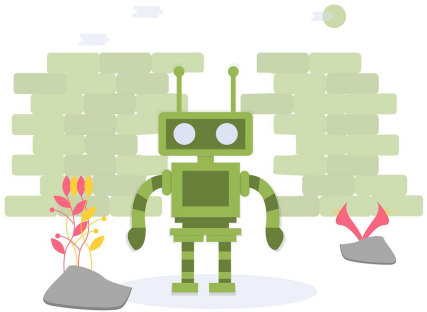
Verteiltes Deep Reinforcement Learning System zum Trainieren von Game AI

Projektgruppe SoSe 2021

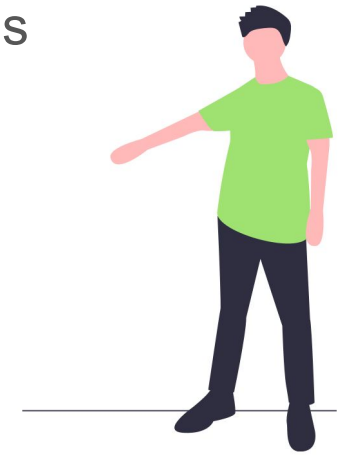


Marco Pleines| Dortmund, 10.02.2021

Hi!

Wir sind Roman Kalkreuth & Marco Pleines

- Lehrstuhl 11 Algorithm Engineering
- Prof. Günter Rudolph
- Deep Learning
- Genetische & Evolutionäre Algorithmen

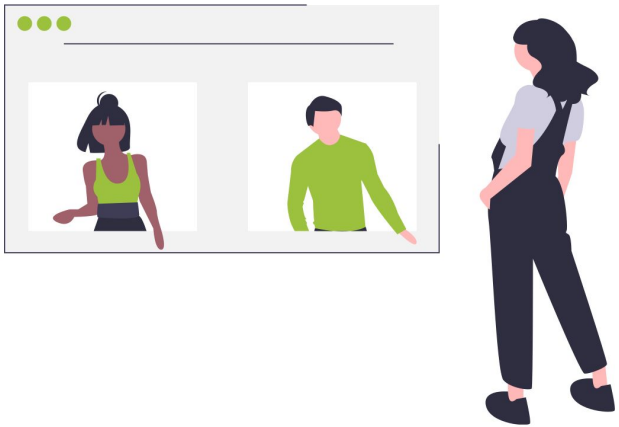


Wer seid Ihr?

- Bachelor wo gemacht?
- Thema Bachelorarbeit?

- Schon mal etwas mit ...
  - … Unity ...
  - … Deep Learning …

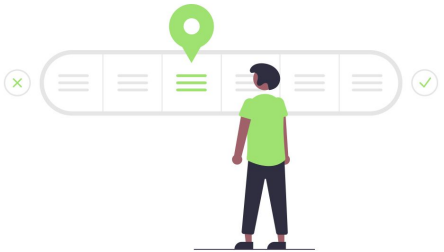
  … gemacht?



Agenda

Kurze DRL Einführung

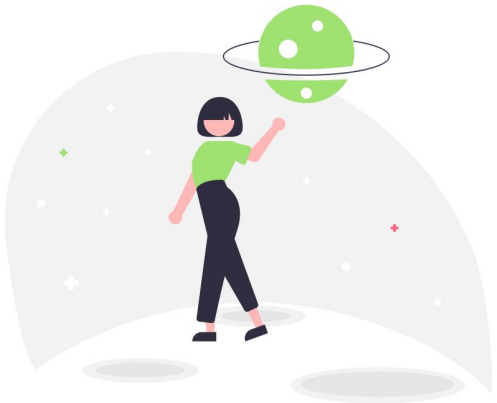
1. Grundlagen
2. Übung: REINFORCE
3. Probleme und Herausforderungen



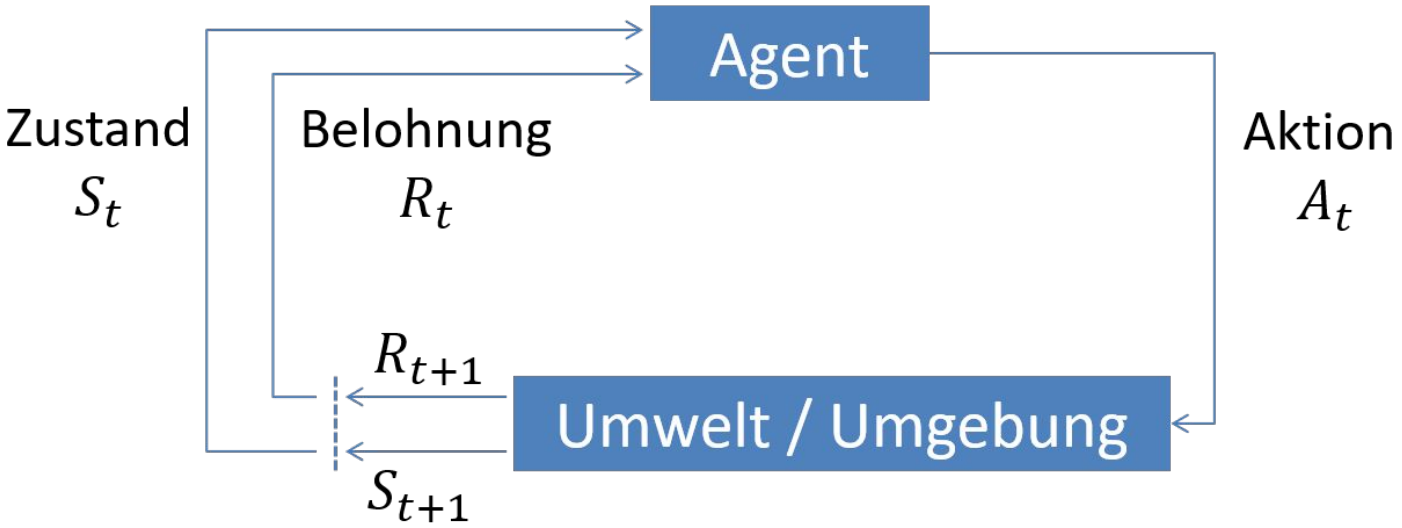
Projektgruppe

4. Vorhaben, Ziele und Vertiefungsmöglichkeiten
5. Seminar- und Praktikumsphase
6. Umfrage

1. Kurze Einführung in DRL



Markov-Entscheidungsprozess



Ziel

- Agent lernt autonom eine Strategie π
- Maximieren des Belohungssignals

- Erwarteter Rückgabewert:

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$$

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- γ - Diskontierungsrate (discount factor)

Value & Action-Value Function

Value Function

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right], \text{ for all } s \in \mathcal{S}$$

Q-Function oder Action-Value Function

$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right]$$

Q-Learning

- Optimieren der Q-Werte
- α - Lernrate (learning rate)

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$$

- Auswahl der Aktion mittels ε-greedy:
  - Wähle nützlichste Aktion oder
  - Zufall mit einer Wahrscheinlichkeit von ε

Traditionelles Reinforcement Learning

- Tabellarische Datenstrukturen für jeden Zustand und jede Aktion (z.B. Q-Learning, SARSA)

- **Warum könnte das nicht ausreichen?**

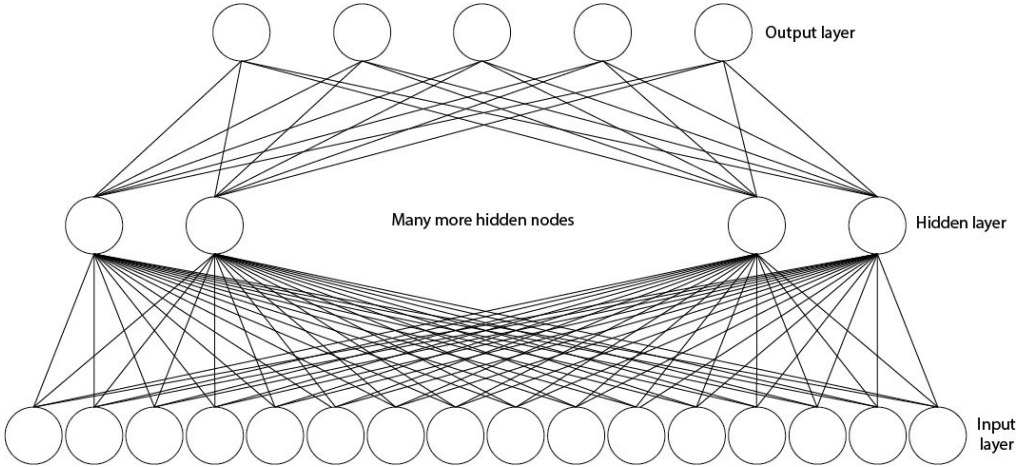
Traditionelles Reinforcement Learning

- Tabellarische Datenstrukturen für jeden Zustand und jede Aktion

- **Warum könnte das nicht ausreichen?**
  - Curse of Dimensionality
  - Zustandsraum zu groß oder kontinuierlich
  - Aktionsraum zu groß oder kontinuierlich

Traditionelles Reinforcement Learning

- Künstliche Neuronale Netze ⇒ Deep Reinforcement Learning
- Potential zur Generalisierung



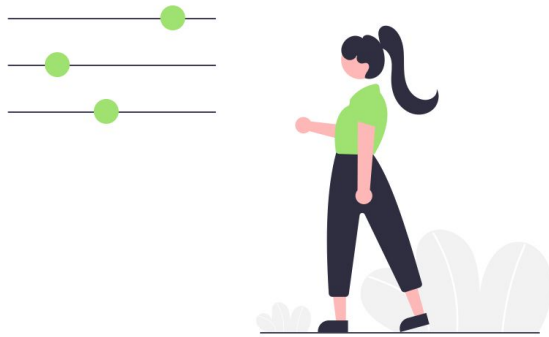
Deep Q Networks (DQN)

- Action-Value Function wird durch ein neuronales Netz approximiert

- Episode Replay Buffer
  - Speichert Transitionen

- Playing Atari with Deep Reinforcement Learning (2013)

2. Übung: REINFORCE



REINFORCE, R. J. Williams, 1992

- Parametrisierte Strategie (Gewichtungen eines neuronalen Netzes)
- Aktionswahrscheinlichkeiten als Ausgabe

- Wahrscheinlichkeiten werden optimiert beim Folgen des Gradientens der Strategie (engl. policy gradient)

REINFORCE

3 essentielle Komponenten:

- Parametrisierte Strategie (Gewichtungen eines neuronalen Netzes)
- Zu optimierende Zielfunktion
- Methode zum Optimieren der Parameter

Parametrisierte Strategie

- Künstliches neuronales Netz
  - Eingabeschicht ← Zustand der Umwelt
  - Versteckte Schicht
  - Ausgabeschicht → Wahrscheinlichkeitswerte für n-Aktionen
  - Aktivierungsfunktion

Zielfunktion

- $\boldsymbol{\tau}$ - Trajektorie (komplette Episode aus Umwelt-Agent-Interaktionen)
- $\boldsymbol{\theta}$ - Parameter des neuronalen Netzes
- t - aktueller Schritt
- T - Episodenlänge

$$J(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[R(\tau)] = \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} \gamma^t r_t]$$

Optimierung

- Anwendung des Gradientenverfahren in aufsteigender Richtung

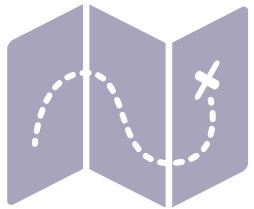
$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\pi_\theta)$$

- Policy Gradient

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} R_t(\tau) \nabla_\theta log \pi_\theta(a_t|s_t)]$$

Algorithmus

1. Initialisiere Lernrate α
2. Initialisiere Gewichtungen (θ) des Strategienwetzwerkes π
3. **for** episode 0, …, MAX_EPISODE **do**
   a. Sammle Trajektorie mittels der Strategie π
   b. **for** t = 0, …, T **do**
      i. Berechne diskontierte Belohnung für t
      ii. Berechne Zielfunktion für t
   c. **end for**
   d. Anwendung des Gradientenverfahren
4. **end for**



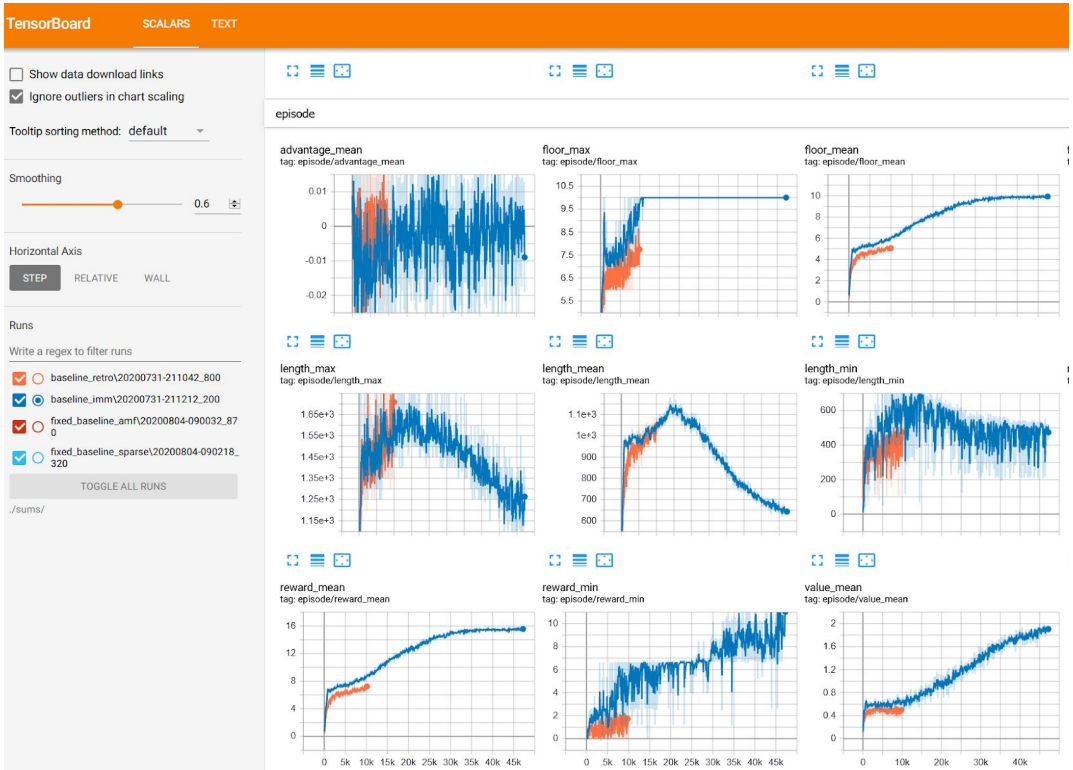
Hands-On REINFORCE

Jupyter Notebook für Google Colab

https://colab.research.google.com/drive/1wrsauUmL0oWH_bA7Y6lUr4oHU2FMa58A?usp=sharing

Tensorboard

Vor jedem Trainingsdurchlauf bitte die run_id ändern!



Lösung a)

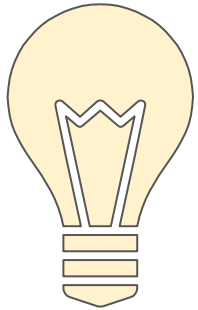
```
pg['lr'] = learning_rate * (1 - (episode / num_episodes))
```

Lösung b)

```
returns = (returns - returns.mean()) / (returns.std() + 1.0e-7)
```

Was machen SOTA Algorithmen besser?

- Mehr Agenten sammeln parallel Daten
- Aktualisierungen zu groß für das Netz?
  - Normalisierung
  - Clipping
- Actor-Critic
  - Hinzunahme der Value Function
  - Anstatt Value -> Advantage Function
- Verwendung mehrerer neuronaler Netze



SOTA:
PPO
SAC

3. Probleme und Herausforderungen

Exploration vs Exploitation Dilemma

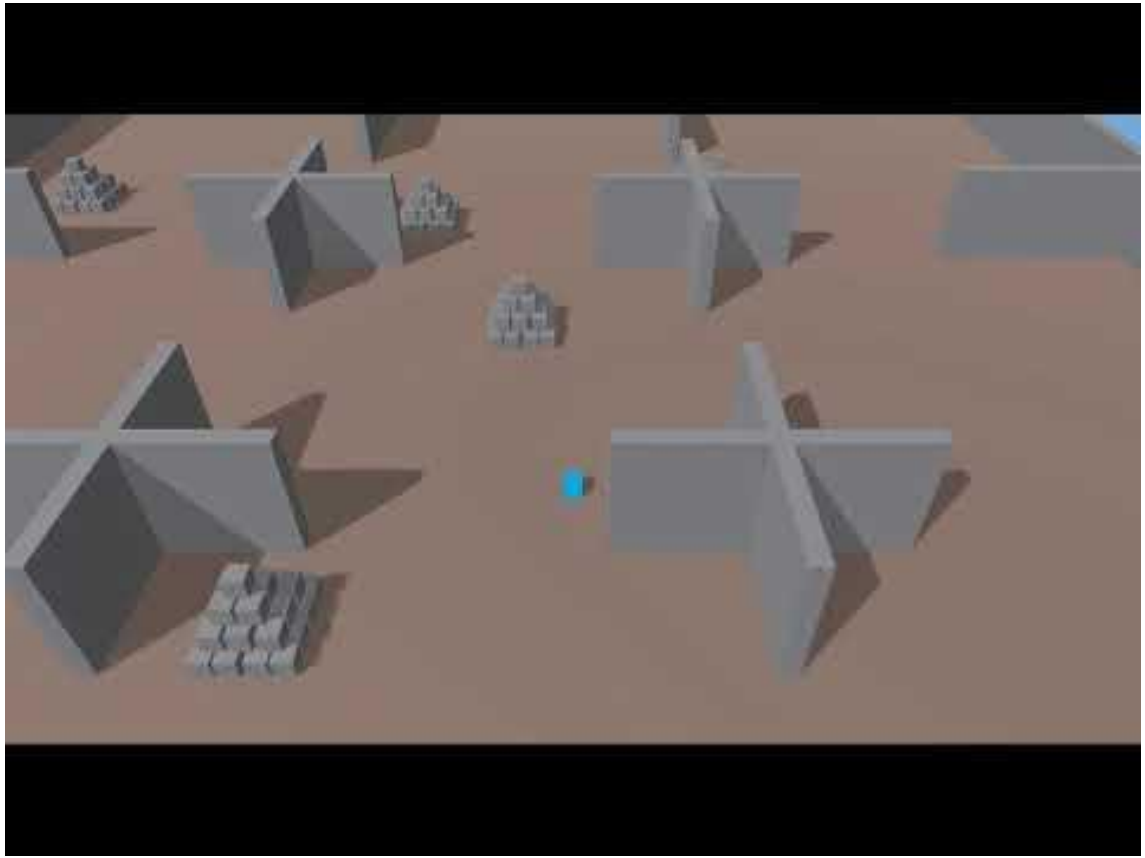
- Trial-and-error Verfahren

Fragestellungen:

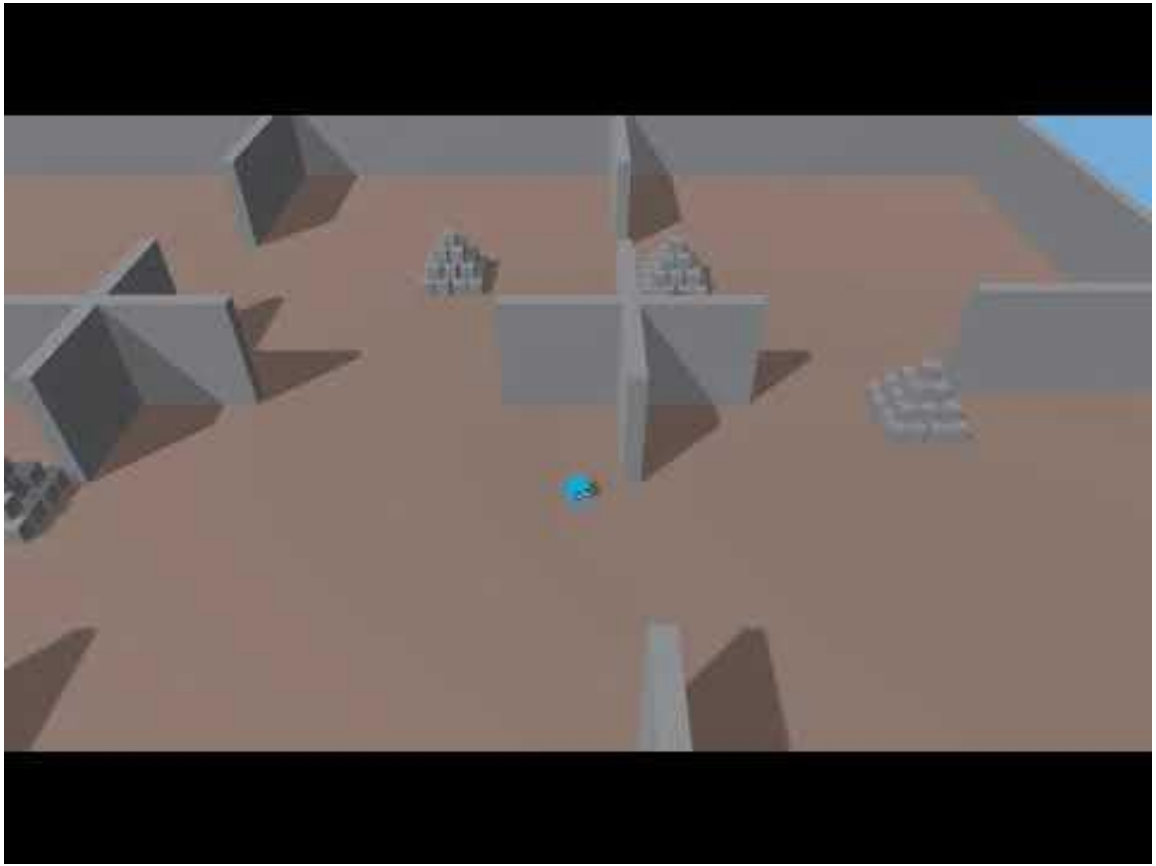
- Wie erkunde ich meine Umwelt?
- Wie lange erkunde ich meine Umwelt?
- Ab wann soll ich meine gelernte Strategie ausnutzen?

- Ansätze der intrinsischen Motivation z.B. Curiosity

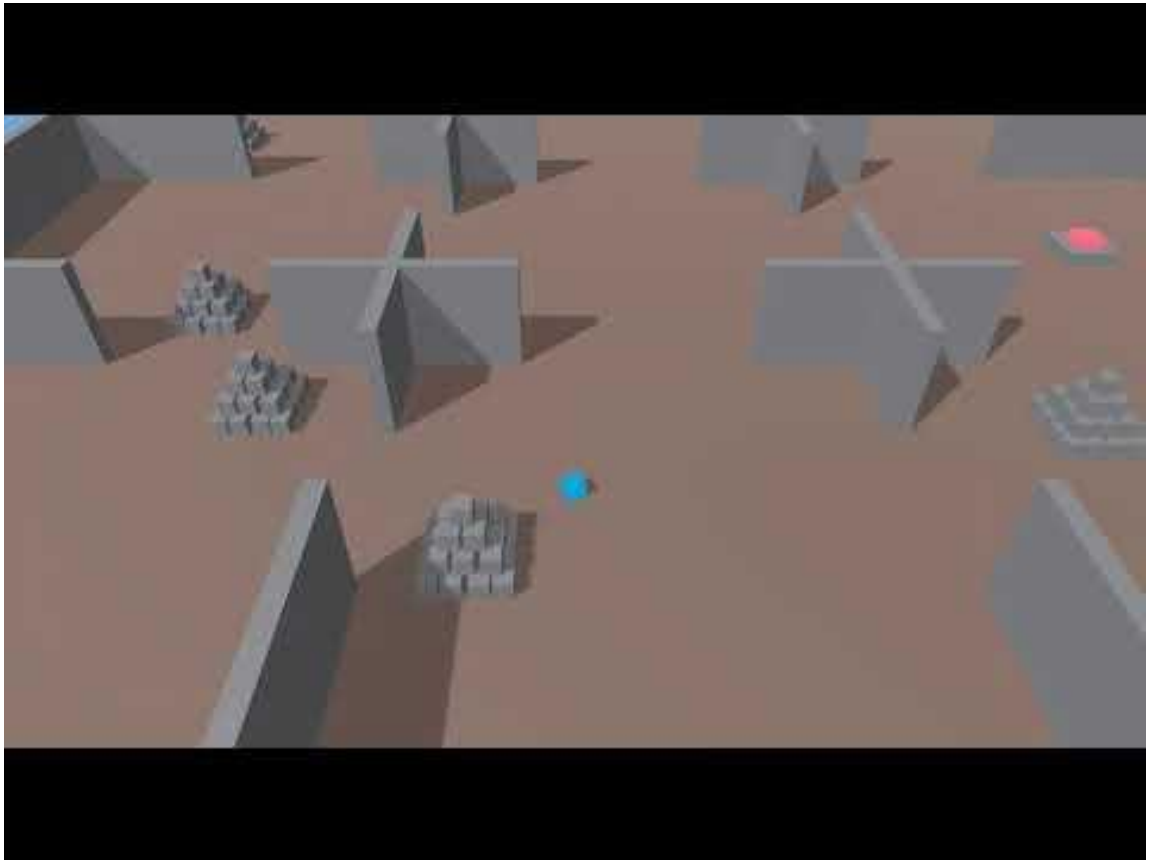
https://www.youtube.com/watch?v=3ro7yXfK_X0&feature=emb_logo



https://www.youtube.com/watch?v=yEcBMCU_VRU&feature=emb_logo



https://www.youtube.com/watch?v=9YomzVTv3Ho&feature=emb_logo



Sparse and Delayed Rewards

- Belohnungen können jeden Schritt signalisiert werden (dense)
- Belohnungen können nur am Ende einer Episode ausgeschüttet werden

- Belohnungen können verzögert signalisiert werden

Credit Assignment Problem

- Welche Aktion war entscheidend zum Erhalt des positiven Belohnungssignals?

Sample Efficiency

- Wie viele Daten / Erfahrungen benötigt der Agent um eine optimale Strategie zu erlernen?
- Können alte Daten wiederverwendet werden?

- On-Policy Algorithmen wie REINFORCE nutzen nur die Daten der aktuellsten Strategie
- Off-Policy verwendet auch Daten anderer Strategien (z.B. Deep Q-Networks)

Curriculum Learning

- Finales Problem vereinfachen und unterteilen
- Agent trainiert zunächst einfache Aufgaben
- Bei Erfolg, trainiere die nächst schwerere

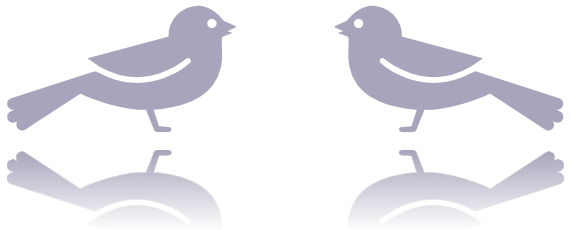
Problem: Welche Aufgabe maximiert den Lernerfolg?

Ansätze: z.B. Automated Curriculum Learning

Nützlich: Procedural Content Generation

Multi-Agent

- Min. 2 Agenten interagieren miteinander in einer Umwelt
- Kompetitiv, kooperativ oder beides in Form von Teams
- Herausforderungen:
  - Schwer vorhersehbar
  - Kommunikation
  - Curse of dimensionality
  - Credit assignment
  - ...



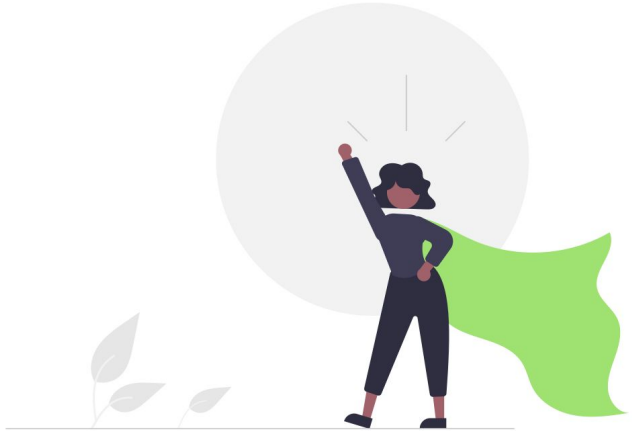
Kurz- und Langzeitgedächtnis

https://de.wikipedia.org/wiki/Motion-Interpolation

Problem: Partiell observierbare Umwelten (unvollständige Informationen)

Ansätze: z.B. Rekurrentes Neuronale Netz

4. Vorhaben, Ziele und Vertiefungsmöglichkeiten

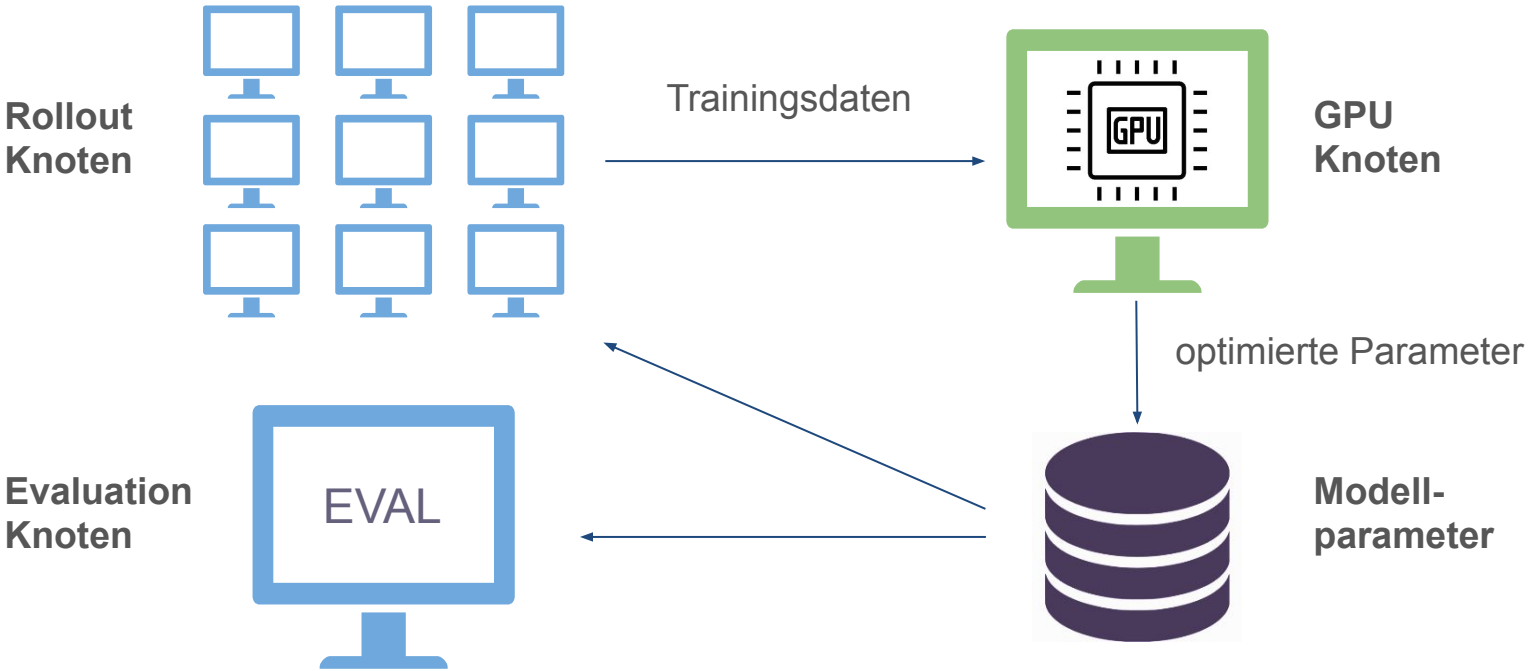


Marco Pleines| Dortmund, 10.02.2021

Vorhaben

- 10x PC-Pool Rechner
  - RTX 2070, 8 Kerne
- 2x LiDo3 Knoten
  - Tesla V100, 64 Kerne
- HPC: Jülich Supercomputing Centre (Auf Antrag)

Vorhaben



nach [5]

Vertiefungsmöglichkeit

Vorhaben

Vertiefungsmöglichkeit

Vorhaben

Vertiefungsmöglichkeit

Vorhaben

PyTorch

unity

Memory

Multi-Agent Competitive

Novel Environments

Verteiltes Trainingssystem
Deep Reinforcement Learning

Procedural Content Generation

Curriculum Learning

Multi-Agent Cooperative

Ziele

- Konzeption, Dokumentation und prototypische Umsetzung eines verteilten Trainingssystemes
- Anwendung und Evaluierung des Trainingssystemes
- Konzeption und prototypische Umsetzung von neuartigen DRL Umwelten (Größenordnung Minispiel)
- Anwedung des Trainingssystemes
- Teilnahme an einem Wettbewerb

Ziele

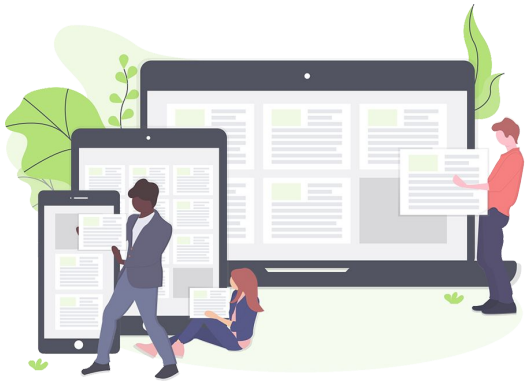
- Folienpräsentation zum Abschluss eines Semesters im Rahmen des LS-Seminars
- Erstellung eines Zwischen- und Abschlussberichtes
- (Fachgespräch unter Vorbehalt)
- Termine TBA

Vertiefungsmöglichkeiten

- Currciulum Learning
- Mutli-Agent Szenario
- Lang- und Kurzzeitgedächtnis

- Transfer Learning
- Intrinsic Motivation
- Model-Based DRL
- (Imitation Learning)

5. Seminar- und Praktikumsphase



Marco Pleines| Dortmund, 10.02.2021

Präsentationen im Seminar

- Dreierteams
- Wahl eines Präsentationsthemas
- Vorbesprechung mit den Betreuern
- Präsentationsdauer: 45 Minuten
- Diskussion: 10 Minuten
- Hands-On Session: 20 Minuten
- Diskussion: 10 Minuten
- Handout: 1 DIN A4 Seite

15. April 2021, 14 Uhr
22. April 2021, 14 Uhr
29. April 2021, 14 Uhr
06. Mai 2021, 14 Uhr

Praktikum

- Bearbeitung von Übungsaufgaben
- Betreuung während der Praktikumszeit (3h)

- Google Colab
- PC-Pool LS11 (remote)
- Live Share Visual Studio Code?

- Unity, ML-Agents
- PyTorch

16. April 2021, TBA
23. April 2021, TBA
30. April 2021, TBA
07. Mai 2021, TBA

Rainbow DQN

- Tabular Q-Learning
- Deep Q-Network
- Double Q-Learning
- Prioritized Replay
- Dueling Networks
- Multi-step Learning
- Noisy Networks

Hessel et al. 2018. Rainbow: Combining Improvements in Deep Reinforcement Learning. AAAI:3215-3222. Arxiv.

Soft Actor-Critic (SAC)

- Deep Deterministic Policy Gradients (DDPG)
- Twin-Delayed DDPG (TD3)
- Maximum Entropy
- Double Q-Learning
- Reparameterization Trick

Haarnoja et al. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. ICML: 1856-1865. Arxiv.

Proximal Policy Optimization (PPO)

- Policy Gradient
- Advantage Actor-Critic (A2C)
- Trust Region Policy Optimization (TRPO)
- Generalized Advantage Estimation
- Surrogate Objectives
- (Phasic Policy Gradient (PPG)?)

Schulman et al. 2017. Proximal Policy Optimization Algorithms. Arxiv.

Never Give Up

- Distributed Agents
- Exploration Strategies
- Short-Term Memory
- Episodic Memory

Badia et al. 2020. Never Give Up: Learning Directed Exploration Strategies. ICRL. Arxiv.

https://deepmind.com/blog/article/Agent57-Outperforming-the-human-Atari-benchmark

Discord

https://discord.com/invite/m2REf2duWN

Weitere nützliche Quellen

https://lilianweng.github.io/lil-log/2018/02/19/a-long-peek-into-reinforcement-learning.html

https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html

https://www.youtube.com/channel/UC58v9cLitc8VaCjrcKyAbrw

Prüfungsleistung

Richtlinien zur Durchführung von Projektgruppen **§7** & **§8**:

https://www.cs.tu-dortmund.de/nps/de/Studium/besondere_Lehrveranstaltungen/Projektgruppen/Rechtliches/index.html

⇒ Engagiert Euch!

6. Umfrage

Umfrage

- git?
- Google Colab, Jupyter Notebook?
- Unity?
- Deep Learning
  - PyTorch?
- Python IDE?

- Discord Server einrichten?

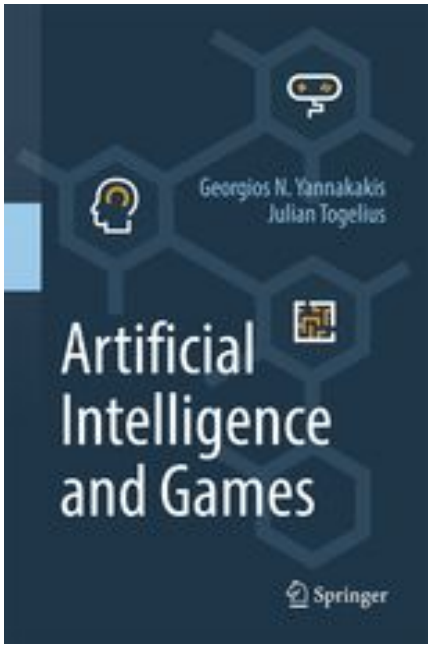
Literaturempfehlungen

Artificial Intelligence and Games

Yannakakis & Togelius

Springer

2018



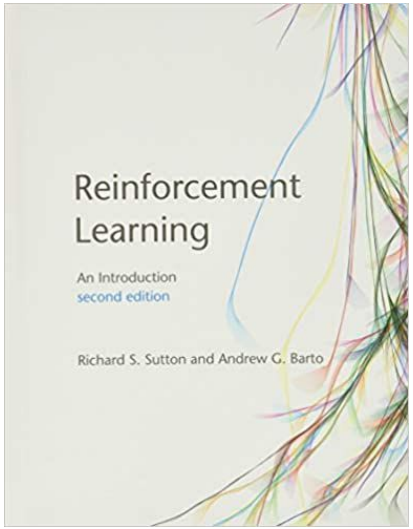
Reinforcement Learning

Sutton & Barto

2nd Edition

MIT Press

2018



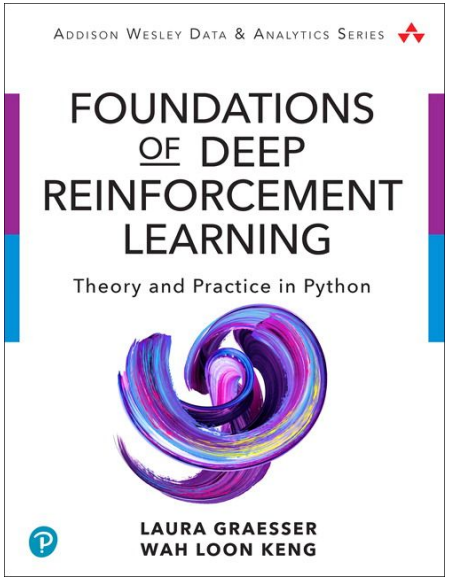
http://incompleteideas.net/book/the-book.html

Foundations of Deep Reinforcement Learning

Graesser & Keng

Pearson

2019

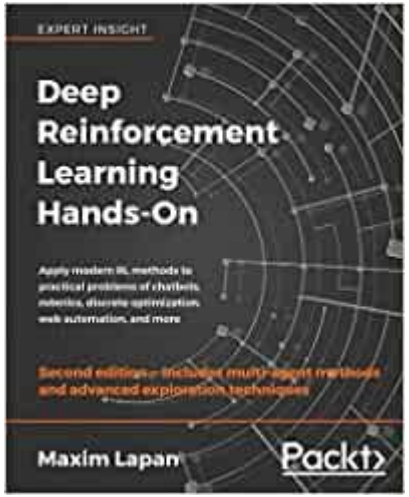


Deep Reinforcement Learning Hands-On

Lapan

PacktPub

2018



Machine Learning with Phil

https://www.youtube.com/channel/UC58v9cLitc8VaCjrcKyAbrw

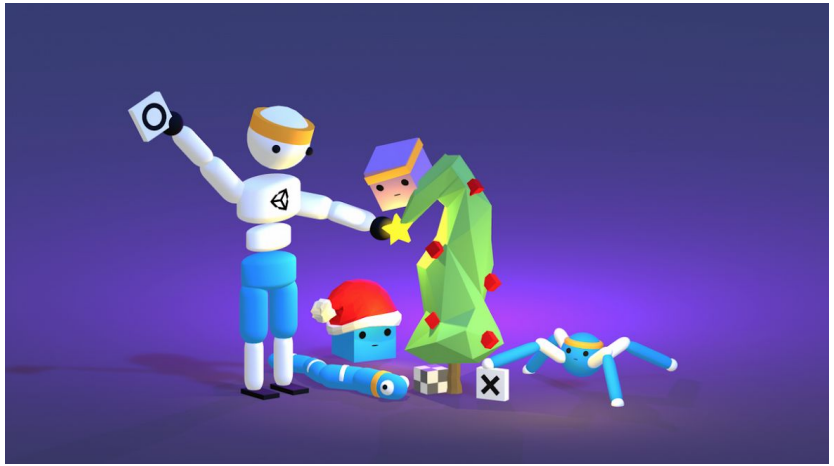
Intuitive RL intro to A2C

https://hackernoon.com/intuitive-rl-intro-to-advantage-actor-critic-a2c-4ff545978752

Unity ML-Agents Toolkit

https://github.com/Unity-Technologies/ml-agents

https://blogs.unity3d.com/category/machine-learning/



Fragen?

Kontakt

Roman Kalkreuth
roman.kalkreuth@tu-dortmund.de

Marco Pleines
marco.pleines@tu-dortmund.de