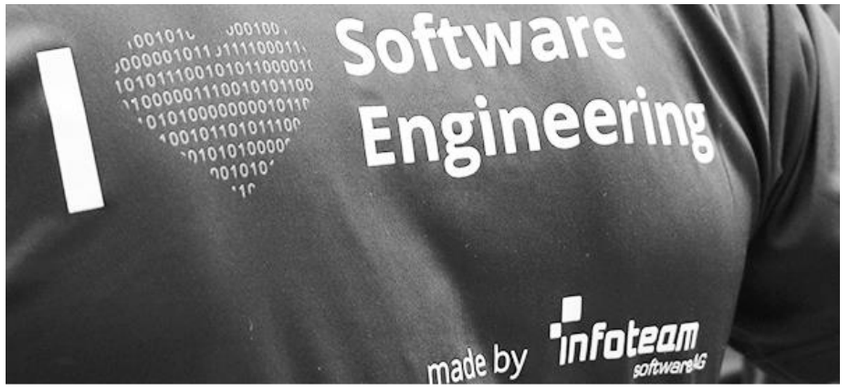
Reinforcement Learning

Ingmar Kieferle & Dr. Sebastian Falkner

infoteam

- Softwaredienstleister seit 1983
- Firmensitz in *Bubenreuth*, weitere Standorte in *Dortmund* und in der Schweiz *(Baar), Tschechien (Prag)* und *China (Peking)*
- Märkte: *Industry, Infrastructure, Life Science und Public Services*
- Aktuell ~ 350 Mitarbeiter, von Schülern und Studenten bis zu Senior-Professionals
- Data Science Abteilung mit ~ 40 Mitarbeitern, davon ~ 10 im Analytics Team

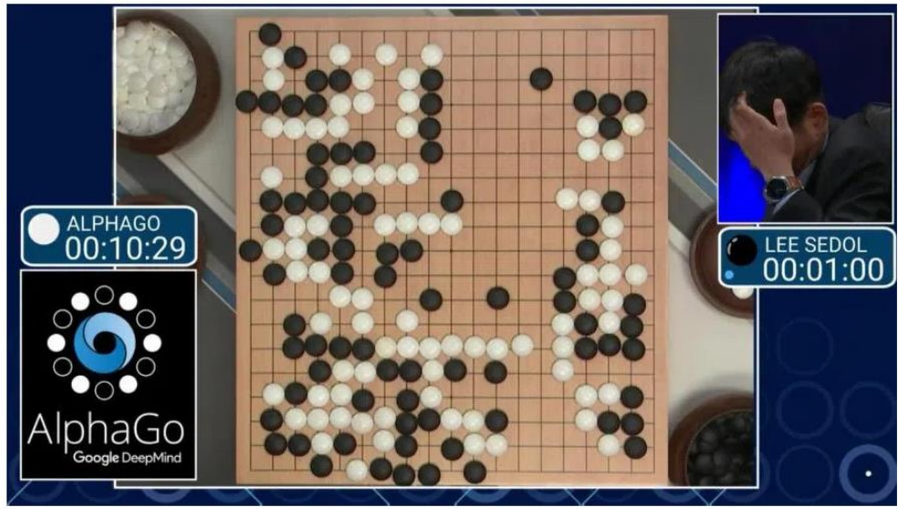


Deep Reinforcement Learning Framework

Deep Reinforcement Learning Framework

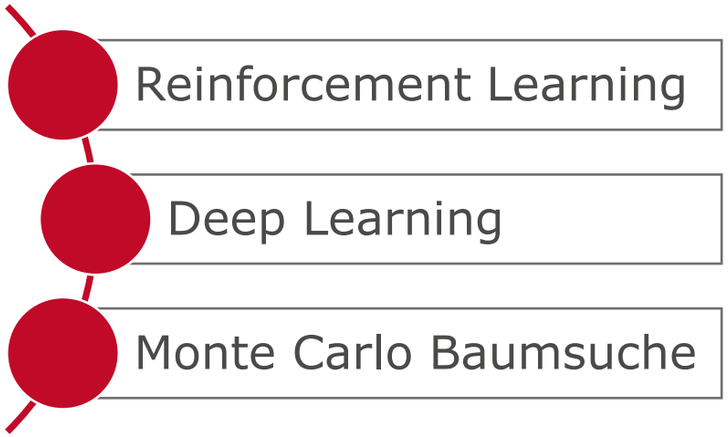
Konzeption und Implementierung eines Lernframeworks

- zur Optimierung von simulierbaren Entscheidungsproblemen
- am Vorbild von *Deepminds Alpha Zero*



Maschine schlägt Mensch: 2016 besiegte Googles Machine Learning System AlphaGo den Weltmeister im Spiel Go.

Foto: Google



Beispiele

Mehrspieler-Spiele:

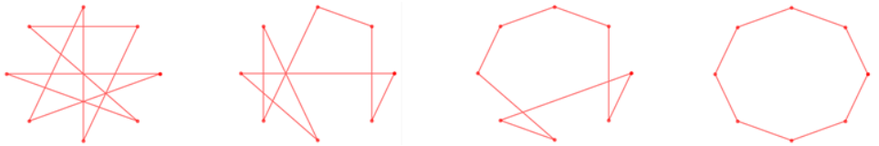
- Brettspiele wie Go, Schach, …
- Kartenspiele wie Poker, etc.
- Würfelspiele, etc.

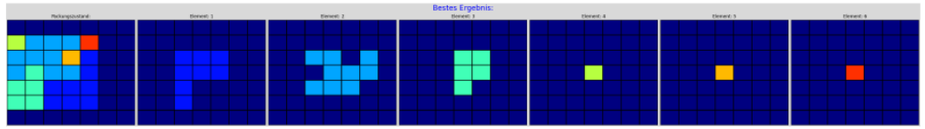
Einspieler-Spiele:

- Logik-Puzzles wie Rubik Würfel

Optimierungsprobleme als Einspieler-Problem:

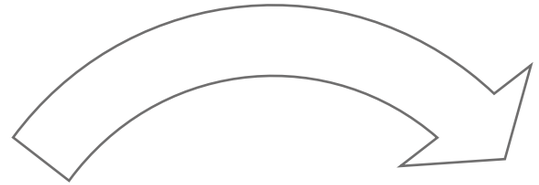
- Finde kostengünstigste Route unter Nebenbedingungen (Travelling Salesman)
- Finde kostengünstige Anordnung von geometrischen Objekten (Bin Packing)
- …





Deep Reinforcement Learning Framework

Trainingsphase

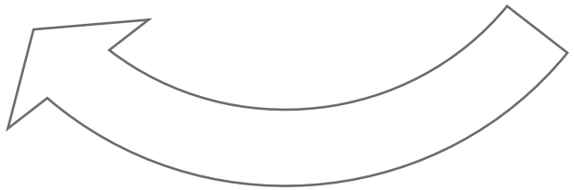


Erstellung der Trainingsdaten

Reinforcement Learning

MC-Baumsuche

Neuronales Netz

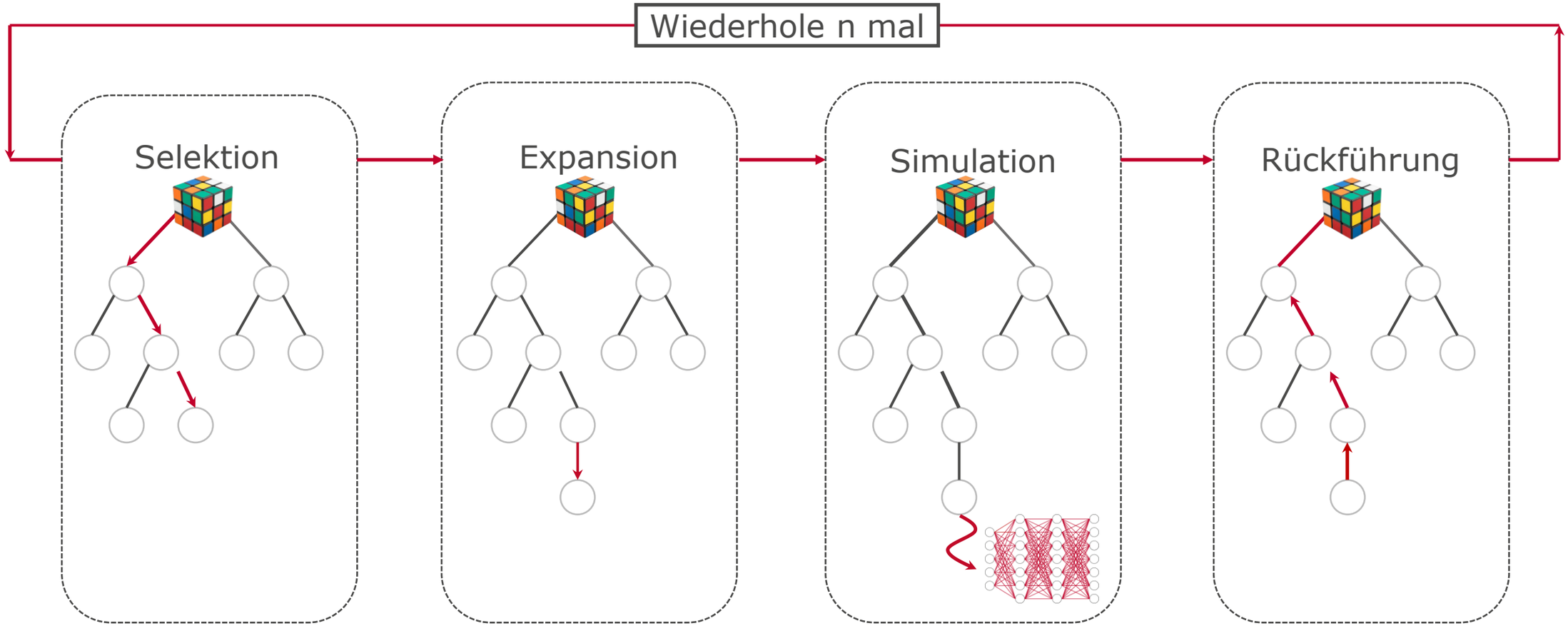


Training des neuronalen Netzes

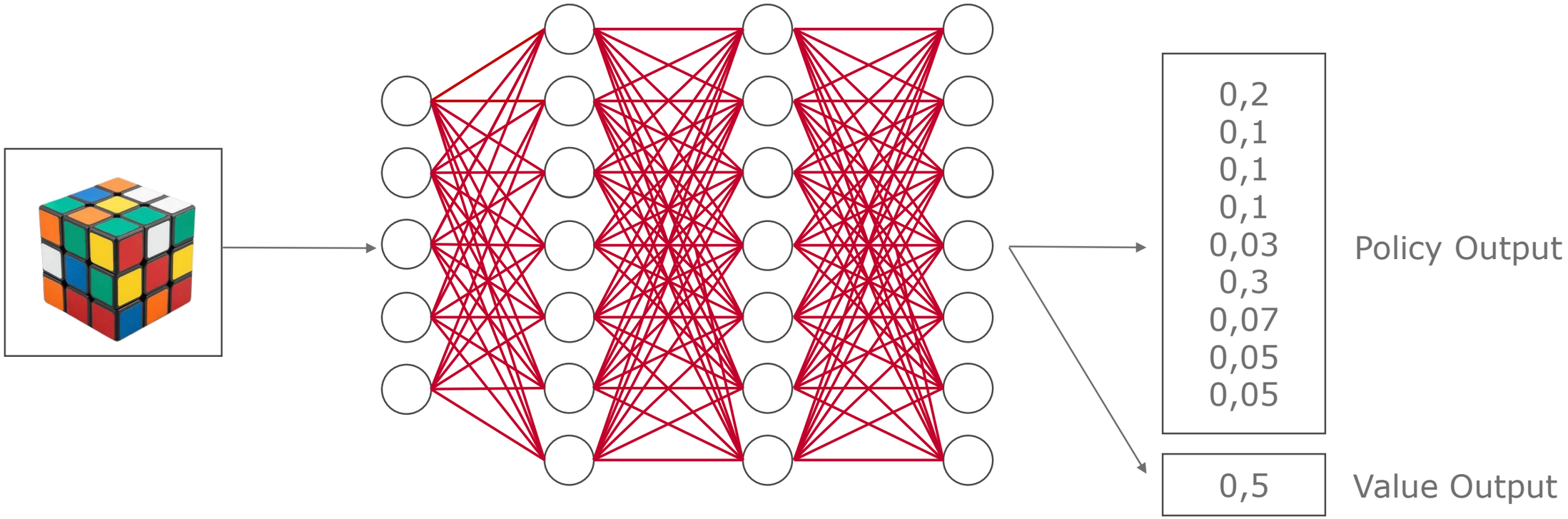
Wettbewerbsphase

„Intelligenter" Agent
(Neuronales Netz + MC-Baumsuche)

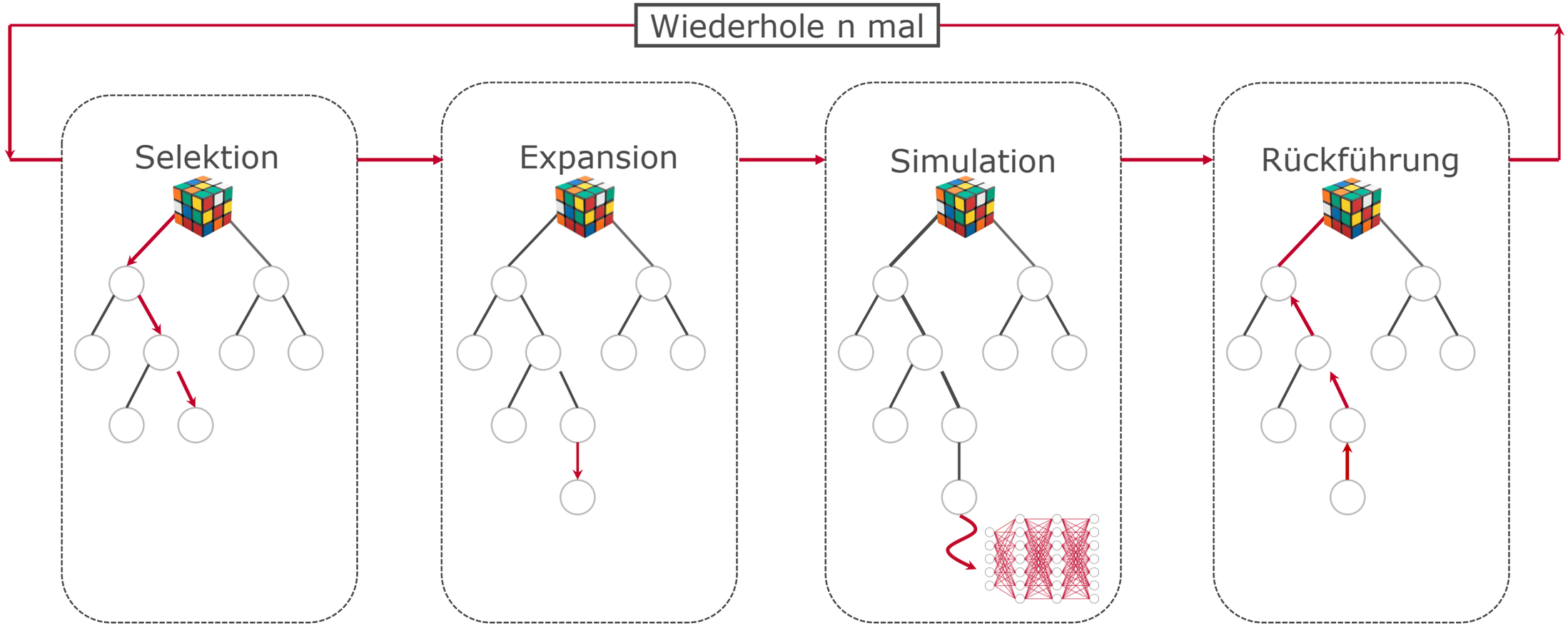
Monte Carlo Baumsuche



Neuronales Netzwerk



Monte Carlo Baumsuche

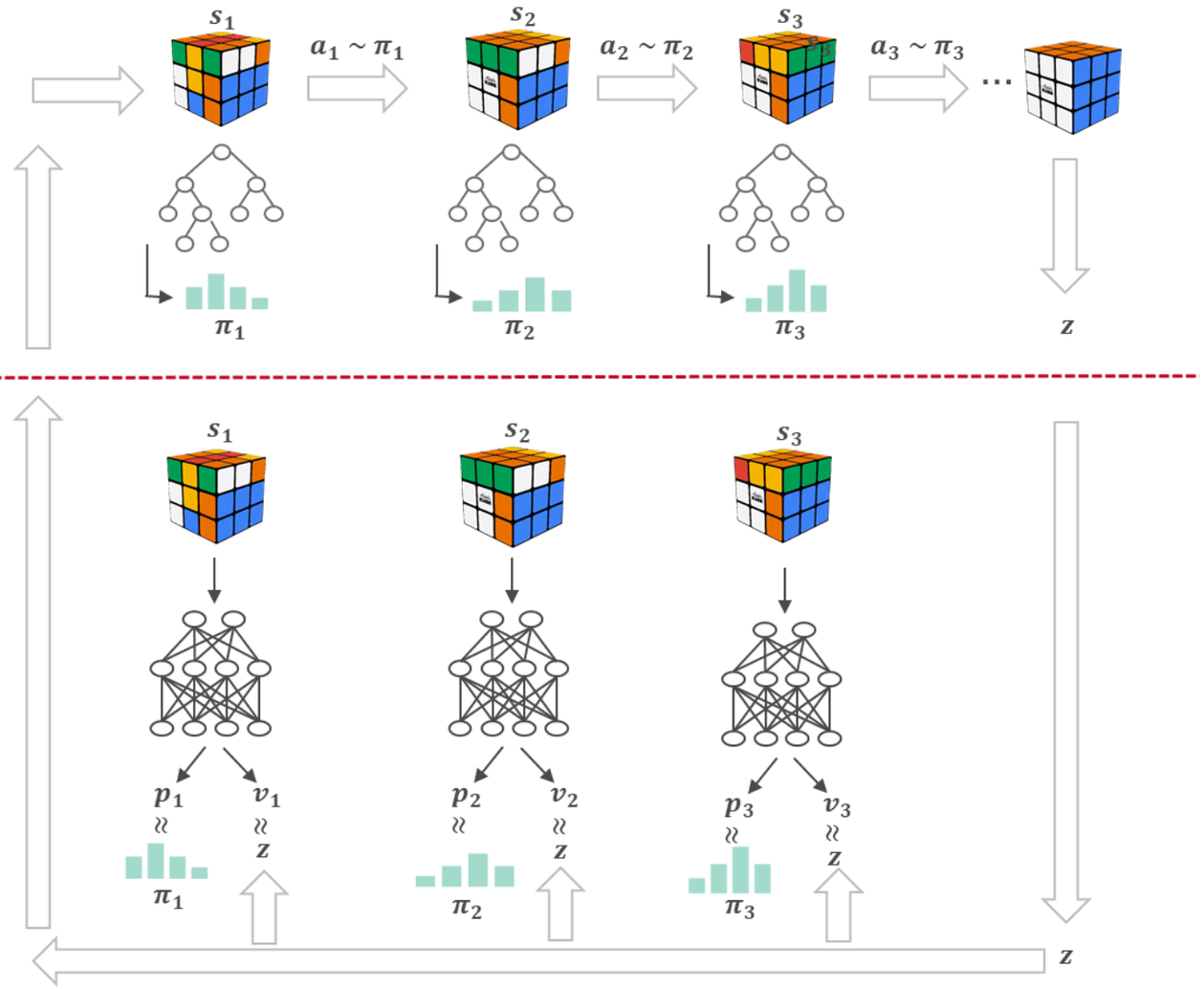


Deep Reinforcement Learning System

Erstellung der Trainingsdaten

Trainingsphase

Training des neuronalen Netzes



Relative Belohnungen

Relativer Bewertungsmechanismus

Ranked Rewards

- Bestimme *Ranked Reward Threshold (RRT)*
- d.h., bestimme Belohnungswert aus den letzten N Belohnungen zum Perzentil p (z.B. N=100, p=0,75)
- Mit anderen Worten: 75% der letzten 100 Spiele haben eine Belohnung r <= RRT
- **Idee:** Bewertung neuer Ergebnisse mittels *RRT* durch:

$$RR(r, RRT, R_{min}, R_{max}; a) = \begin{cases} -1, \text{ falls } r \leq R_{min}, \\ 1, \text{ falls } r \geq R_{max}, \\ -a - (1-a)\frac{RRT - r}{RRT - R_{min}}, \text{ falls } r < RRT, \\ a + (1-a)\frac{r - RRT}{R_{max} - RRT}, \text{ falls } r > RRT, \\ z \sim U(\{-a, a\}), \text{ sonst.} \end{cases}$$

Ranked Reward: Enabling Self-Play Reinforcement Learning for Combinatorial Optimization

Alexandre Laterre a.laterre@instadeep.com
Yunguan Fu y.fu@instadeep.com
Mohamed Khalil Jabri mk.jabri@instadeep.com
Alain-Sam Cohen as.cohen@instadeep.com
David Kas d.kas@instadeep.com
Karl Hajjar k.hajjar@instadeep.com
Hui Chen h.chen@instadeep.com
Torbjørn S. Dahl t.dahl@instadeep.com
Amine Kerkeni ak@instadeep.com
Karim Beguir kb@instadeep.com

Abstract

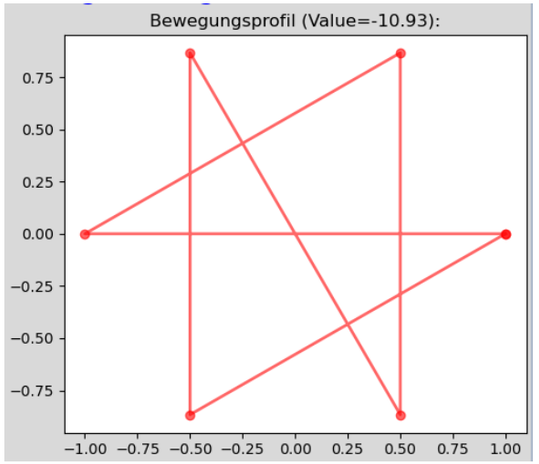
Adversarial self-play in two-player games has delivered impressive results when used with reinforcement learning algorithms that combine deep neural networks and tree search. Algorithms like AlphaZero and Expert Iteration learn *tabula-rasa*, producing highly informative training data on the fly. However, the self-play training strategy is not directly applicable to single-player games. Recently, several practically important combinatorial optimization problems, such as the traveling salesman problem and the bin packing problem, have been reformulated as reinforcement learning problems, increasing the importance of enabling the benefits of self-play beyond two-player games. We present the Ranked Reward (R2) algorithm which accomplishes this by ranking the rewards obtained by a single agent over multiple games to create a relative performance metric. Results from applying the R2 algorithm to instances of a two-dimensional and three-dimensional bin packing problems show that it outperforms generic Monte Carlo tree search, heuristic algorithms and integer programming solvers. We also present an analysis of the ranked reward mechanism, in particular, the effects of problem instances with varying difficulty and different ranking thresholds.

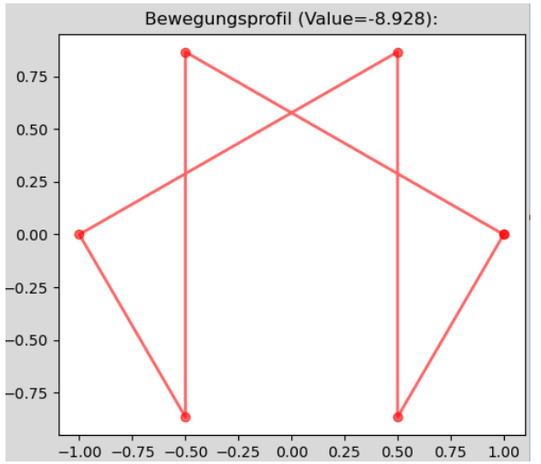
https://arxiv.org/abs/1807.01672

Beispiel: Travelling Salesman

Beispiel: Travelling Salesman

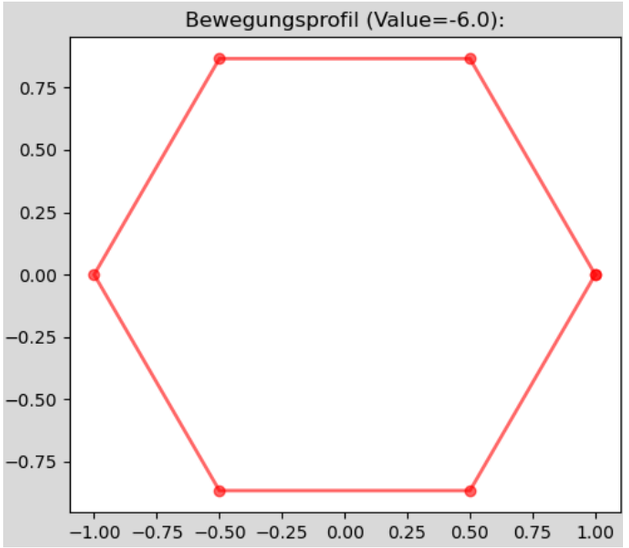
- N Orte mit paarweisen Distanzen bzw. Kosten D_{ij}
- Reise von einem Startort aus zu allen anderen Orten, so dass jeder Ort genau einmal besucht wird
- Am Ende wieder Reise zum Startort
- Finde Sequenz von Orten, so dass Gesamtreisekosten minimal sind





Beispiel:

- Reguläres N-Eck mit Euklidischem Abstand
- Optimale Kosten durch Umfang des N-Ecks
- Optimale Sequenz: Besuche Orte im Kreis-Sinn
- Also 2 optimale Lösungen



Beispiel: Travelling Salesman

Für ein festes Travelling Salesman Problem:

- (N-1)! Möglichkeiten
- Für N = 7 also 720 Möglichkeiten, die alle berechnet werden können
- Für N = 13 bereits ca. 48 Millionen (10e6) Möglichkeiten
- Für N = 19 ca. 640 Billionen (10e12) Möglichkeiten

Spielregeln:

- Gegeben Matrix mit pw. Kosten (insb. Sortierung)
- Erster Ort ist Startort und muss als erstes gewählt werden
- Wähle nächsten Ort, der noch nicht besucht wurde
- Kehre zum Startort zurück, wenn keine weiteren Orte besucht werden können

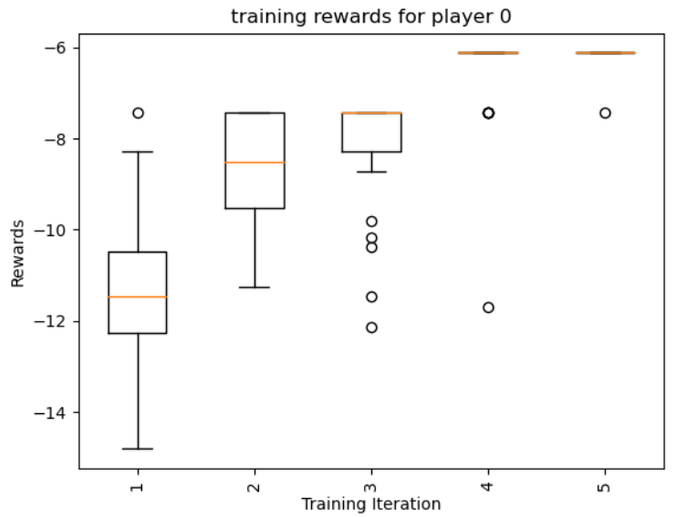
Beobachtungen

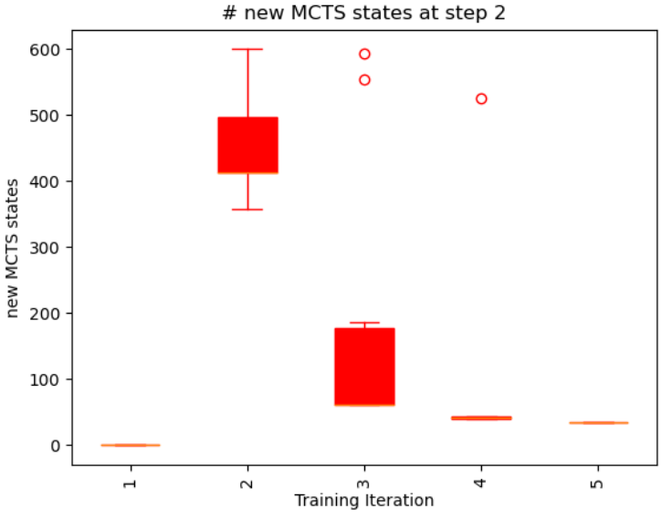
- Sequenz aus bereits besuchten Orten
- Paarweise Kosten D_{ij}

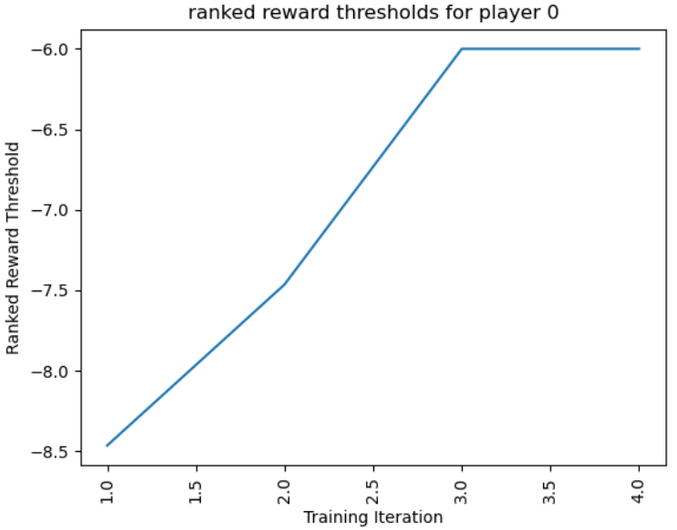
Erster unvollständiger Ansatz:

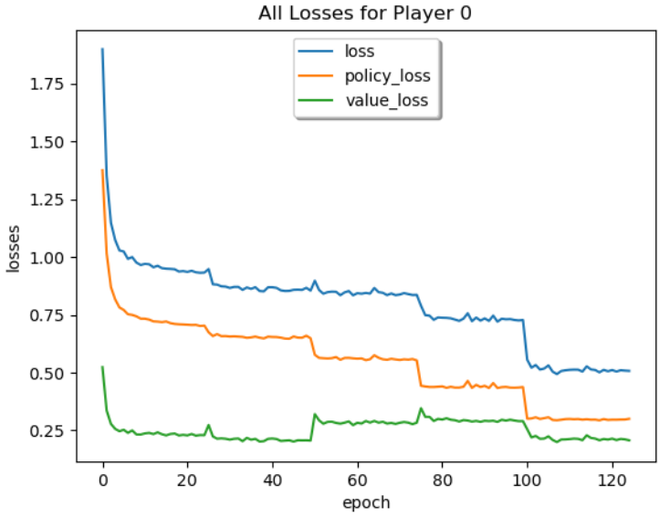
- Verwende nur Sequenz aus bereits besuchten Orten
- In Form einer N x N Matrix
- One-hot Vektoren für besuchte Orte bzw. Null-Vektoren für zukünftige Entscheidungen
- Kosten nur über Berechnung der Gesamtkosten in terminaler Beobachtung

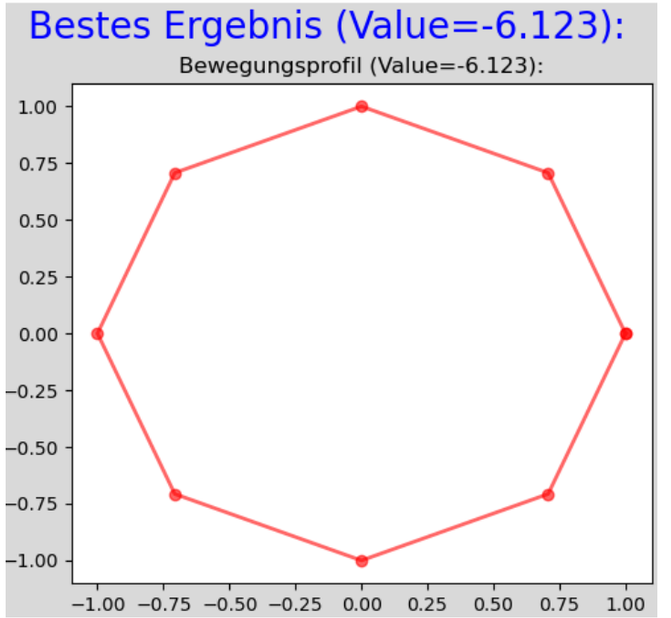
Beispiel: Travelling Salesman (8-Eck)



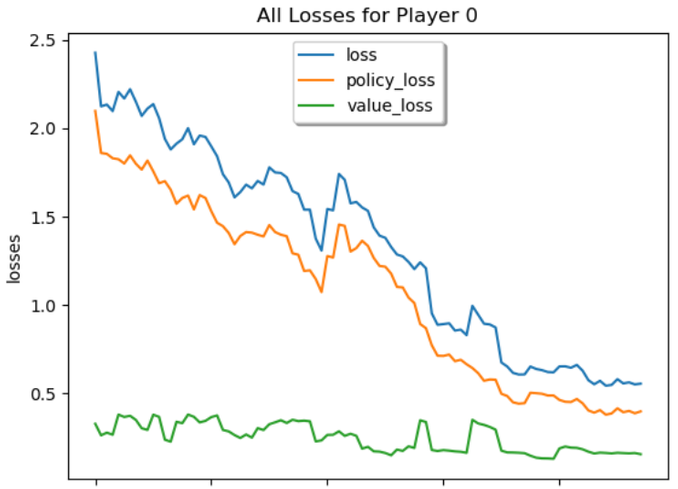


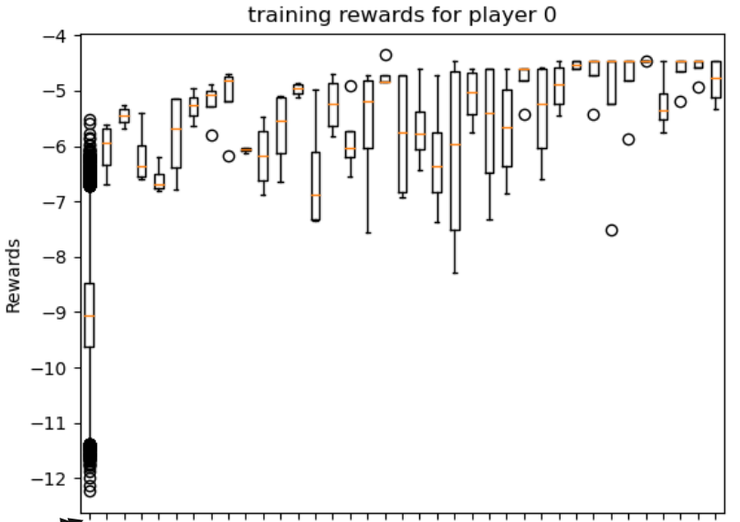


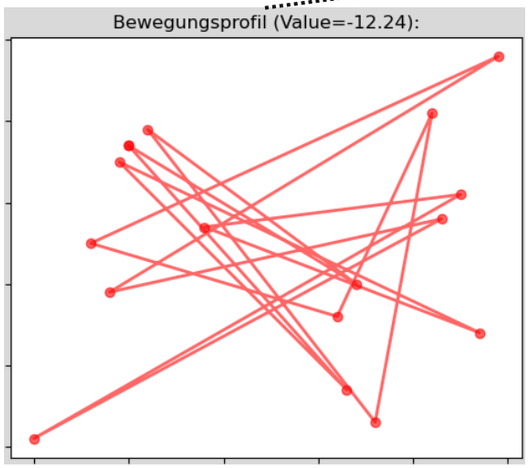


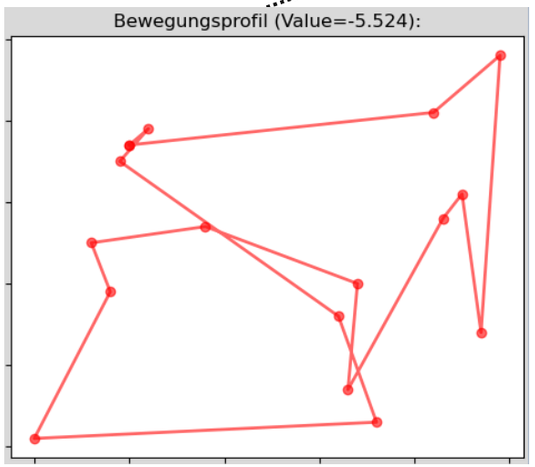


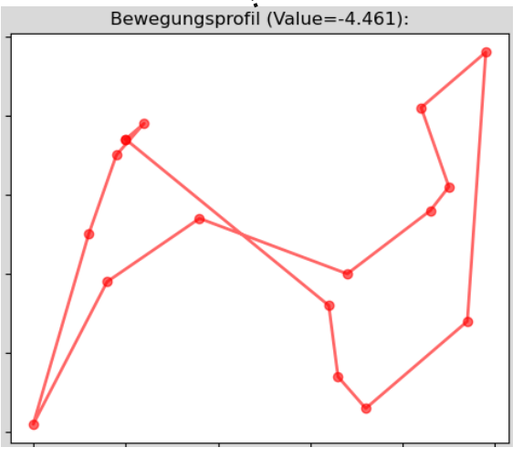
Beispiel: Travelling Salesman (16 zufällige Punkte)











Beispiel: Travelling Salesman

- Bisher Lösung des (für uns relativ einfachen) Travelling Salesman Problem teilweise schwierig für den Agenten
- Beobachtungen bisher nur als Sequenz vergangener Entscheidungen
- Agent hat keinen „Sinn“ für Lage der Orte zueinander
- Benötigt geeignetes Modell, das paarweise Kosten berücksichtigt
- Keine Bestimmung der aktuellen Kosten, sondern nur Bestimmung der Gesamtkosten zum Schluss

Letztlich gewünscht:

- Modell, das auf unterschiedlichen Travelling Salesman Problem trainiert wurde
- Um für ein *neues* Travelling Salesman Problem möglichst schnell zu einem optimalen Ergebnis zu gelangen
- Hier: Eingabe unsortierte Liste an Orten bzw. paarweise Kosten
- Modell sollte in gewisser Weise invariant sein bzgl. Permutationen der Orte

Kontakt

infoteam Software AG
Am Bauhof 9 | 91088 Bubenreuth | Deutschland

Telefon: +49 9131 78 00-0
Telefax: +49 9131 78 00-50
info@infoteam.de | www.infoteam.de