
iteratec

iteratec

Machine Learning with a smile 😀

Eine kurze Einführung in die Programmierung von neuronalen Netzwerken mit Python und Tensorflow

Yves Schubert

07.10.2022 | XP Days Germany 2022

iteratec

AGENDA

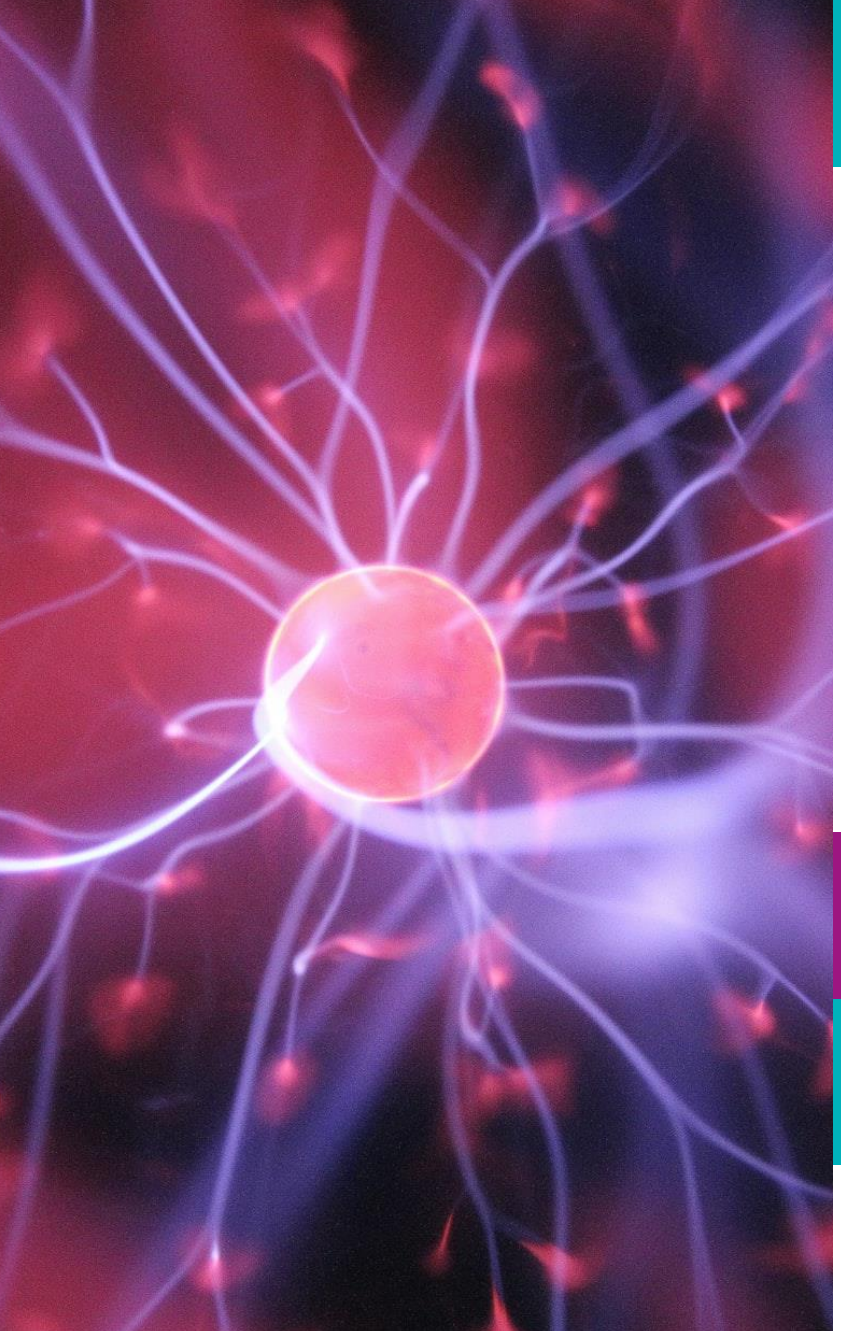
- Kurze Einführung in die Theorie
- Das Projekt
- Erste Schritte
- Jetzt wird gefaltet
- Unser Werkzeugkoffer

Zu meiner Person

- Yves Schubert
  - Verheiratet, 2 Kinder
  - Dipl. Inf. in Softwaretechnik an der Universität Stuttgart 2010
  - "Senior Software Architect" bei der iteratec GmbH
- Technologische Interessen:
  - Softwarearchitektur und Softwarequalität im Allgemeinen
  - Machine Learning
  - Internet Of Things

- Speaker bei Konferenzen, Trainer für DevOps und QS, Dozent an der DHBW (Web-Programmierung)

- Twitter: https://twitter.com/YvesSchubert
- Github: https://github.com/scyv



iteratec

Neuronale Netzwerke

Simple, yet so powerful

Zunächst: Was ist Machine Learning

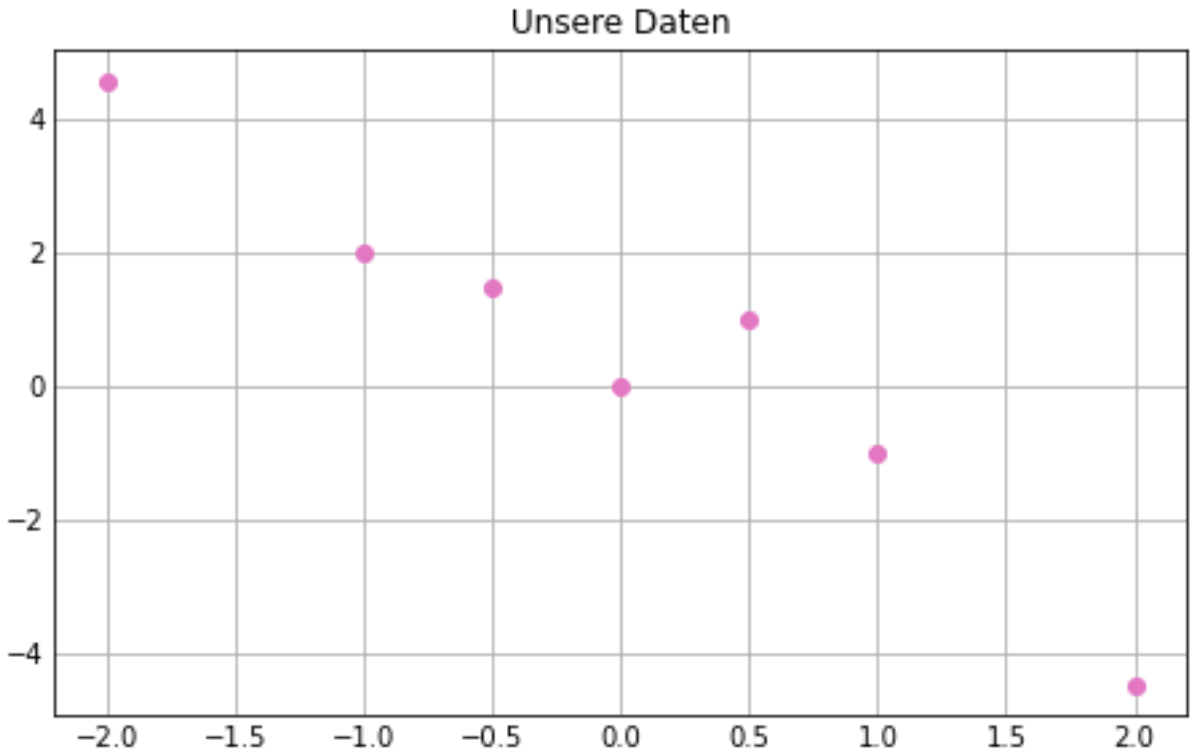
Eine Definition

- Ein System, das vorgegebenes Wissen abstrahiert ("lernt"), sodass es bisher unbekanntes Wissen einordnen kann.
- Ziel: Abstraktion von Daten mit Hilfe eines mathematischen Modells
- Zutaten:
  - Daten (!)
  - Ein Modell
  - Eine Optimierungsfunktion („Loss Function“)
  - Lernphase

- Mathematisch ausgedrückt: **Machine Learning** ist der Versuch, durch **schrittweises Anpassen von Parametern ("lernen")**, eine **Funktion** zu finden, die möglichst gut **bereits bekannte Datenpunkte** beschreibt und darüber hinaus in der Lage ist, **die Werte unbekannter Datenpunkte vorherzusagen**.

So funktioniert Machine Learning

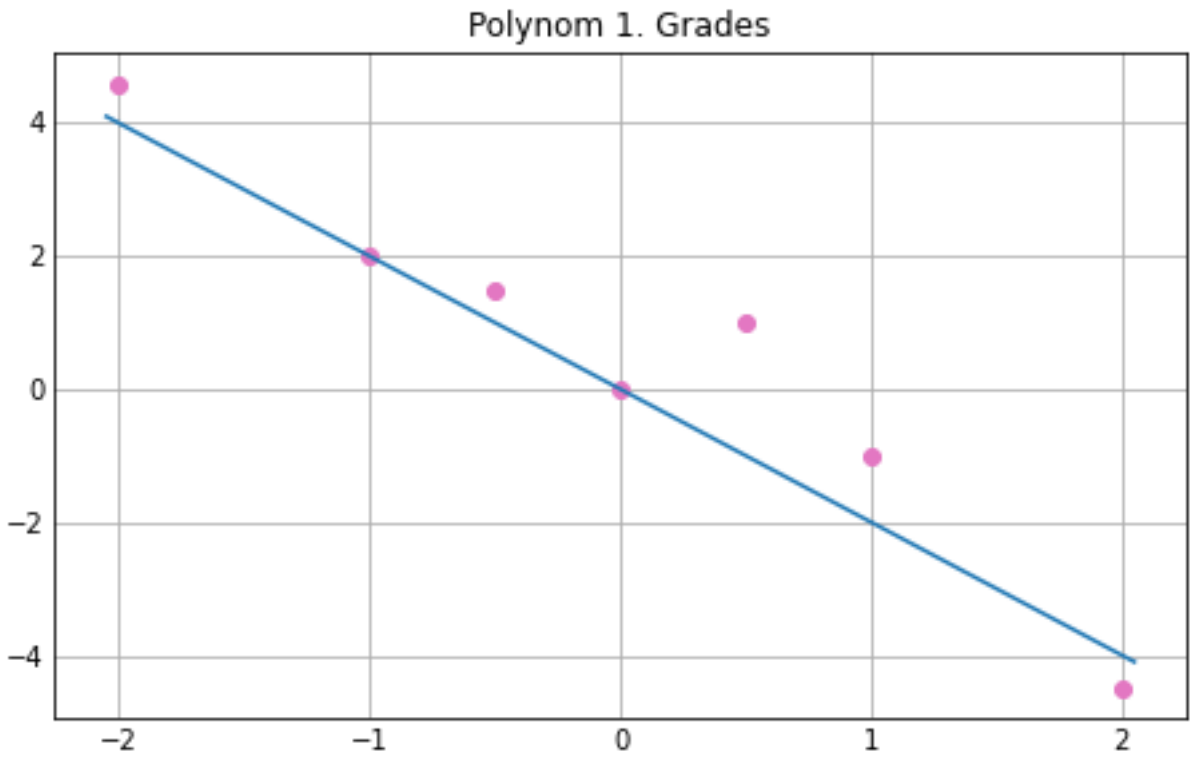
Stark vereinfacht



- Zunächst braucht man eine Menge von Daten

So funktioniert Machine Learning

Stark vereinfacht

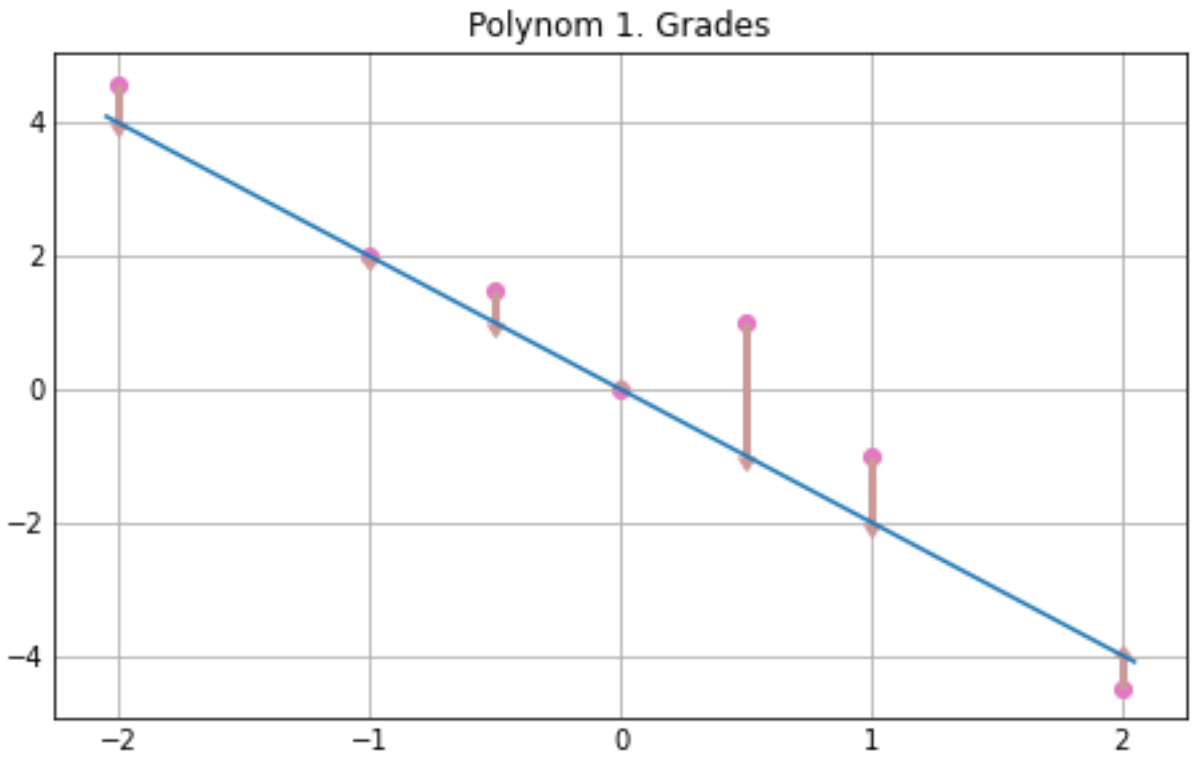


$$y = -2x$$

- Zunächst braucht man eine Menge von Daten
- Dann fängt man mit einer beliebigen Funktion an (das Modell)

So funktioniert Machine Learning

Stark vereinfacht

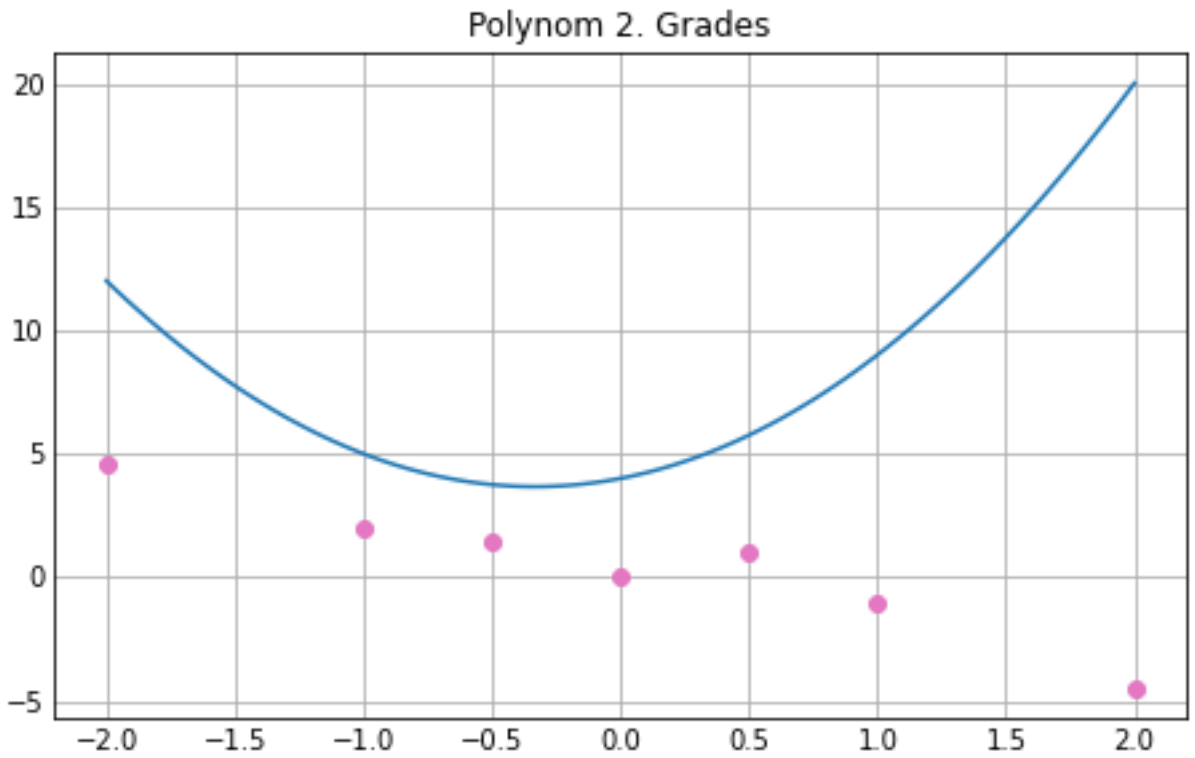


$$y = -2x$$

- Zunächst braucht man eine Menge von Daten
- Dann fängt man mit einer beliebigen Funktion an (das Modell)
- Man bestimmt eine Fehler Funktion („Loss Function“)

So funktioniert Machine Learning

Stark vereinfacht

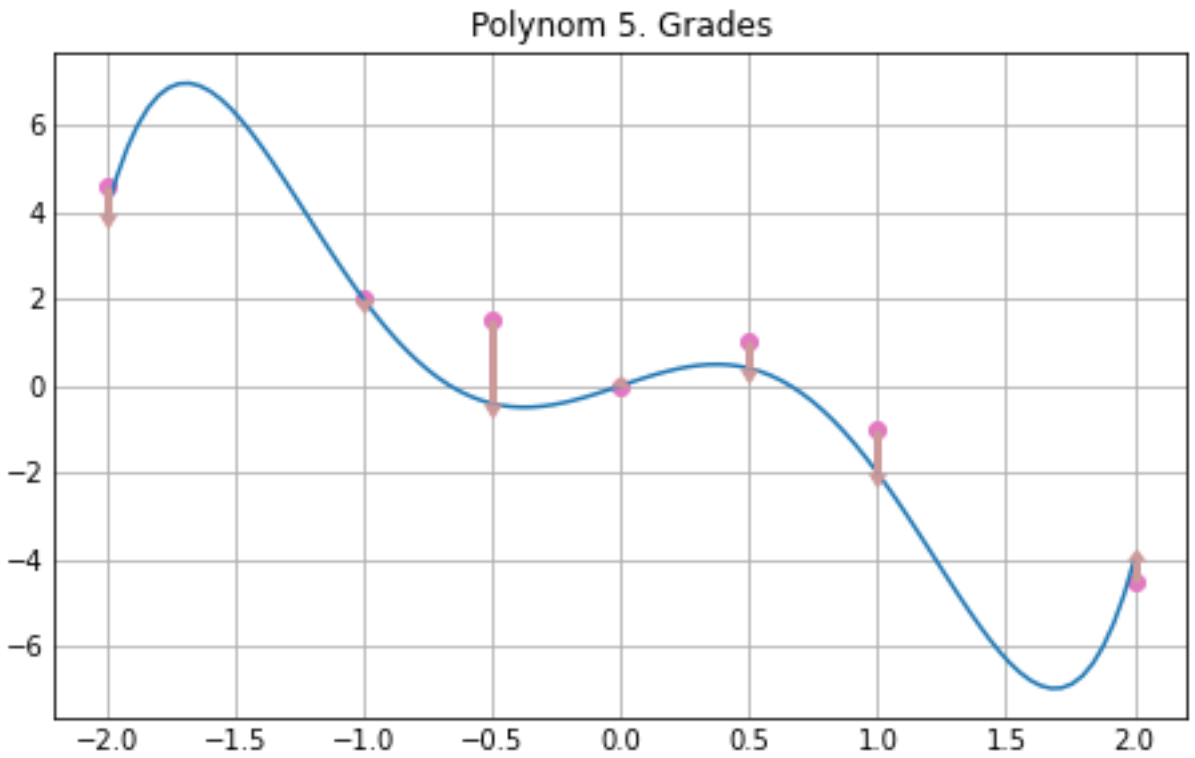


$$y = 2x + 3x^2$$

- Zunächst braucht man eine Menge von Daten
- Dann fängt man mit einer beliebigen Funktion an (das Modell)
- Man bestimmt eine Fehler Funktion („Loss Function“)
- Nun nähert man sich (numerisch) schrittweise einem Optimum (möglichst kleiner Fehler), indem man Parameter anpasst (hier der Grad des Polynoms und die Koeffizienten)

So funktioniert Machine Learning

Stark vereinfacht

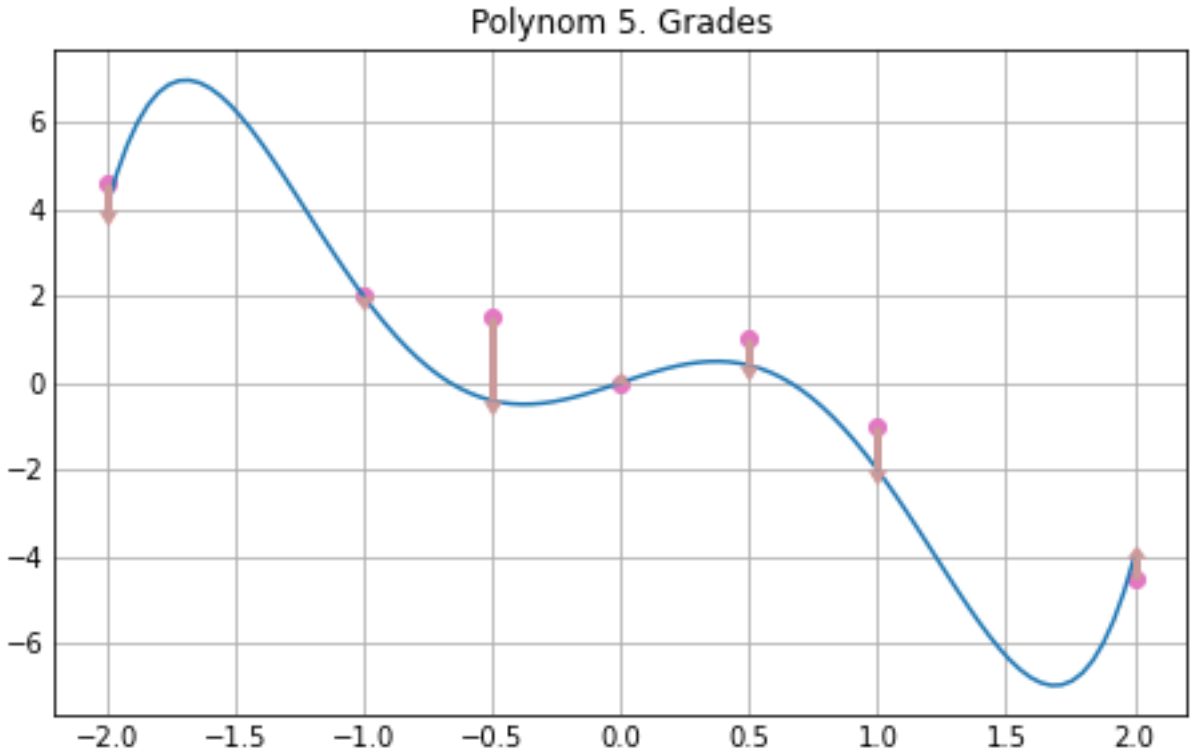


$$y = 2x - 5x^3 + x^5$$

- Zunächst braucht man eine Menge von Daten
- Dann fängt man mit einer beliebigen Funktion an (das Modell)
- Man bestimmt eine Fehler Funktion („Loss Function“)
- Nun nähert man sich (numerisch) schrittweise einem Optimum (möglichst kleiner Fehler), indem man Parameter anpasst (hier der Grad des Polynoms und die Koeffizienten)

So funktioniert Machine Learning

Stark vereinfacht



$$y = 2x - 5x^3 + x^5$$

- Je mehr Datenpunkte vorhanden sind, umso besser wird das Modell
- Vorsicht vor Underfitting und Overfitting
  - Underfitting: Das Modell hat noch nicht genug trainiert
  - Overfitting: Das Modell hat sich zu stark an die Trainingsdaten angepasst und ist nicht mehr abstrakt genug => „Keine Mustererkennung mehr“

iteratec

Herausforderung: Bias

- Bias ist der größte Feind im Machine Learning
- Systematische Effekte, die dazu führen, dass das System bereits „Vorurteile“ lernt
  - Unbeabsichtigtes Verfälschen der Daten
  - Erkennung von Gesichtern, aber alle Bilder sind bei Tag erstellt
  - Erkennung von Gesichtern, aber alle sind Jung und Hellhäutig
- Verschlechtert die Qualität der Daten

- Kein allgemeines Rezept, Bias zu erkennen oder zu entfernen

Strategien des Lernens

Supervised Learning

Lernen anhand von festen Beispielen und Vorgeben der Ausgaben

- Klassifikation, Regression

Unsupervised Learning

Lernen durch Mustererkennung und Clustering von Daten ohne Konkrete Angabe der Ausgaben

- Clustering, Zusammenhänge erkennen

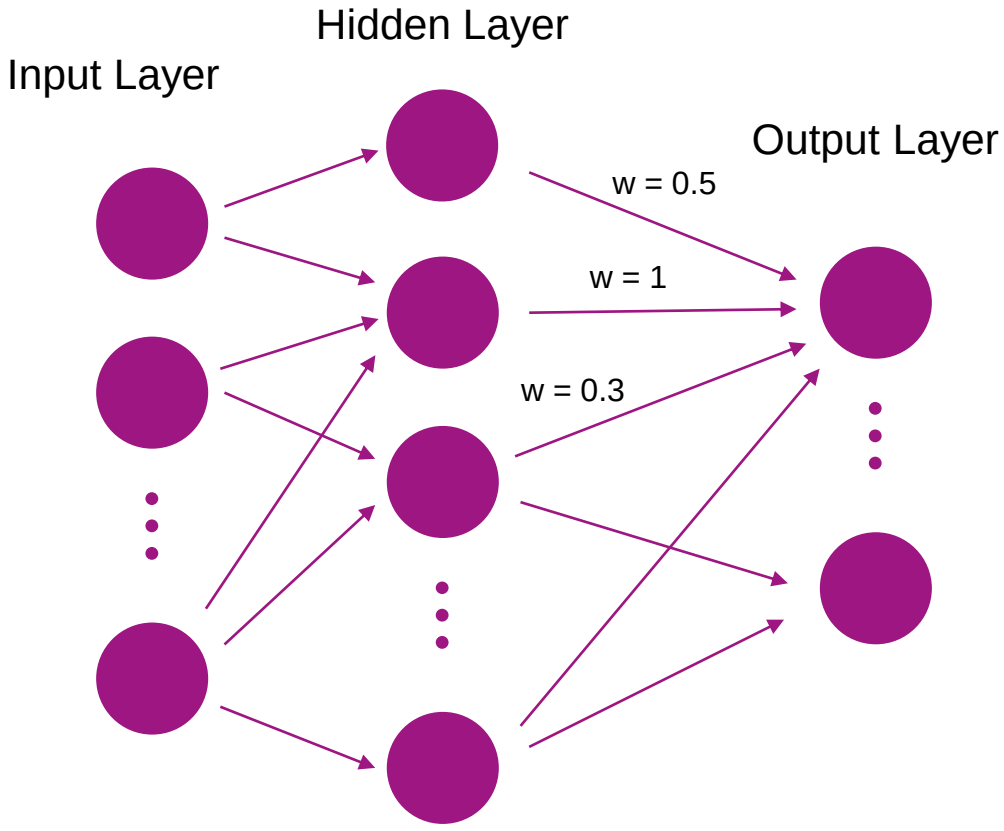
Reinforcement Learning

Lernen unter Verwendung eines Belohnungssystems (Keine Angabe von Eingaben oder Ausgaben)

- Simulation, „AlphaGo“

Neuronale Netzwerke

Die Natur kopiert

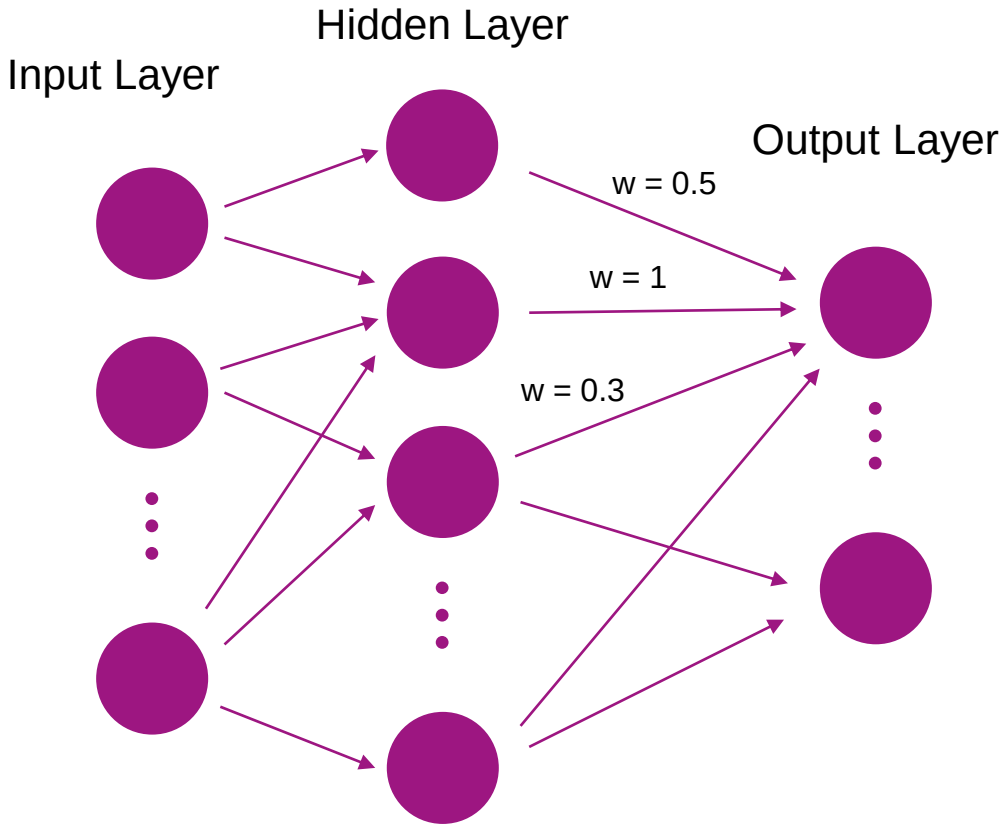


- Ein Netzwerk aus Neuronen
  - Input Layer: Eingabedaten (z.B. Farbwerte aller Pixel eines Bilds)
  - 0-n Hidden Layer: Beliebige Anzahl von Ebenen mit einer beliebigen Anzahl von Neuronen
  - Output Layer: Zielkategorien (z.B. „Hund“, „Katze“, „Maus“
- Die Neuronen erhalten eine Menge an Inputs (Zahlen) und geben einen verrechneten Wert an die Folgeneuronen weiter: „Aktivierungsfunktion“
- Die Kanten haben eine Gewichtung, die den Ausgabewert vom vorherigen Neuron entsprechend verstärken oder abschwächen

=> Das ist das statische Modell oder die „Netzwerkarchitektur“

Neuronale Netzwerke

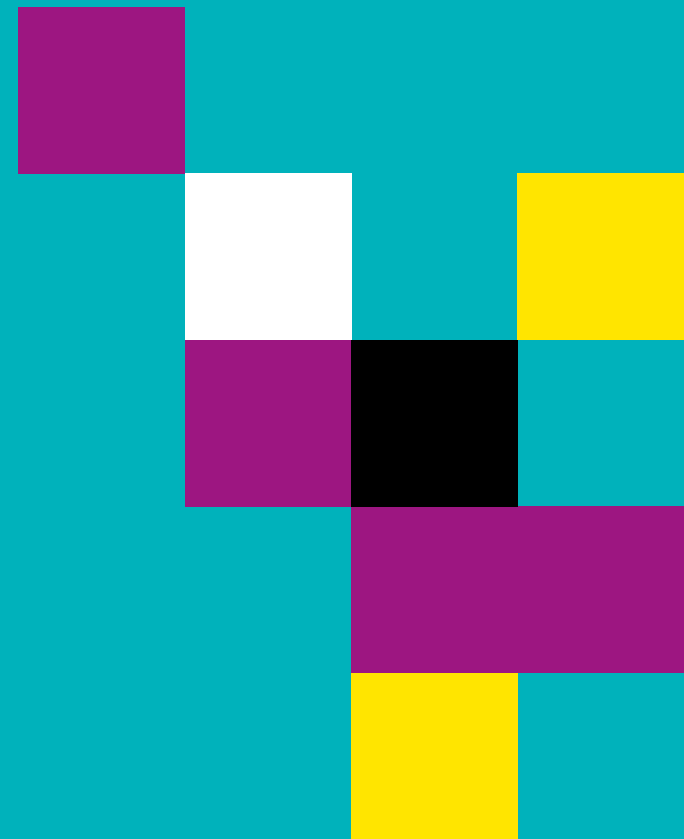
Wie funktioniert hier das „Lernen“?



- Das neuronale Netzwerk lernt dadurch, dass schrittweise die Gewichte und bestimmte Parameter der Layer so angepasst werden, dass die gewünschten Neuronen im Output Layer am meisten „feuern“ (Stichwort: Backpropagation)
- Auch hier gilt: Die richtigen Trainingsdaten sorgen die Qualität des Modells hoch zu halten
- Vertreter von „Supervised Learning“

iteratec

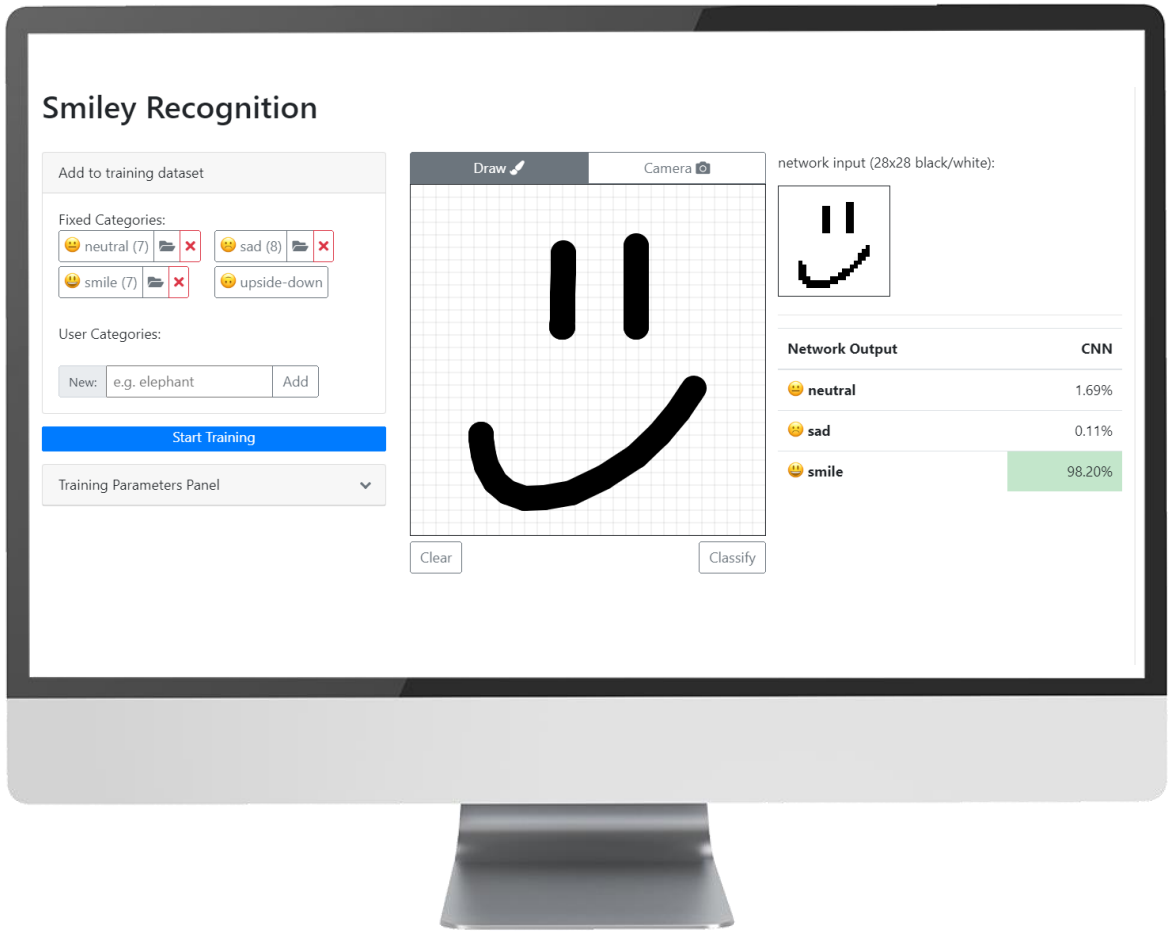
Puh…



Jetzt zu den Smileys! 😀

Das Projekt

Python mit Tensorflow/Keras und Flask



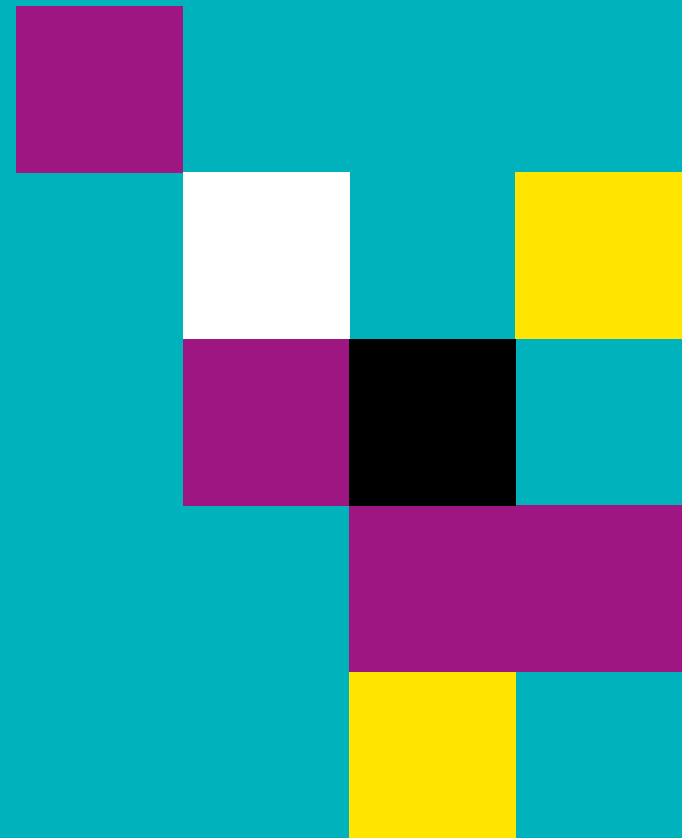
- Das Modell eines Neuronalen Netzwerks ist in code gegossen(unter Verwendung der Tensorflow/Keras API)
- Die Weboberfläche kann:
  - Bilder für Kategorien erstellen
  - Parameter setzen
  - Das Training anstoßen
  - Bilder erkennen

Sourcecode: https://github.com/scyv/Smiley

iteratec

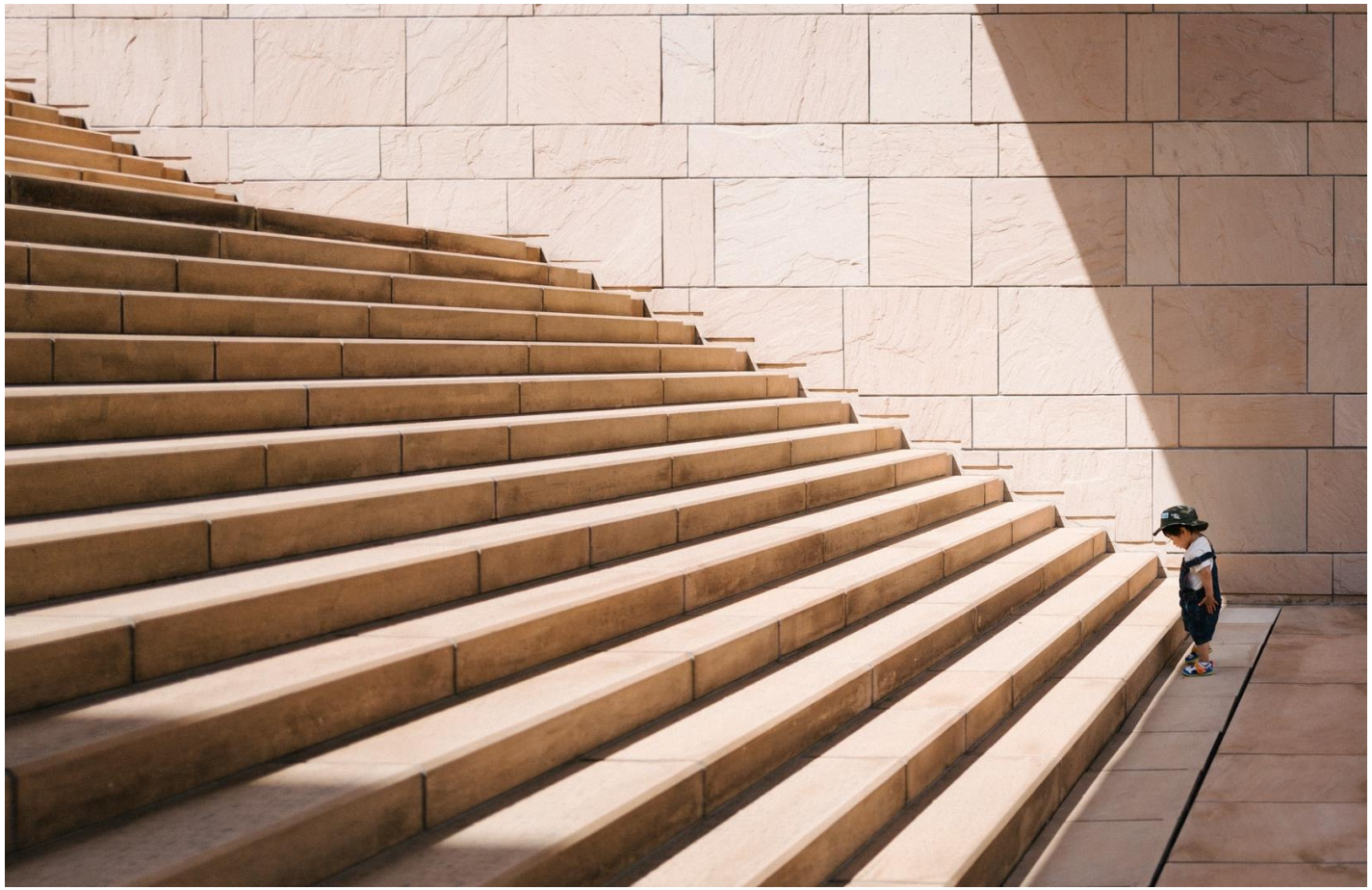
Erste Schritte

Ein einfaches Neuronales Netzwerk



Erste Schritte

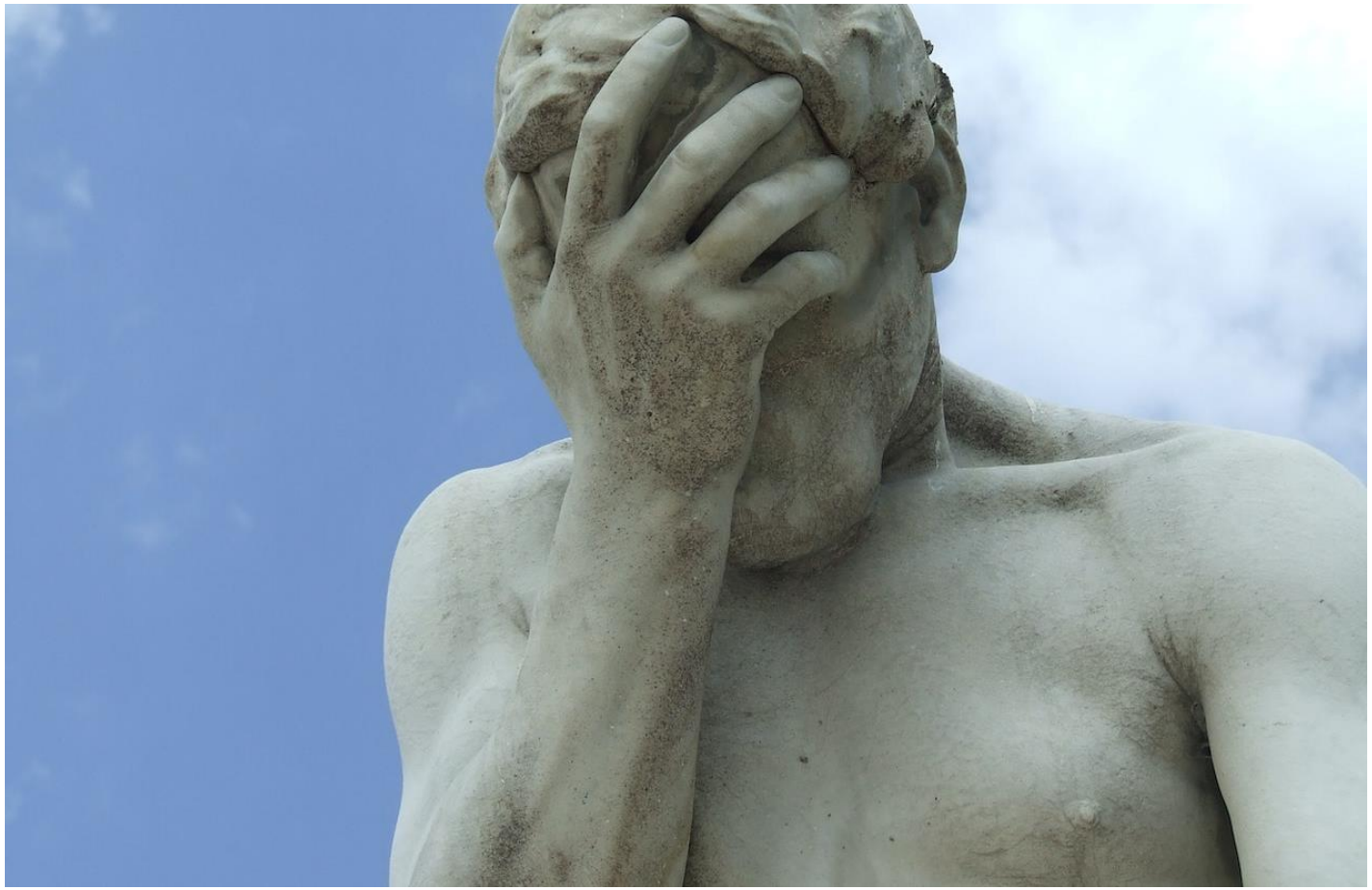
Jeder fängt mal klein an



- Wir bauen ein neuronales Netzwerk mit Hilfe der Tensorflow Keras API
- Drei Layer die komplett miteinander Verknüpft sind (Dense)
- 200 Neuronen pro Layer

Erste Schritte

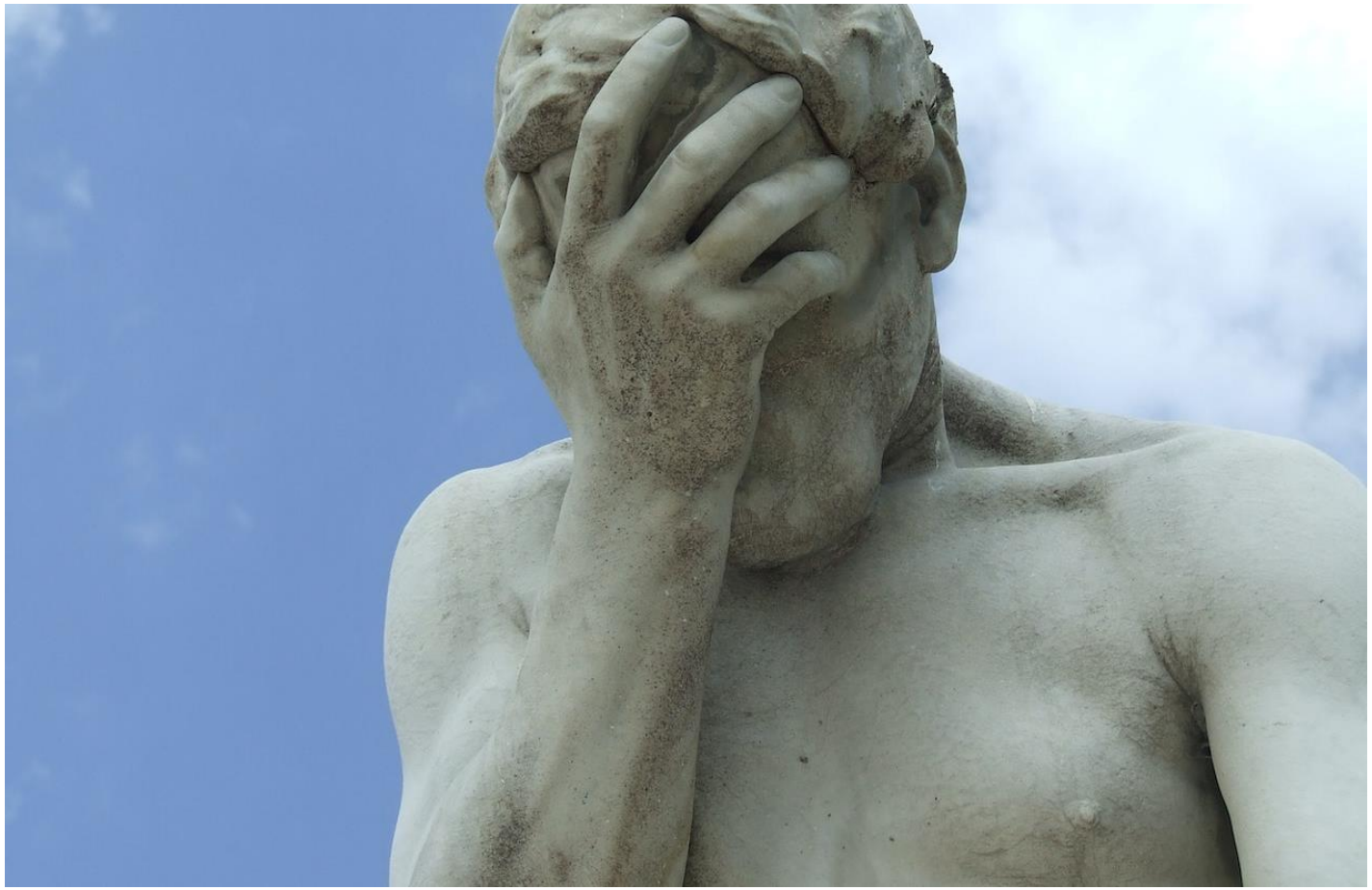
Dense Layer Network



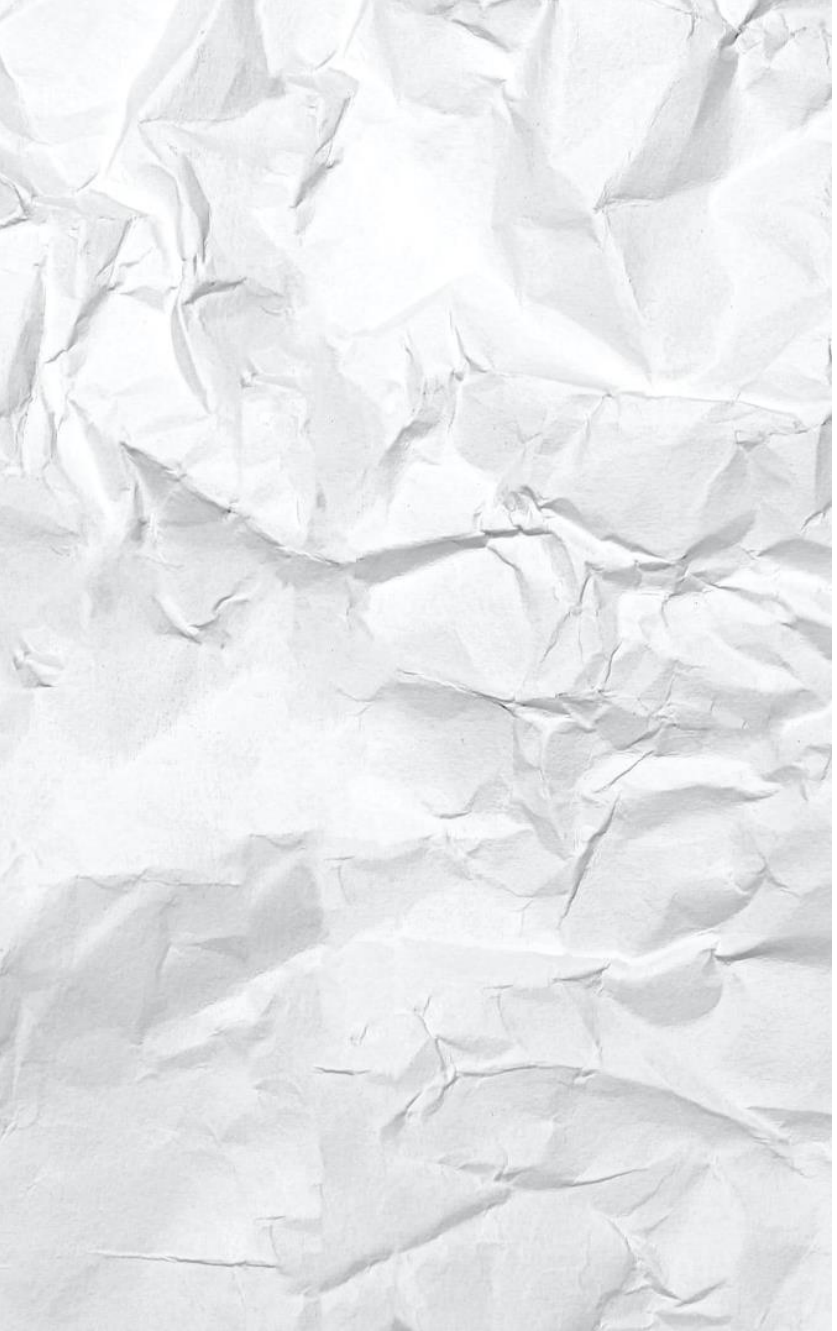
- Funktioniert leider nicht so gut ☹
- Warum?
  - Diese Art von Netzwerk ist nicht in der Lage, bestimmte Teilaspekte der Eingabebilder zu abstrahieren

Erste Schritte

Dense Layer Network



- Funktioniert leider nicht so gut ☹
- Was können wir tun?
  - Experimentieren!
  - Nachdenken!



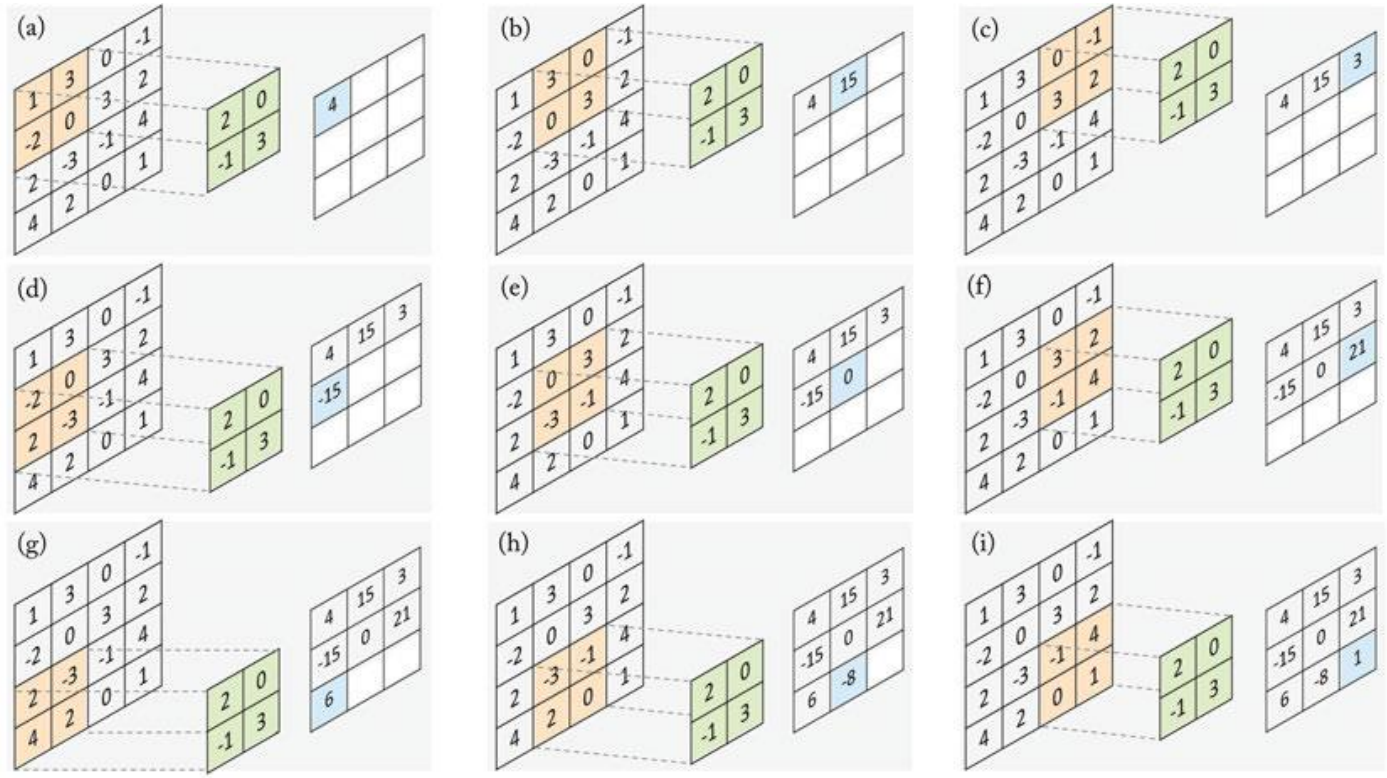
Faltung

Convolutional Neuronal Network (CNN)

- Mit Hilfe von Convolutional Layern, werden aus den Bildern bestimmte Merkmale herausgearbeitet (Features)
- Mit der mathematischen Faltung werden auf den Eingabebildern Filter angewendet (Kantendetektion, Weichzeichner, Scharfzeichner)
- Der Trick ist, dass das Netzwerk selbst die Parameter für die Filter lernt (man also gar nicht vorgibt, ob z.B. eine Kantendetektion stattfinden soll)

Faltung

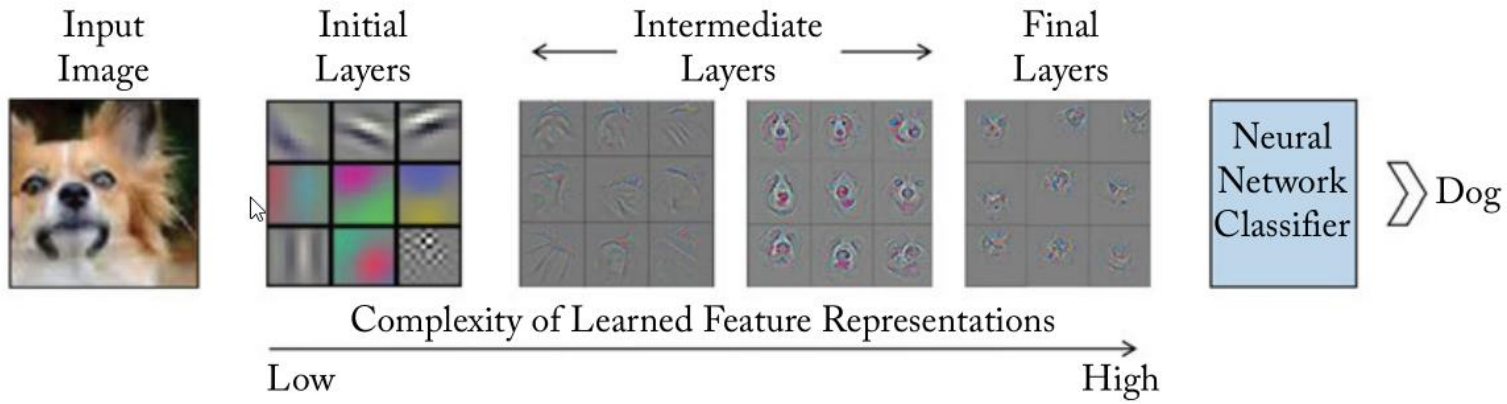
Convolutional Neuronal Network (CNN)



Quelle: A Guide to Convolutional Neural Networks for Computer Vision, Salman Khan, Hossein Rahmani, Syed Afaq Ali Shah, Mohammed Bennamoun

Faltung

Convolutional Neuronal Network (CNN)



Quelle: A Guide to Convolutional Neural Networks for Computer Vision, Salman Khan, Hossein Rahmani, Syed Afaq Ali Shah, Mohammed Bennamoun

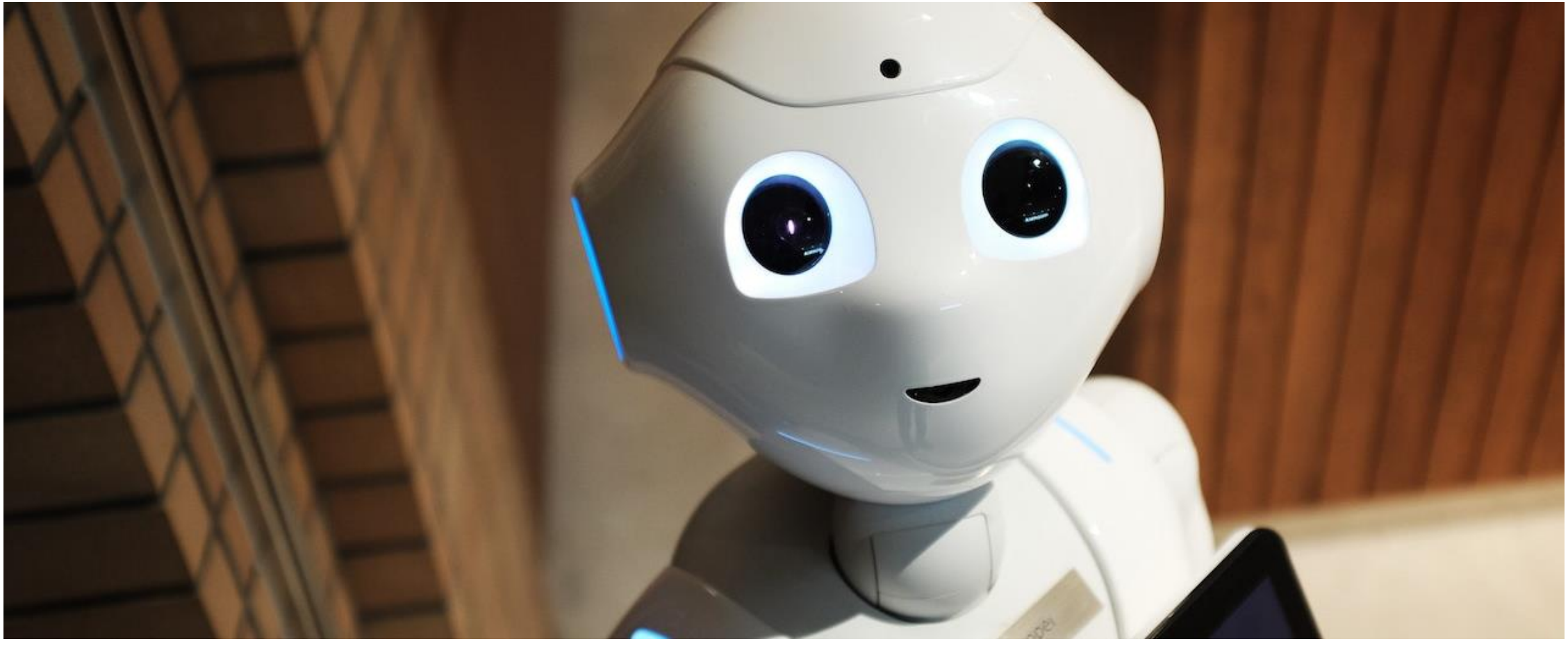
Zweiter Versuch

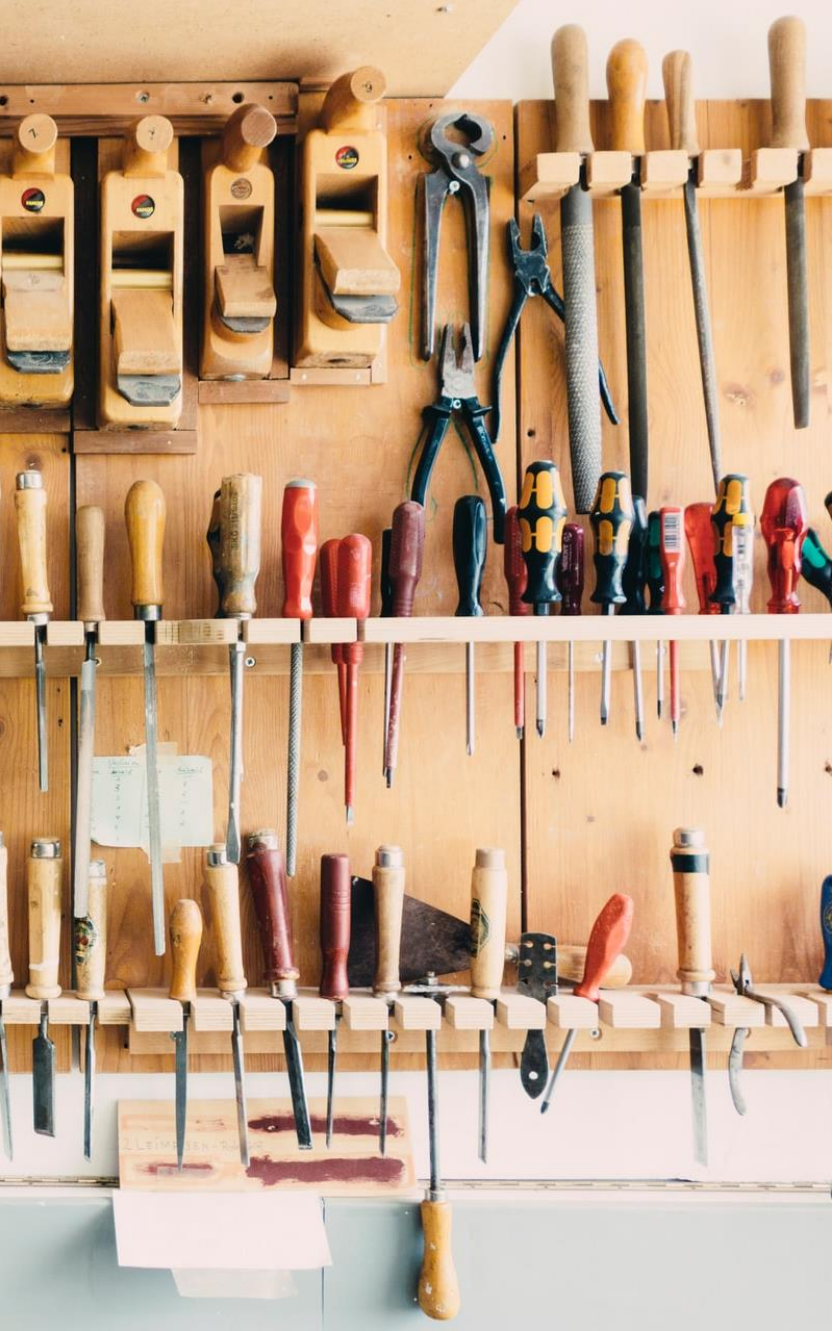
CNN!

- Drei Convolutional Layer
  - Jeweils mit 50 Filtern
- Gemischt mit Max-Pooling Layern
  - Verbessern die Abstraktion/verringern das „Rauschen“
- Am Ende noch zwei Dense Layer
  - Schrittweise Annäherung an die Anzahl der Kategorien

Geht doch!

Unser CNN funktioniert wesentlich besser!





Zusammenfassung

Unser Handwerkszeug

- Python mit Tensorflow/Keras
- Kleines Grundsetup mit ein paar Zeilen Python Code
- Definition eines Modells
- Gute (!) Trainingsdaten
- Intuition und Erfahrung
- Zeit

Vielen Dank für Ihre Aufmerksamkeit!

Bitte bewerten ☺

DEVELOPING
DIGITAL
CHAMPIONS

iteratec

Bereit, digitaler Champion zu werden?
Wir entwickeln Sie weiter!

Yves Schubert

Senior Software Architect

iteratec GmbH
Zettachring 6 | 5. OG
70567 Stuttgart

Tel: +49 151 542 411 61
Email: yves.schubert@iteratec.com