

# Suchalgorithmen im Machine Learning

Optimierungsverfahren für Ingenieure

---

Jan Hoegen

1. Dezember 2025

Hochschule Karlsruhe  
University of Applied Sciences

1. Problemstellung
2. Lösung mit naiven Methoden
3. Lösung mit Bayesian Search
4. Fazit

# Problemstellung

---

# Ein einfaches Beispiel

Blackbox  $b$ :

- Eingabe: Bild
- Ausgabe: Klasse (Hund, Katze, Auto, ...)
- Ergebnis jedoch nicht immer korrekt

$$b(\text{Bild}) \rightarrow \text{Klasse}$$

Training der Blackbox:

- Anpassen der Parameter von  $b$ , wenn die falsche Klasse vorhergesagt wurde
- Größe: ca. 500 000 Parameter
- Datensatz: 60 000 Bilder

Zielfunktion  $f \rightarrow$  Accuracy:

Prozess: Trainiert die Blackbox  $b$   
über alle Bilder

Ausgabe: Genauigkeit von  $b$  nach  
dem Training

# Genauigkeit der Blackbox

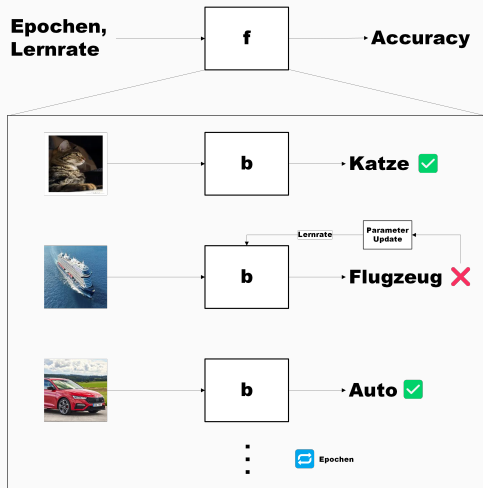


Abbildung 1: Darstellung der Zielfunktion.

$f(\text{Epochen, Lernrate}) \rightarrow \text{Accuracy}$ :

Prozess: Trainiert die Blackbox  $b$  über alle Bilder

Ausgabe: Genauigkeit von  $b$  nach dem Training

Eingaben:

- Epochen: Wie häufig wird der gesamte Datensatz durchlaufen
- Lernrate: Stärke der Parameteranpassung

Ziel:

- Höchste Genauigkeit für den Datensatz

Problem:

- Berechnung von  $f$  mit einer einzelnen Kombination von Eingabewerten kann mehrere Stunden dauern

Lösung:

- Effiziente Suchstrategien nutzen

## Lösung mit naiven Methoden

---



# Suchalgorithmus Grid Search

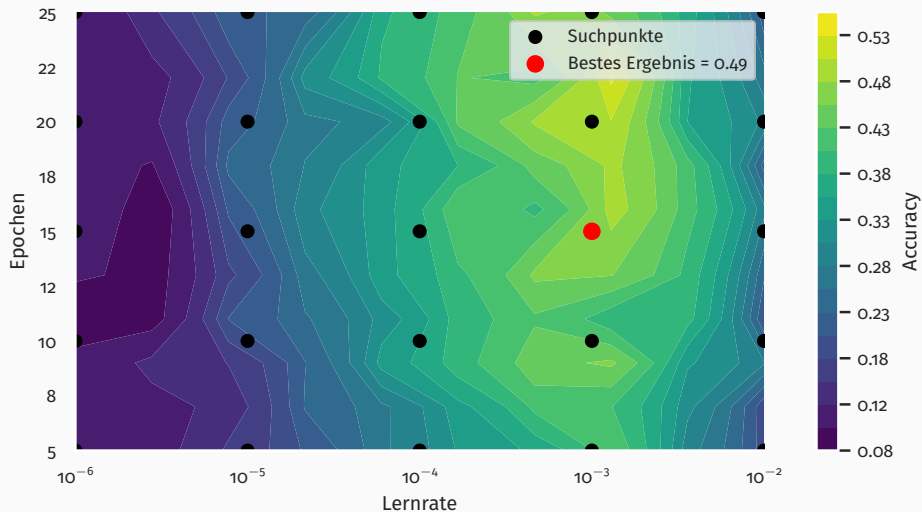


Abbildung 2: Suchpunkte für Grid Search

# Suchalgorithmus Random Search

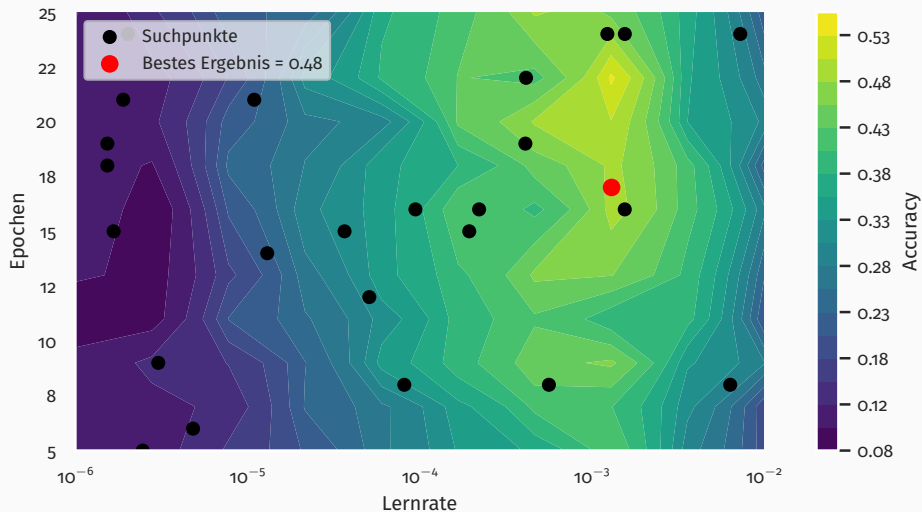


Abbildung 3: Suchpunkte für Random Search

## Lösung mit Bayesian Search

---

# Erklärung Bayesian Search

Wir bauen ein Modell  $m(\text{epochen}, \text{lernnrte})$ . Wir hoffen, dass es sich annähernd wie unsere zielfunktion  $f(\text{epochen}, \text{lernnrte})$  verhält.

Wir wählen einen Suchpunkt aus, der in unserem Modell  $m$  den den maximalert ausgibt und berechnen  $f$ . Da die sich  $m$  und  $f$  ähnlich verhalten, sollte das ergebnis auch der maximalwert oder nahe dran für  $f$  sein.

Mit dem Ergebnis können wir unser Modell  $m$  anpassen, um die zielfunktion besser zu modellieren.

Dieser Vorgang wird wiederholt, bis wir keine Versuche mehr übrig haben. Als Ergebnis sollte unser Modell recht gut die Maximalwerte von  $f$  vorhergesagt haben.

Zusätzlich wird ab und zu statt den aktuell vorhergesagten Bestwert, ein ganz anderer Punkt ausgetestet. Damit testen wir, ob in bisher nicht gesuchten Bereichen unser Modell stark abweicht und dort tatsächlich bessere Werte zu

## Abbildung 4: Bayes'sche Scoreoptimierung für einen Random-Forest-Klassifizierer

Quelle: [1]

**Legende:** *x-Achse:* Parameter des Random-Forest-Klassifizierers. *Schwarz:* Zielfunktion. *Lila:* Modellierte Funktion mit Unsicherheitsbereich  $\pm 1$  Standardabweichung. *Expected Improvement:* Erwarteter Zugewinn gegenüber dem aktuellen Bestwert. *Upper Confidence Bound:* Suche vielversprechende, aber unerkundete Bereiche. *Probability of Improvement:* Wahrscheinlichkeit, dass ein neuer Punkt besser ist als der bisherige Bestwert.

# Suchalgorithmus Bayesian Search

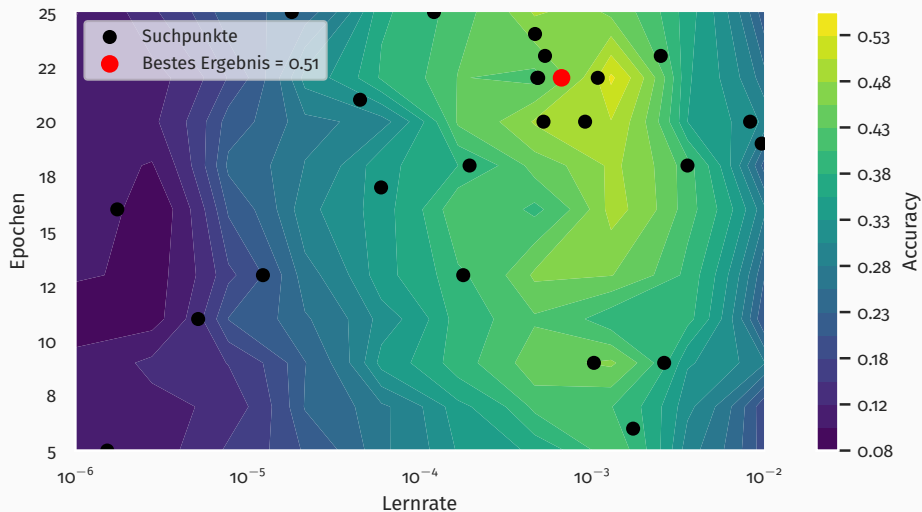


Abbildung 5: Suchpunkte für Bayesian Search

## Fazit

---

# Vergleich der Methoden

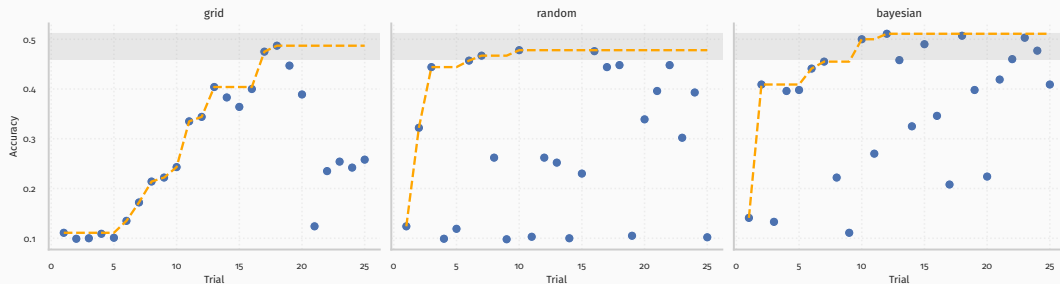


Abbildung 6: Vergleich der Suchalgorithmen

Legende: grau: Top 10% der Ergebnisse; orange gestrichelt: aktueller Bestwert



Fragen?

- [1] AnotherSamWilson, „**Bayesian optimization of a function with a Gaussian process**“, besucht am 27. Okt. 2025. Adresse:  
<https://github.com/AnotherSamWilson/ParBayesianOptimization>