CIFAR10 mit PyTorch klassifizieren

Neuronale Netze in der Bildverarbeitung

Jan Hoegen Nico Weber

27. Oktober 2025

Hochschule Karlsruhe
University of Applied Sciences

Inhaltsverzeichnis

IMPROVEremove all plt titles from figures. use latex caption

Allgemeines zu PyTorch

PyTorch im Überblick

Was ist PyTorch?

- Python-Bibliothek für Deep Learning
- Stark verbreitet in Forschung und Lehre [1]
- Unterstützt dynamische Berechnungsgraphen („Define-by-Run“)

Warum PyTorch?

- Einfache und flexible Modellimplementierung
- Direkte Nutzung von GPU-Beschleunigung
- Große Community, viele Tutorials und Ressourcen

Grundlegende Konzepte von Pytorch

Autograd Berechnet Gradienten automatisch für Backpropagation

`nn.Module` Basis für selbstdefinierte Modelle, enthält vordefinierte Layers

DataLoader Einfaches Laden, Batchen und Parallelisieren von Datensätzen

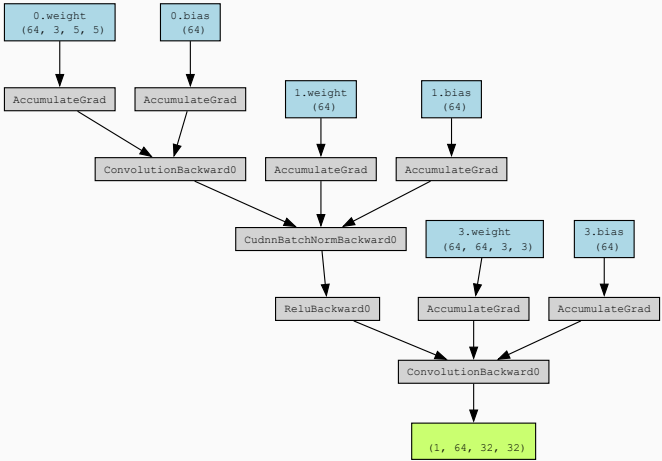
Optimizer Vorgefertigte Optimierer, z.B. **`Adam`**

Eigenes Modell zu CIFAR 10

Aufgabenstellung

- **Aufgabe:** Klassifikation von CIFAR-10 Bildern
- **Datensatz:**
  - 10 Klassen
  - 60.000 Bilder, Größe $32 \times 32 \times 3$
  - Trainingsset: 49.000 Bilder
  - Validierungsset: 1.000 Bilder
  - Testset: 10.000 Bilder
- **Ziel:** Modell in *maximal 10 Epochen* trainieren, um Bilder korrekt zu klassifizieren

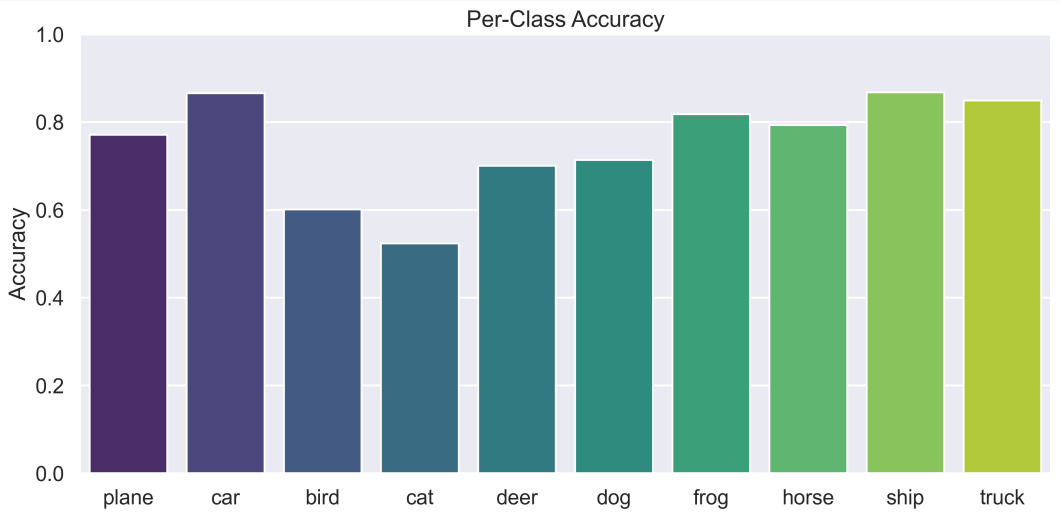
Architektur des Modells



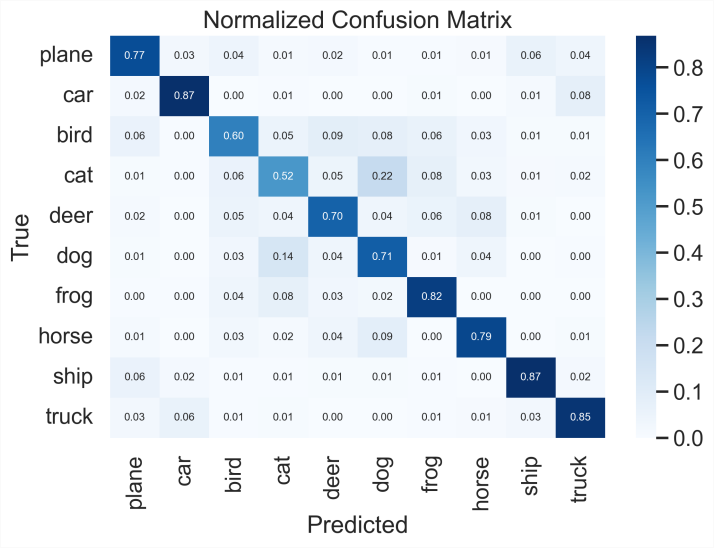
Ergebnisse des besten Modells

- Gewählte Hyperparameter:
  - ch1 = 64, ch2 = 64, ch3 = 64
  - Lernrate = 0.00086
  - Weight Decay = 3.81e-6
- Best Validation Accuracy: 75.70%
- Trainings-Accuracy: 73%
- Test Accuracy: 75.05%

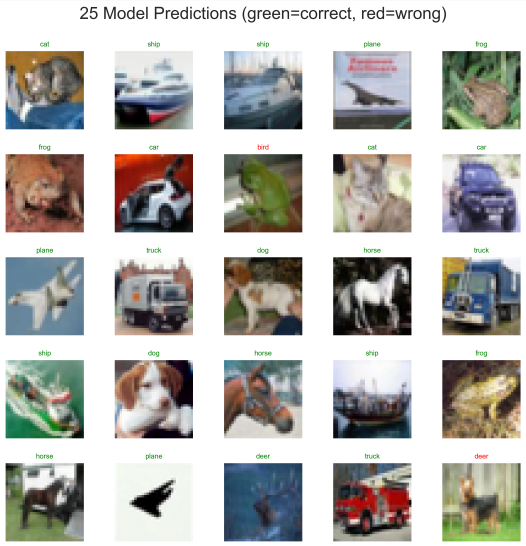
Accuracy für verschiedene Klassen



Confusion Matrix



Beispielvorhersagen



Training und Hyperparameter

Einstellungen für das Training

Trainingsdaten:

- 49.000 Trainingsbilder, Größe $32 \times 32 \times 3$
- Batchgröße: 256
- DataLoader-Worker: 12 (hohe Parallelisierung)

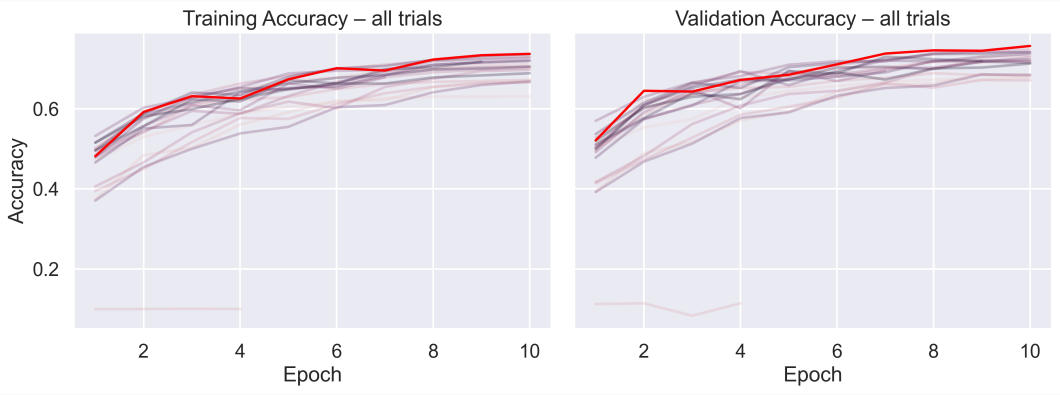
Training:

- Trainierbare Parameter: 604.810
- Iterationen pro Epoche: $\lceil 49{,}000/256 \rceil = 192$
- Epochen pro Trial: 10

Hyperparameter:

- Bayesian Search mit 20 Trials
- Optimierte Parameter: ch1, ch2, ch3, Lernrate, Weight Decay

Genauigkeit während des Trainings



Early Stopping

Motivation:

- Verhindert Overfitting
- Stoppt Training, wenn Validierungsgenauigkeit über mehrere Epochen nicht steigt

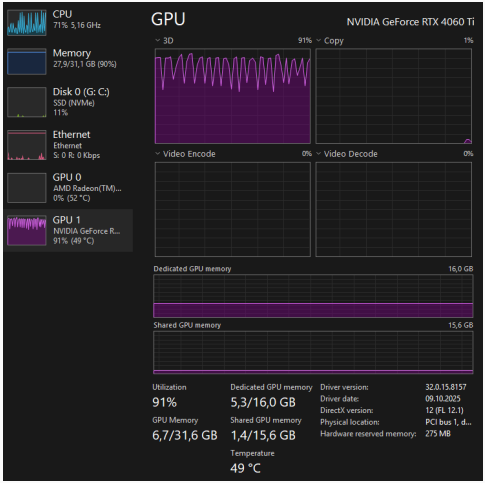
Prinzip:

- Wähle Grenzwert: `patience`
- Nach jeder Epoche: Behalte den besten Validierungswert
- Zähle Epochen ohne Verbesserung: `epochs_no_improvement`
- Stoppe, wenn $\texttt{epochs\_no\_improvement} \geq \texttt{patience}$

Hardware-Auslastung

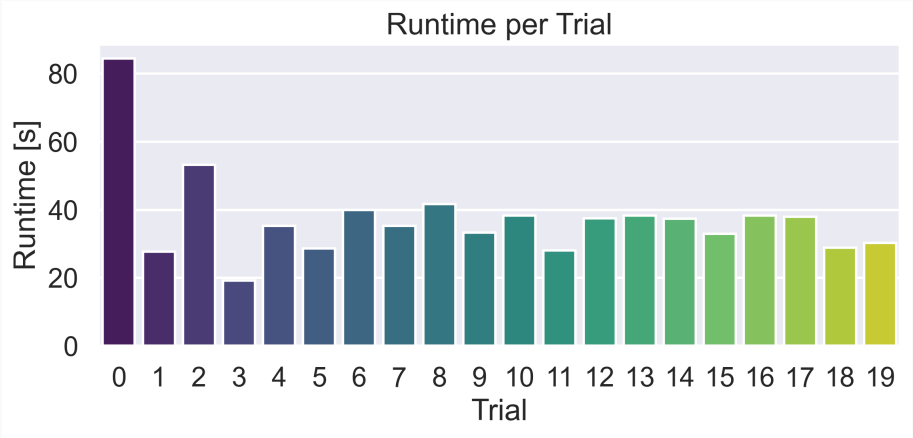
Hardware:

- GPU: RTX 4060 Ti, 16 GB VRAM
- CPU: Ryzen 5 7600X, 12 Kerne
- RAM: 32 GB DDR5



Laufzeiten pro Trial

Gesamtdauer: ca. 15 Minuten

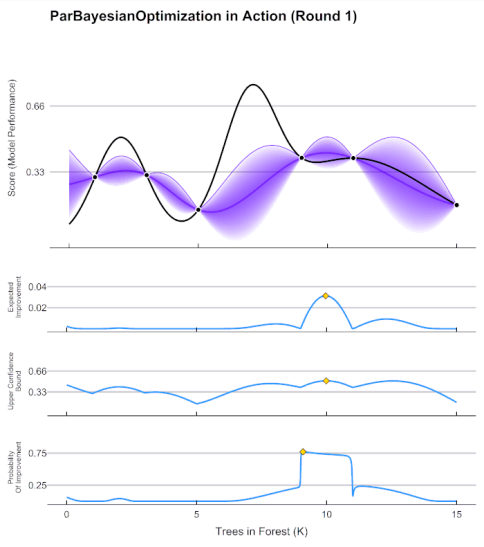


Hyperparameter mit Bayesian Search

Bayesian Search Erklärt

- Bayesian Search ist eine intelligente Methode zur Optimierung von Hyperparametern.
- Ziel: Maximierung der Validierungsgenauigkeit
  $f(\text{Parameter}) \rightarrow \text{Validation Accuracy}$.
- Idee:
  - Es wird ein Modell erstellt, das die unbekannte Zielfunktion f abschätzt.
  - Nach jeder Evaluierung wird dieses Modell mit den neuen Ergebnissen aktualisiert.
  - Eine *Acquisition Function* wählt die nächsten Parameterwerte aus – als Kompromiss zwischen **Exploration** (neue Bereiche testen) und **Exploitation** (bestehende gute Bereiche verfeinern).
- Vorteil: Findet gute Parameter mit deutlich weniger Versuchen als Random oder Grid Search.

Animation zu Bayesian Search



Anwendung auf unser Modell

- Anstatt zufällig Parameterkombinationen zu testen (Random Search) oder alle möglichen Kombinationen (Grid Search), wurde **Bayesian Search** verwendet.
- Das Modell der Funktion Validation Accuracy $=$ f(Hyperparameter) wird kontinuierlich angepasst.
- Neue Vorschläge für Hyperparameter werden auf Basis bisheriger Ergebnisse erzeugt.

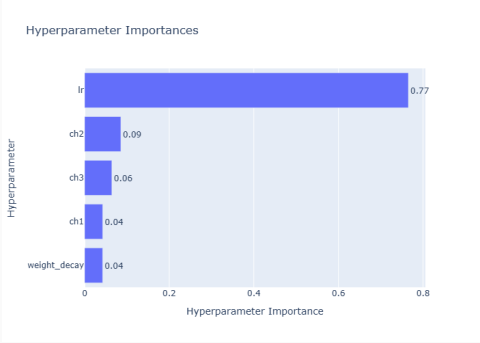


Abbildung 1: Relative Wichtigkeit der Hyperparameter

Bayesian Search Ergebnisse

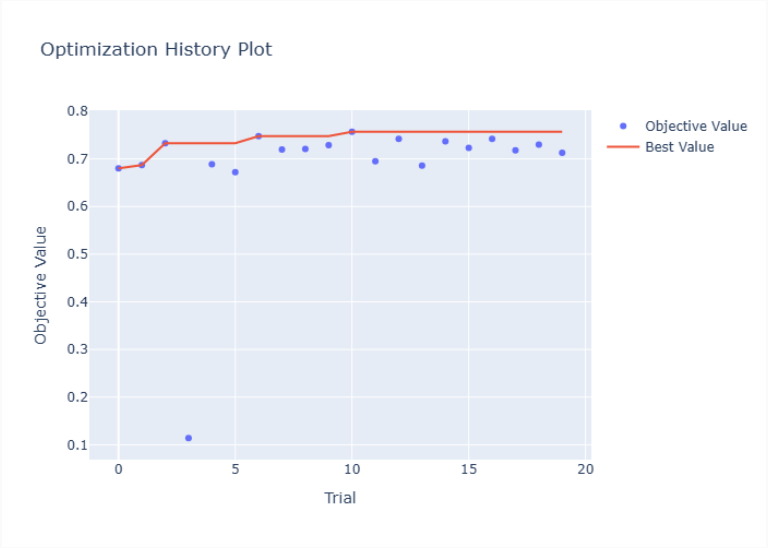


Abbildung 2: Verlauf der Validierungsgenauigkeit über die Trials (je höher, desto besser)

Ausblick

Ausblick

quantitaiver Vergleich bayesian mit anderen methoden

andere modelle

etc

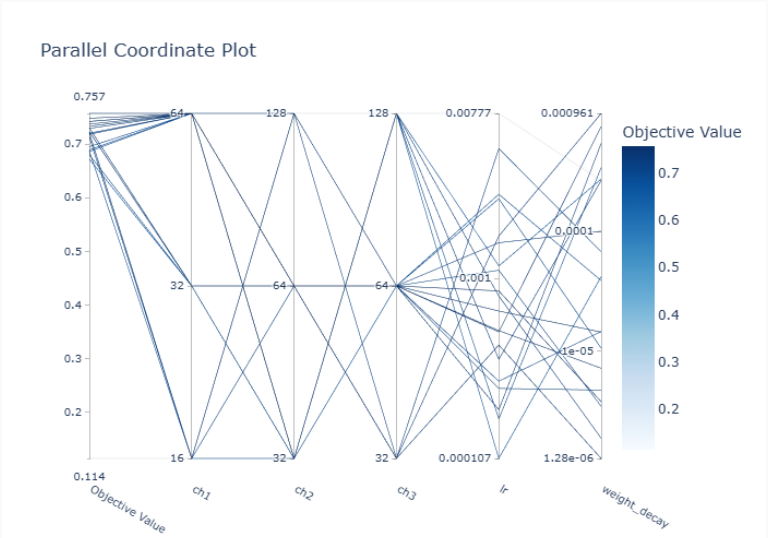
Fragen?

Literatur i

[1] J. Bauer, *ComputerVision2: Neuronale Netze in der Bildverarbeitung*. 18. Juni 2025.

[2] AnotherSamWilson, **„Bayesian optimization of a function with a Gaussian process“,** besucht am 27. Okt. 2025. Adresse: https://commons.wikimedia.org/wiki/File:GpParBayesAnimationSmall.gif

[3] S. Subramanian, S. Juarez, C. Breviu, D. Soshnikov und A. Bornstein, **„PyTorch Tutorials: Beginner Basics“,** besucht am 26. Okt. 2025. Adresse: https://docs.pytorch.org/tutorials/beginner/basics/intro.html


Parallel Coordinate Plot
0.757
0.7
0.6
0.5
0.4
0.3
0.2
0.114
64
32
16
128
64
32
128
64
32
0.00777
0.001
0.000107
0.000961
0.0001
1e-05
1.28e-06
Objective Value
0.7
0.6
0.5
0.4
0.3
0.2
Objective Value
ch1
ch2
ch3
lr
weight_decay