



Grundlagen Künstliche Intelligenz

Schulung am 21.10.2024 und 22.10.2024

Ihr Trainer: Nicolas Kuhaft

Überblick Zeiten

9:00 – 10:30



10:40 – 12:00



13:00 – 14:45



15:00 – 16:30

16:30 – 17:30 **OpenSpace**



Vorstellungsrunde

Inhalt

1. Einführung und Motivation
2. Daten als Grundlage von Künstlicher Intelligenz
3. Methoden der Künstlichen Intelligenz
4. Anwendungen von Künstlicher Intelligenz
5. Grenzen und Risiken
6. Künstliche Intelligenz in der Praxis
7. Ausblick

Inhalt

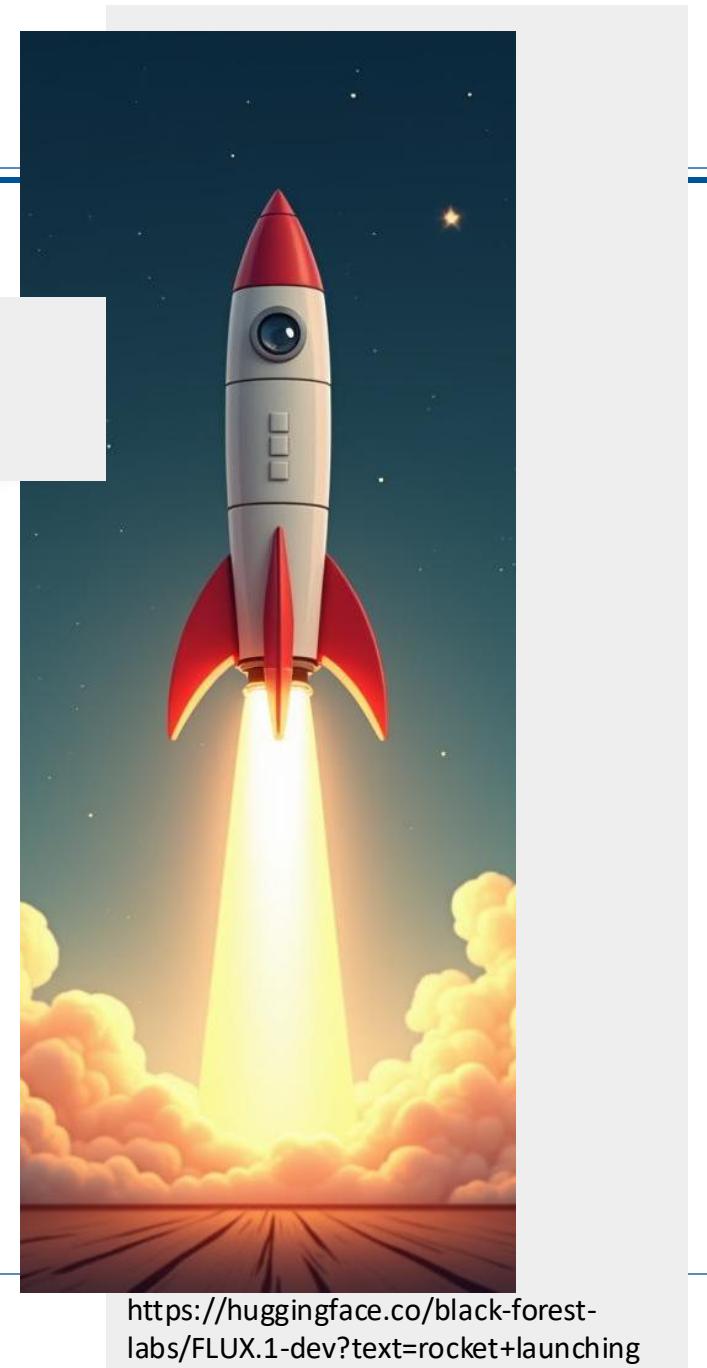
1. Einführung und Motivation

2. Daten als Grundlage von Künstlicher Intelligenz
3. Methoden der Künstlichen Intelligenz
4. Anwendungen von Künstlicher Intelligenz
5. Grenzen und Risiken
6. Künstliche Intelligenz in der Praxis
7. Ausblick

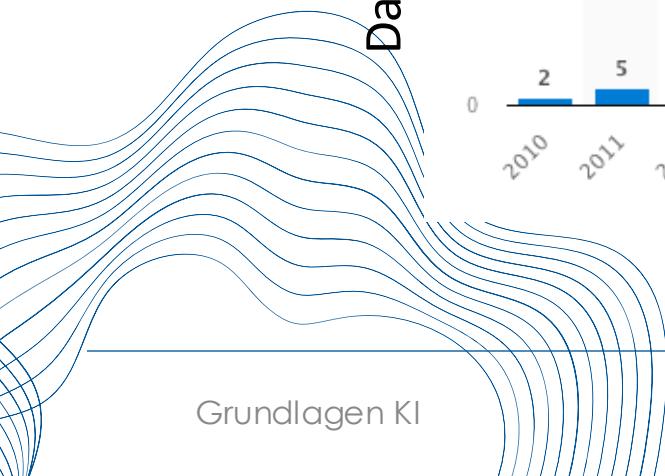
1. KI – Wieso jetzt?



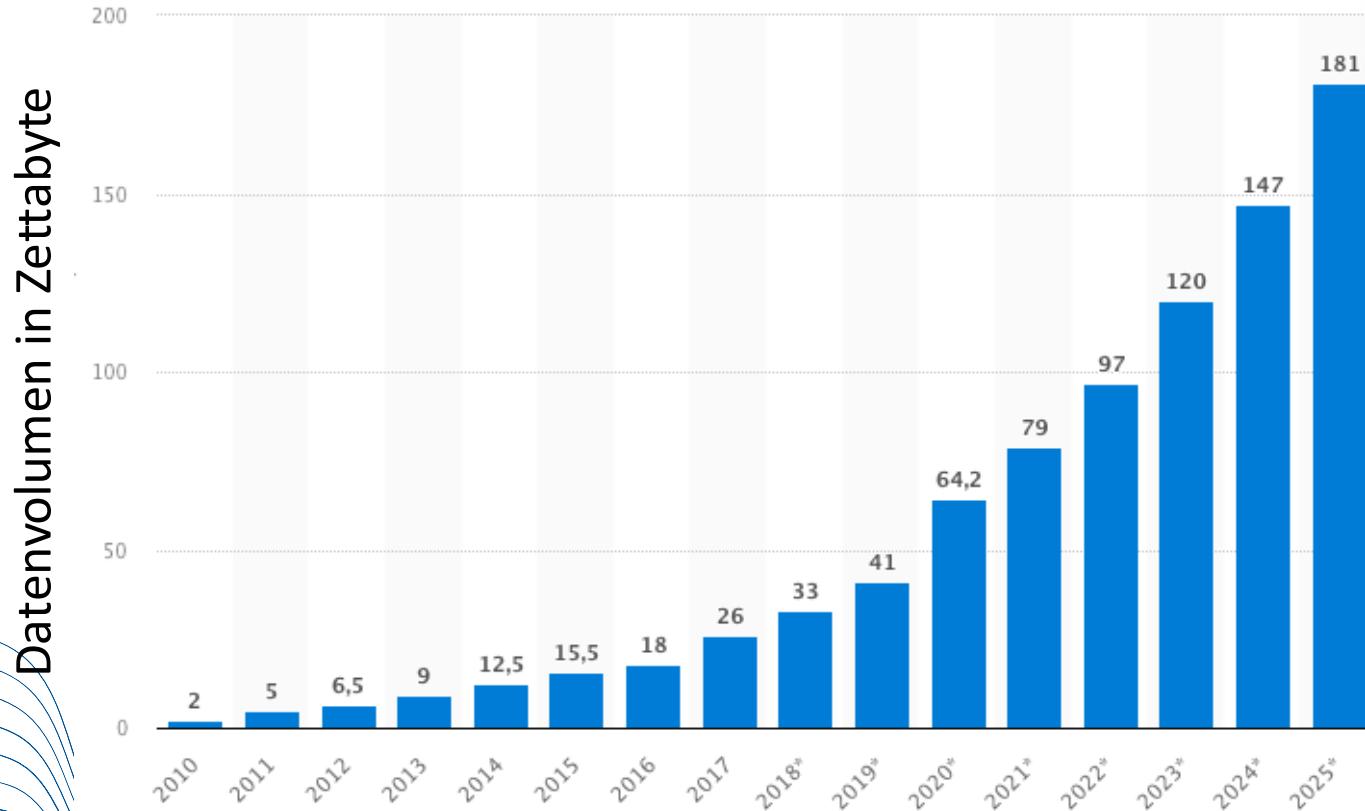
Treibstoff - Daten



1. KI – Wieso jetzt?



Grundlagen KI



<https://de.statista.com/statistik/daten/studie/1420298/umfrage/prognose-weltweites-erstelltes-datenvolumen/>



<https://huggingface.co/black-forest-labs/FLUX.1-dev?text=rocket+launching>

1. KI – Wieso jetzt?

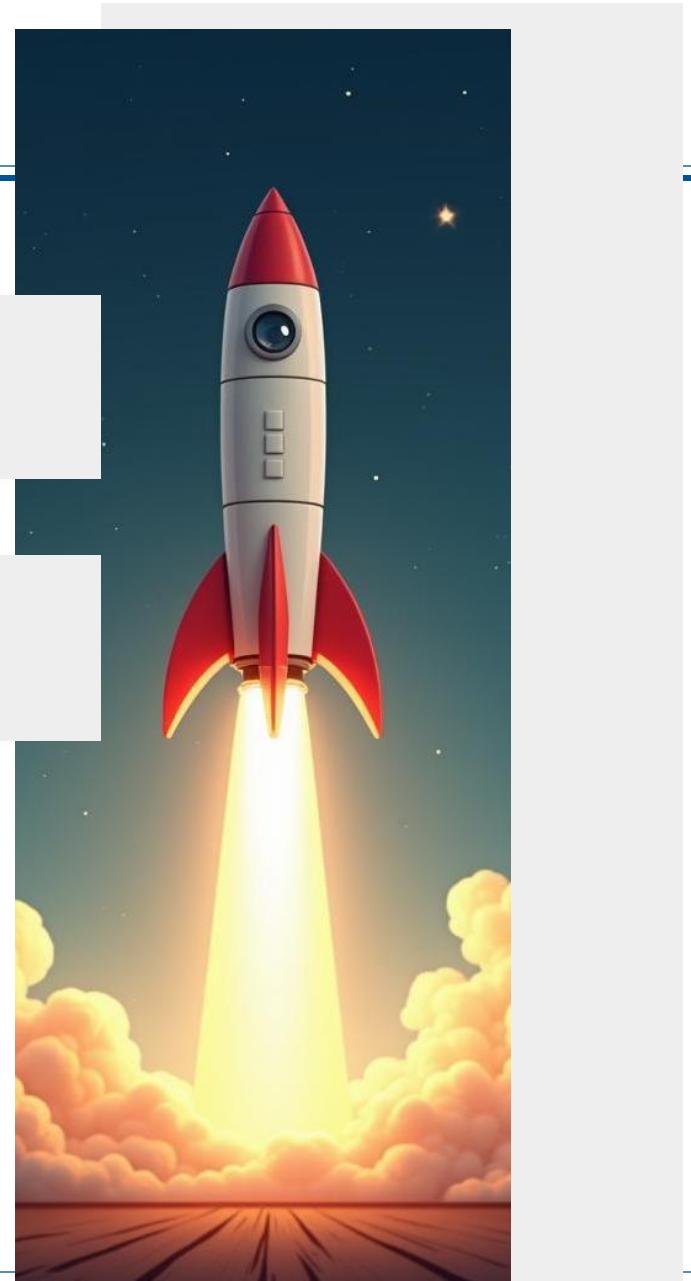
Treibstoff - Daten

Starke Maschinen - GPUs

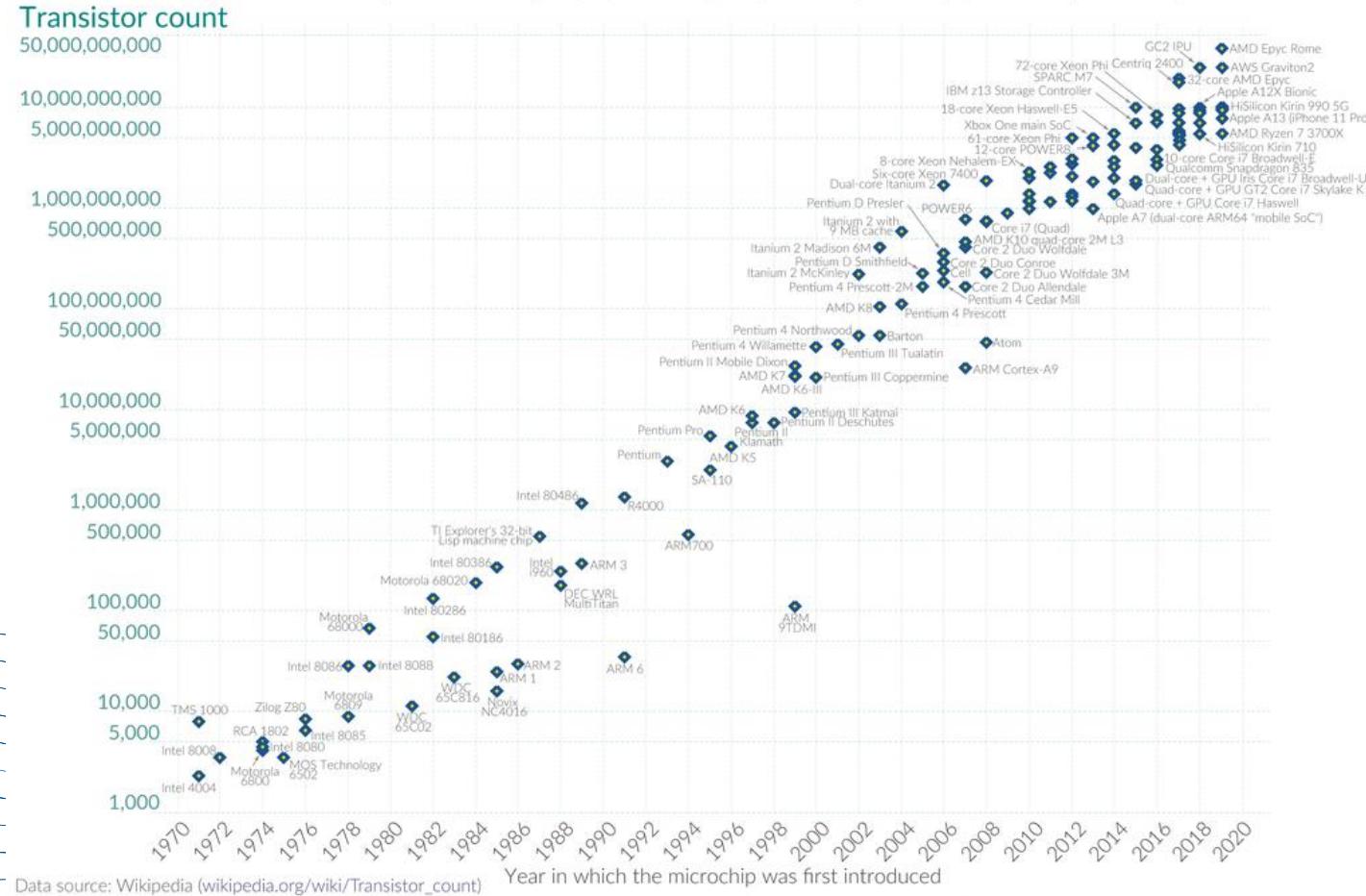
Grundlagen KI

8

<https://huggingface.co/black-forest-labs/FLUX.1-dev?text=rocket+launching>



1. KI – Wieso jetzt?



1. KI – Wieso jetzt?

Treibstoff - Daten

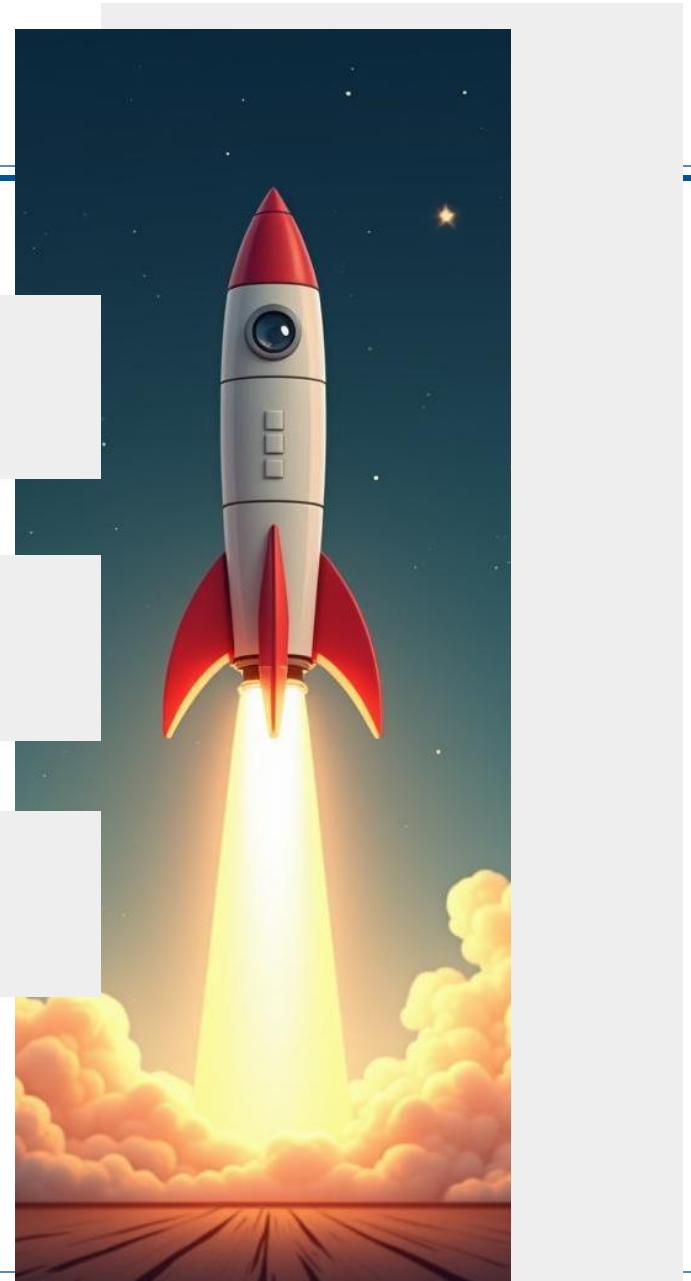
Starke Maschinen - GPUs

Rocket Science – KI-Algorithmen

Grundlagen KI

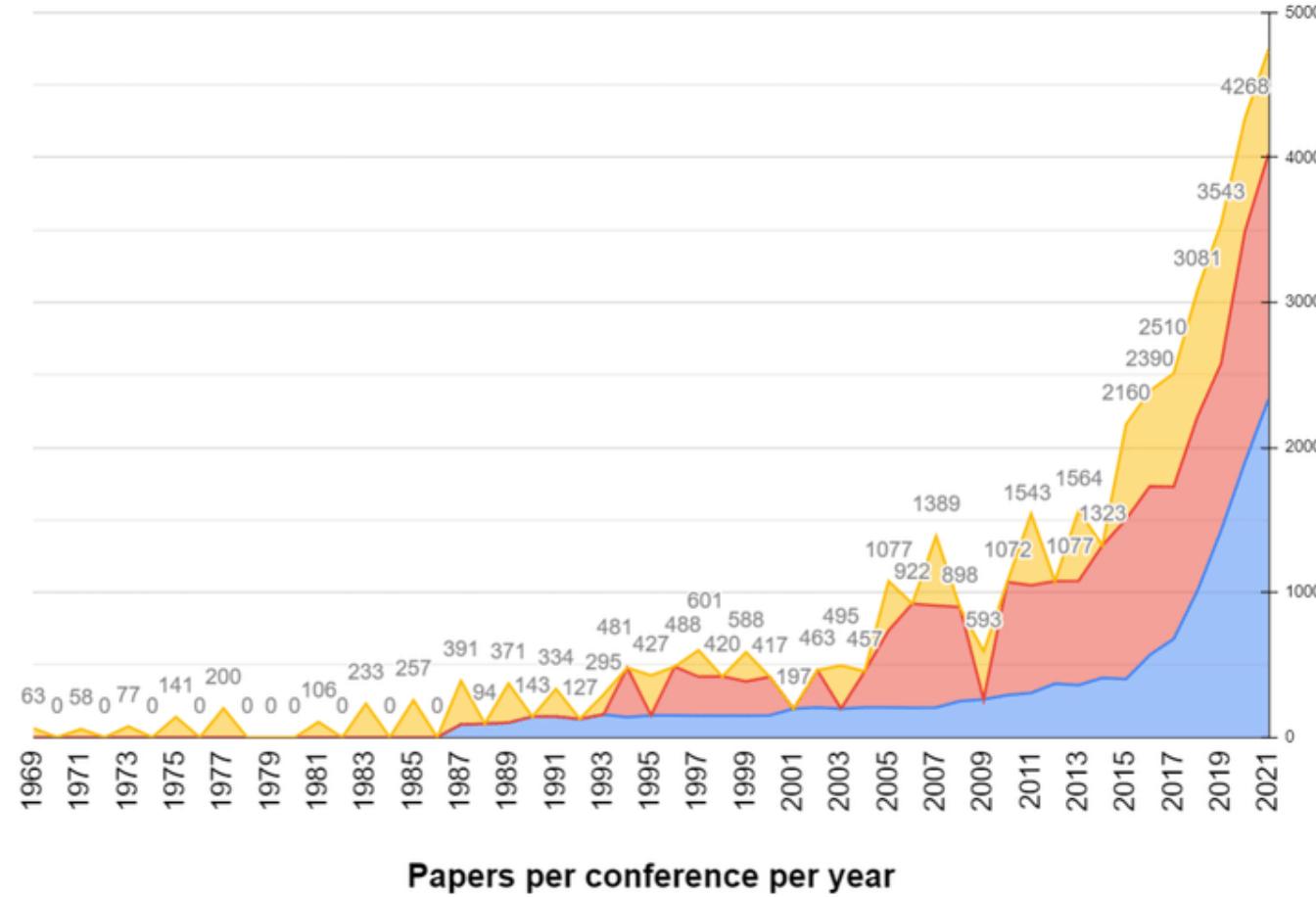
10

<https://huggingface.co/black-forest-labs/FLUX.1-dev?text=rocket+launching>



1. KI – Wieso jetzt?

- IJCAI
- AAAI
- NeurIPS



https://www.researchgate.net/figure/Manual-paper-count-per-year-in-AAAI-NeurIPS-and-IJCAI_fig3_360888073



<https://huggingface.co/black-forest-labs/FLUX.1-dev?text=rocket+launching>

1. KI – Wieso jetzt?

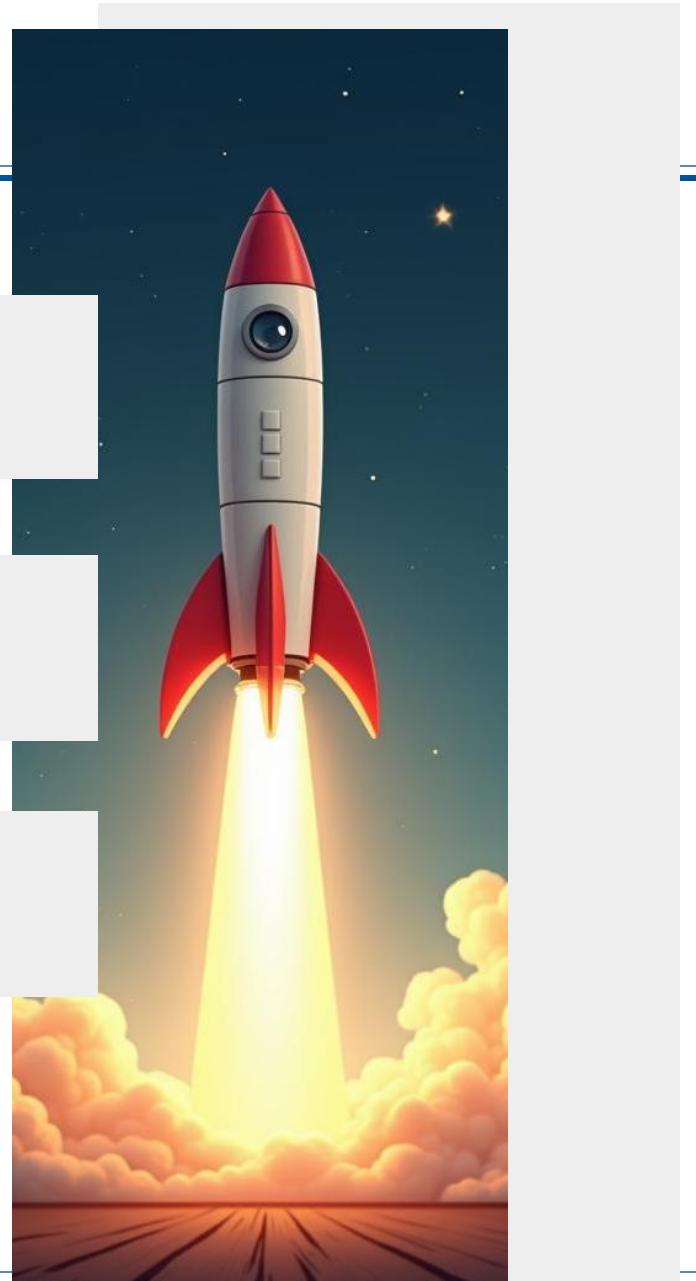
Treibstoff - Daten

Starke Maschinen - GPUs

Rocket Science – KI Algorithmen

Grundlagen KI

12



<https://huggingface.co/black-forest-labs/FLUX.1-dev?text=rocket+launching>

“In my little group chat with my tech CEO friends there’s this betting pool for the first year that there is a one-person billion-dollar company.”

Sam Altman



Inhalt

1. Einführung und Motivation
- 2. Daten als Grundlage von Künstlicher Intelligenz**
3. Methoden der Künstlichen Intelligenz
4. Anwendungen von Künstlicher Intelligenz
5. Grenzen und Risiken
6. Künstliche Intelligenz in der Praxis
7. Ausblick

2. Strukturierte Daten

= Daten, die in Tabellenform vorliegen

SQL Datenbanken

.csv Dateien

Excel Dateien

| ID | Name | Alter | Wohnort |
|----|------|-------|---------|
| | | | |
| | | | |

2. Unstrukturierte Daten

= Daten ohne Struktur

PDF Dateien

Bilder

Webseiten (HTML
Dateien)

Word Dateien

2. Semistrukturierte Daten

= zum Teil strukturiert, zum Teil unstrukturiert

Log Daten

XML, JSON, YAML

NoSQL
Datenbanken

E-Mails

```
{  
  "_id" : ObjectId("5ad88534e3632e1a35a58d00"),  
  "name" : {  
    "first" : "John",  
    "last" : "Doe" },  
  "address" : [  
    { "location" : "work",  
      "address" : {  
        "street" : "16 Hatfields",  
        "city" : "London",  
        "postal_code" : "SE1 8DJ"},  
      "geo" : { "type" : "Point", "coord" : [  
        51.5065752, -0.109081]}},  
  ]  
}
```

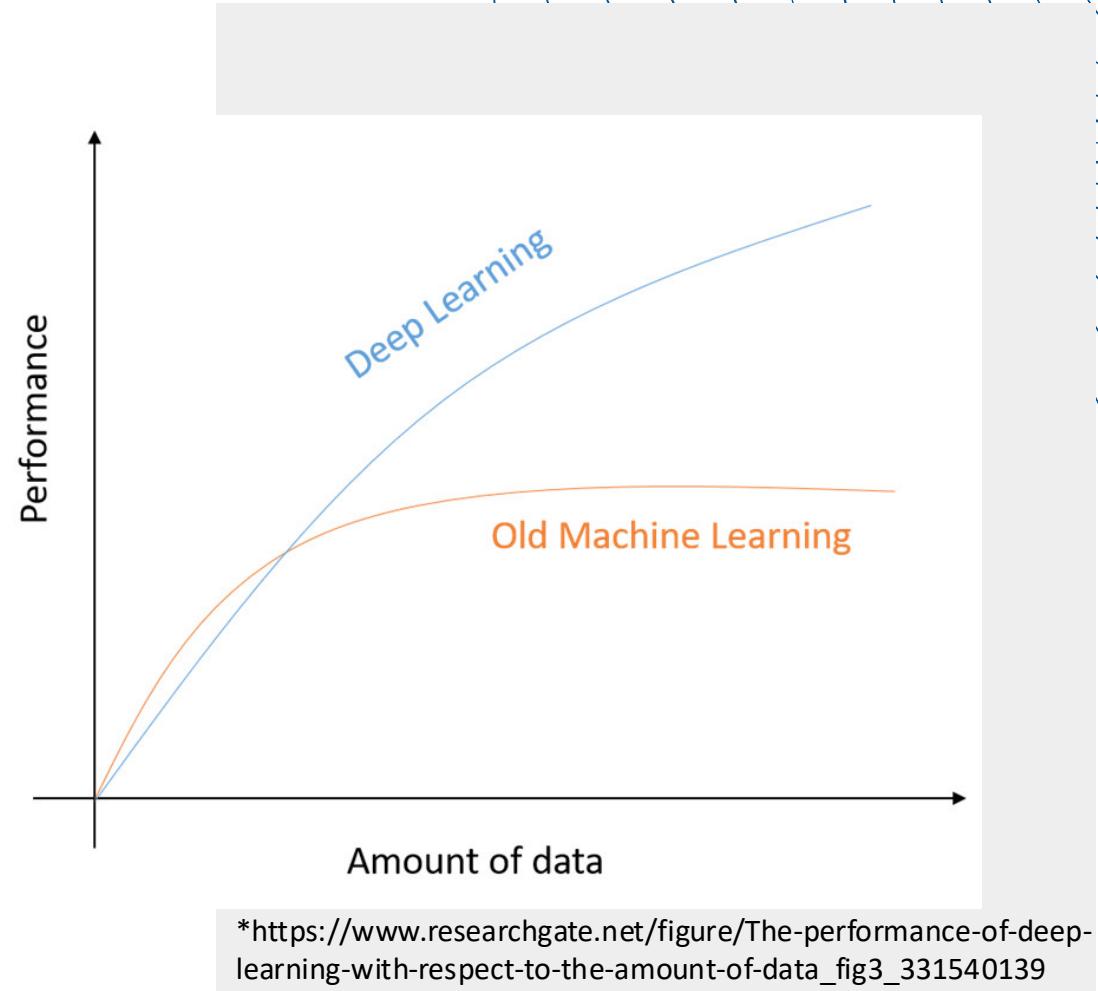
2. Daten im KI Kontext

Daten-Qualität ist wichtig



Garbage in – Garbage out

Je mehr – desto besser



2. Von Daten zur Information

- Strukturiert > semistrukturiert > unstrukturiert
- Der Mehrwert liegt in der Information. Bei strukturierten Daten ist der Weg von den Daten zur Information oft kurz.



2. Von Daten zur Information

Datenkontext: Wichtig zur Interpretation der Daten

- Z.B. Zeitstempel, Einheiten

Codierung: Dateiformate oder numerische Codierung

- z.B. 1 für „Ja“, 2 für „Nein“, Serialisierungen, Unicode

2. Daten als Grundlage für KI

Dateiformate

- Effizienz
- Kompatibilität

.CSV

.JPG

Datenbanken

Datenquelle

- Zuverlässigkeit
- Aktualität

Open Data

Interne Quellen
(z.B. CRM System)

Manuelle
Eingabe

Sensordaten

2. Datalake als Zentrale

- Zentraler Speicherort für (möglichst) alle Daten im Unternehmen
- Daten in verschiedenen Verarbeitungsstufen
- Grundlage für Datenanalysen und KI-Projekte
- Hier nimmt das Thema Datenkontext, Dateiformaten etc. wichtigen Stellenwert ein

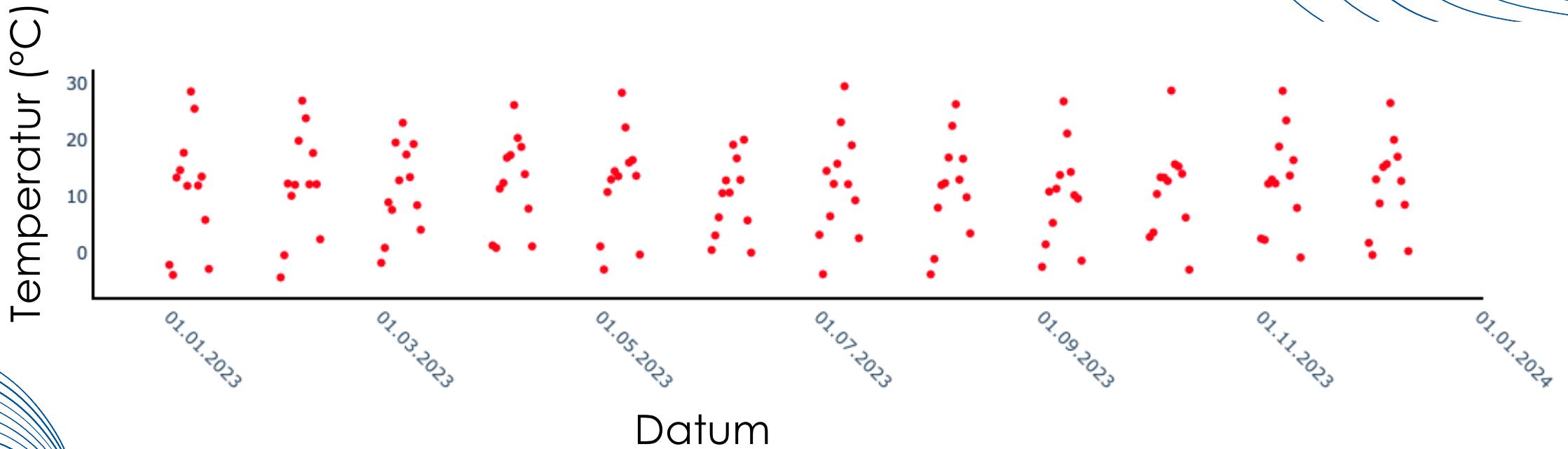
2. Daten – Was kann schief gehen?

... aus dem Nähkästchen:

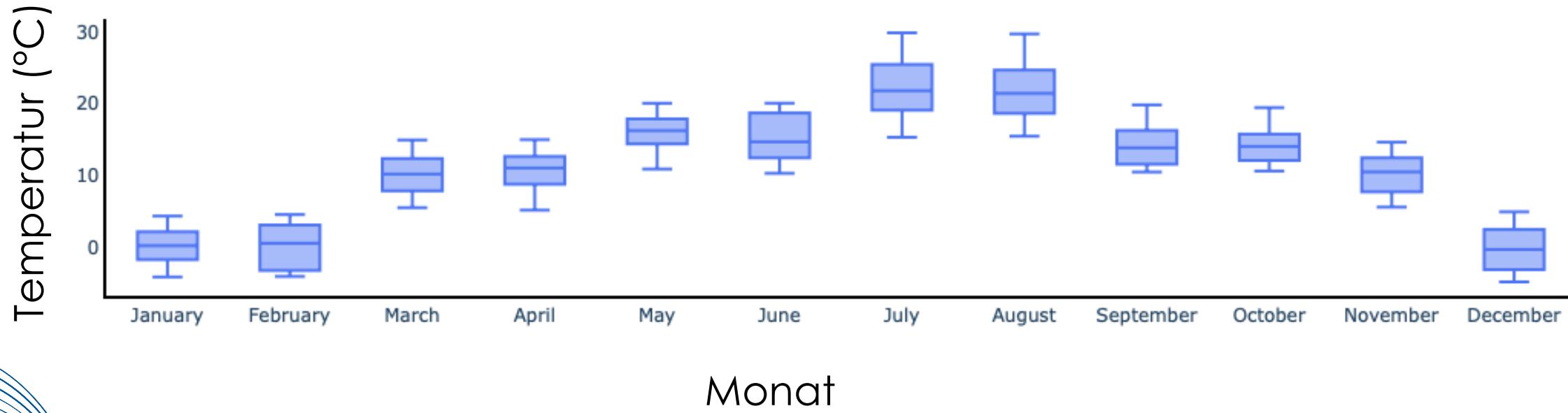
Ein KI-Algorithmus zum Handel von Energiemengen an der Energiebörse wird erstellt. Das GPU Cluster ist heiß gelaufen und mehrere Varianten wurden erstellt. Bisher ohne Erfolg. Der Algorithmus zeigt ein unklares Verhalten. Es wird in Zweifel gezogen, ob der KI-Algorithmus überhaupt funktionieren kann. Eine (späte) Analyse der Daten zeigt aber ein anderes Problem...

Datetimes wurden teilweise im deutschen Format (01-10-2024) bereitgestellt und – wo möglich – falsch herum transformiert.

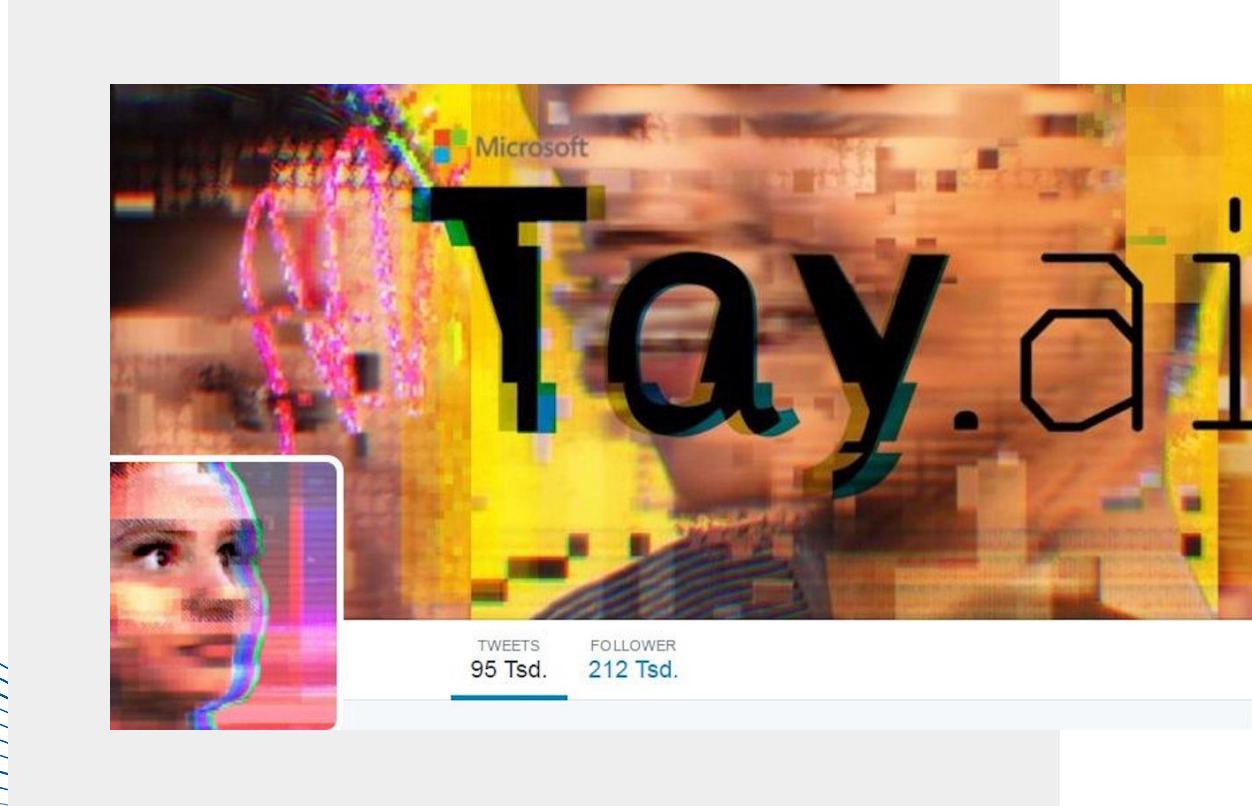
2. Daten visualisieren



2. Daten visualisieren



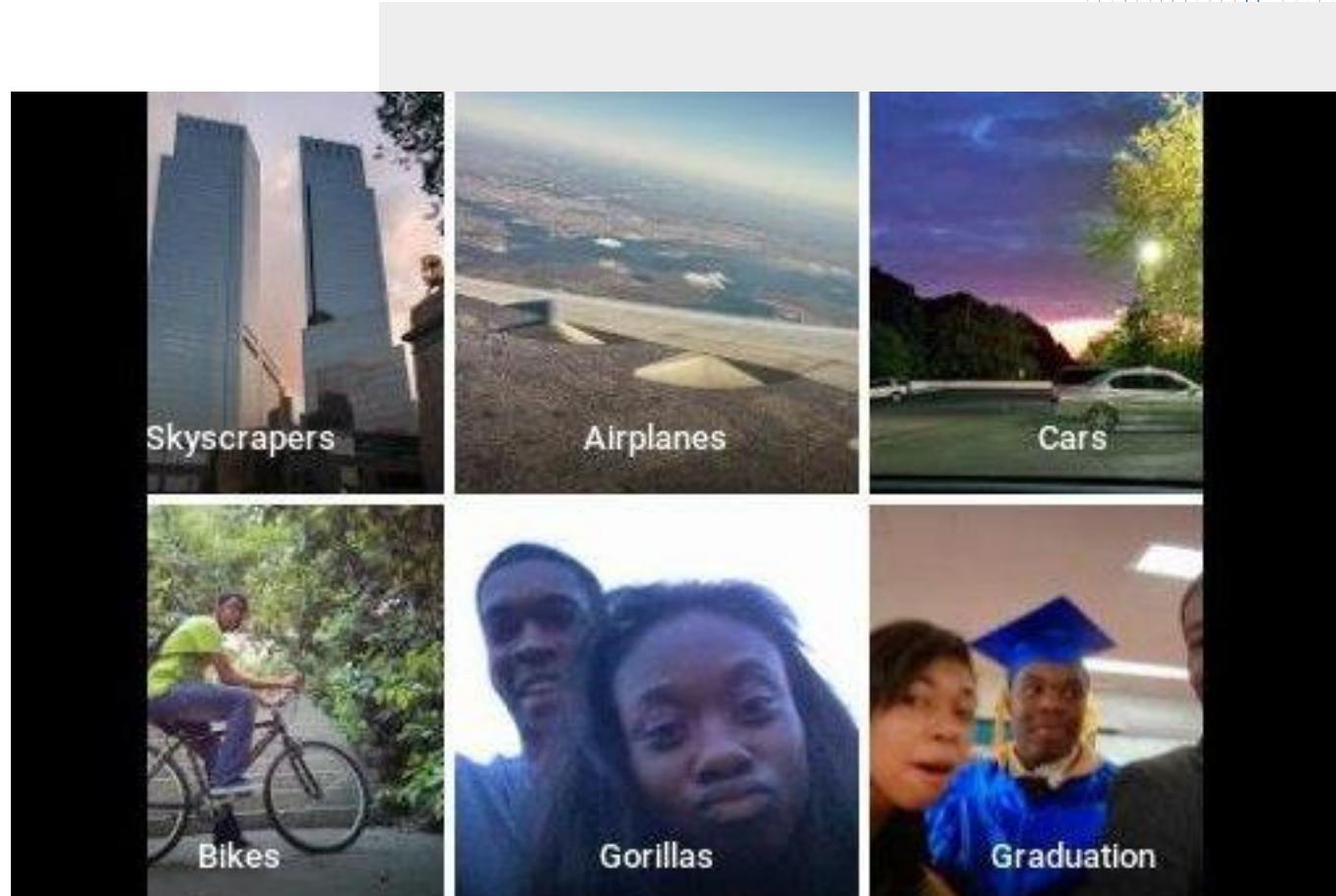
2. Daten – Was kann schief gehen?



Garbage in – Garbage out:
Twitter-Bot Tay hat durch
Chat mit Usern gelernt. Und
wurde dann rassistisch.

2. Daten – Was kann schief gehen?

- Google Photos in 2015
- Problem: Es müssen nicht nur insgesamt genug Daten vorhanden sein, sondern auch für jede Kategorie (Klasse) müssen genug Beispiele im Datensatz vorhanden sein.



*<https://www.bbc.com/news/technology-33347866>

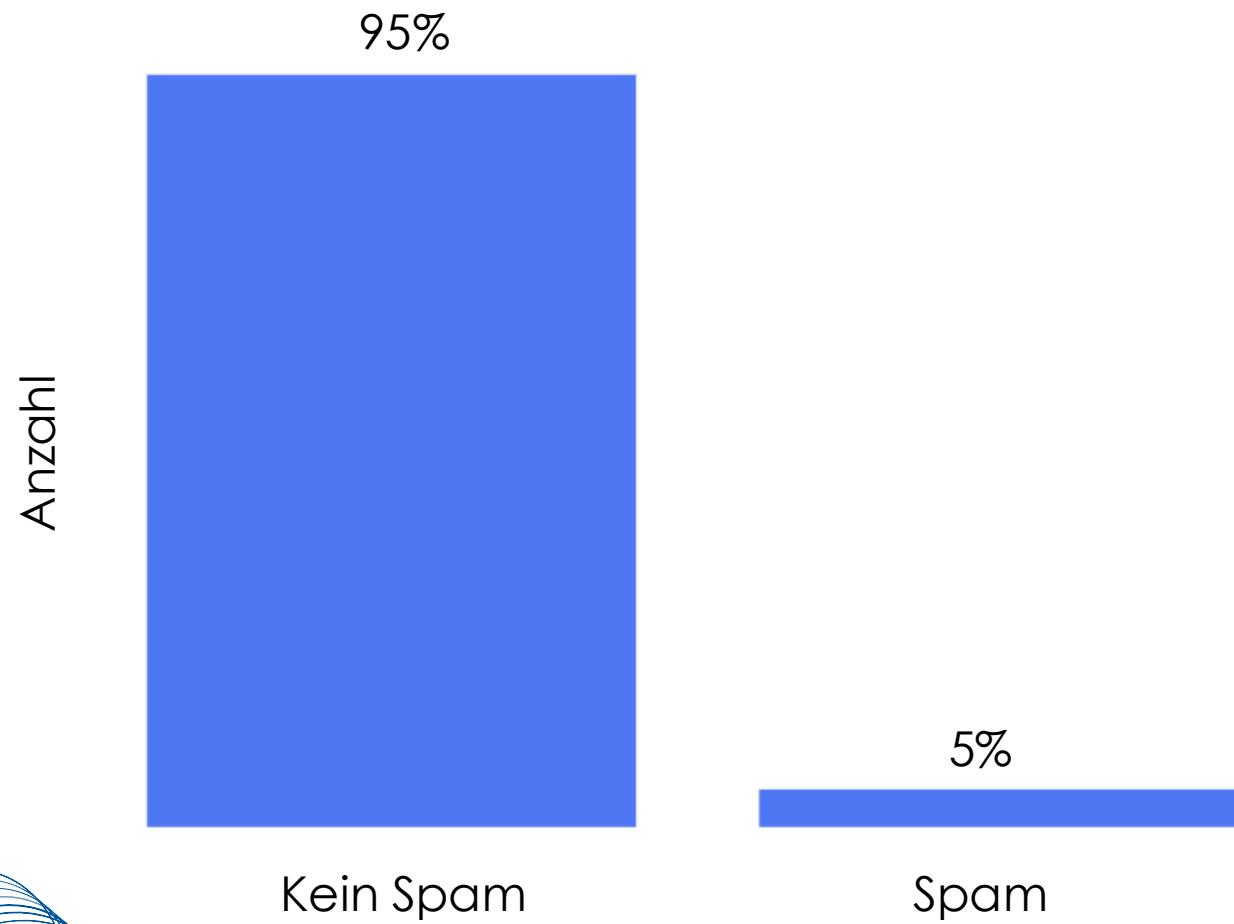
2. Daten – Was kann schief gehen?



Ein Spam Filter hat eine Treffsicherheit von 95%. Nur in 5% der Fälle wird die Mail fehlerhaft sortiert. Ist der Spam Filter gut?

Es kommt drauf an...

2. Daten – Beispiel Spam Filter



Daten visualisieren

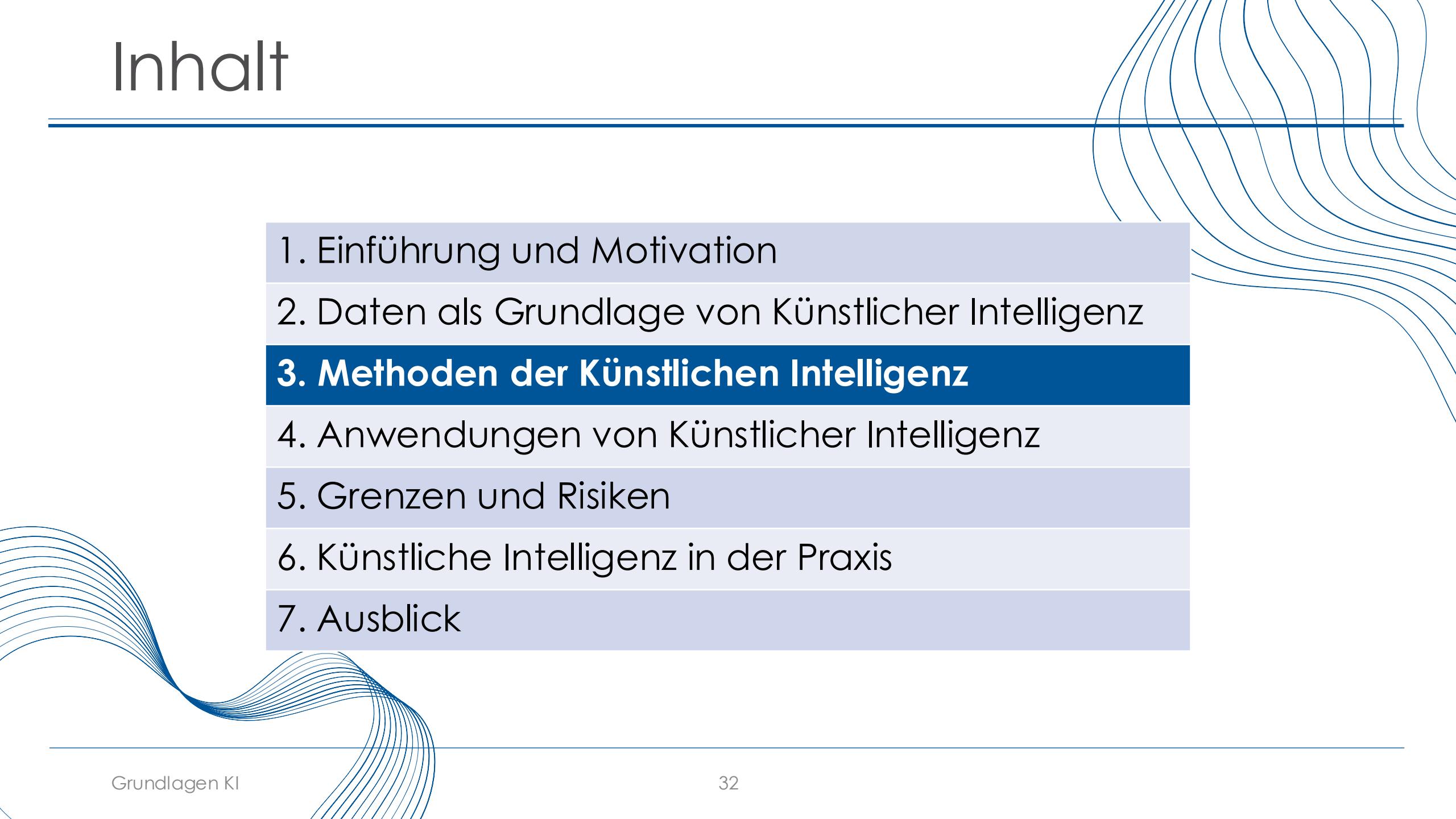
<https://app.rawgraphs.io>

<https://raw.githubusercontent.com/datasets/titanic/master/titanic.csv>

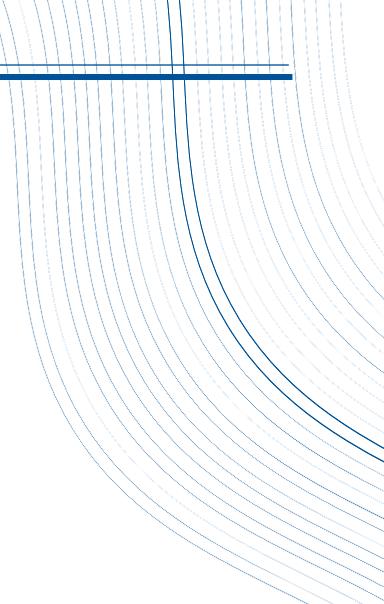
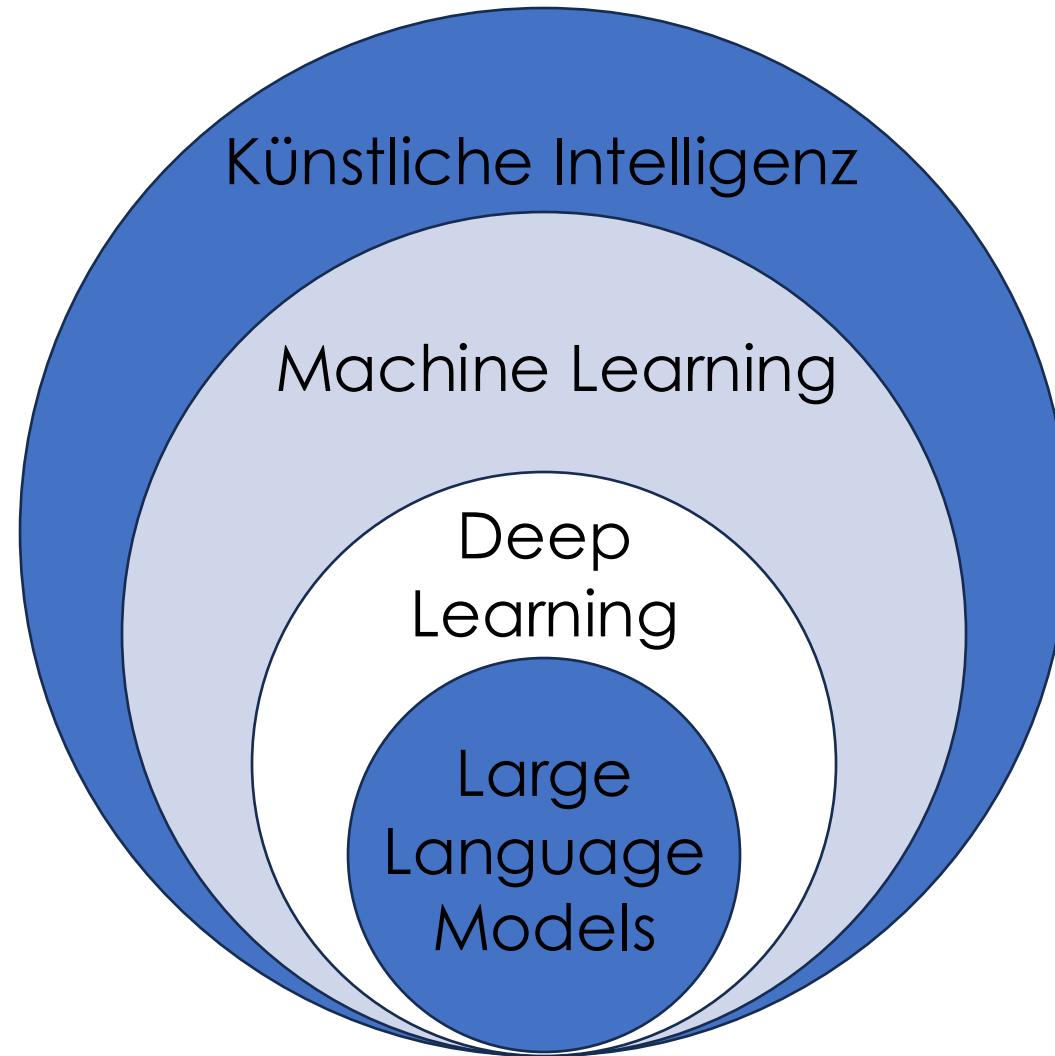
Daten visualisieren

1. Visualisiere mit einem Bar Chart die Überlebensrate der Passagiere getrennt nach Geschlecht.
2. Zeige mit einem Violin Plot die Altersverteilung nach Passagierklasse.
3. Zeige mit einem Sankey Diagram, wie sich die Passagierklassen nach Embark (Einstiegs-Hafen) unterscheiden.
4. Erstelle ein Sunburst Diagram. Ziehe in das Feld Hierarchy „Sex“ und „Pclass“. Die Größe soll durch die Anzahl der Überlebenden bestimmt werden. Die Farbe durch den Durchschnitt der Überlebenden.

Inhalt

- 
1. Einführung und Motivation
 2. Daten als Grundlage von Künstlicher Intelligenz
 - 3. Methoden der Künstlichen Intelligenz**
 4. Anwendungen von Künstlicher Intelligenz
 5. Grenzen und Risiken
 6. Künstliche Intelligenz in der Praxis
 7. Ausblick

3. Methoden



3. Künstliche Intelligenz

„An AI system is a machine-based system that can **operate autonomously** and **adapt after deployment**, generating **outputs like predictions or decisions**.“

- Definition im Europäischen KI-Act

3. Machine Learning

Machine Learning ist die Entwicklung von Algorithmen und Modellen, die selbstständig Muster in Daten finden – und lernen.

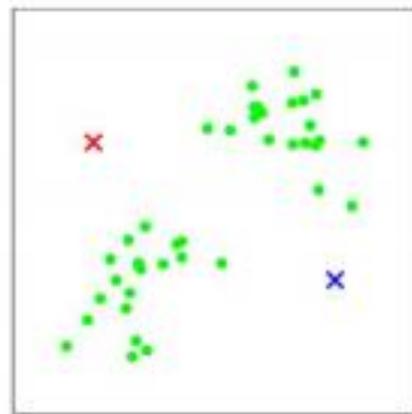
Für welche Art von Problemen eignet sich Machine Learning?

- Wenn das Definieren von Regeln schwer oder nicht möglich ist
 - z.B. Erkennen von Bildern, Verarbeitung von Sprache
 - Im Gegensatz zu Multiplikation von Zahlen

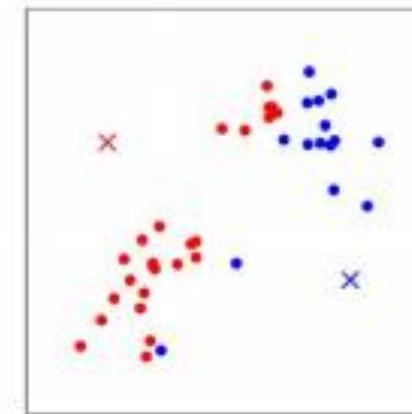
3. Beispiel „Aus Daten lernen“: KMeans



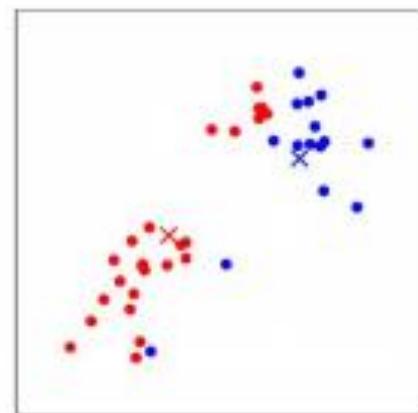
(a)



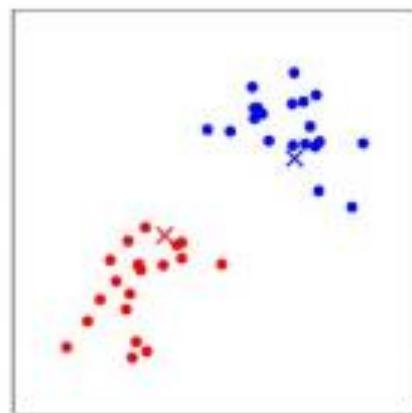
(b)



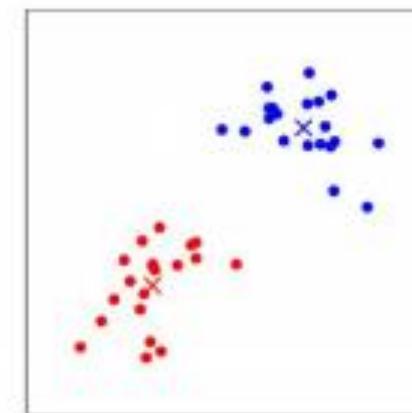
(c)



(d)



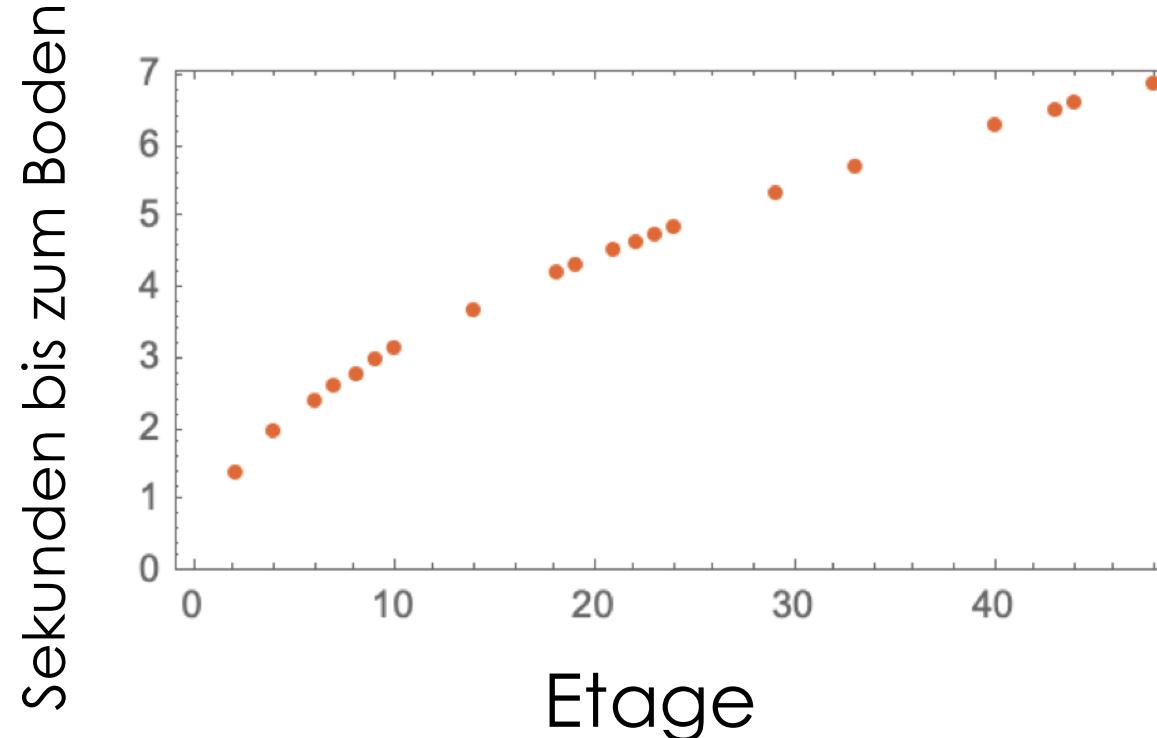
(e)



(f)

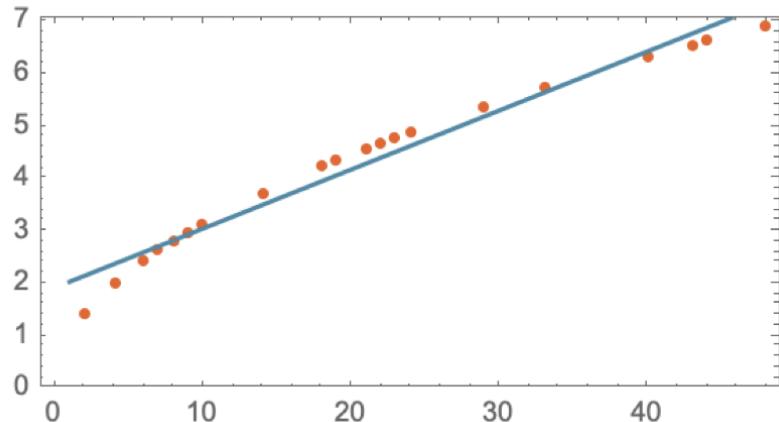
3. Was ist ein Modell?

Ein Modell stellt den Zusammenhang zwischen **Features** (Input Variablen) und der Zielgröße her.

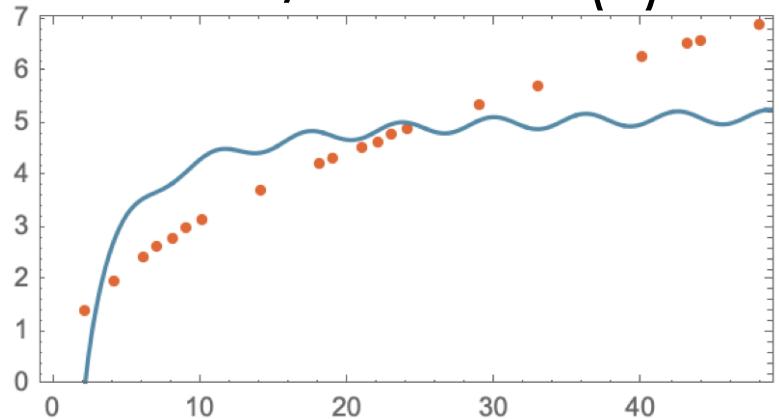


3. Was ist ein Modell?

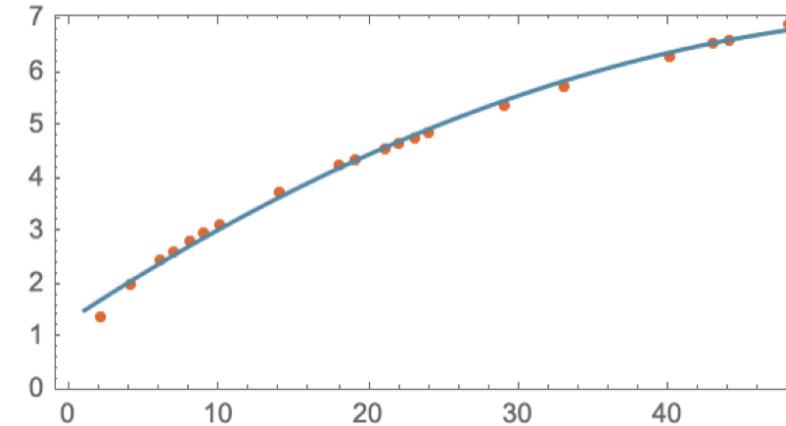
$$a^*x + b$$



$$a + b/x + c^*\sin(x)$$



$$a^*x + b^*x^2 + c$$



a, b und c sind Parameter, die man einstellen kann, damit das Modell die Daten möglichst gut trifft.

3. Supervised Learning

= Überwachtes Lernen

Das korrekte Ergebnis ist vorgegeben und der Algorithmus lernt den Zusammenhang.

Klassifikation

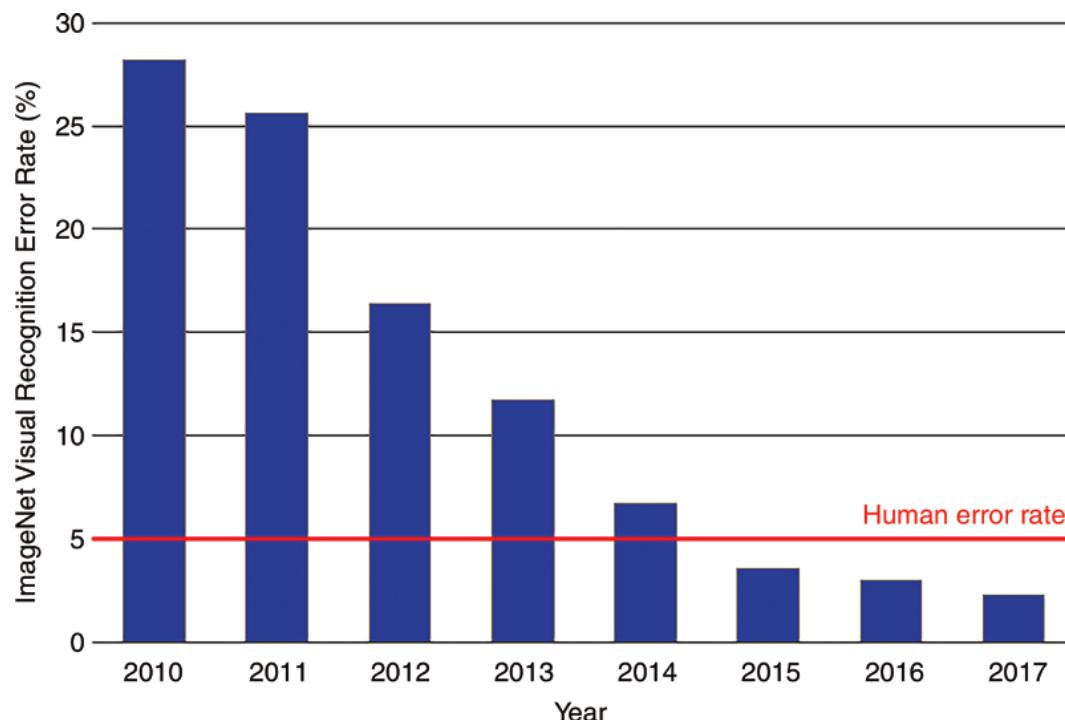
Output ist eine diskrete Klasse, z.B. Vorhersage „Regen-Tag“ oder „kein Regen-Tag“.

Regression

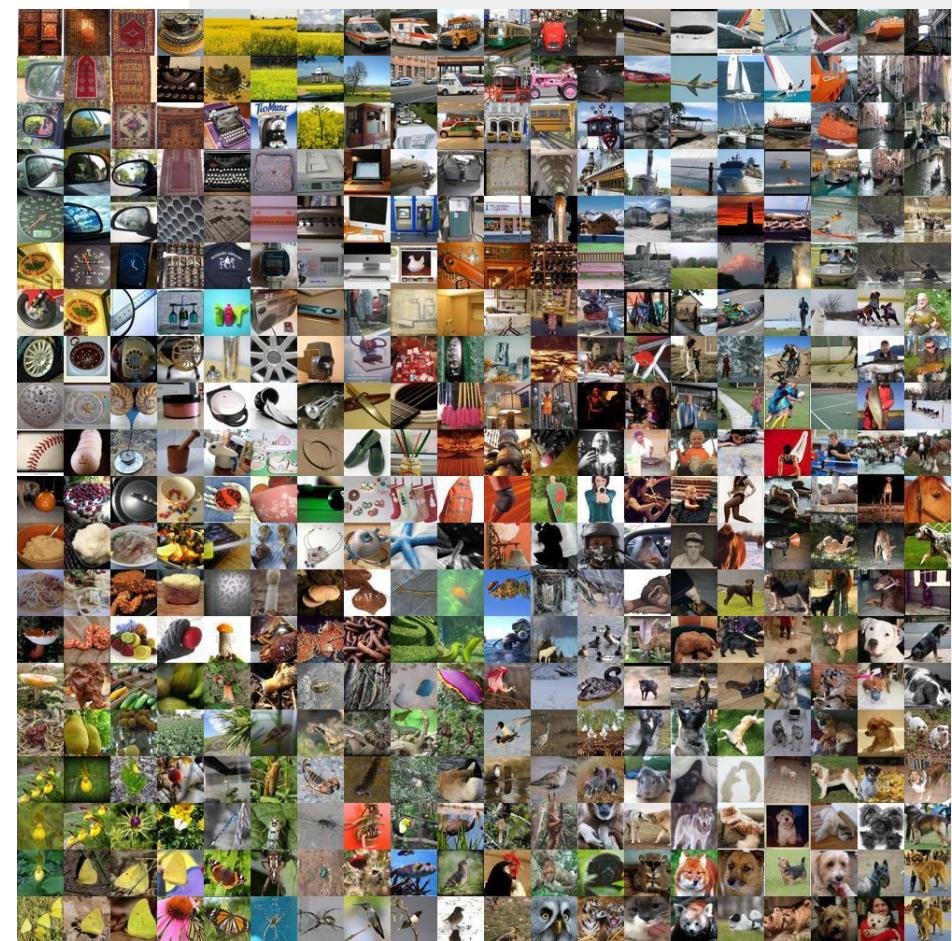
Output ist ein kontinuierlicher Wert, z.B. Temperatur.

3. Klassifikation - Imagenet

Einteilung in 1000 Klassen mit über einer Millionen Bildern.



https://www.researchgate.net/figure/Error-rates-on-the-ImageNet-Large-Scale-Visual-Recognition-Challenge-Accuracy_fig1_332452649



<https://image-net.org/>

3. Klassifikation - MNIST

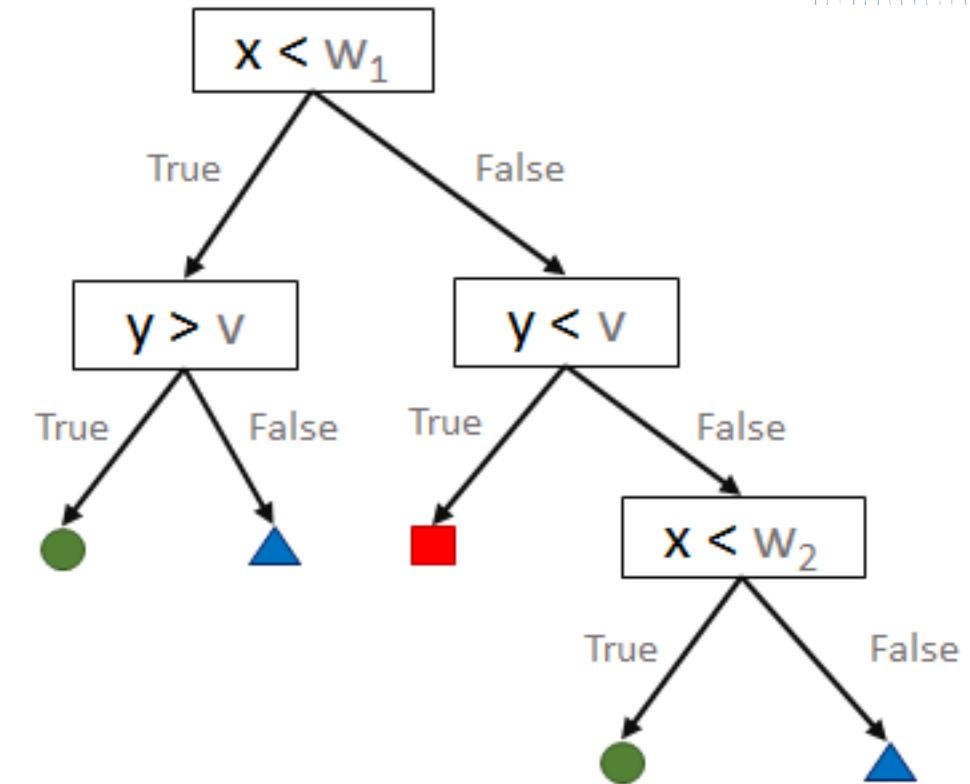
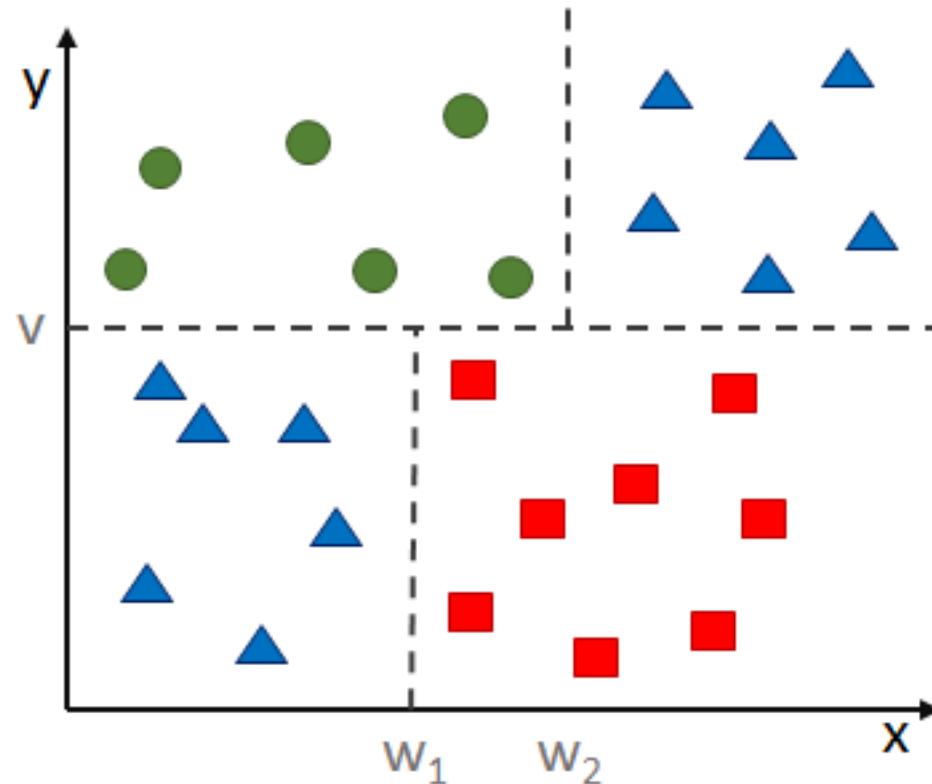
10 Klassen, handgeschriebene Zahlen

Beliebter Datensatz für
Bildklassifikation



<https://de.wikipedia.org/wiki/MNIST-Datenbank>

3. Klassifikation mit Bäumen



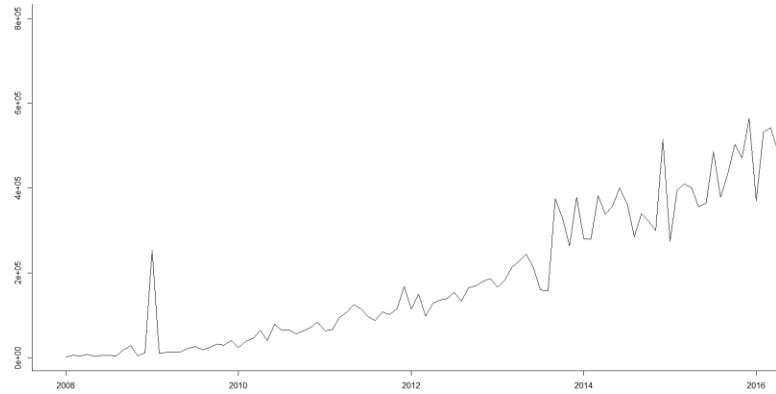
<https://data-science-blog.com/blog/2017/02/13/entscheidungsbauverfahren-artikelserie/>

3. Anwendungen Klassifikation

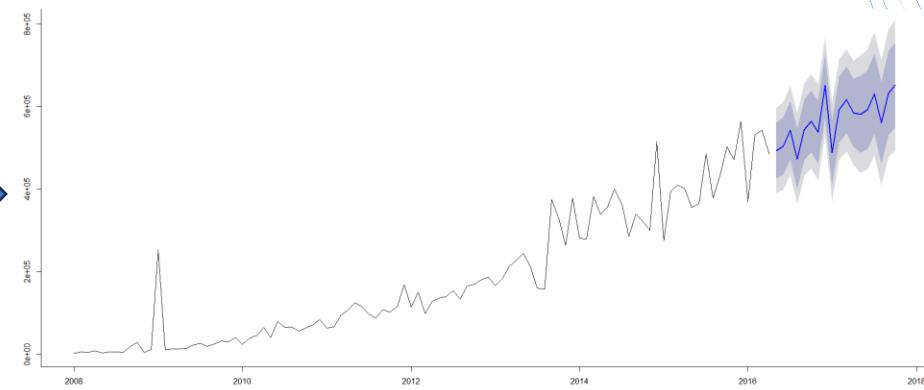
- Predictive Maintenance
 - Input Sensor Maschinen Daten
 - ↓
 - Output: Soll gewartet werden oder nicht (mit Wahrscheinlichkeits-Score)
- Sentiment Analysis
 - Input: Text, z.B. Kundenbewertung
 - ↓
 - Output: Klasse positiv, negativ, neutral

3. Regression - Zeitreihenprognose

Ziel: Vorhersage des nächsten Zeitpunkts

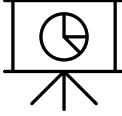
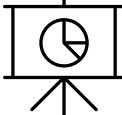


Input



Output

3. Anwendungen Regression

- Zeitreihenprognosen für Waren-Käufe
 - Input: Warenbestand über die letzten Jahre
 -  Output: Vorhersage der Käufe in den nächsten Tagen
- Hauspreis-Bestimmung
 - Input: Baujahr, Größe, Anzahl Zimmer etc.
 -  Output: Preis

Entscheidungsbäume Playground

<https://nlhlong01.github.io/playground/decisiontree.html>

1. Variiere die Max-depth. Beobachte dabei den Fehlerwert RMSE (Root Mean Squared Error). Wie verhält sich der Trainings Error und der Test Error beim Erhöhen der Depth?
2. Wechsle zu „2D Random Forest“. Wie verändert sich das Verhalten?
3. Finde einen Datensatz und Einstellung, bei dem kein Overfitting stattfindet (RMSE Test ist kleiner oder gleich RMSE Train)

3. Mögliche Lösung 3



3. Unsupervised Learning

= Unüberwachtes Lernen

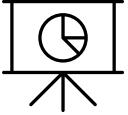
Es gibt kein vorgegebenes bzw. richtiges Ergebnis.

- Cluster identifizieren (siehe Kmeans)
- Anomalie Erkennung
- Empfehlungsdienste (Recommender Systems)

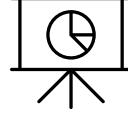
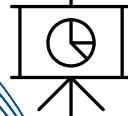
3. Anwendungen Cluster Analyse

- Als Teil der Datenanalyse (Intelligenz Data Analytics)
- Verstehen der Kundengruppen
 - Input: Alter, Geschlecht, gekaufte Waren
 - Output: Einteilung der Kunden in Gruppen mit speziellen Merkmalen

3. Anwendungen Anomalie Erkennung

- Überwachung eines Maschinenparks um ungewöhnliches Verhalten zu erkennen
 - Input: Sensordaten
 -  Output: Ein Score, der den Grad der Abweichung von der Norm angibt.
- Überwachung der Input Daten für einen anderen Machine Learning Algorithmus um Meßfehler zu erkennen
 - Input: Wetter Daten
 -  Output: Ein Score, der den Grad der Abweichung von der Norm angibt.

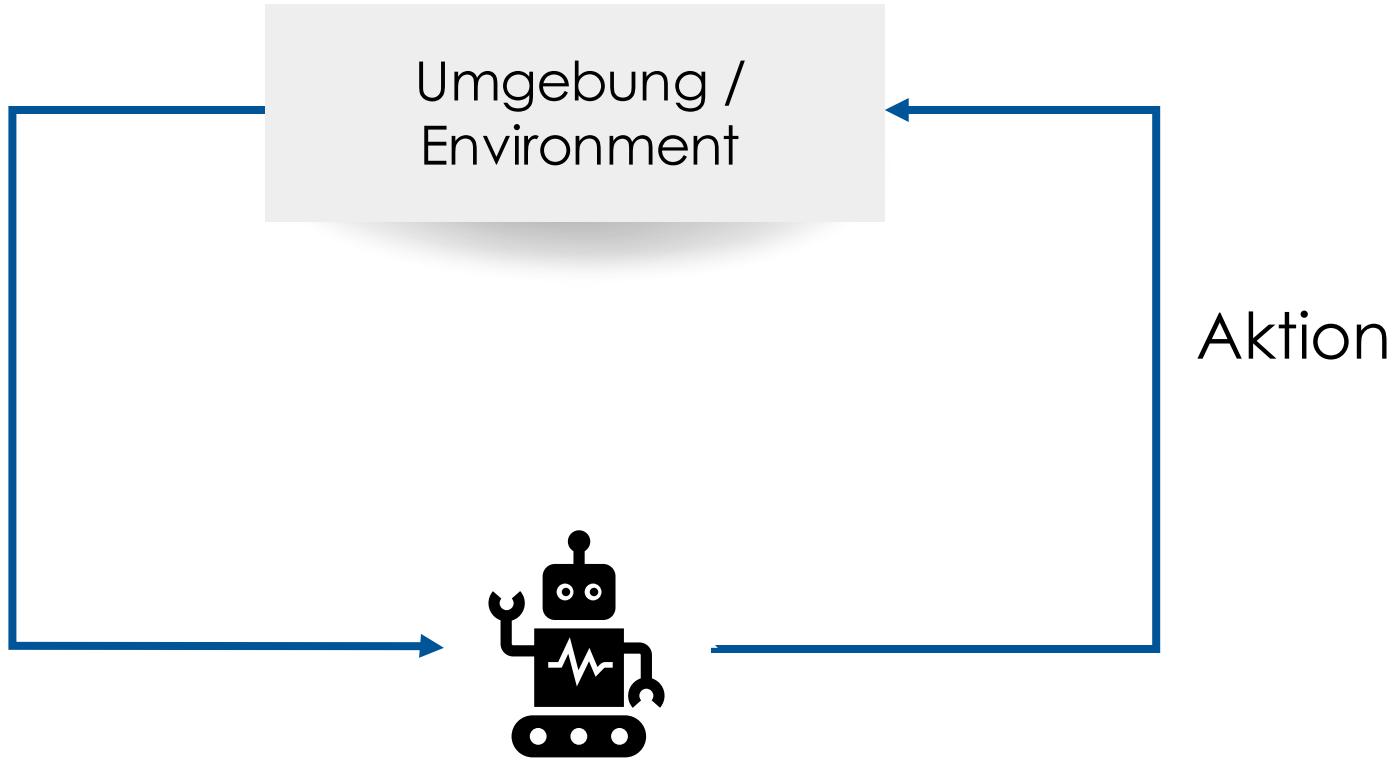
3. Anwendungen Empfehlungssystemen

- Empfehlung für zu Kaufende Waren
 - Input: Alter, Geschlecht, bisher gekaufte Waren
 -  Output: Liste von Waren
- Empfehlung für den nächsten Film
 - Input: Alter, Geschlecht, bisher gesehene Filme
 -  Output: Liste von Filmen

3. Reinforcement Learning

= bestärkendes Lernen

Belohnung (Reward)
+
Beschreibung der
Umgebung (State)



3. Reinforcement Learning mit Atari



Reinforcement Learning

https://alazareva.github.io/rl_playground/

1. Setze Epsilon auf 0. Welches Verhalten zeigt der Agent?
2. Wie wirkt sich der Discount Faktor auf die Q-Werte aus?

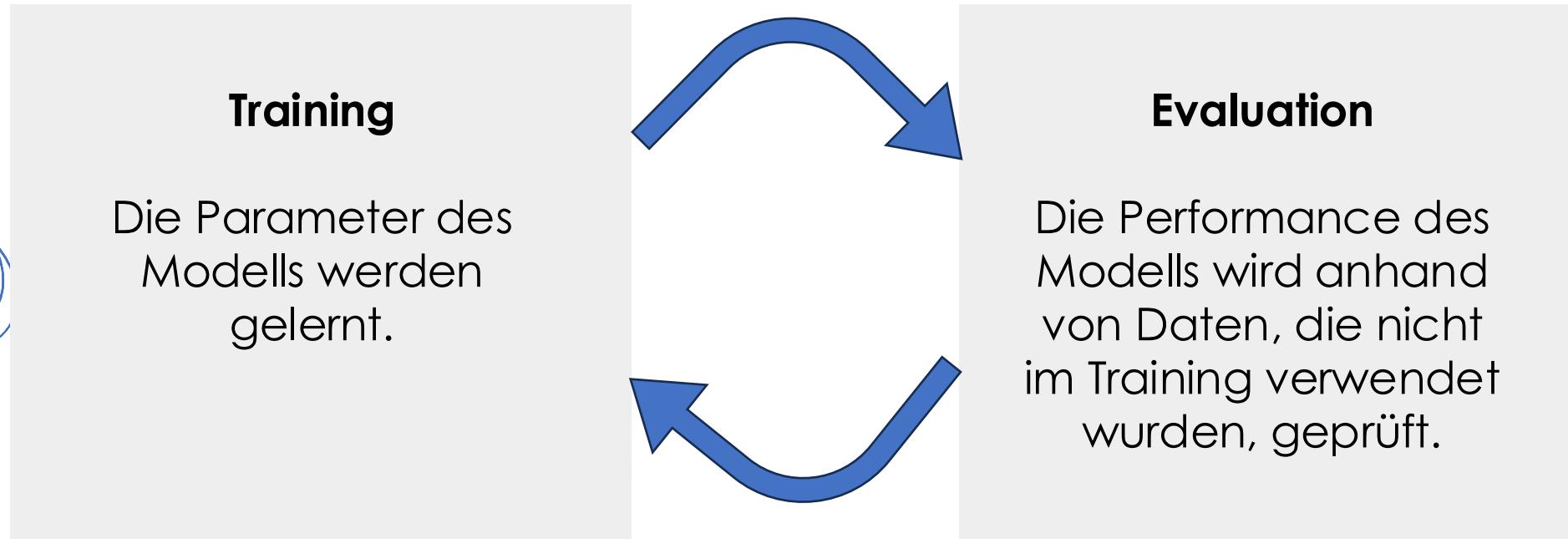
3. Anwendung Reinforcement Learning

- Automatisierter Aktienhandel
 - Input: Aktienkurse der letzten Stunden, aktuelles Portfolio, etc.
 - Output: Aktion Kaufen, Verkaufen, Nichts tun
- Autonomes Fahren
 - Input: Kamera Bild der Umgebung, aktuelle Geschwindigkeit etc.
 - Output: Aktion Beschleunigen / Bremsen, Lenken

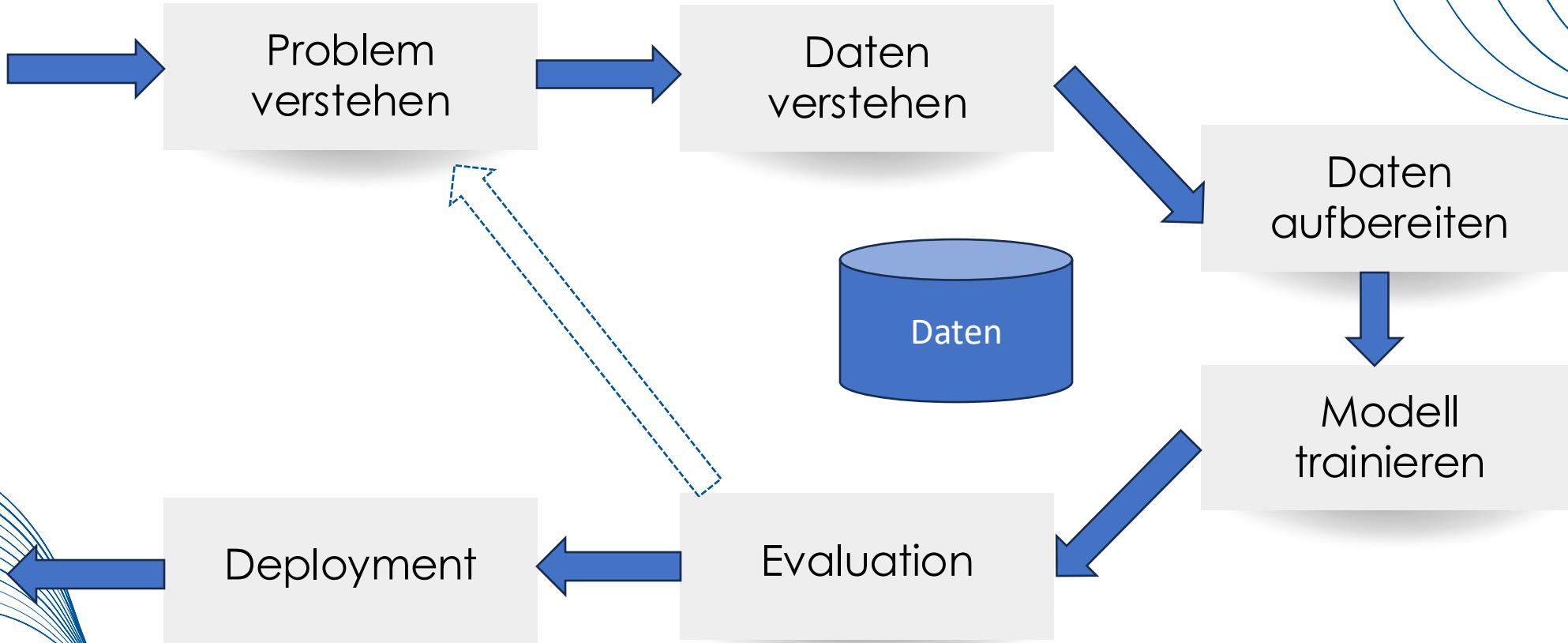
3. Vorgehen im Machine Learning

- Daten werden in 3 Teile aufgeteilt:

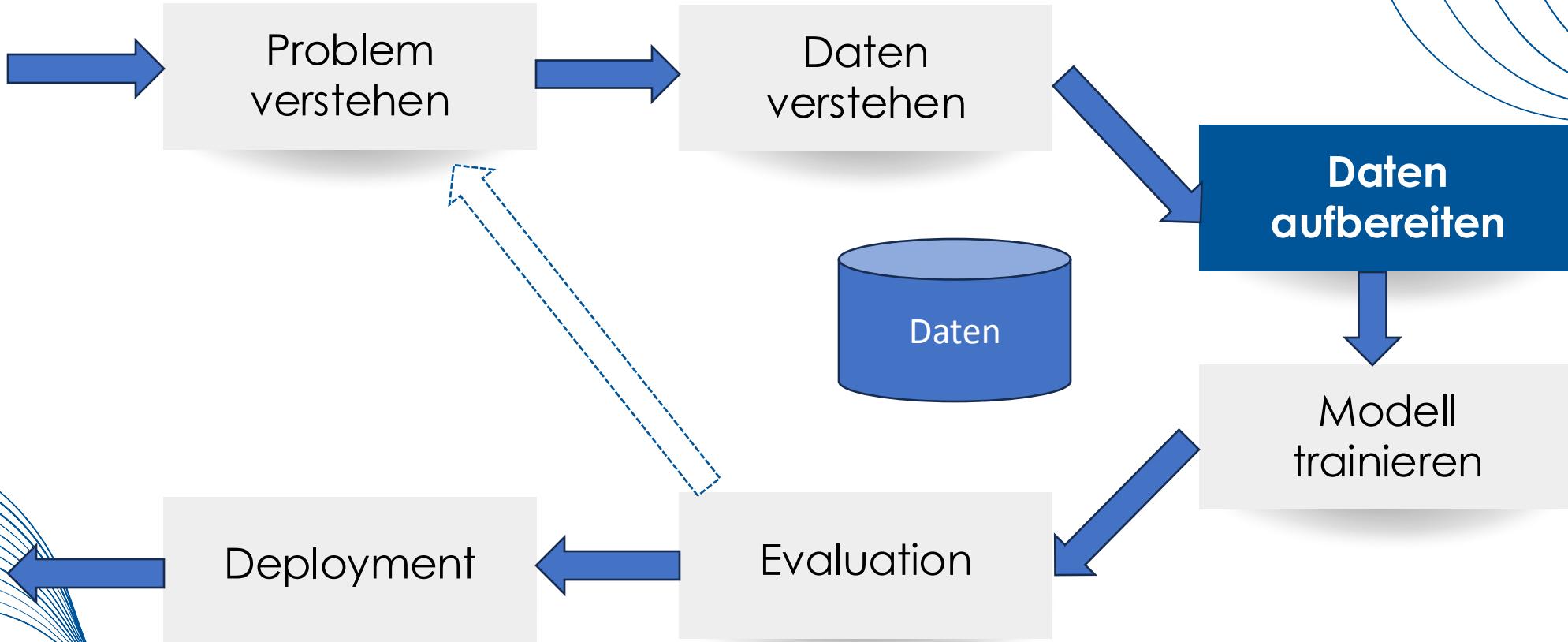
Trainings- / Validierungs- / Test- Datensatz



3. Vorgehen nach CRISP-DM



3. Vorgehen nach CRIPS-DM



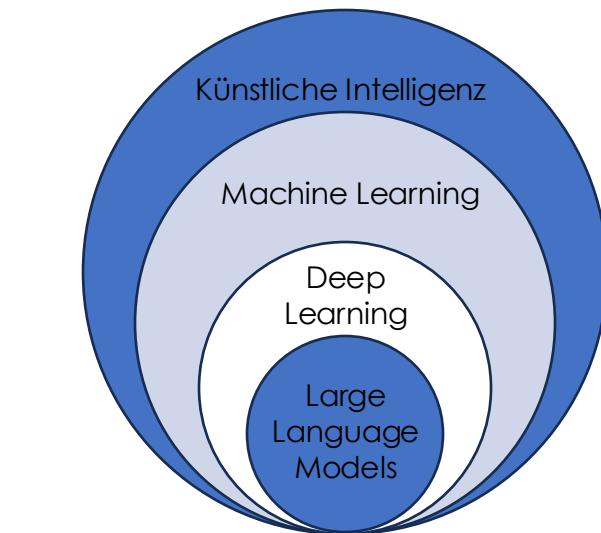
3. Daten aufbereiten – Feature Engineering

- Typische Probleme:
 - Teilweise fehlende Werte
 - Werte liegen als Kategorie vor (z.B. m / w / d)
 - Werte müssen skaliert werden
- Mehrwert durch Erstellung neuer Features
 - z.B. aus Datetime Werte das Feature Tag (=0) oder Nacht (=1)
 - Kombination von bestehenden Features (z.B. Aggregationen pro Produkt-Kategorie)

Python Machine Learning Demo

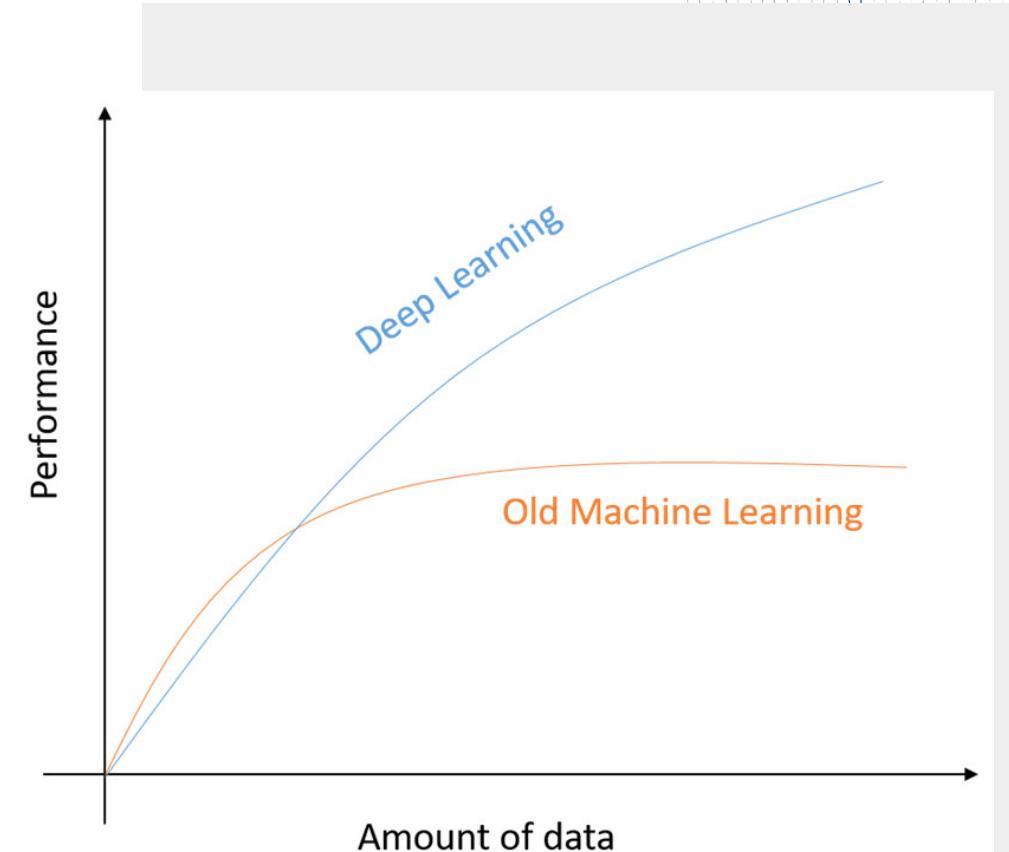
3. Neuronale Netze

- Das aktuell erfolgreichste Modell im Machine Learning sind Neuronale Netze.
- Neuronale Netze können für alle vorgestellten Algorithmen Typen (Supervised, Unsupervised, Reinforcement) eingesetzt werden.
- Neuronale Netze sind in Schichten aufgebaut. Ab 2 Schichten spricht man von einem „tiefen“ Neuronalen Netz: **Deep Learning**.



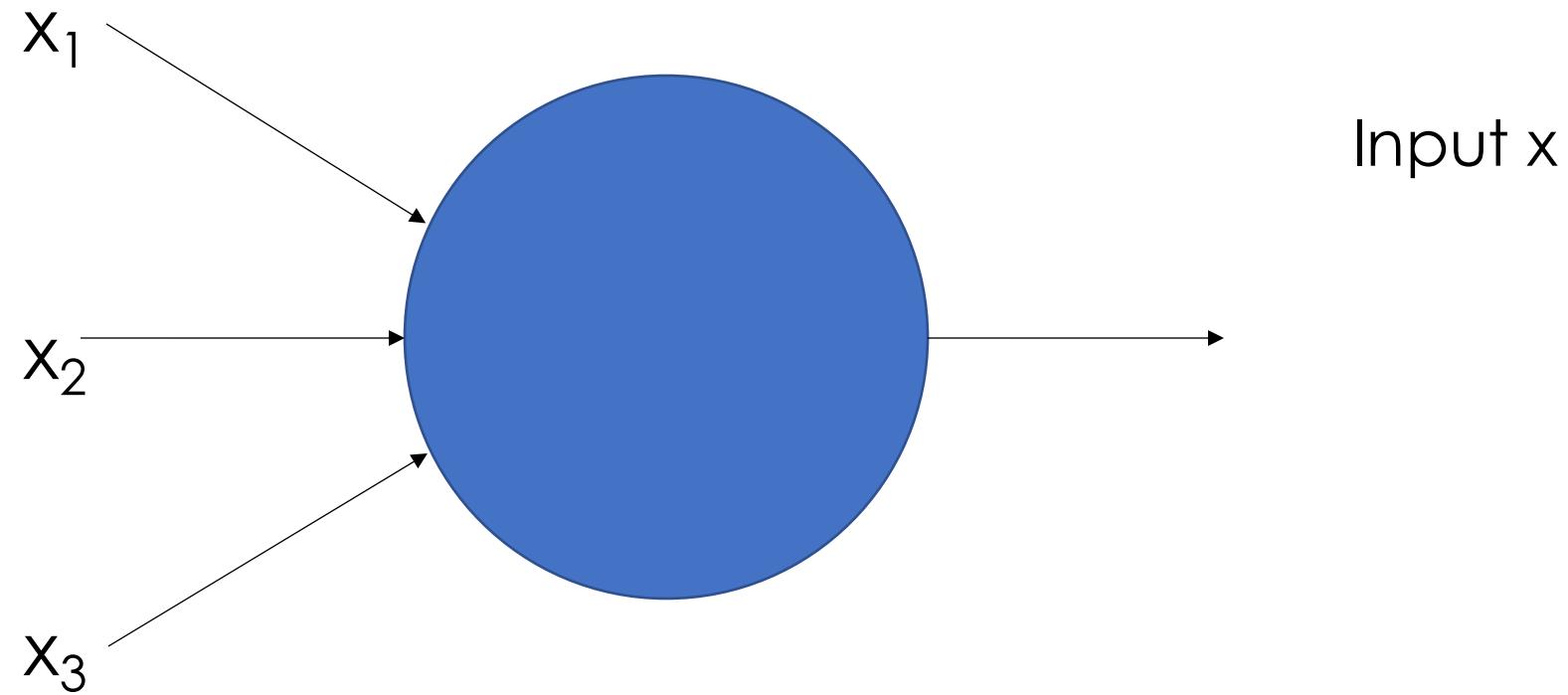
3. Eigenschaften Neuronale Netze

- Feature Engineering ist einfacher
- Performance steigt mit Größe des Datensatzes und aktuell kein Ende in Sicht
- Training eines Modells ist aufgrund der Infrastruktur Anforderung teuer

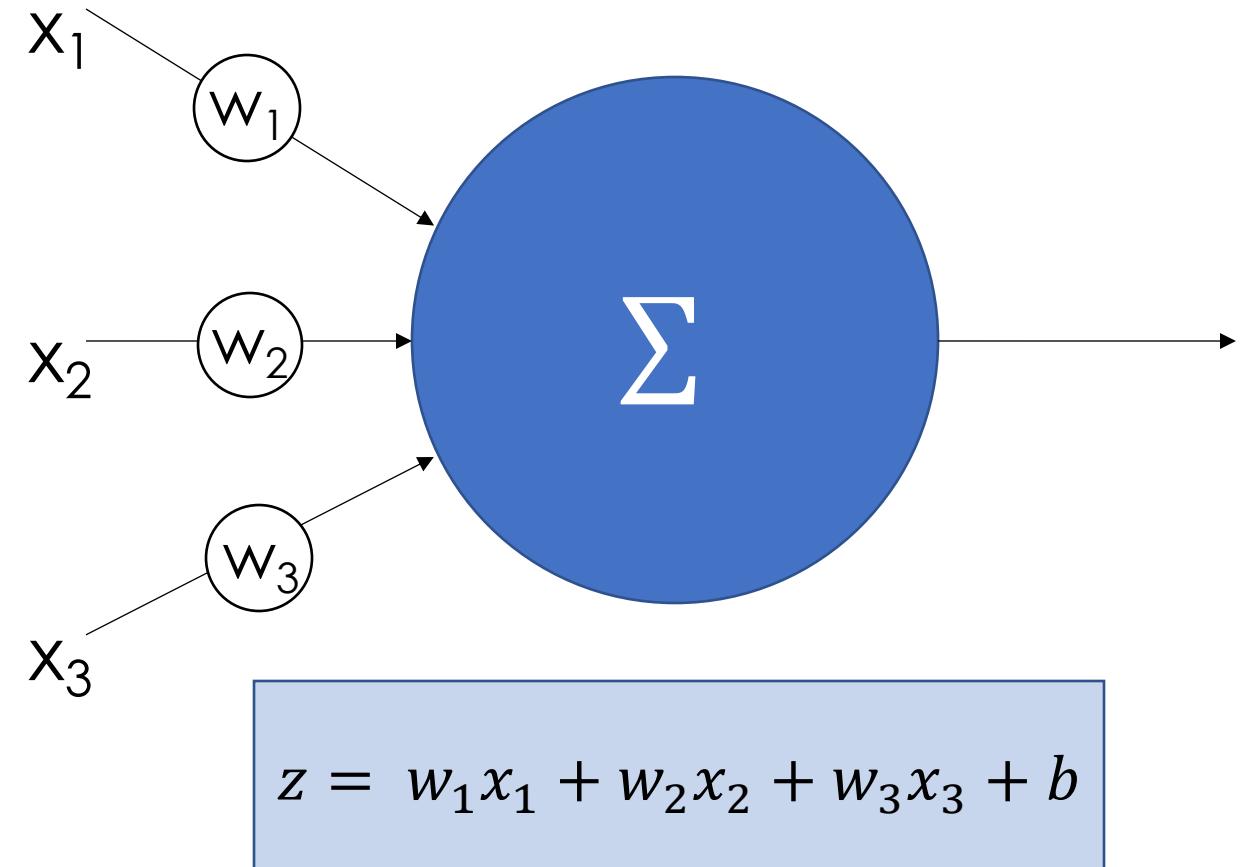


*https://www.researchgate.net/figure/The-performance-of-deep-learning-with-respect-to-the-amount-of-data_fig3_331540139

3. Neuronale Netze – das Neuron



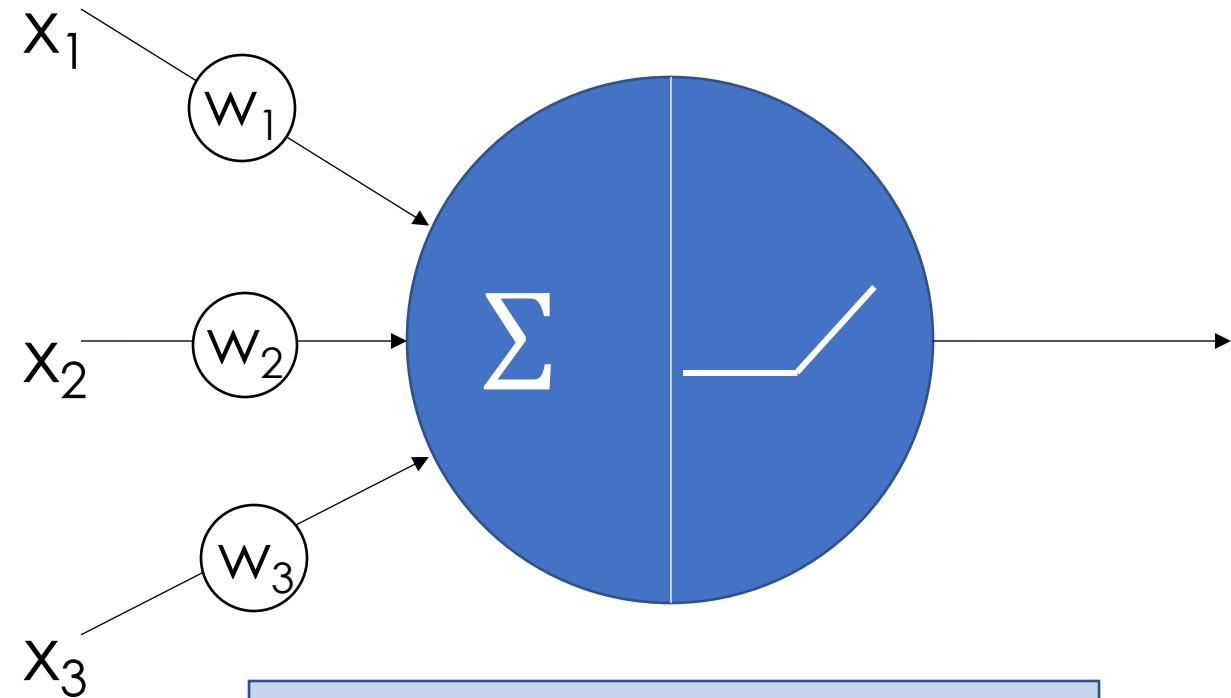
3. Neuronale Netze – das Neuron



Input x

Gewichte oder
Parameter w

3. Neuronale Netze – das Neuron



Input x

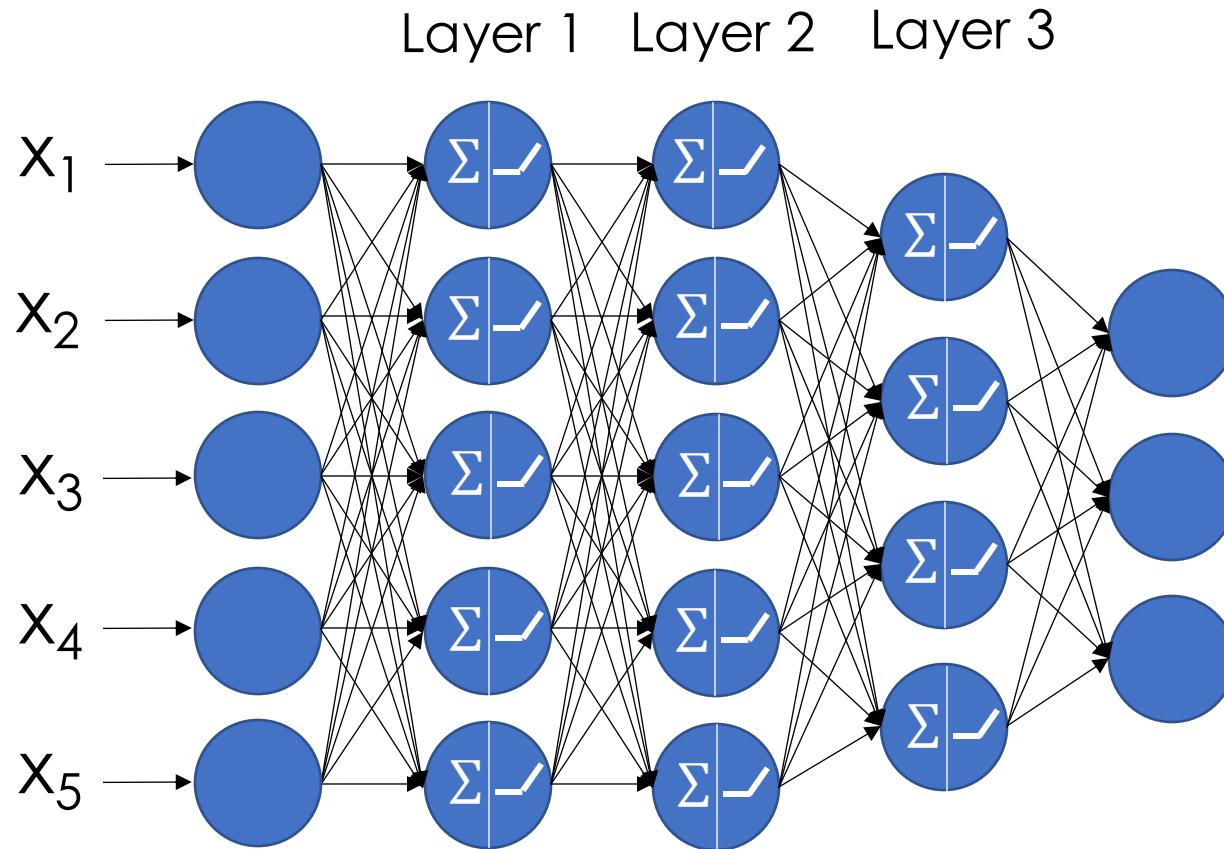
Gewichte oder
Parameter w

$$z = w_1x_1 + w_2x_2 + w_3x_3 + b$$
$$\text{ReLU: } h(z) = \max(0, z)$$

Playground

<https://playground.tensorflow.org/>

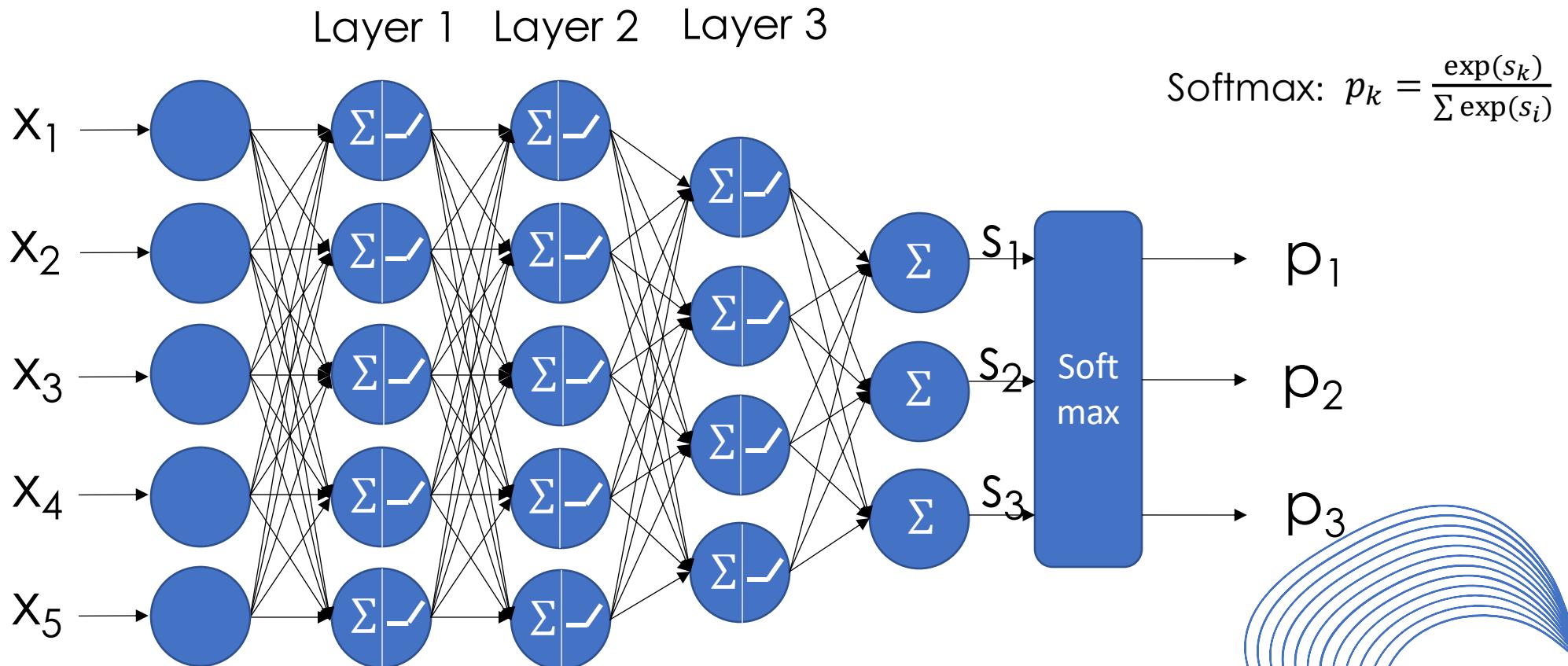
3. Neuronale Netze – Deep Learning



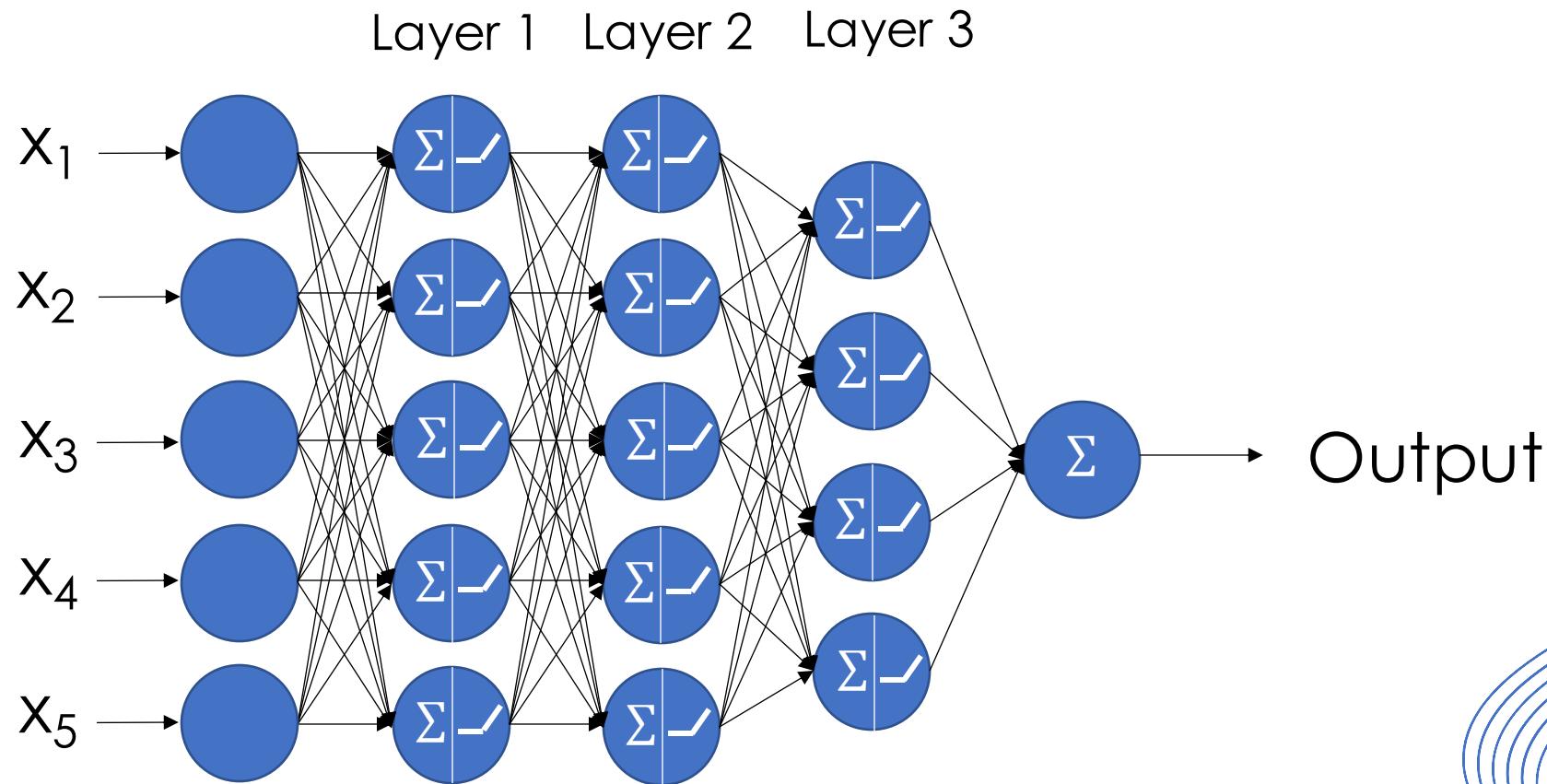
Fully-Connected: alle Neuronen in aufeinanderfolgenden Layers sind miteinander verbunden

Multi-Layer Perceptron (MLP): Fully-Connected Neuronales Netz mit >2 Layer.

3. Deep Learning für Klassifikation

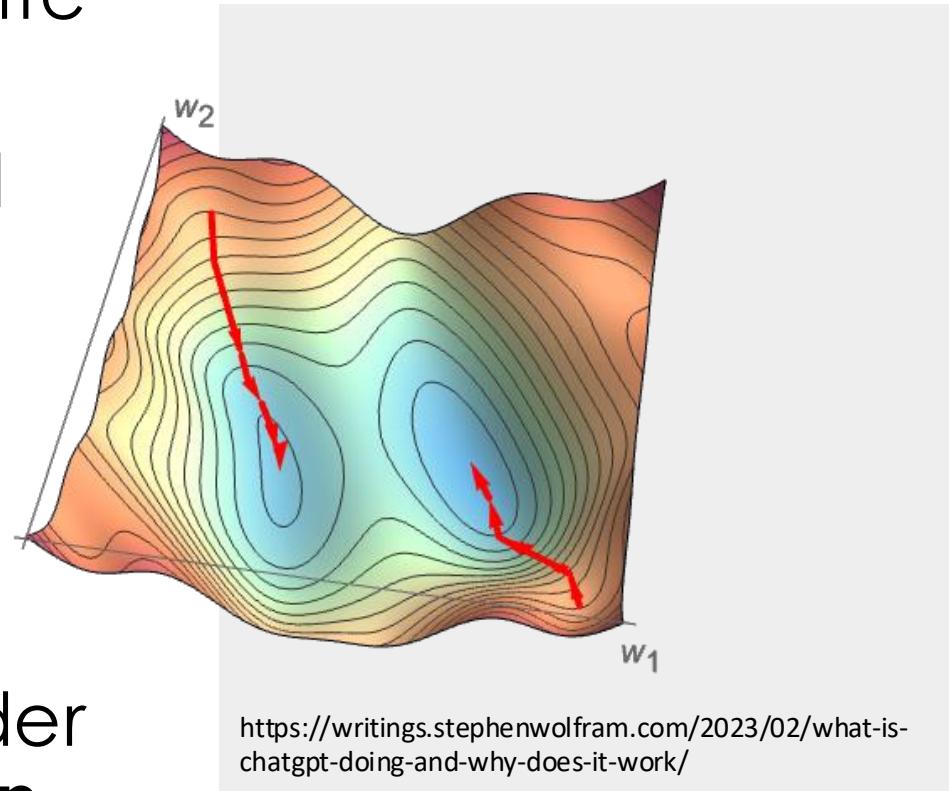


3. Deep Learning - Regression



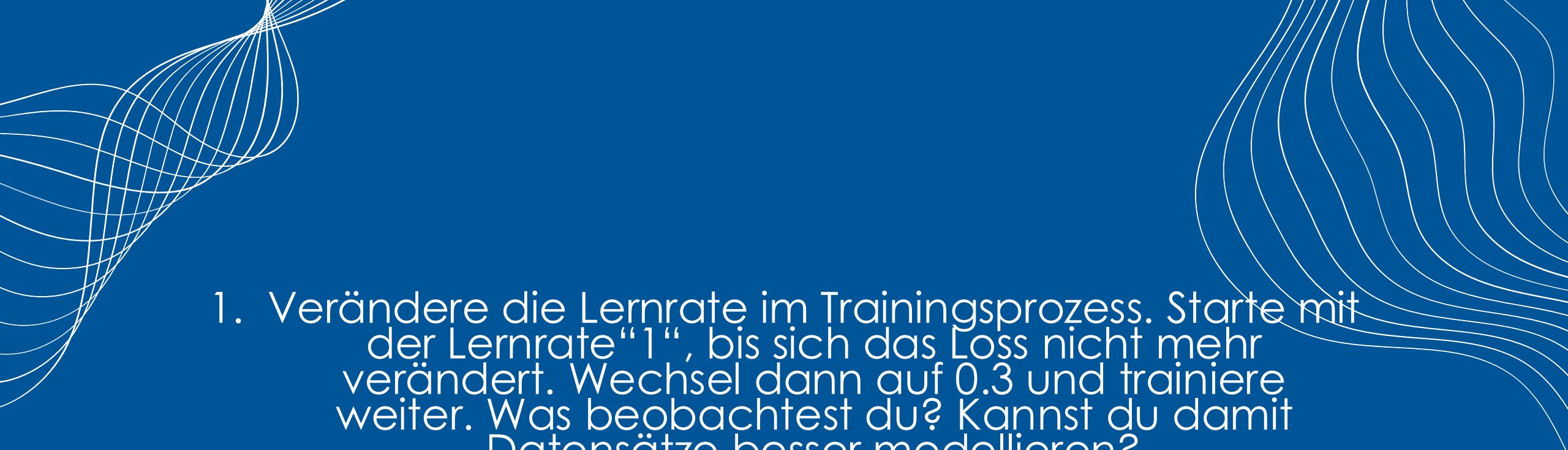
3. Wie findet man die richtigen Gewichte?

- Das Finden der richtigen Gewichte nennt man **Training**
- Das Training ist eine iterative und ggf. lange Berechnung
- Dabei werden die Daten dem Neuronalen Netzwerk mehrfach gezeigt und die Gewichte Schrittweise angepasst
- Der Algorithmus zur Anpassung der Gewichte heißt **Backpropagation**

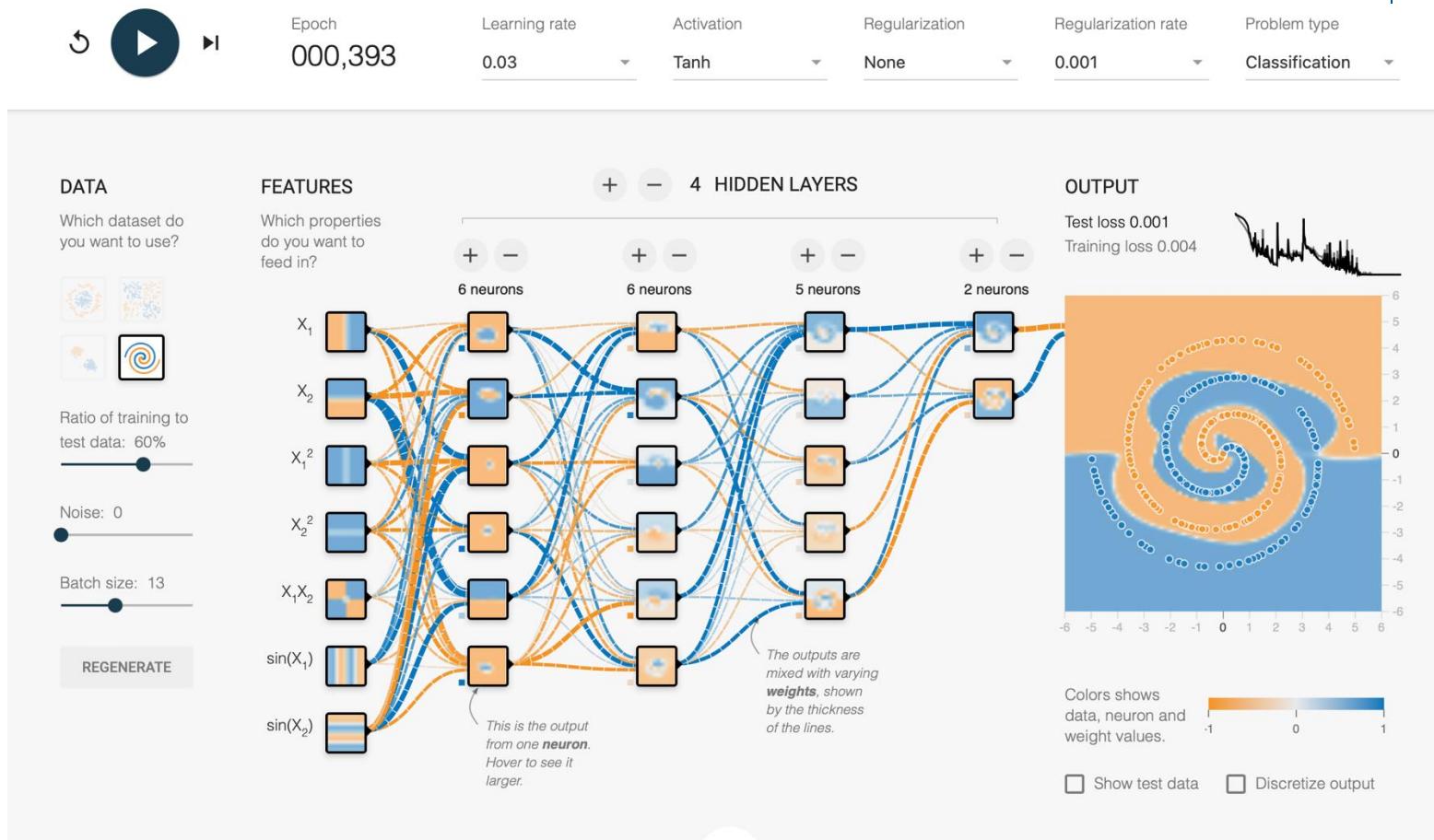


Playground

<https://playground.tensorflow.org/>

- 
1. Verändere die Lernrate im Trainingsprozess. Starte mit der Lernrate "1", bis sich das Loss nicht mehr verändert. Wechsel dann auf 0.3 und trainiere weiter. Was beobachtest du? Kannst du damit Datensätze besser modellieren?
 2. Nutze bei Klassifikation den letzten Datensatz (Spirale) und trainiere ein Neuronales Netz, das die Aufgabe löst. Welche Tricks helfen?
 3. Erhöhe die Anzahl der Neuronen und Layer auf das Maximum. Wie verhält sich das Neuronale Netz?
 4. Wie wirkt sich das „Noise“ auf das Loss aus? Warum?

3. Mögliche Lösung Aufgabe 2



3. Deep Learning – Was noch?

Größe des
Netzes

Hyperparameter

Jede Menge Tricks

Architektur

3. Deep Learning – Was noch?

Größe des
Netzes

Hyperparameter

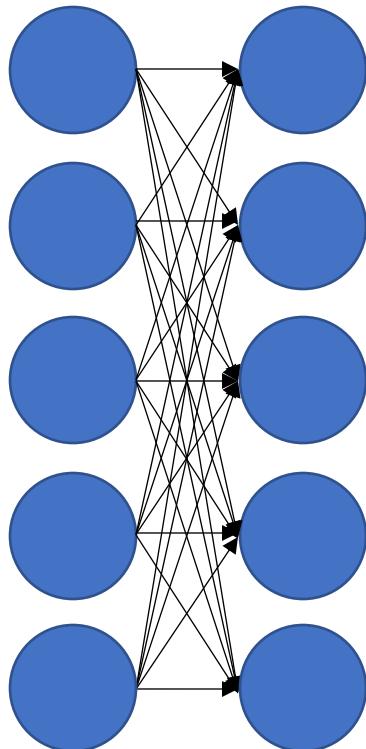
Jede Menge Tricks

Architektur

3. Deep Learning – Convolutions

Bisher im Neuronalen Netz

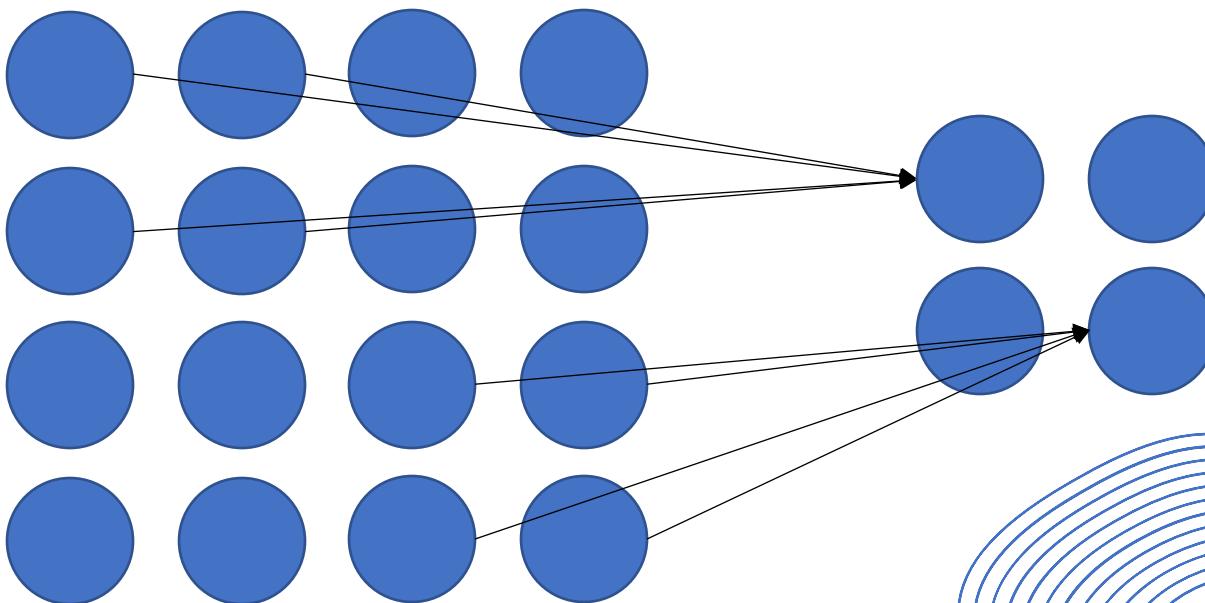
Layer 1 Layer 2



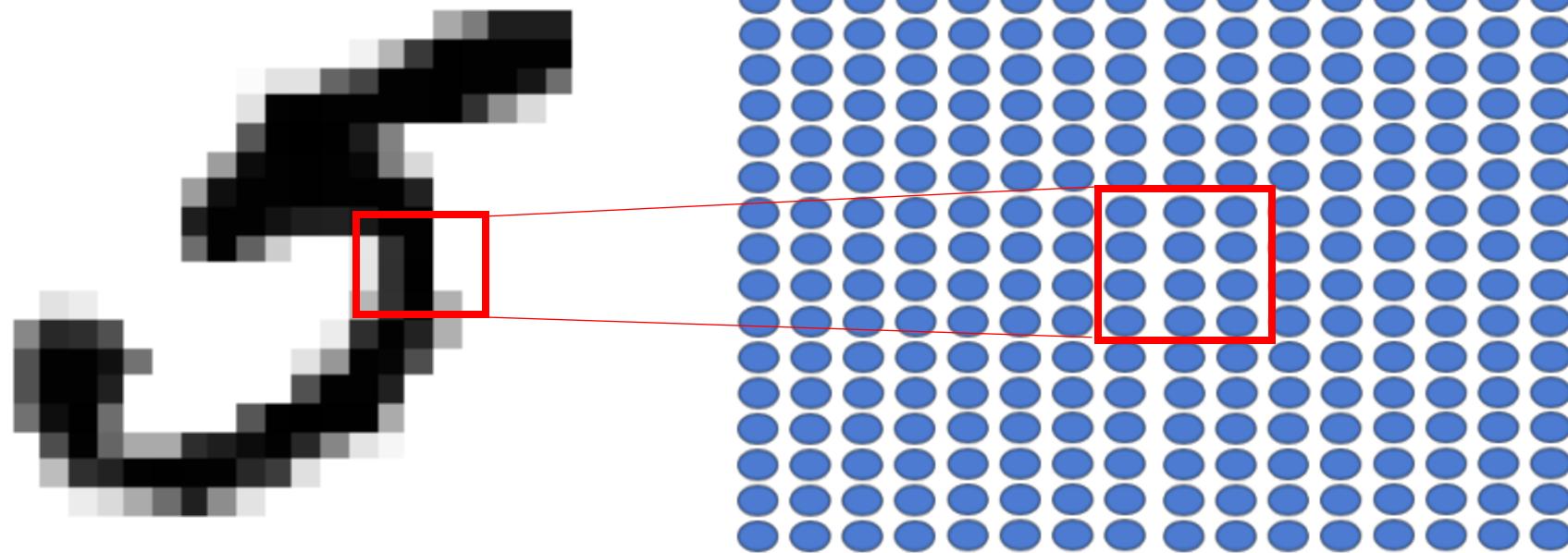
Im Convolutional Neuronal Net (CNN)

Layer 1

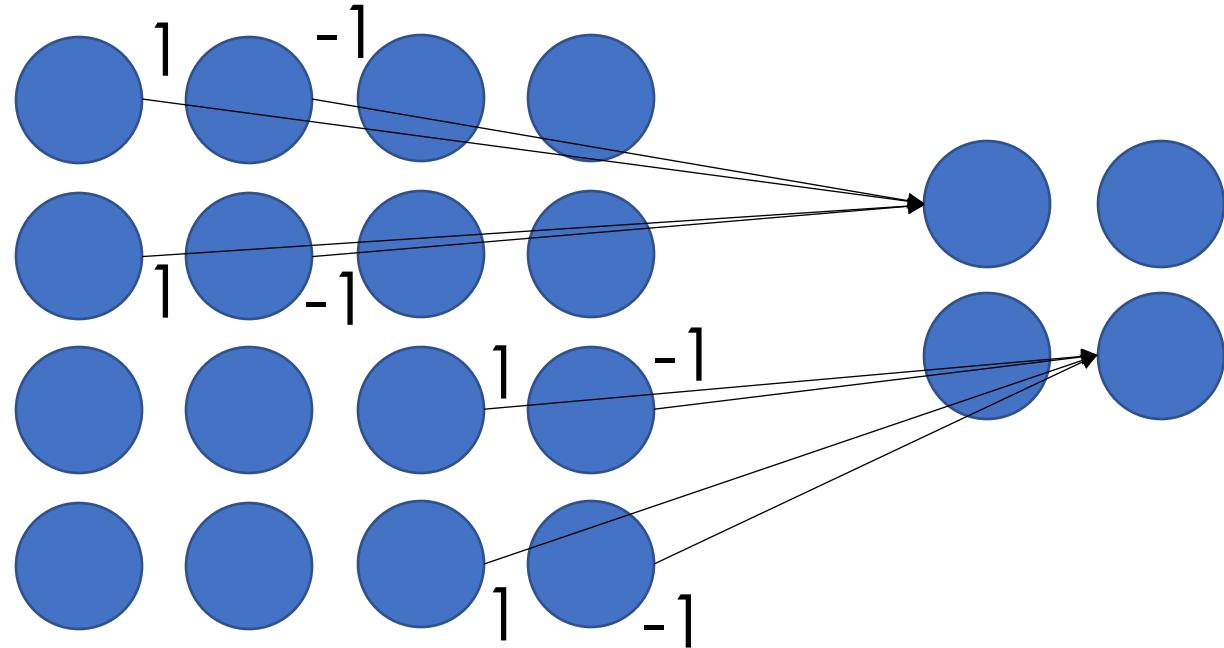
Layer 2



3. Deep Learning - Convolutions

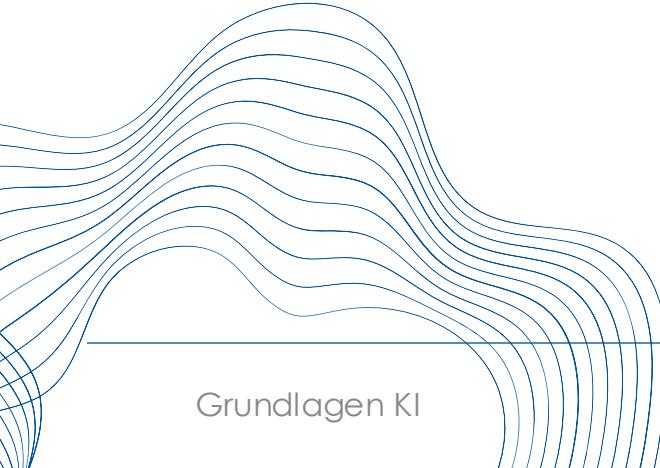


3. Convolutions als Filter



Convolutions können auch als Filter verstanden werden. In diesem Fall, um vertikale Kanten zu erkennen.

3. Deep Learning - Convolutions



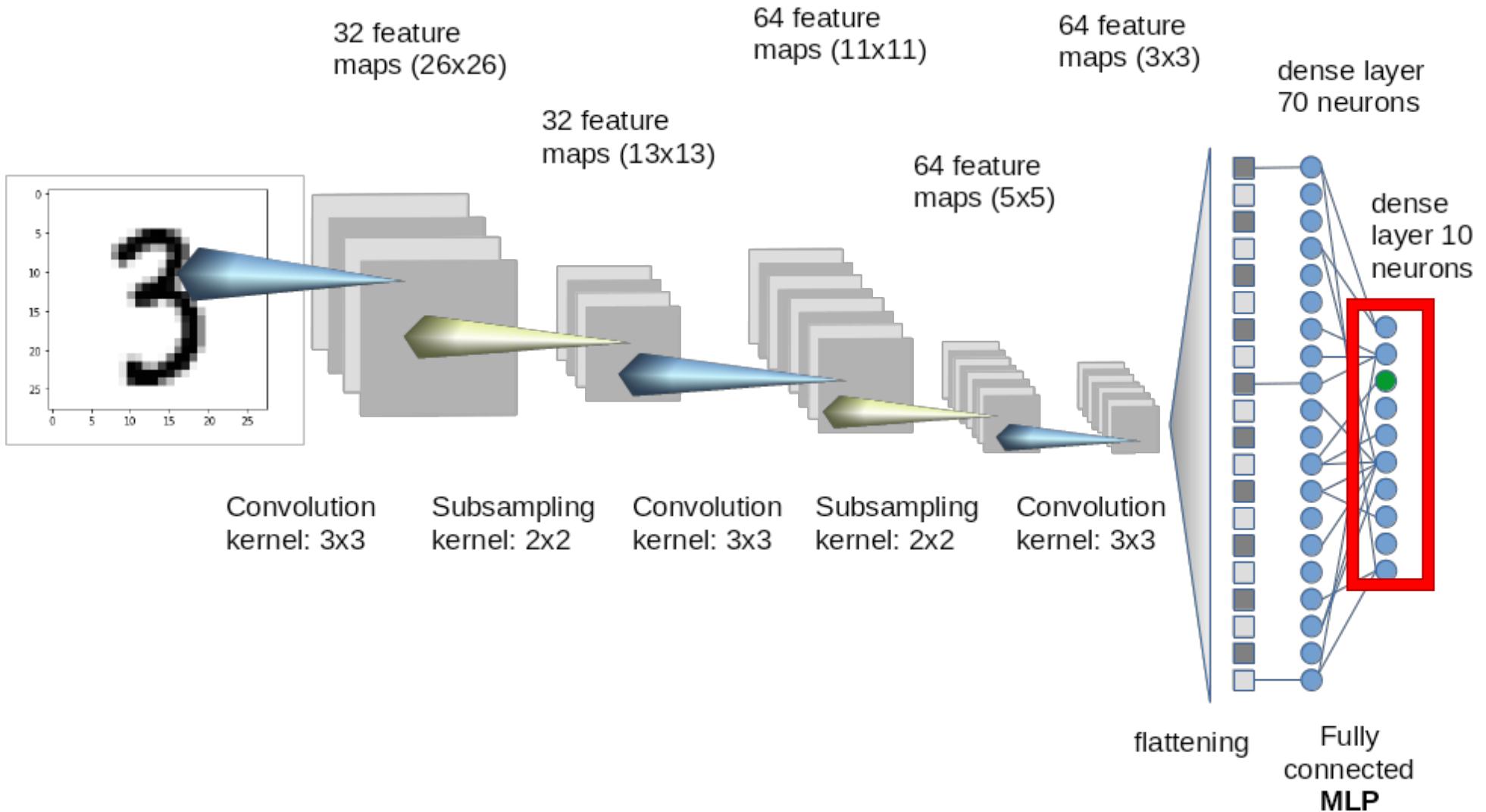
<https://writings.stephenwolfram.com/2023/02/what-is-chatgpt-doing-and-why-does-it-work/>



CNNs visualisiert

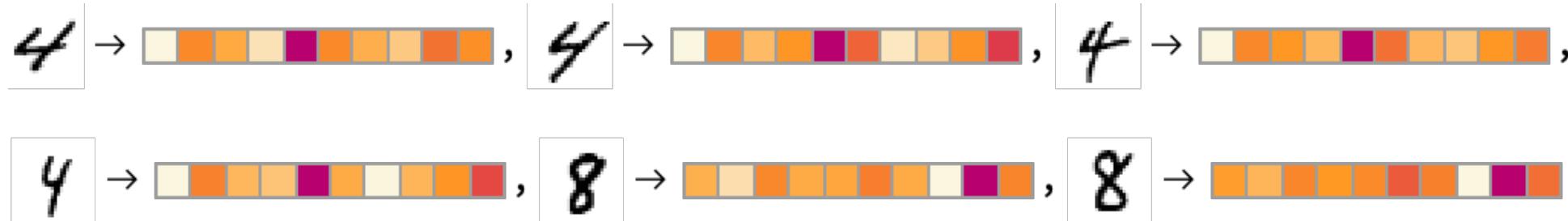
<https://poloclub.github.io/cnn-explainer/>

3. CNNs – Embeddings



3. CNNs – Embeddings

Embedding = Repräsentation (des Bildes) mit einem Vektor / Array von Zahlen



Embeddings visualisiert

<https://projector.tensorflow.org>

CNN Training Demo

Python + Tensorflow + Tensorboard
MNIST Datensatz

3. CNNs – Fine Tuning

Herausforderung

1. Das Training von CNNs ist aufwendig
2. Wenige Bilder vorhanden

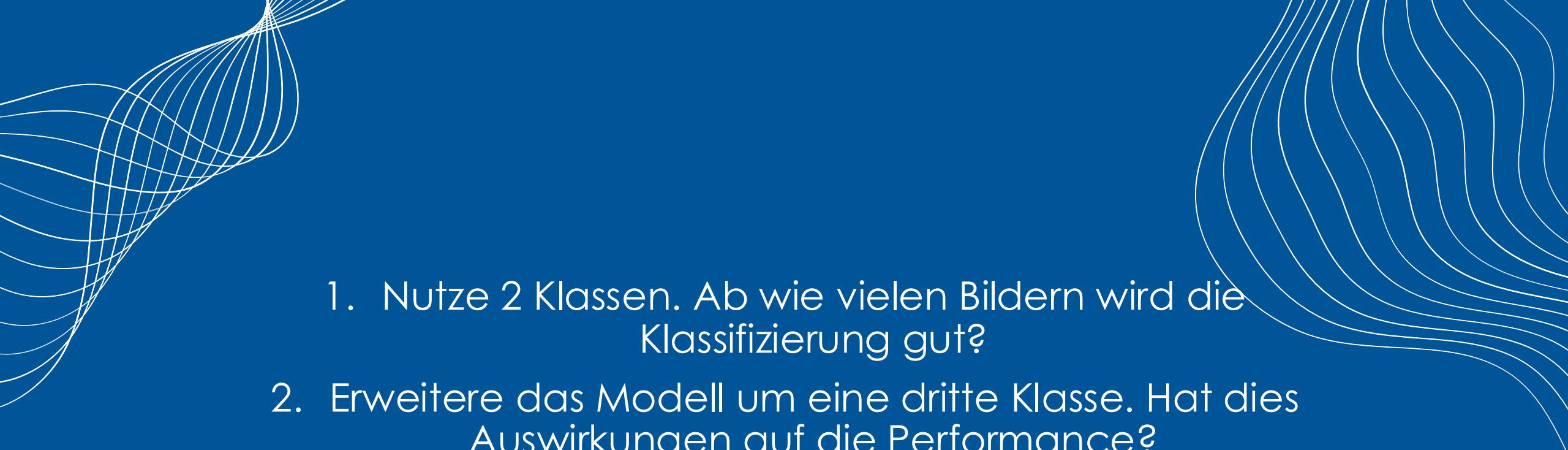


Lösung

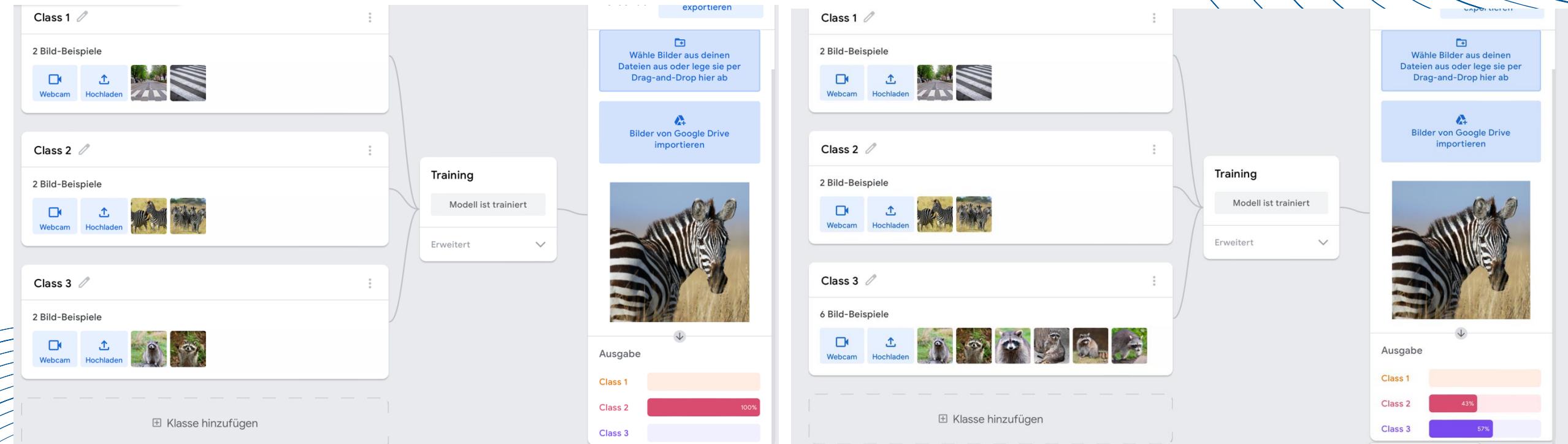
Ein CNN, das auf imagenet trainiert wurde, hat schon nützliche Convolutions gelernt. Nutze das CNN (und entferne gegebenenfalls die letzten Layers) und trainiere mit eigenen Daten weiter.

Fine Tune

<https://teachablemachine.withgoogle.com/train>

- 
1. Nutze 2 Klassen. Ab wie vielen Bildern wird die Klassifizierung gut?
 2. Erweitere das Modell um eine dritte Klasse. Hat dies Auswirkungen auf die Performance?
 3. Trainiere das Modell mit ungleicher Anzahl je Klasse. Stellst du einen Unterschied fest?

3. Aufgabe 3



3. Deep Learning mit Text – Large Language Models

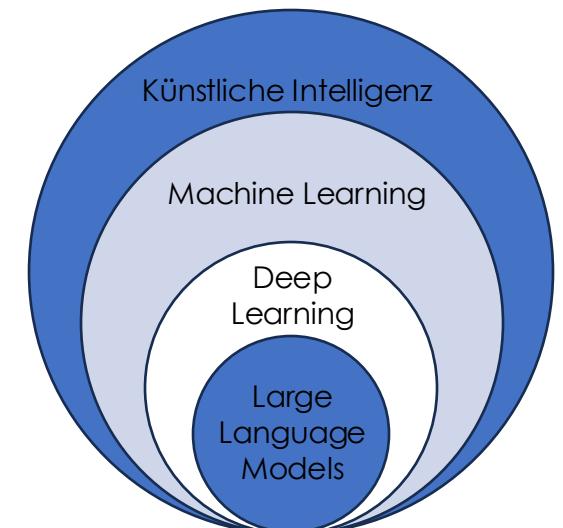
1. Von Wörtern zu Tokens

- Tokens können Wörter, Wortteile oder einzelne Zeichen sein

“Das ist ein Beispiel” → [“Das”, “ist”, “ein”, “Bei”, “spiel”]

Vorteile:

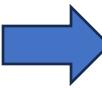
- Kleinere Vokabular-Größe, das auch seltene Wörter beinhalten (da diese in Sub-Wörter zerlegt werden)
- Auch Subfixe (z.B Beugungen von Verben) können erhalten bleiben



3. Deep Learning mit Text – Large Language Models

2. Von Tokens zu Encoding

- Jedes Token bekommt eine eigene ID

[“Das”, “ist”, “ein”, “Bei”, “spiel”]  [42, 13, 81, 32, 96]

- Anschließend Umwandlung in „One Hot“ Vektoren

42  [0, 0, ..., 0, 1, 0, ..., 0]

42. Stelle im Vektor

Tokenizer in Action

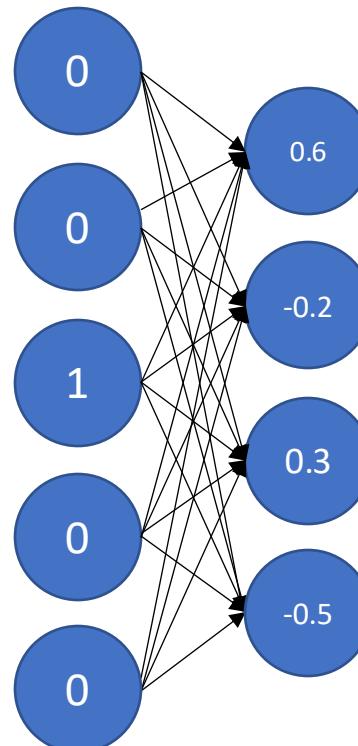
<https://huggingface.co/spaces/Xenova/the-tokenizer-playground>

<https://gptforwork.com/tools/tokenizer>

3. Deep Learning mit Text – Large Language Models

3. Von Token-Encoding zu Embeddings

Input: One-Hot
Encoded Vektor

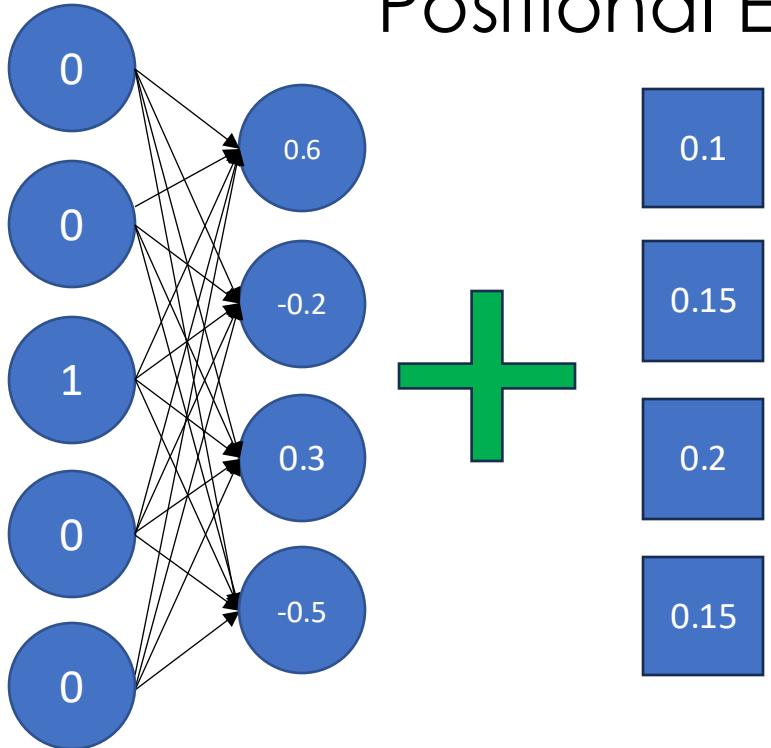


Embedding Layer:
durch Training des
Neuronalen Netzes
gelernte Gewichte

3. Deep Learning mit Text – Large Language Models

3. Von Token-Encoding zu Embeddings

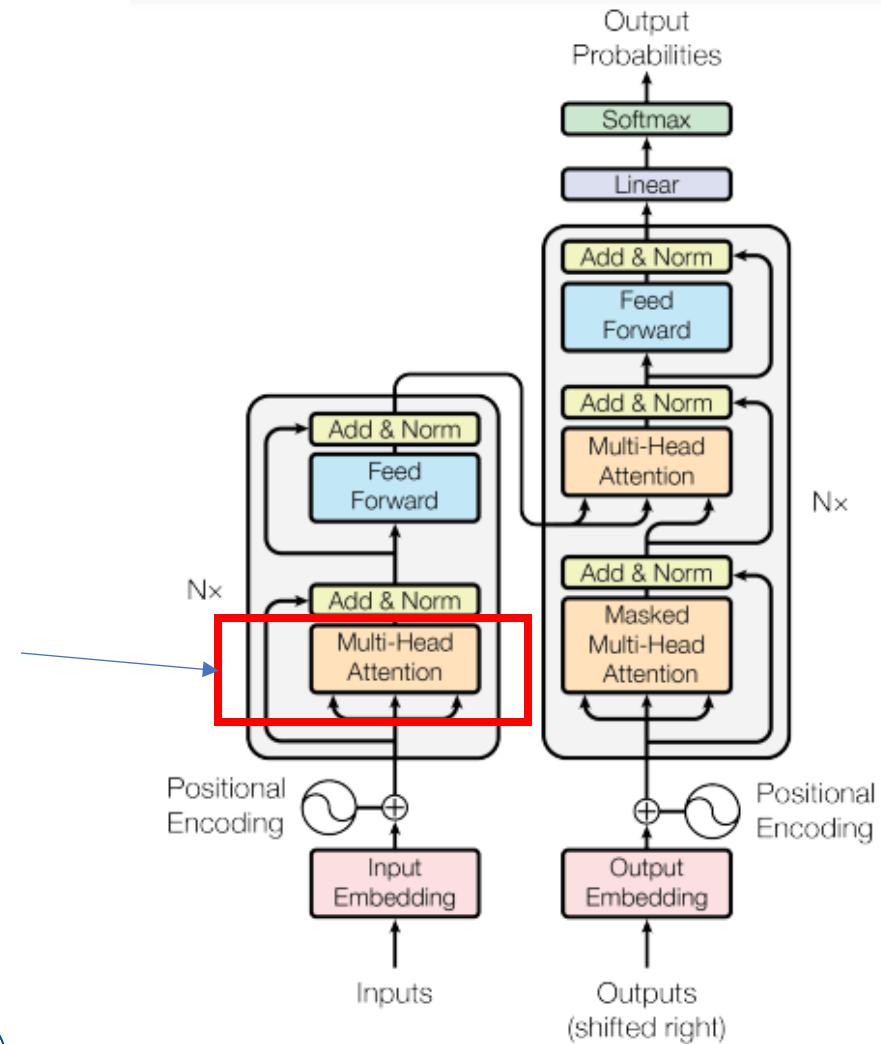
Positional Encoding



Berechnungsvorschrift
aus der Position des
Tokens im Satz

3. LLMs - Transformer

Attention is all
you need

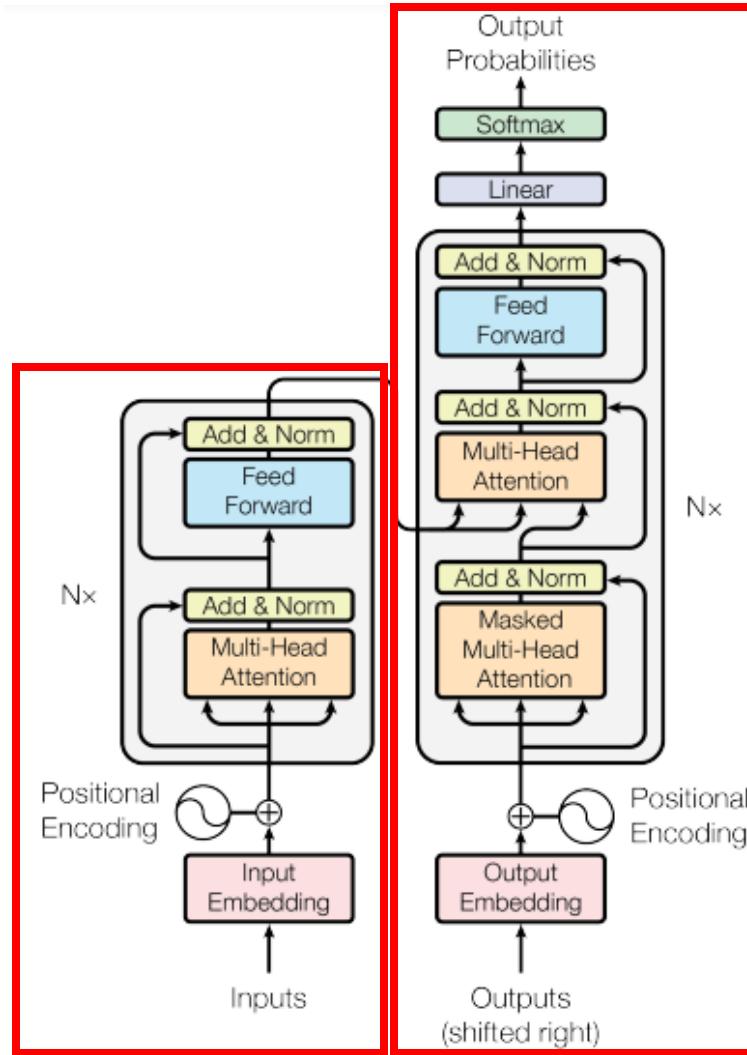


Attention Mechanismus

<https://poloclub.github.io/transformer-explainer/>

3. LLMs - Transformer

Encoder



Decoder

3. Encoder und Decoder Architektur

Encoder

Erkennen der
Eigenschaften des
Textes

Klassifikation des
Textes,
Embeddings
erstellen

BERT

Decoder

Input Sequenz
erweitern

Textvorhersage,
Next Token
Prediction

GPT

Encoder Decoder

Sequenz zu
Sequenz

Übersetzung,
Zusammenfassung

T5

3. Bidirectional Encoder Representations from Transformers (BERT)

- Mit BERT werden mehrere Ideen eingeführt



Encoder only
Architektur



Pre Training mit 2
Auxiliary Tasks



Bidirectional: Attention
auch nach vorne



[CLS] Token für
Embedding der
Sequenz

3. BERT: Masked Language Model

[CLS] Ich ging zur [MASK], um Geld abzuheben

- CLS ist ein spezielles Token, das den Start einer Sequenz markiert.
- Beim Auxiliary Task MLM wird auf den Transformer ein Neuronales Netz zur Klassifikation des maskierten Tokens angefügt.
- Der Attention Mechanismus kann auch nach vorne gerichtet werden.

3. BERT: Next Sentence Prediction

[CLS] Ich mag Musik. Ich gehe zum Konzert.

- Beim Auxiliary Task NSP wird auf den Encoder beim [CLS] Token ein Neuronales Netz zur Klassifikation für zwei Klassen angefügt.
- Input sind zwei Sätze. Die Klassen geben aus, ob die Sätze in richtiger Reihenfolge stehen.
- Damit soll im [CLS] Token die Gesamte Sequenz repräsentiert werden.

3. BERT – Fine Tuning

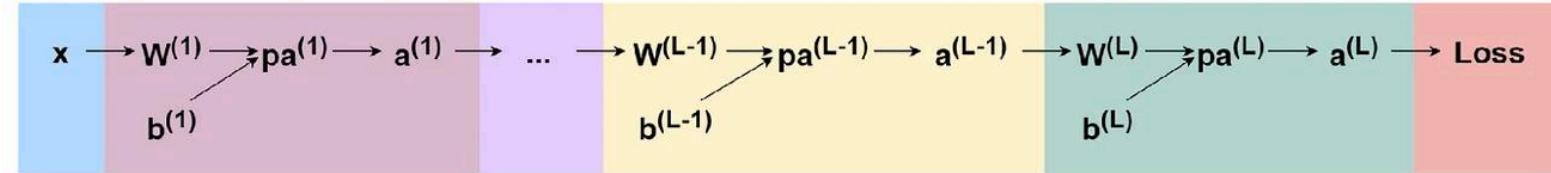
- Nach dem Training auf die beiden Auxiliary Tasks, kann das Training für den eigentlichen Zweck stattfinden, z.B.
 - Text Klassifikation
 - Sentiment Analysis
 - Named Entity Recognition
- Vorteil: das Neuronale Netz hat bereits ein gutes Verständnis von Sprache (gute Repräsentationen von Tokens gelernt).
- Der letzte Klassifikations-Layer aus den Auxiliary Tasks wird nicht weiter benötigt. Stattdessen muss für das Fine Tuning ein neuer Kopf angefügt werden.

BERT Visualisierung

3. Nachtrag - Backpropagation

Ziel: Gewichte anpassen

$$W_{neu}^{(L)} = W_{alt}^{(L)} - \eta \frac{\partial Loss}{\partial W^{(L)}}$$



Wie berechnet sich $\frac{\partial Loss}{\partial \mathbf{W}^{(L)}}$: ?

$$Loss := \frac{1}{2}(\mathbf{a}^{(L)} - \mathbf{y})^2$$

Dafür muss zunächst die Ableitung des Loss in Richtung $\mathbf{a}^{(L)}$ berechnet werden:

$$\frac{\partial Loss}{\partial \mathbf{a}^{(L)}} = \frac{\partial \frac{1}{2}(\mathbf{a}^{(L)} - \mathbf{y})^2}{\partial \mathbf{a}^{(L)}} = \mathbf{a}^{(L)} - \mathbf{y}$$

Im nächsten Schritt wird die Ableitung des Loss nach $\mathbf{pa}^{(L)}$ berechnet:

$$\frac{\partial Loss}{\partial \mathbf{pa}^{(L)}} = \frac{\partial Loss}{\partial \mathbf{a}^{(L)}} \frac{\partial \mathbf{a}^{(L)}}{\partial \mathbf{pa}^{(L)}} = \frac{\partial Loss}{\partial \mathbf{a}^{(L)}} \sigma'(\mathbf{pa}^{(L)})$$

Nun kann das Loss in Richtung $\mathbf{W}^{(L)}$ berechnet werden:

$$\frac{\partial Loss}{\partial \mathbf{W}^{(L)}} = \frac{\partial Loss}{\partial \mathbf{pa}^{(L)}} \frac{\partial \mathbf{pa}^{(L)}}{\partial \mathbf{W}^{(L)}}$$

$$\frac{\partial \mathbf{pa}^{(L)}}{\partial \mathbf{W}^{(L)}} = \frac{\partial (\mathbf{b}^{(L)} + \mathbf{W}^{(L)} \mathbf{a}^{(L-1)})}{\partial \mathbf{W}^{(L)}} = \mathbf{a}^{(L-1)}$$

BERT Fine Tuning Demo

3. LLMs zur Generierung von Text

- Auf Basis der bestehenden Token wird der nächste Token vorhergesagt
- Der vorhergesagte Token wird wieder in das Neuronale Netz „zurückgespeist“
- Text-Generierung ist Klassifikation: das Neuronale Netz weist den Tokens Wahrscheinlichkeiten zu. Im letzten Layer ist für jeden Token 1 Neuron mit entsprechendem Score.
- Aus Text kann so ein supervised Training erstellt werden

3. Large Language Models - Temperatur

Das Beste an KI ist die Fähigkeit zu

| Wort | Wahrscheinlichkeit |
|-----------------|--------------------|
| lernen | 5.4 % |
| verstehen | 4.1 % |
| prognostizieren | 3.6 % |
| entdecken | 1.1 % |
| ... | ... |

- Die Wahl des wahrscheinlichsten Wortes (Temperatur =0) erzeugt oft schlechte Text
- mit einer hohen Temperatur (> 1) werden auch unwahrscheinliche Worte gewählt
- Damit lässt sich die „Kreativität“ des Modells steuern
- In der Praxis wird oft eine Temperatur von 0.7 gewählt

3. Large Language Models – Context Length

- Context Length (oder auch Window) ist eine Beschränkung der Anzahl an Tokens, die der Transformer verarbeiten kann
- Die Context Length Inkludiert den Input, den Output und gegebenenfalls den vorigen Chatverlauf
- Aktuell GPT-4: 128000 Tokens, Gemini: 2 Millionen Tokens
- Zum Vergleich: die Bibel hat ~1.4 Millionen Token

3. LLMs – ein paar Zahlen

Gewichte im LLM

Llama 3.1: 405 Milliarden

PaLM 2: 340 Milliarden

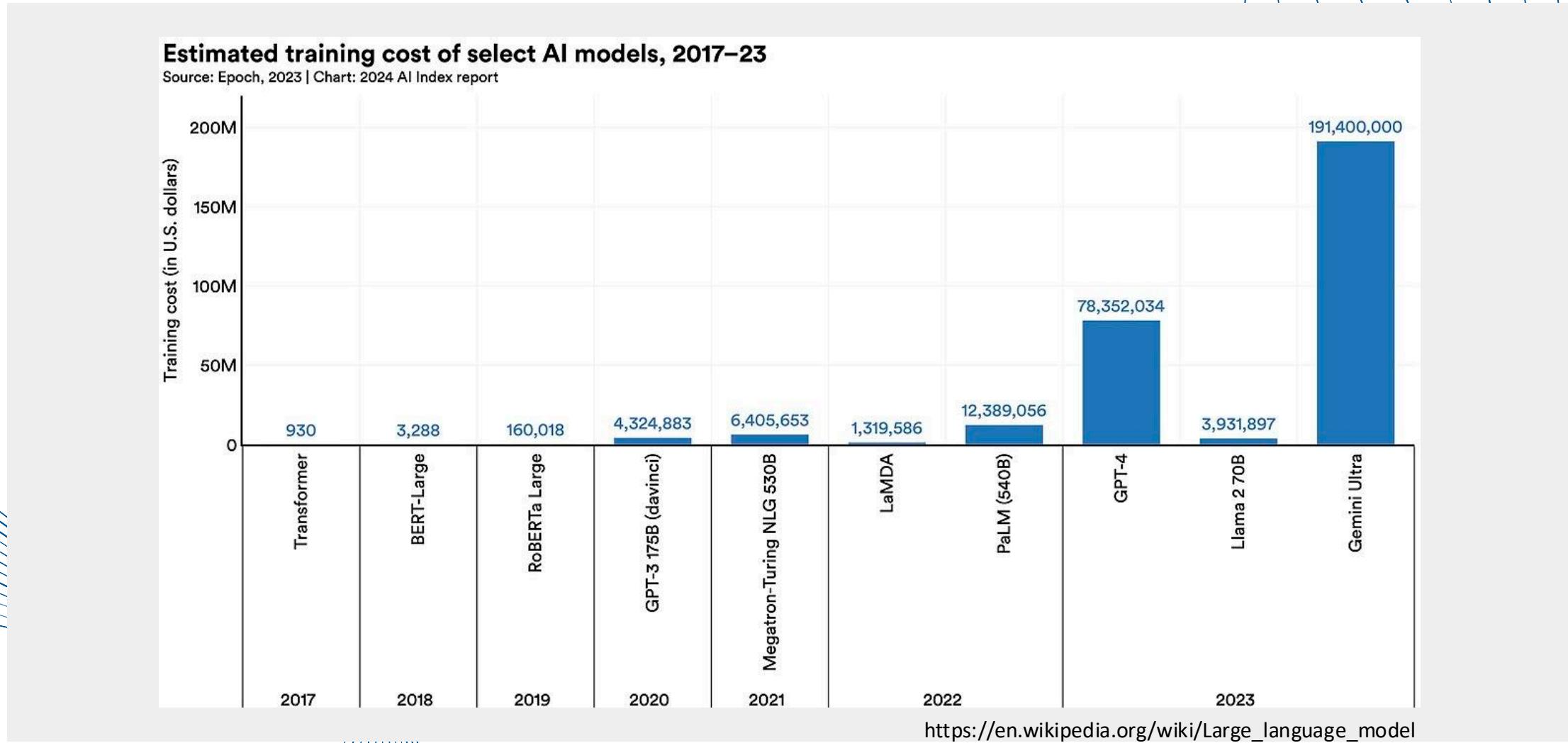
GPT-4: ~1 Billionen (geschätzt)

Zum Vergleich: im Gehirn

100 Milliarden – 1 Billionen
Nervenzellen

> 100 Billionen Synapsen
(Verbindungen)

3. LLMs – Training



3. LLMs – die Hardware

Datacenter xAI

100.000 Nvidia H100

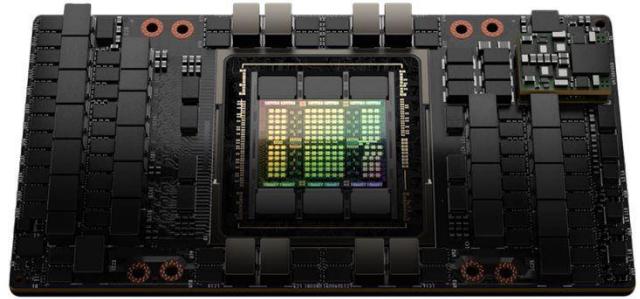
Stromverbrauch pro H100

~1 Haushalt

Kosten pro H100

~40.000 €

Nvidia H100



3. LLMs – Datensatz

Tokens

100 Milliarden – 20
Billionen

Speicherplatz

Einstellig Terabyte

Für GPT-3

| Dataset | Anzahl Tokens | Anteil am Training |
|--------------|----------------|--------------------|
| Common Crawl | 410 Milliarden | 60% |
| WebText2 | 19 Milliarden | 22% |
| Books1 | 12 Milliarden | 8% |
| Books2 | 55 Milliarden | 8% |
| Wikipedia | 3 Milliarden | 3% |

Deep Dive Podcast

[https://www.dropbox.com/scl/fi/z0aogv7dpkc5hap8ngq5i/
Deep-Dive-Tag-
1.mp3?rlkey=br2uv7c27yv9c5cvyhvjfs36u&st=75o8fizi&dl=0](https://www.dropbox.com/scl/fi/z0aogv7dpkc5hap8ngq5i/Deep-Dive-Tag-1.mp3?rlkey=br2uv7c27yv9c5cvyhvjfs36u&st=75o8fizi&dl=0)