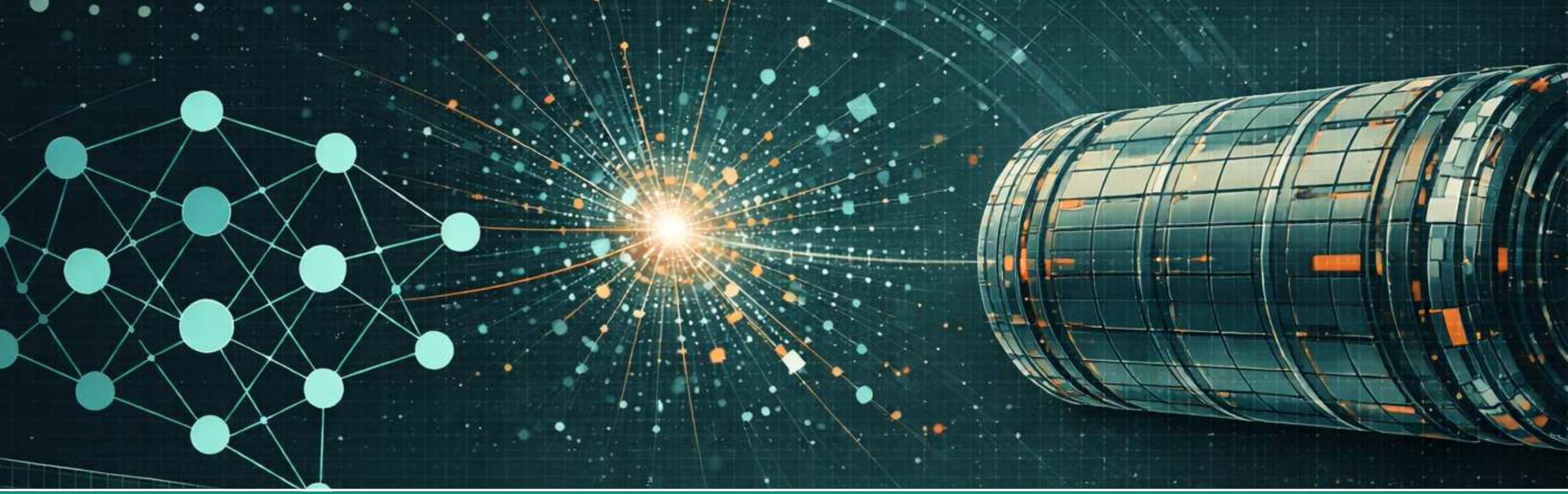
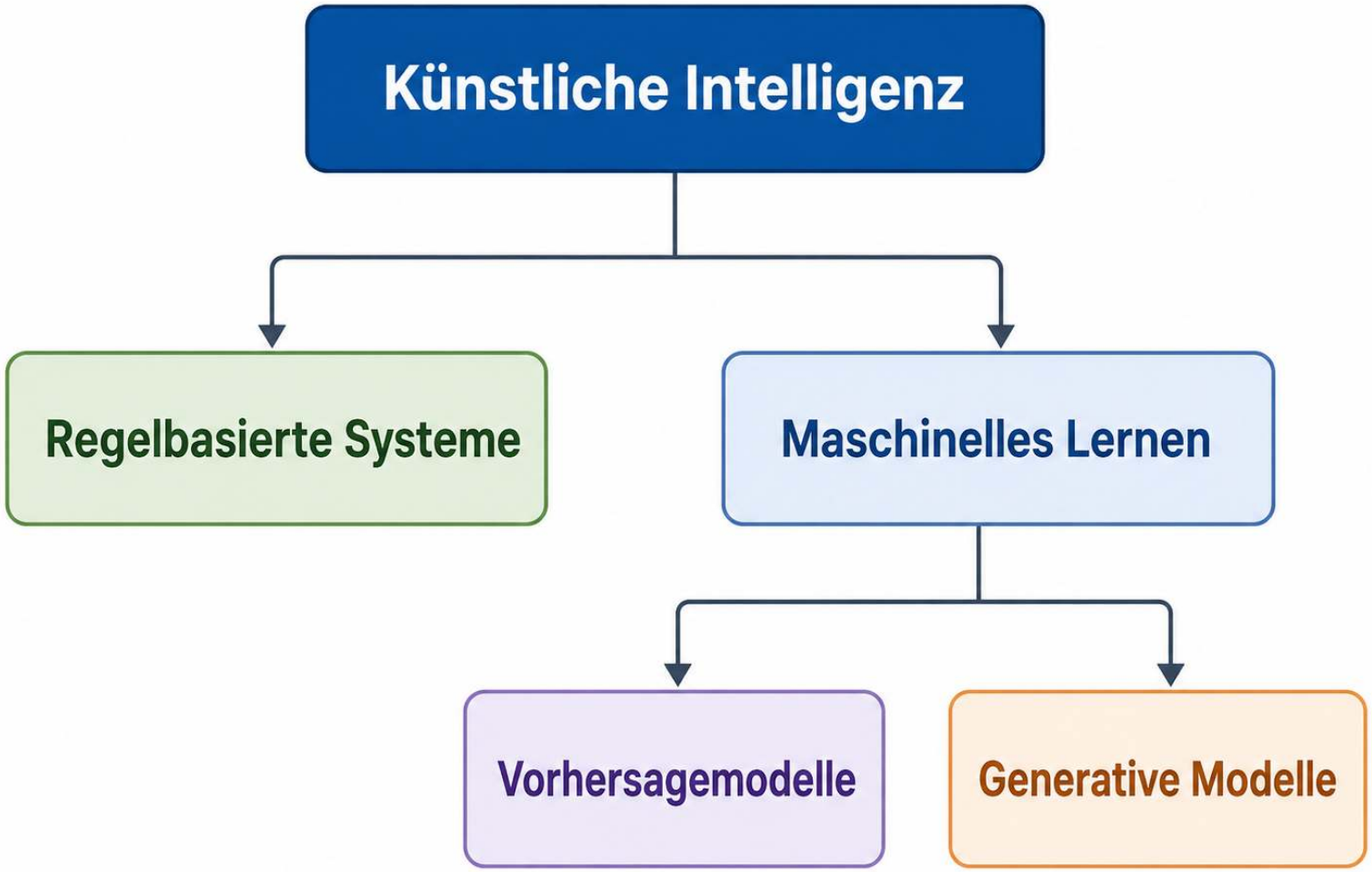


Künstliche Intelligenz
in der (Teilchen-) Physik

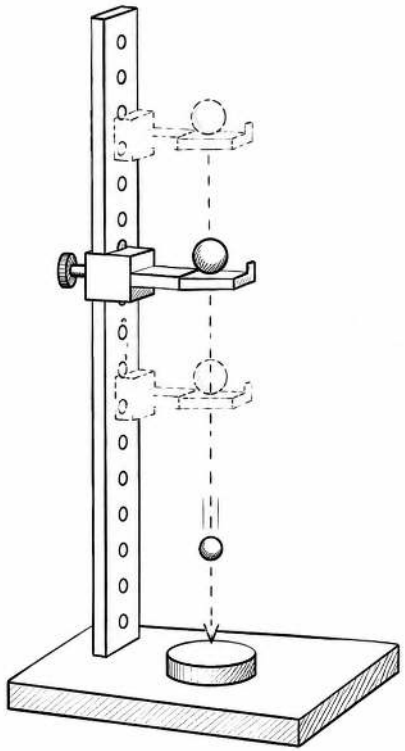
10.06.26

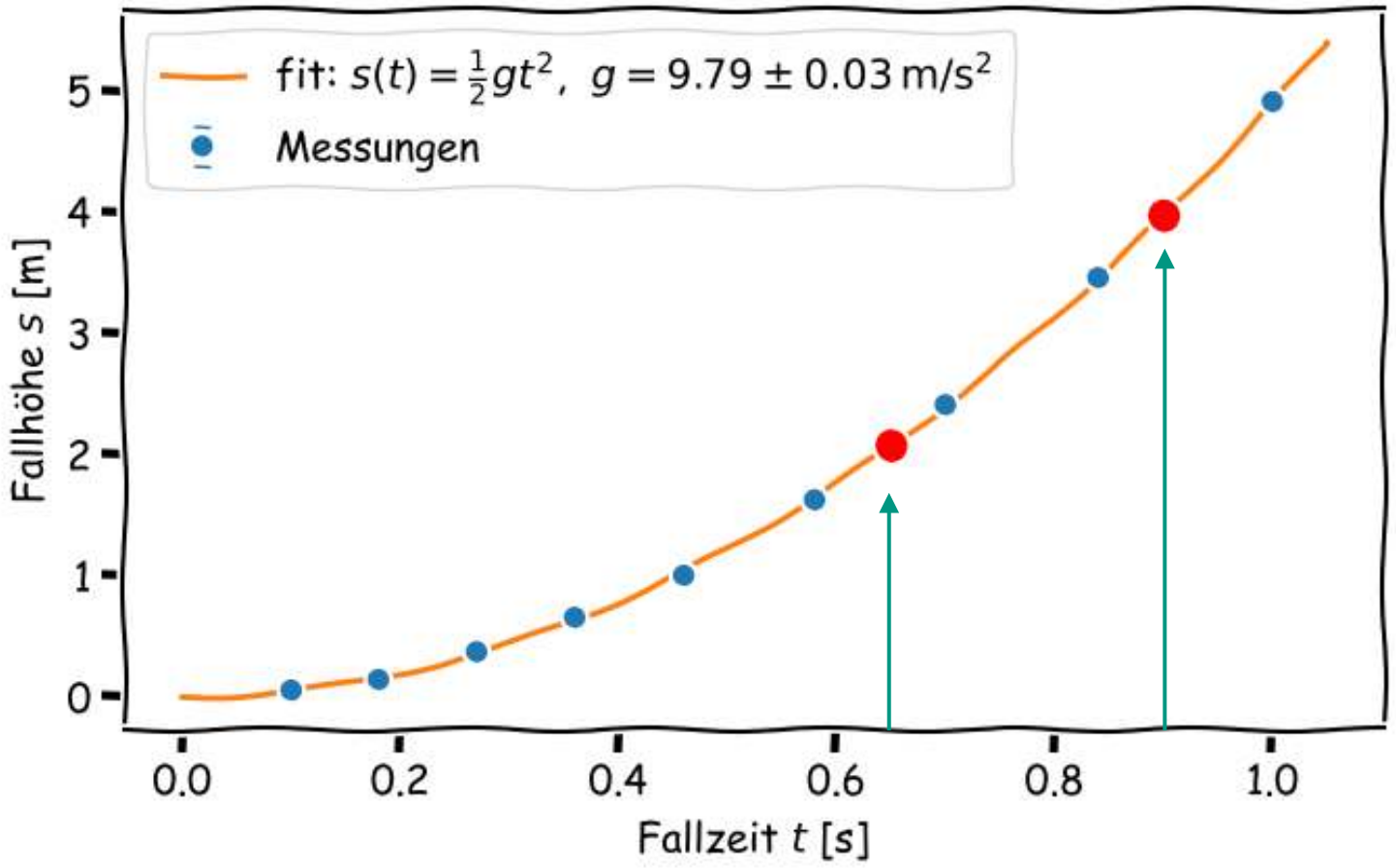
Prof. Dr. Jan Kieseler

KI ist nicht nur Chat-Bots



Modelle und Fits

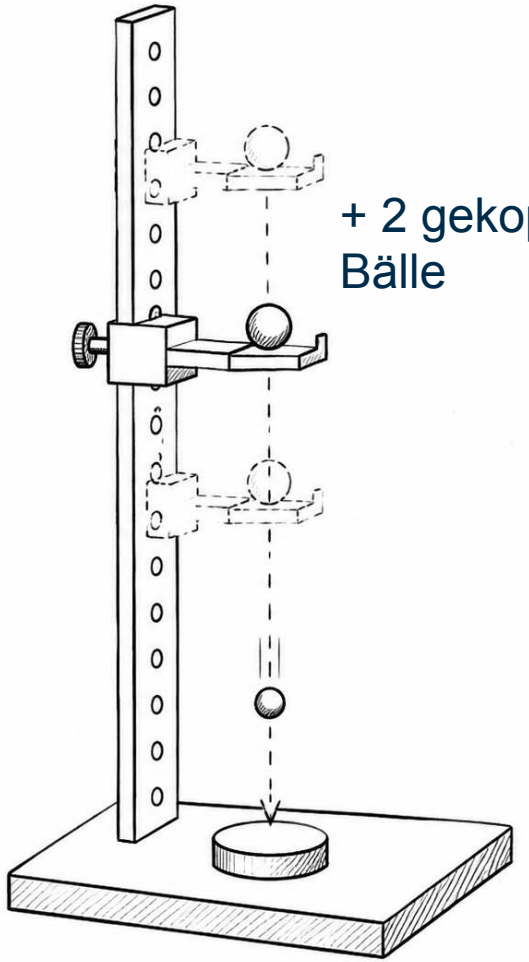




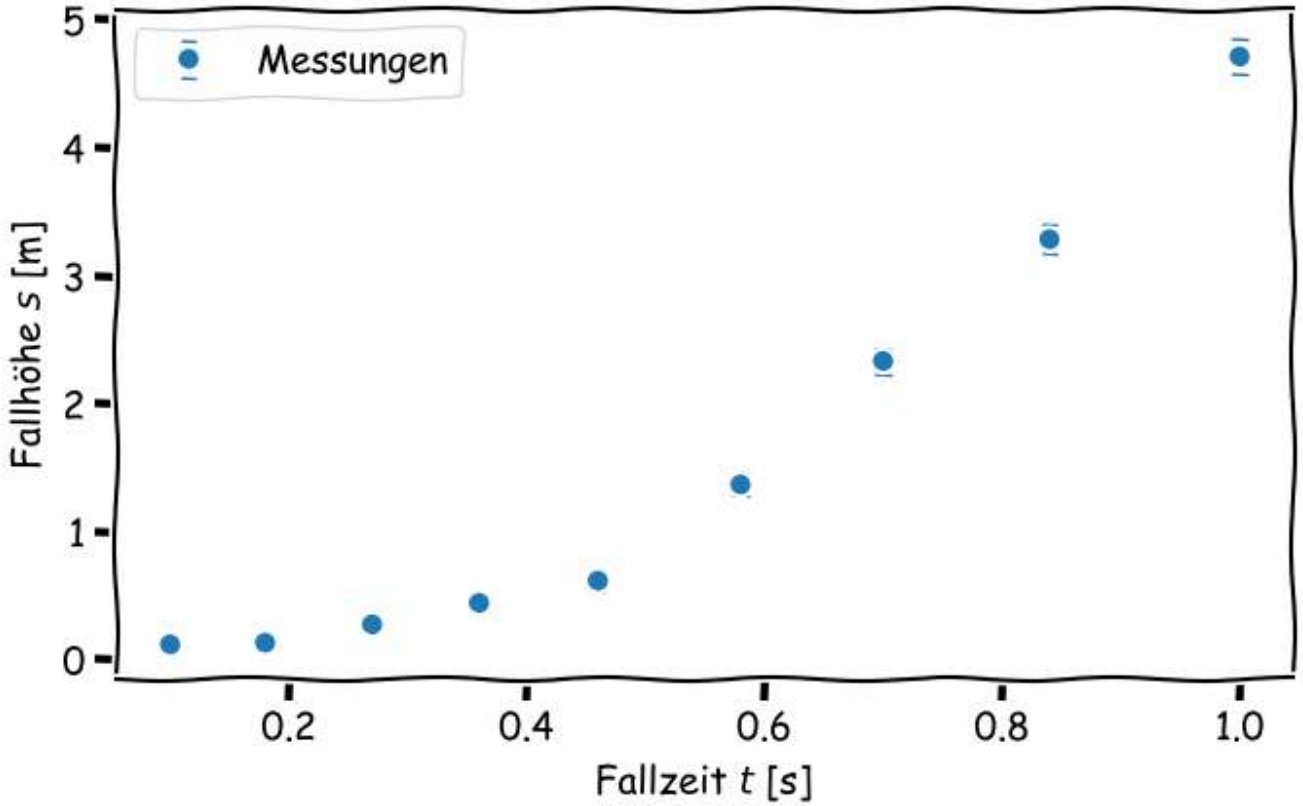
1) Bestimmen der Modellparameter: g

2) Nutzung des Modells für Vorhersagen

Komplexere Modelle



+ 2 gekoppelte Bälle

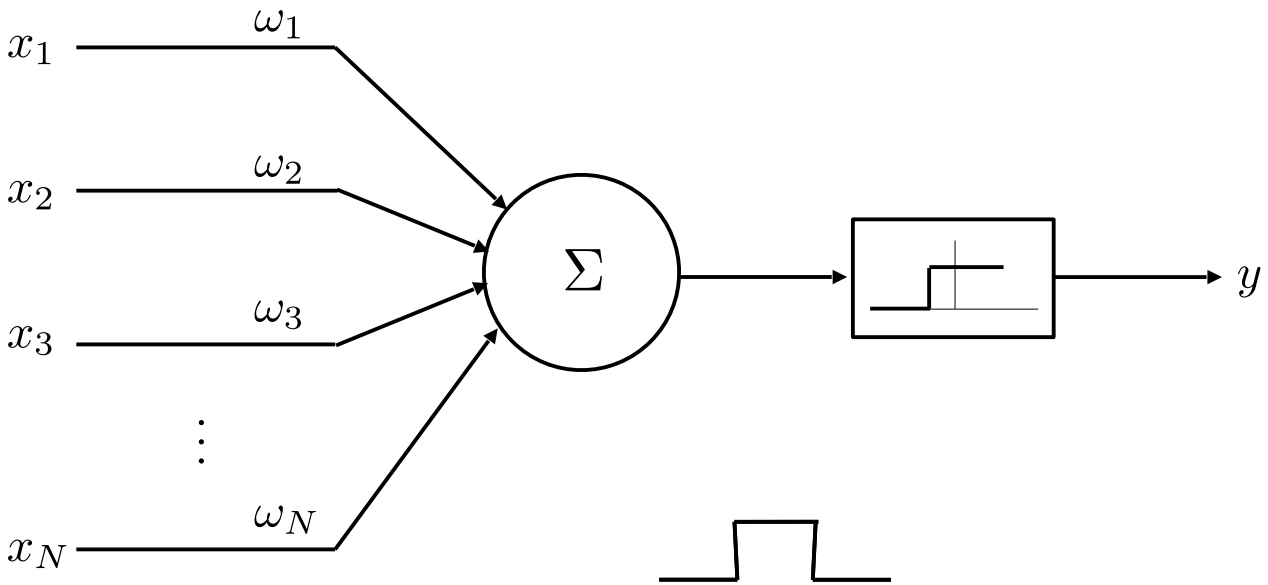


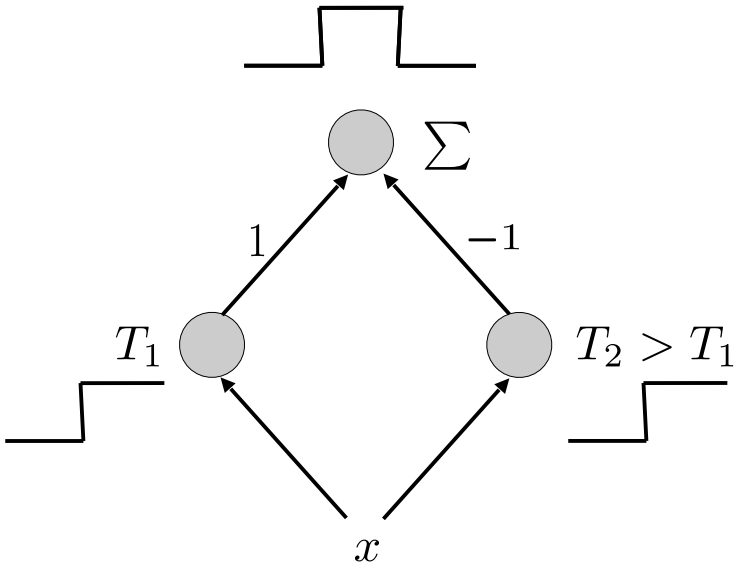
$$s(t) = \ldots$$

Eine genügend komplexe Funktion, mit genügend vielen freien Parametern kann die Abhängigkeit approximieren.

Welche Funktion nutzen wir hier?

Das Perceptron



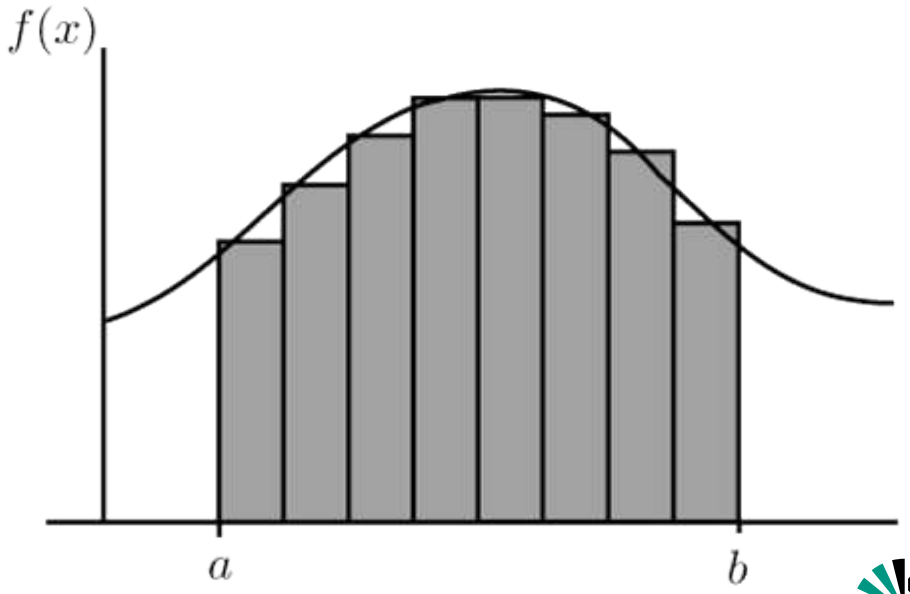


Eingangswerte werden gewichtet addiert

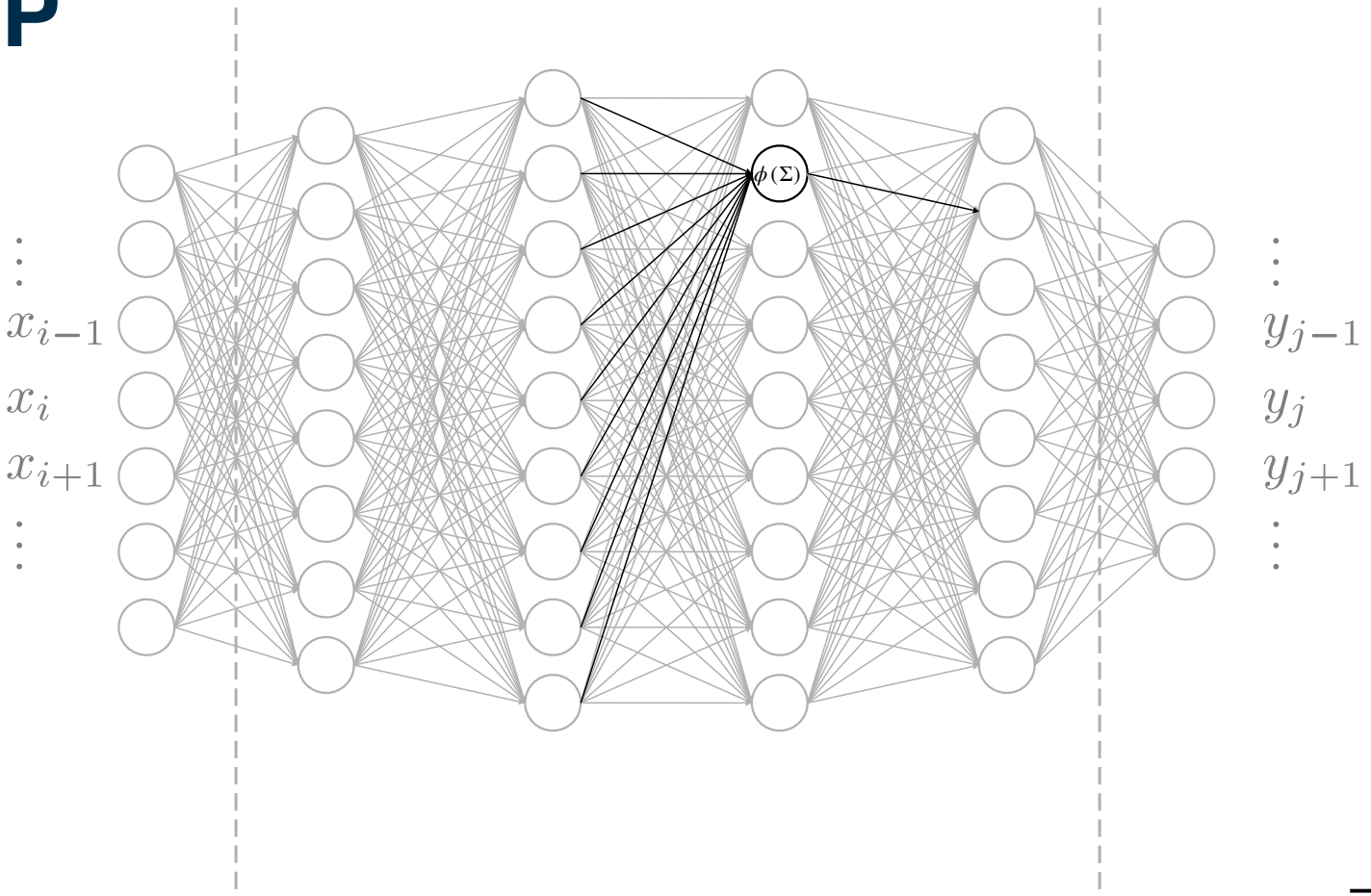
Der Ausgang “schaltet”, wenn eine Schwelle überschritten wird

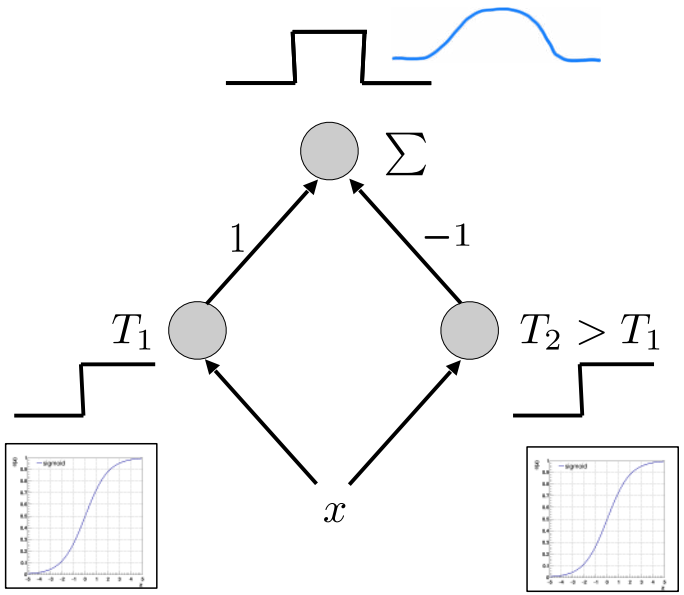
Freie Parameter:

- Gewichte ω
- Schwelle b



Das MLP

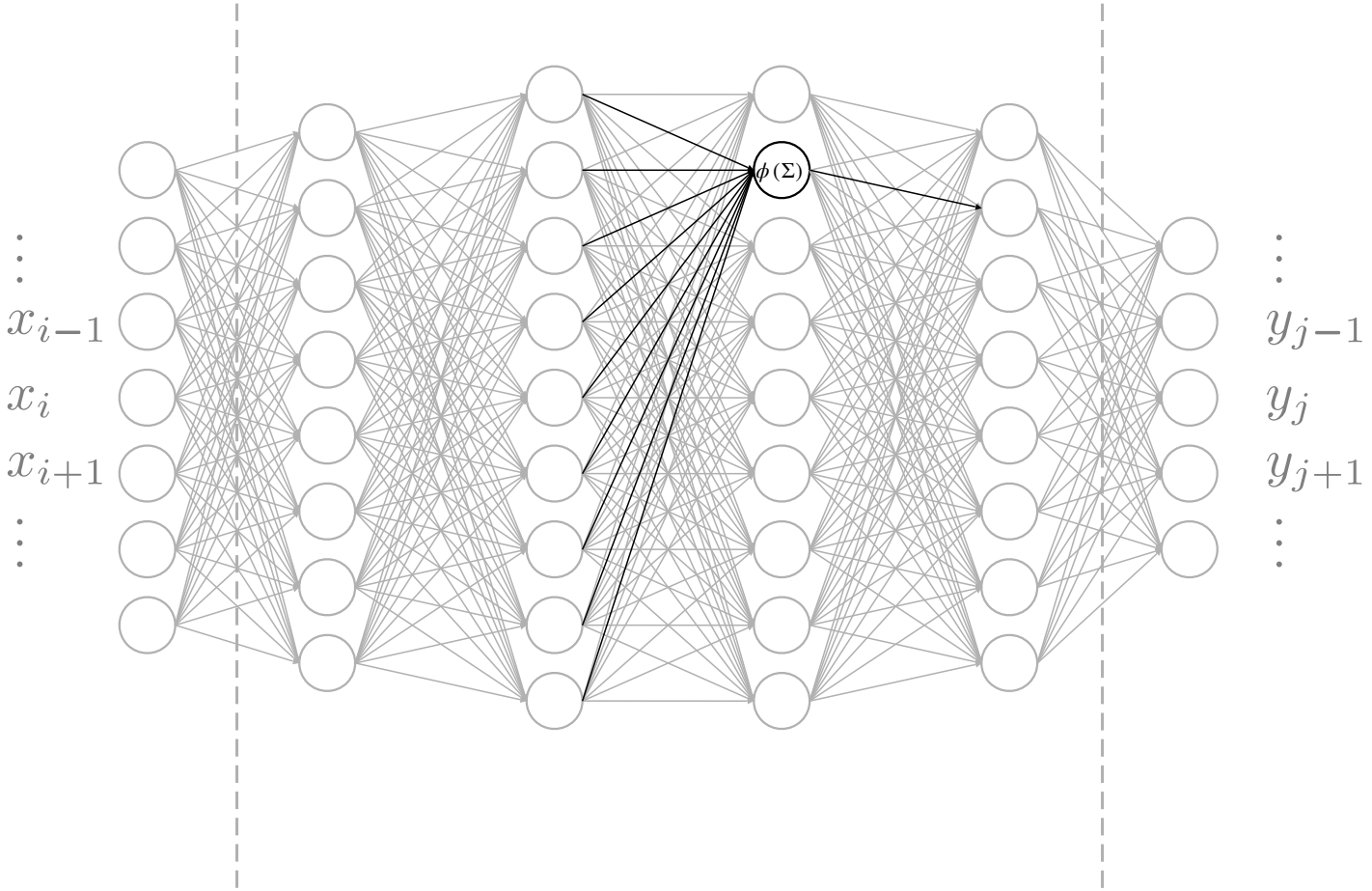


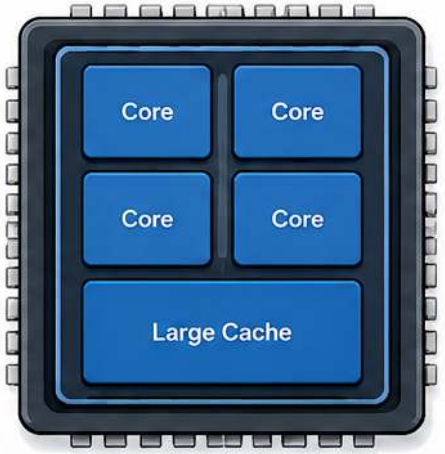


Durch Zusammenschalten von vielen Perceptrons zu einem Multi-Layer-Perceptron können auch komplexe Funktionen approximiert werden.

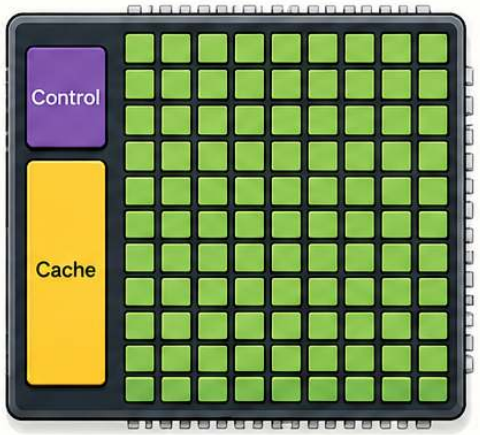
Oft wird dabei eine weichere Schwelle - eine Aktivierungsfunktion - verwendet

Die Hardware: GPUs





CPU
wenige komplexe Operationen



GPU
Viele ähnliche & einfache Operationen

Die Verfügbarkeit starker GPUs hat wesentlich dazu beigetragen, große neuronale Netze praktisch nutzbar zu machen.

Warum wir die Aktivierungsfunktion brauchen

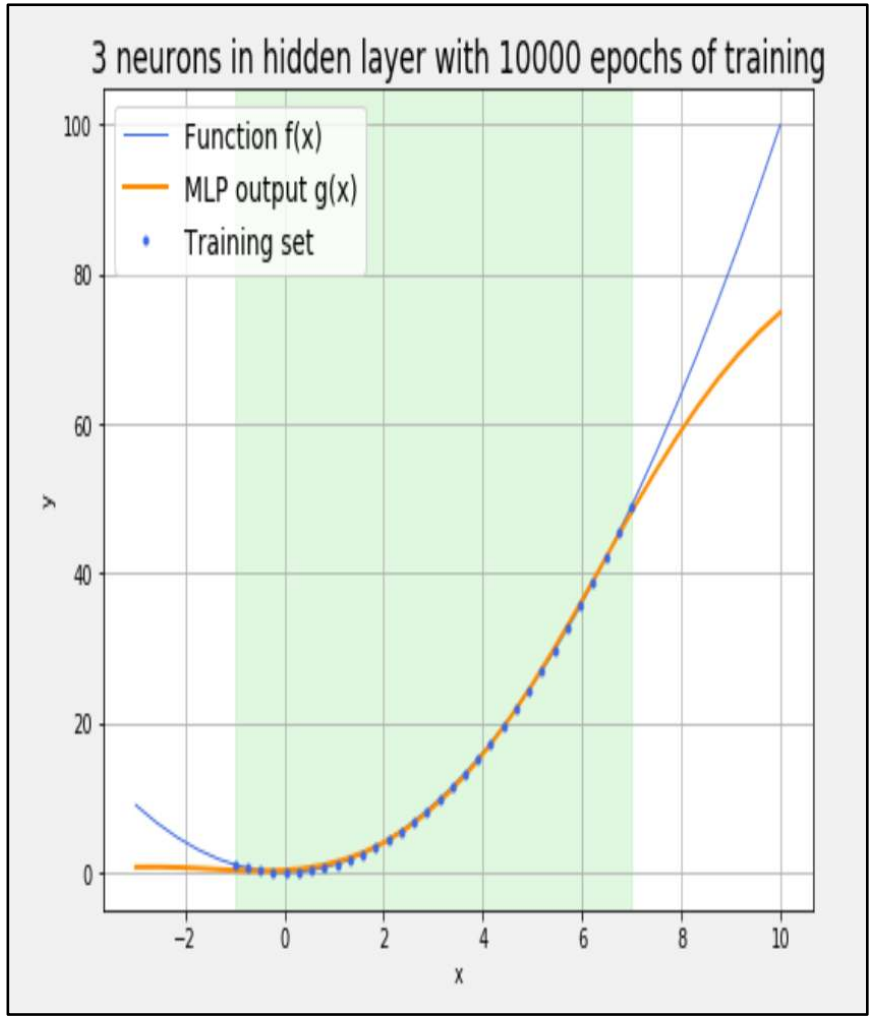
Ohne Aktivierungsfunktion

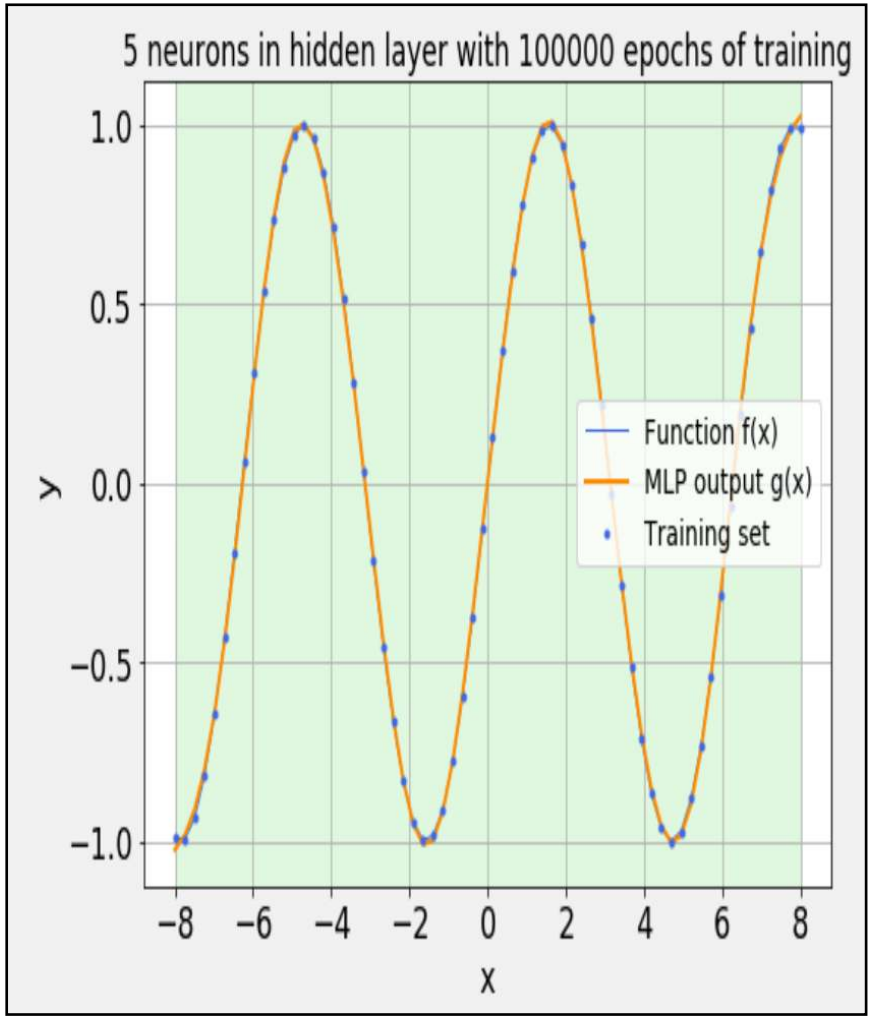
$$h = W_1 x + b_1$$
$$\hat{y} = W_2 h + b_2$$
$$\hat{y} = W_2(W_1 x + b_1) + b_2$$
$$\hat{y} = (W_2 W_1)x + (W_2 b_1 + b_2)$$
$$W_2(W_1 x + b_1) + b_2 = W' x + b'$$

Maybe I remove this one; it is intuitive already from the previous slide

$$h = \sigma(W_1 x + b_1)$$
$$\hat{y} = W_2 h + b_2$$
$$\hat{y} = W_2 \sigma(W_1 x + b_1) + b_2$$

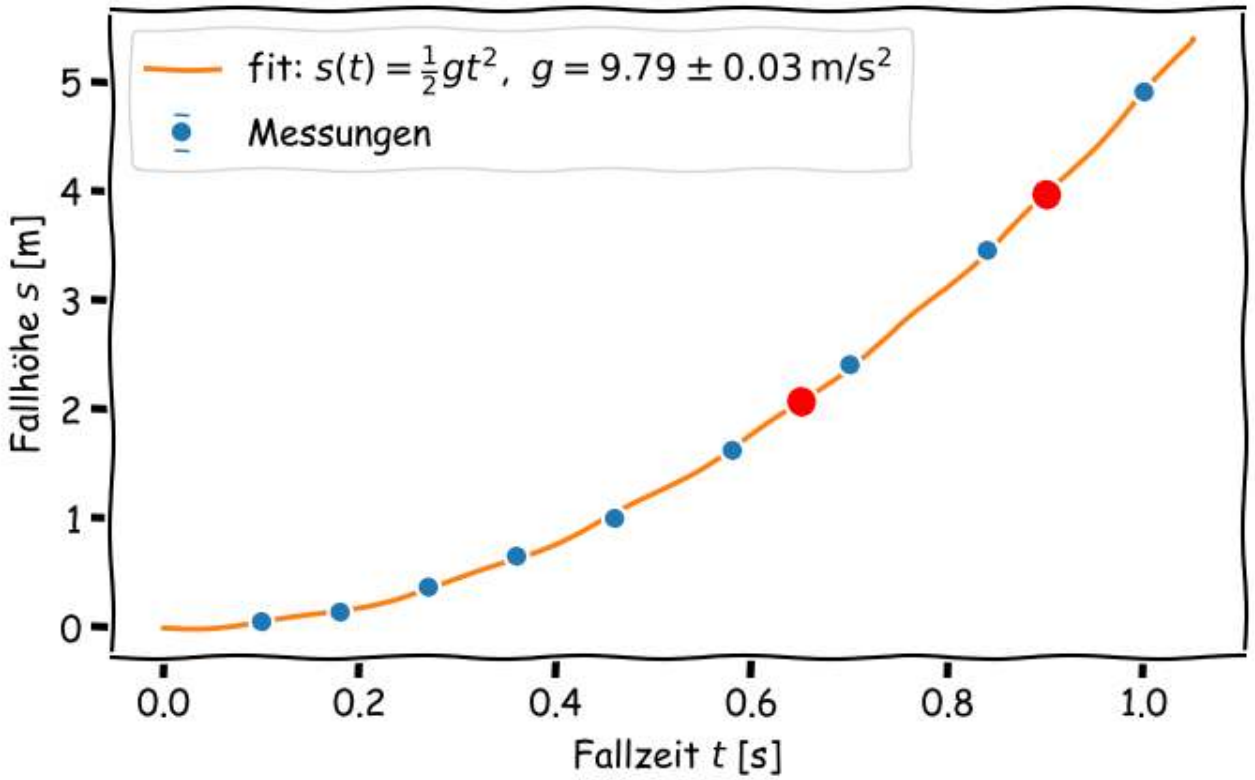
Das MLP als universaler Funktions-Approximator





https://notebook.community/kit-cel/lecture-examples/mloc/ch3_Deep_Learning/pytorch/function_approximation_with_MLP

Parameter bestimmen: ein hochdimensionaler Fit



Neuronale Netze haben bis zu Milliarden Parametern

Nutze **Trainingsdaten** um diese zu bestimmen

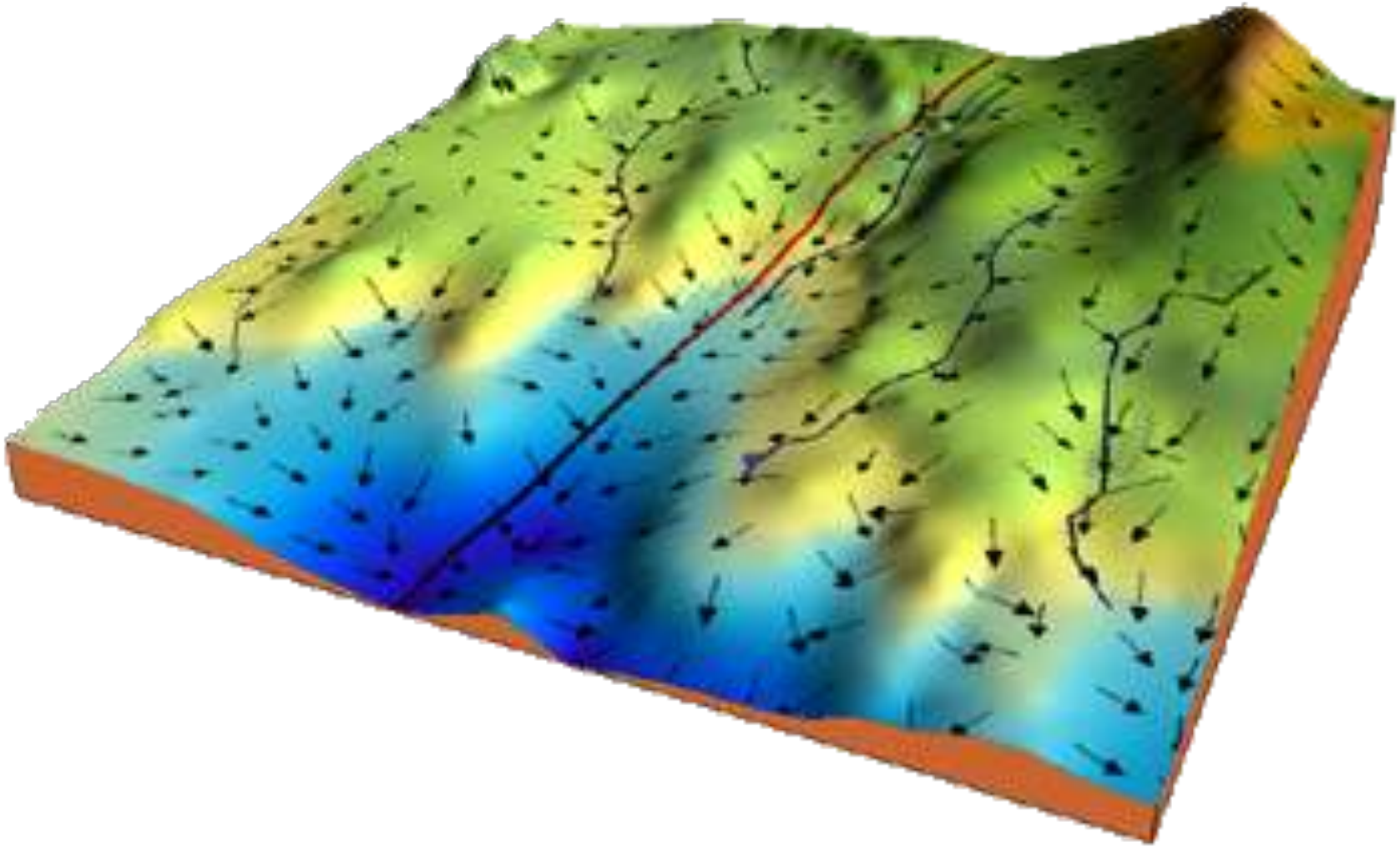
Gradient Descent

$$\hat{y}_i = f_\omega(x_i)$$

$$L(\omega) = \frac{1}{N}\sum_{i=1}^{N}\left(f_\omega(x_i) - y_i\right)^2$$

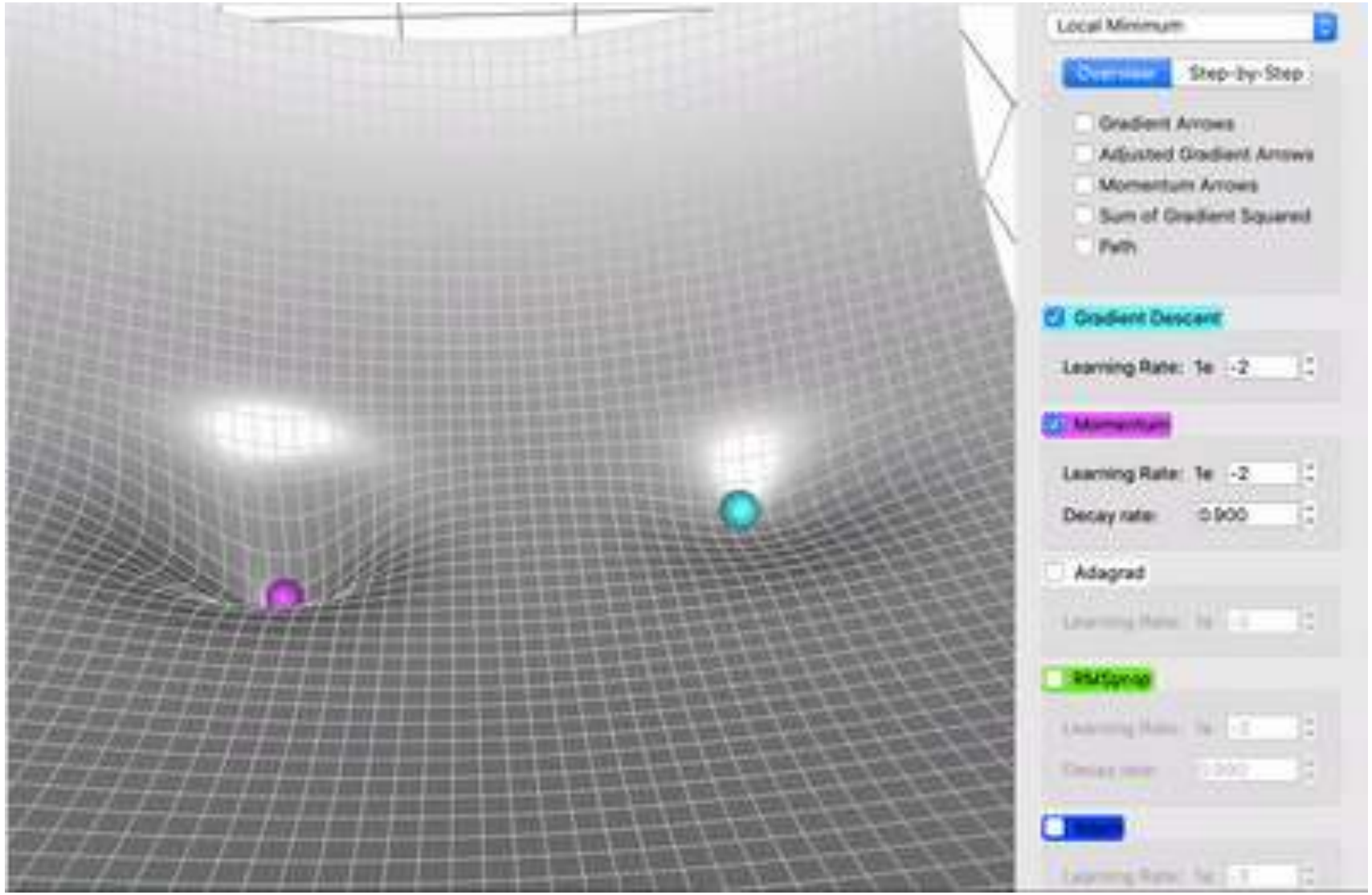
$$\omega \leftarrow \omega - \eta \nabla_\omega L(\omega)$$

$$\nabla_\omega L(\omega) = \left(\frac{\partial L}{\partial \omega_1}, \frac{\partial L}{\partial \omega_2}, \ldots, \frac{\partial L}{\partial \omega_M}\right)$$



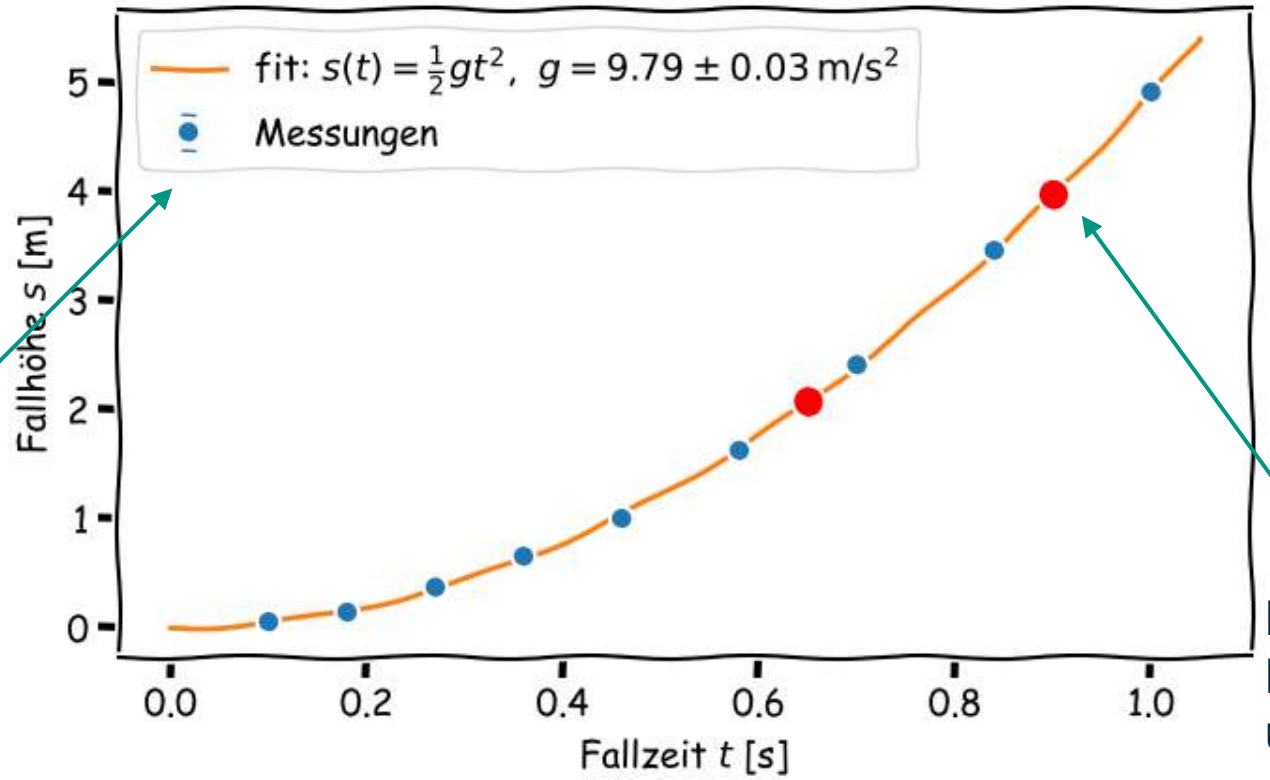
Wir ändern alle Parameter in die Richtung, in der der Fehler kleiner wird.

Momentum: nicht nur Steigung, sondern auch Trägheit



https://towardsdatascience.com/a-visual-explanation-of-gradient-descent-methods-momentum-adagrad-rmsprop-adam-f898b102325c

Training ist nicht Anwendung



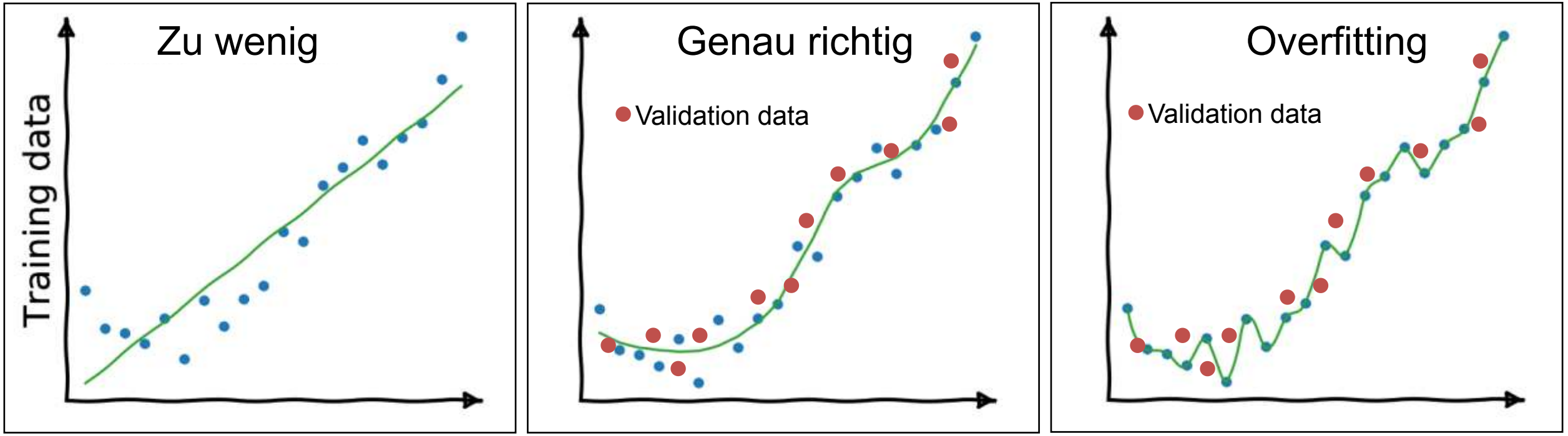
x und y bekannt
bestimmen Parameter

Trainings-Daten

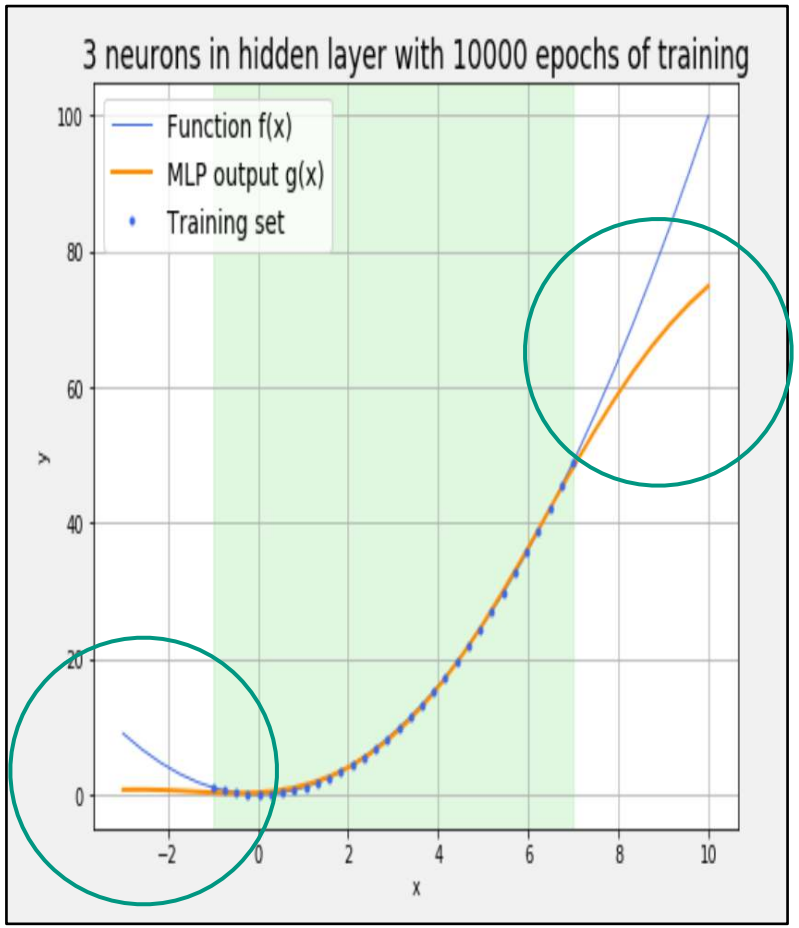
Nur x bekannt:
Nutzen trainiertes Modell
um Verhalten vorauszusagen

Test-Daten

Warum und wo KI manchmal scheitert (1)



Warum und wo KI manchmal scheitert (2)

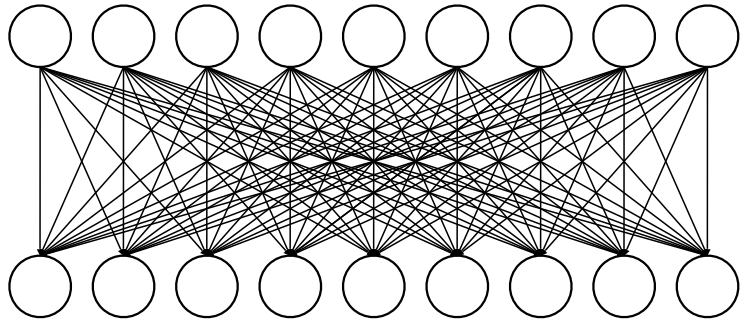


Neuronale Netze interpolieren sehr präzise: sie werden aber unvorhersehbar außerhalb ihres Trainings-Regimes

Netzwerkarchitektur kann Extrapolation in der Rohdarstellung zu Interpolation in einer sinnvolleren Darstellung machen.

Nicht Sätze auswendig lernen, sondern Beziehungen nutzen

Der Apfel fällt vom Baum



Der Apfel fällt vom Baum

Apfel <-> fällt
fällt <-> *Baum*
Der <-> *Apfel*
vom <-> *Baum*

Der Ball fällt vom Tisch.

Ball <-> fällt
fällt <-> *Tisch*
Der <-> *Ball*
vom <-> *Tisch*

Attention Is All You Need

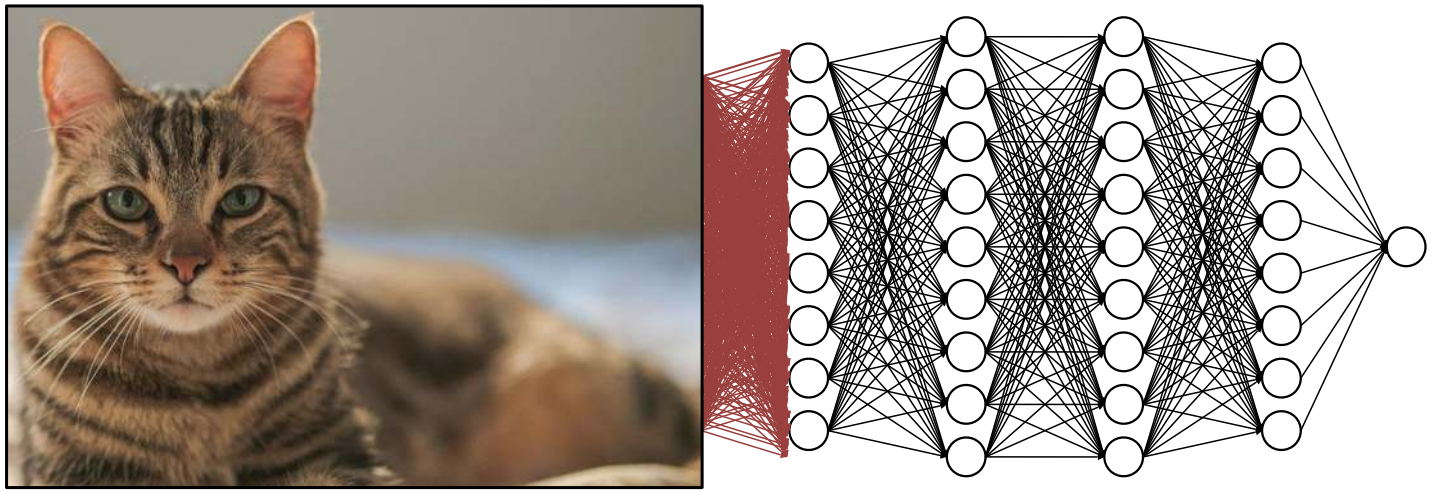
Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

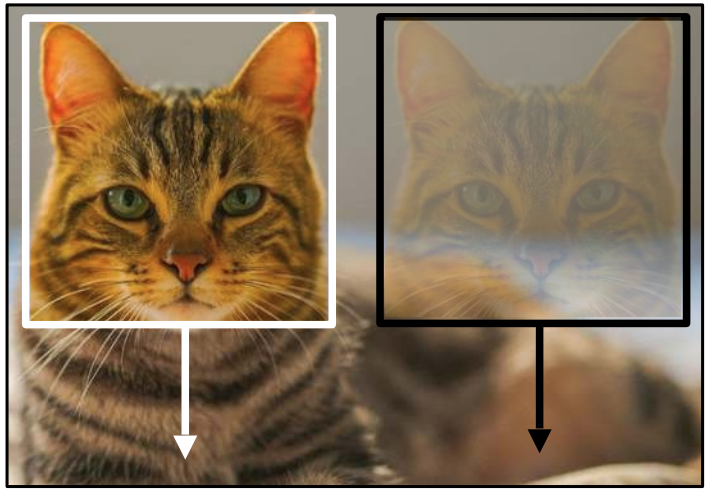
Nicht mehr nur feste Gewichte zwischen Eingaben, sondern ein lernbarer Attention-Mechanismus

Attention: Welche anderen Wörter sind für dieses Wort gerade wichtig?
Mathematisch: das *Gewicht* ist eine lernbare Funktion von ***beiden*** Worten

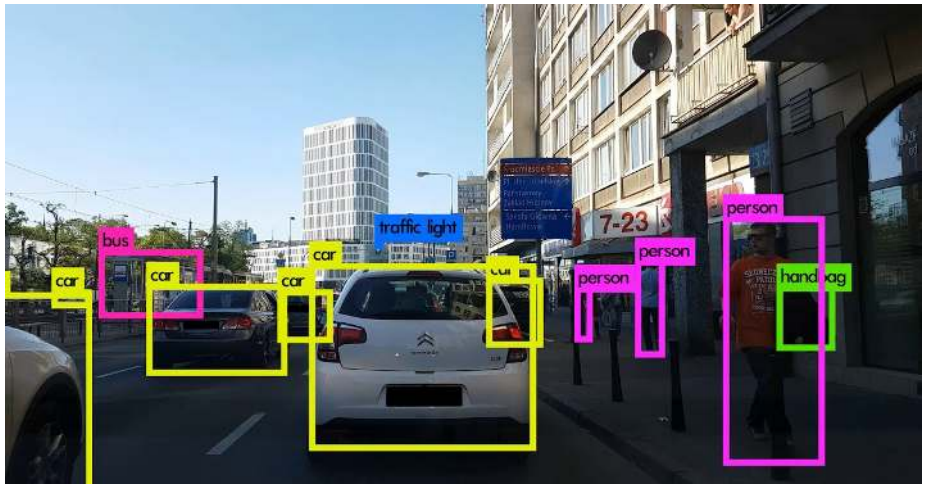
Gute ML-Architekturen bauen passende Strukturannahmen in das Modell ein.

Lokale Muster erkennen statt ganze Bilder interpretieren



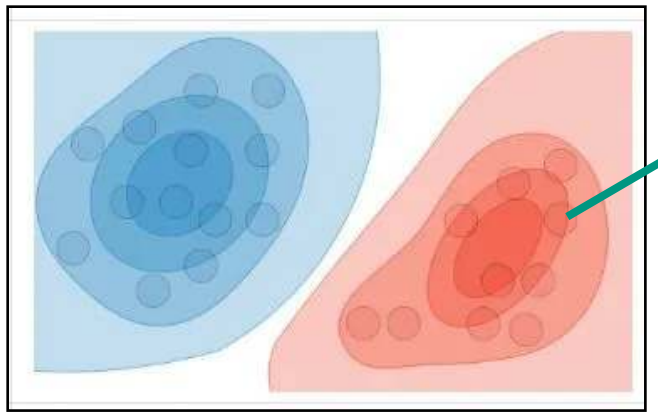


Katze Nicht Katze

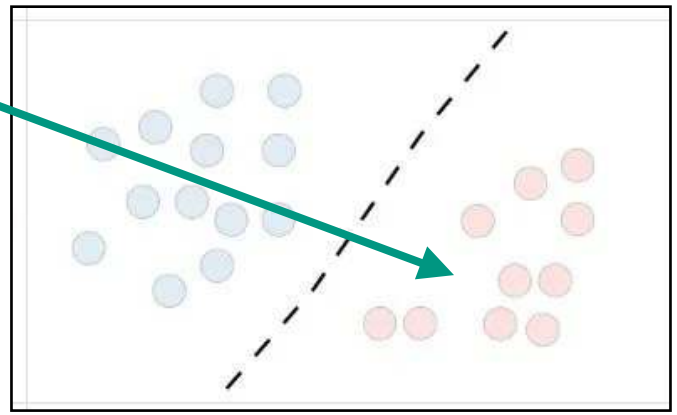


Generative KI vs Vorhersagende KI

“Erzeuge ein Bild einer Katze”



Ist das eine Katze?



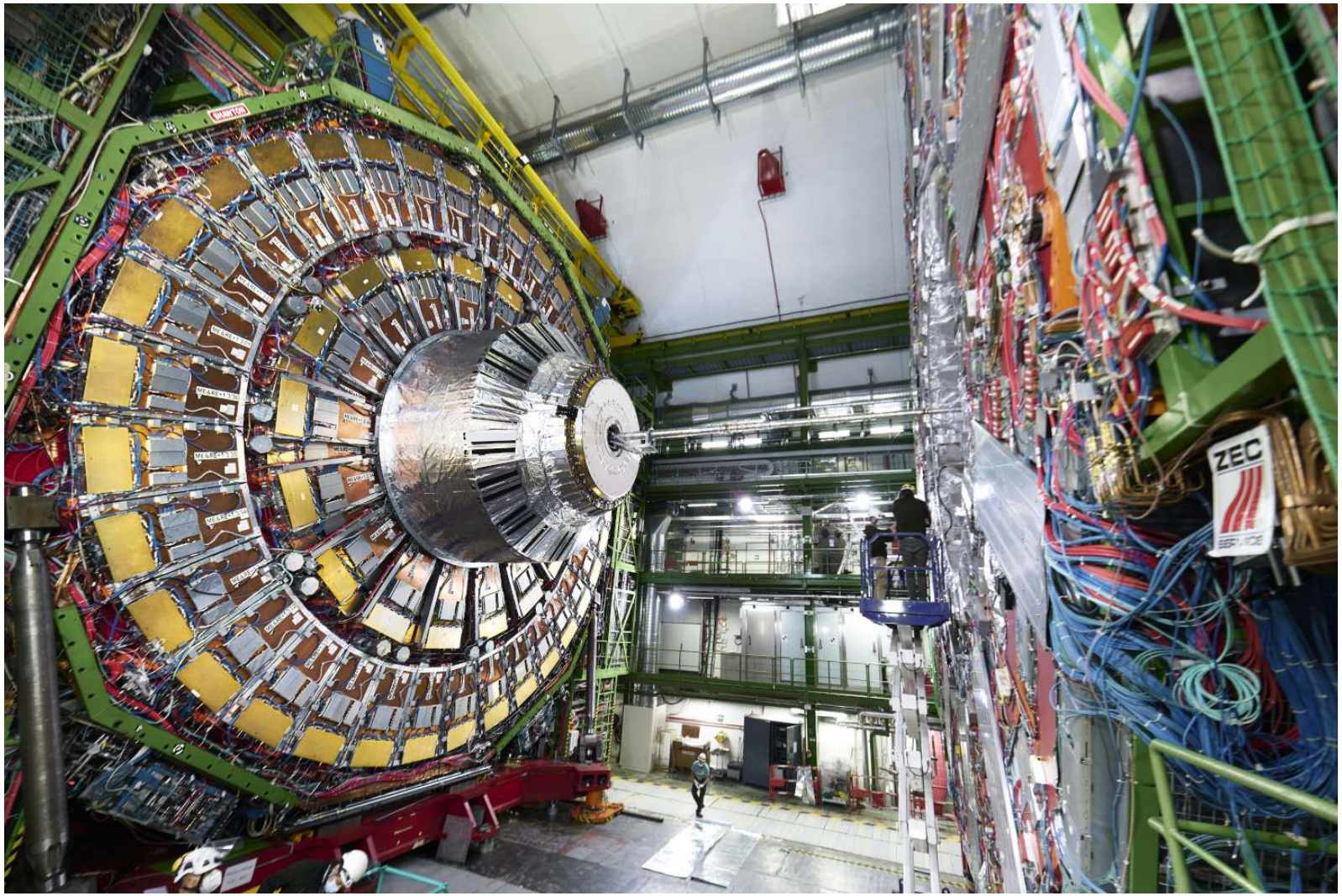
Die Eingabe ist unterspezifiziert:
Viele Bilder wären eine gültige Antwort.

Das Modell erzeugt ein Beispiel aus einer gelernten Wahrscheinlichkeitsverteilung.

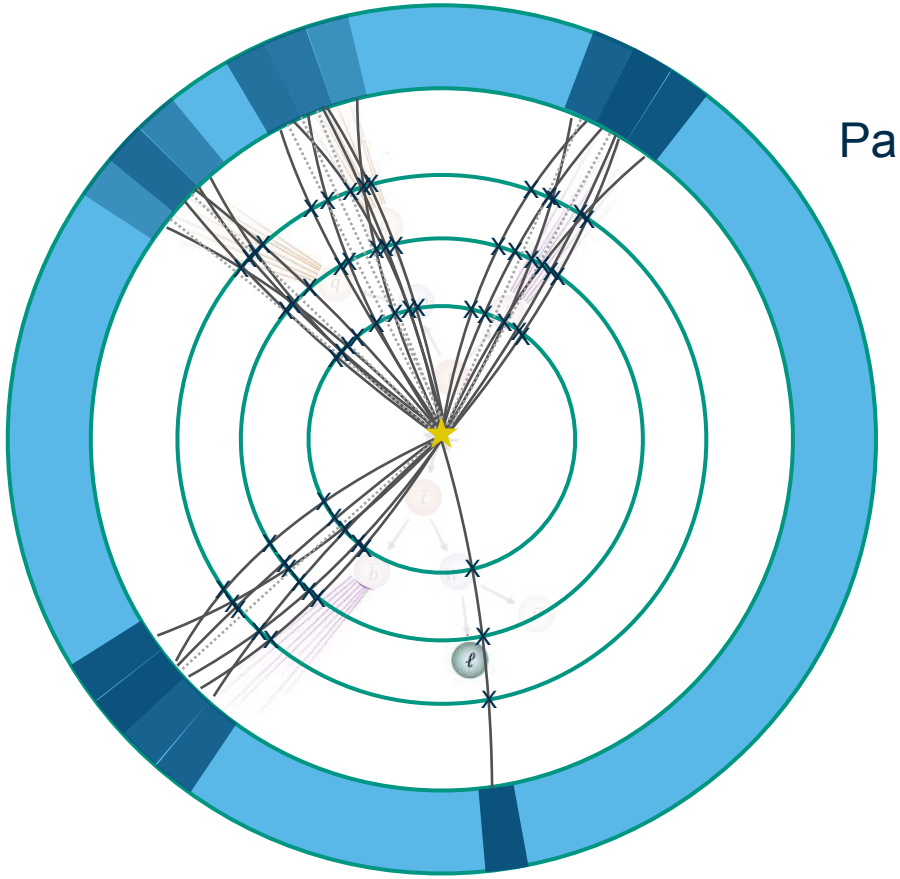
Die Eingabe ist gegeben;
das Modell entscheidet über eine Eigenschaft.

Das Modell schätzt eine Antwort, z.B. die Wahrscheinlichkeit für “Katze”.

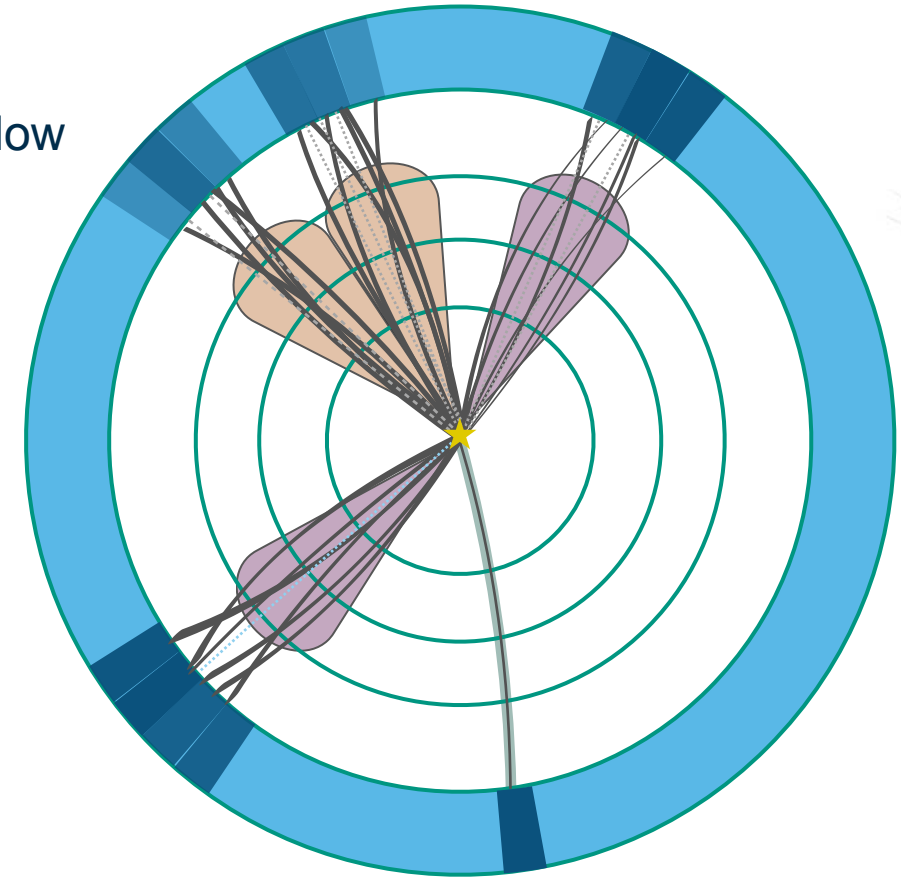
Die Verbindung zur Teilchenphysik

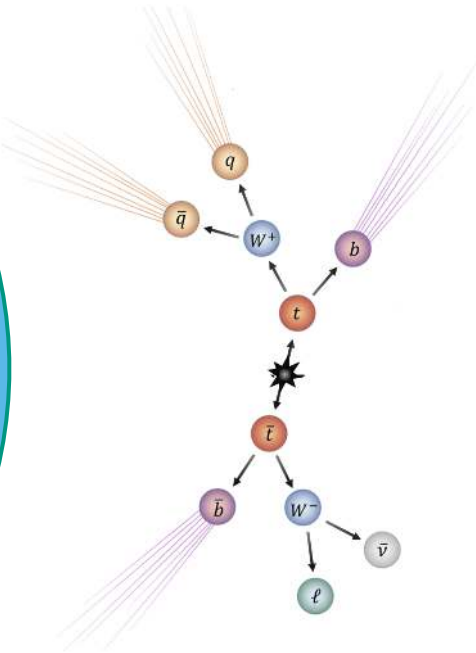


Datenanalyse: eine Reihe von Mustererkennungen

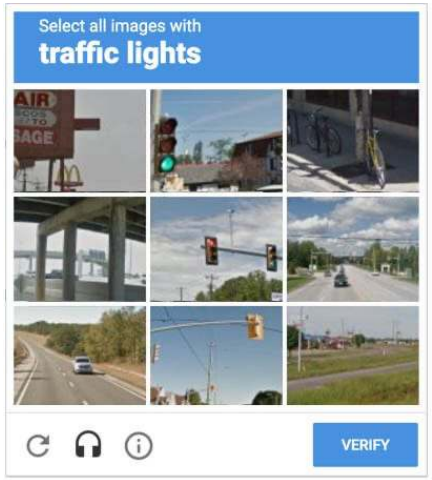


Particle Flow



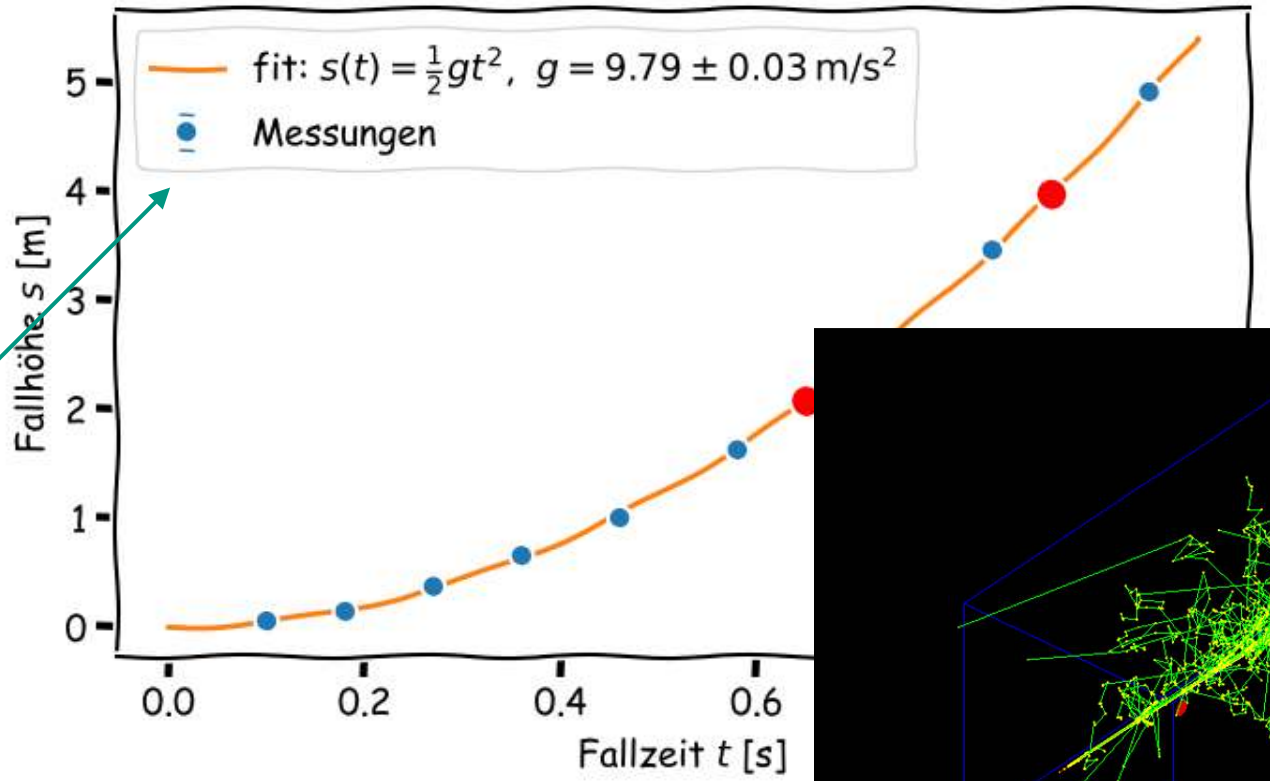


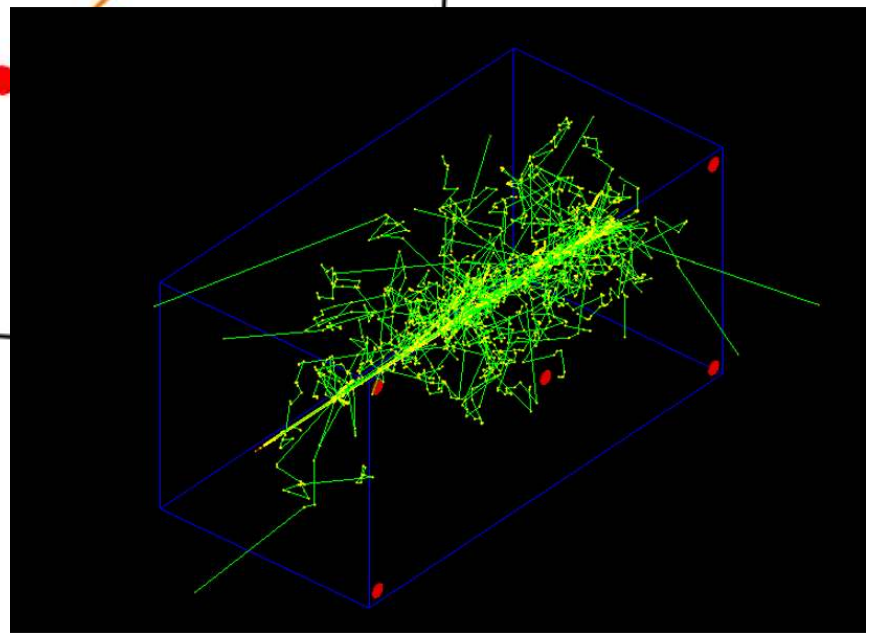
Physikalische Modelle machen KI in der Physik besonders attraktiv



x und y bekannt
bestimmen Parameter

Trainings-Daten

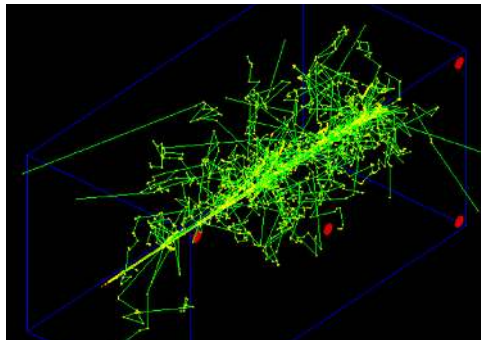


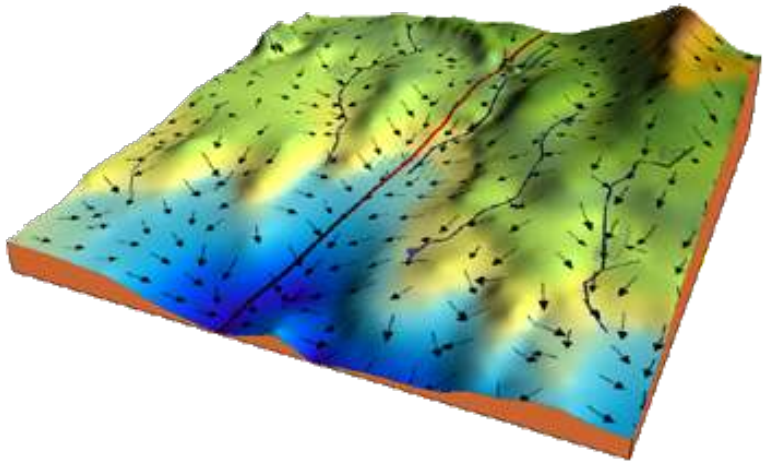


Wir können mit hoher Präzision *vorwärts* simulieren und Trainingsdaten erzeugen

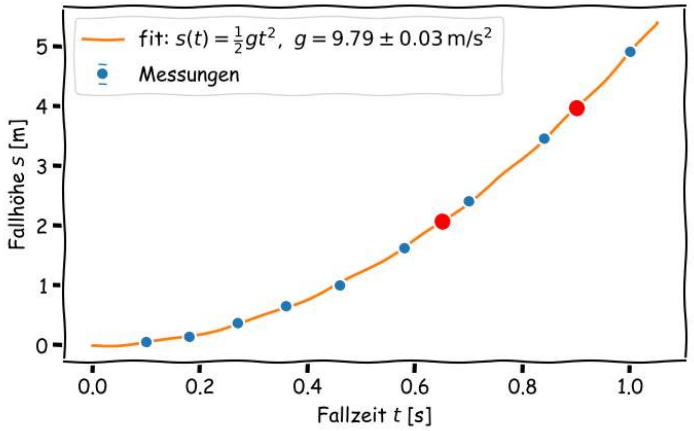
KI als Detektor-Designer

2042
2041 **Start detector installation**
2040
2039
2038
2037
2036 **Detector component production**
2035 **Four detector TDRs completed**
2034
2033
2032 **Detector CDRs (>4) submitted to FC³**
2031
2030 **End of HL-LHC upgrade: more detector experts available**
2029
2028 **FC³ formation, call for CDRs, collaboration forming**
2027
2026 **European Strategy Update: FCC Recommendation**
2025 **Detector EoI submission by the community**



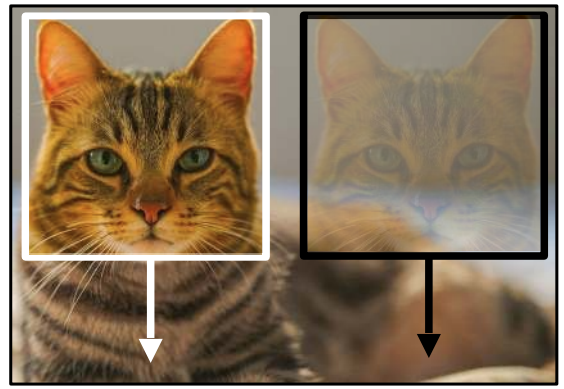


Zusammenfassung

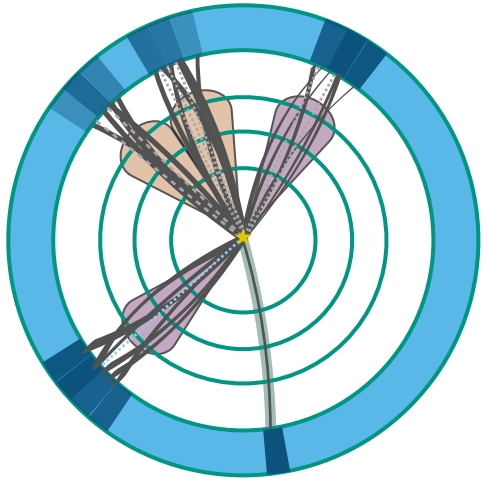


KI ist nicht gleich Chatbot:
KI umfasst viele Ansätze, von regelbasierten Systemen bis zu maschinellem Lernen

Maschinelles Lernen ist im Kern ein sehr flexibler Fit:
Statt eine einfache Formel vorzugeben werden viele freie Parameter aus Trainingsdaten bestimmt, um komplexe Abhängigkeiten zu approximieren.



Neuronale Netze lernen durch Optimierung:
Training bedeutet, Parameter so zu verändern, dass ein Fehlermaß kleiner wird.



KI braucht Struktur und Validierung:
Gute Architekturen nutzen die Struktur der Daten, aber Modelle können überfitten, außerhalb ihres Trainingsbereichs unzuverlässig werden, und Training ist nicht dasselbe wie Anwendung.

Teilchenphysik ist ein natürlicher KI-Anwendungsfall:
Viele Aufgaben bestehen aus Mustererkennung in komplexen Detektordaten; gleichzeitig können wir durch präzise Simulationen große Trainingsdatensätze erzeugen und ML-Modelle für Rekonstruktion, Analyse und perspektivisch auch Detektoroptimierung nutzen.