit gradient descent statt Hebbian learning)
- Verknüpfungen zwischen Neuronen (feed forward vs recurrent)

Künstliche NN sind parametrisierte Funktionen



Wir können ein neuronales Netz als eine parametrisierte Funktion auffassen:

$$f_{\theta} : X \rightarrow Y, x \mapsto y$$

Hierbei beschreibt θ die Parameter des neuronalen Netzes (z.B. die synaptischen Gewichte w_i und die Bias-Terme b).

X und Y sind die Definitions- und Zielmenge von f_{θ} und entsprechen dem Eingaberaum des Inputlayers bzw dem Ausgaberaum des Outputlayers. Sie werden üblicherweise durch die Problemstellung (z.B. Klassifizierung / Segmentierung / Bildgenerierung / etc) vorgegeben.

⇒ Unser Ziel ist es nun, die Parameter θ so zu optimieren, sodass die Abbildung das Problem gut löst. (Mehr dazu später...)