Karlsruher Institut für Technologie

Maschinelles Lernen I - Grundverfahren
V08 Principles of Reinforcement Learning (Wiederholung)

Rec

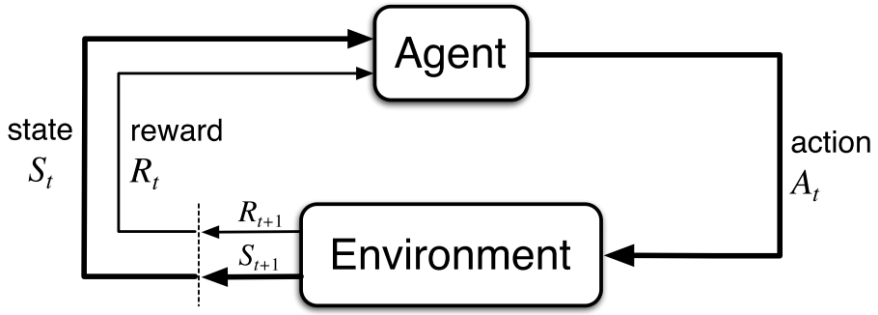
Sommersemester 2018/2019

Prof. Dr. J.M. Zöllner, Karam Daaboul & Karl Kurzer

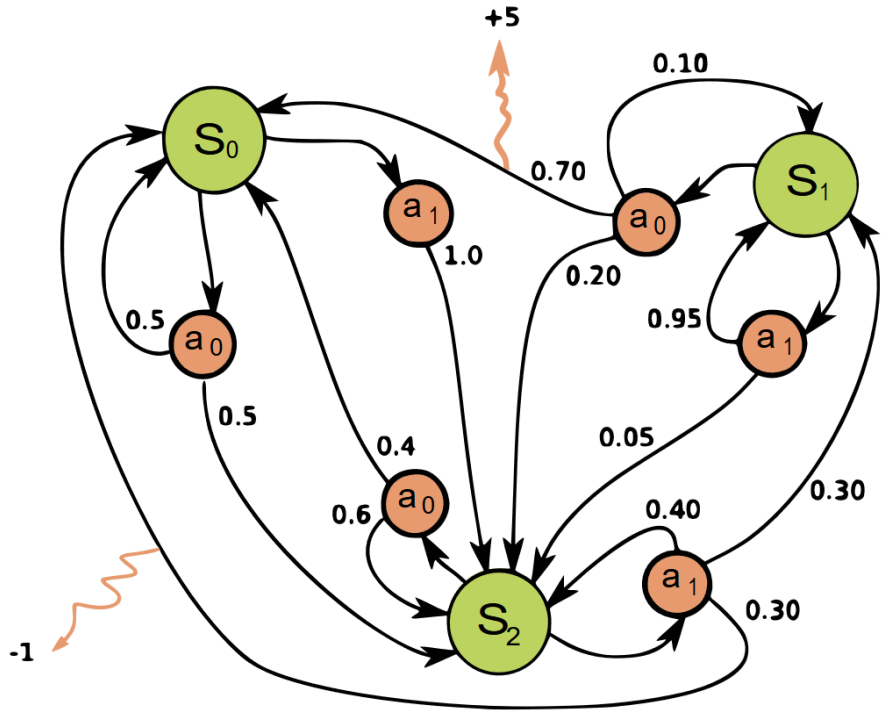
INSTITUT FÜR ANGEWANDTE INFORMATIK UND FORMALE BESCHREIBUNGSVERFAHREN
INSTITUT FÜR ANTHROPOMATIK UND ROBOTIK

Markov'scher Entscheidungsprozess (MDP)

- Formalisierung der sequentiellen Entscheidungsfindung
  - S ist ein endlicher Satz von Zuständen
  - A ist ein endlicher Satz von Aktionen
  - $\mathcal{P}$ ist die Transitionswahrscheinlichkeitsmatrix
    $\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$
  - $\mathcal{R}$ ist die Belohnungsfunktion
    $\mathcal{R}_s^a = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$
  - γ is a discount factor $\gamma \in [0,1]$



Quelle: Sutton2017



Quelle: Wikipedia

Aktionswertfunktion (Action Value Function)

Q-Funktion → $Q(s, a)$

- Die Bewertungsfunktion ist eine Vorhersage der zukünftigen Belohnung
- Bewertet die Güte einer Aktion in einem Zustand

$$Q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in S} \mathcal{P}_{ss'}^a \sum_{a' \in A} \pi(a'|s') Q_\pi(s', a')$$

Definition

Die Aktionswertfunktion eines MDP ist die erwartete kummulierte Belohnung ausgehend vom Zustand s und der Aktion a wenn im folgenden die Strategie π verfolgt wird.

$$Q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma\, Q_\pi(S_{t+1}, A_{t+1}) |\, S_t = s, A_t = a]$$

Optimale Wertfunktionen

- Die optimale Zustandswertfunktion ist der Maximalwert über alle Strategien

$$V^*(s) = \max_\pi V_\pi(s)$$

- Die optimale Aktionswertfunktion ist der Maximalwert über alle Strategien

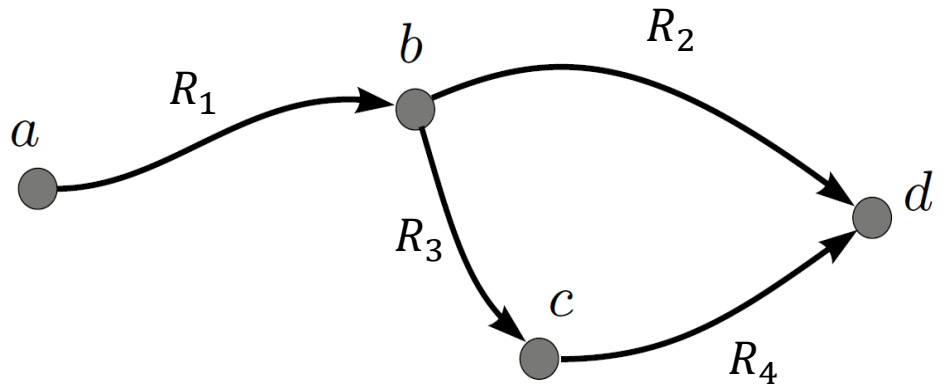
$$Q^*(s,a) = \max_\pi Q_\pi(s,a)$$

- Die optimale Wertfunktion gibt den bestmöglichen Wert eines MDP an
- Ein MDP ist gelöst, wenn die optimale Wertfunktion bekannt ist

Optimalitätsprinzip von Bellman

- Ziel: Berechnung der optimalen Strategie π^* unter Verwendung des Optimalitätsprinzips von Richard E. Bellman:

"An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."



Definition

Wenn der Weg $a \rightarrow b \rightarrow d$ die Verbindung von a nach d mit maximaler Belohnung $R^* = R_1 + R_2$ ist, dann muss $b \rightarrow d$ die Verbindung von b nach d mit maximaler Belohnung sein.

Bellman Optimalitätsgleichung

- Bellman Optimalitätsgleichung für die Zustandswertfunktion

$$V^*(s) = \max_a \left(\mathcal{R}_s^a + \gamma \sum_{s' \in S} \mathcal{P}_{ss'}^a V^*(s') \right)$$

- Bellman Optimalitätsgleichung für die Aktionswertfunktion

$$Q^*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in S} \mathcal{P}_{ss'}^a \max_{a'} Q^*(s', a')$$

Q-Learning

Initialisiere $Q(s,a)\ \forall\ s \in S, a \in A(s)$ zufällig

Wiederhole (für jede Episode):

Initialisiere s

Wiederhole (für jeden Schritt der Episode):

Wähle a von s mit Strategie abgeleitet von Q (z.B. ϵ-greedy)

Nimm Aktion a, beobachte r, s′

$\mathrm{T} = \mathrm{r} + \gamma \max_{a'} Q(\mathrm{s}', \mathrm{a}')$

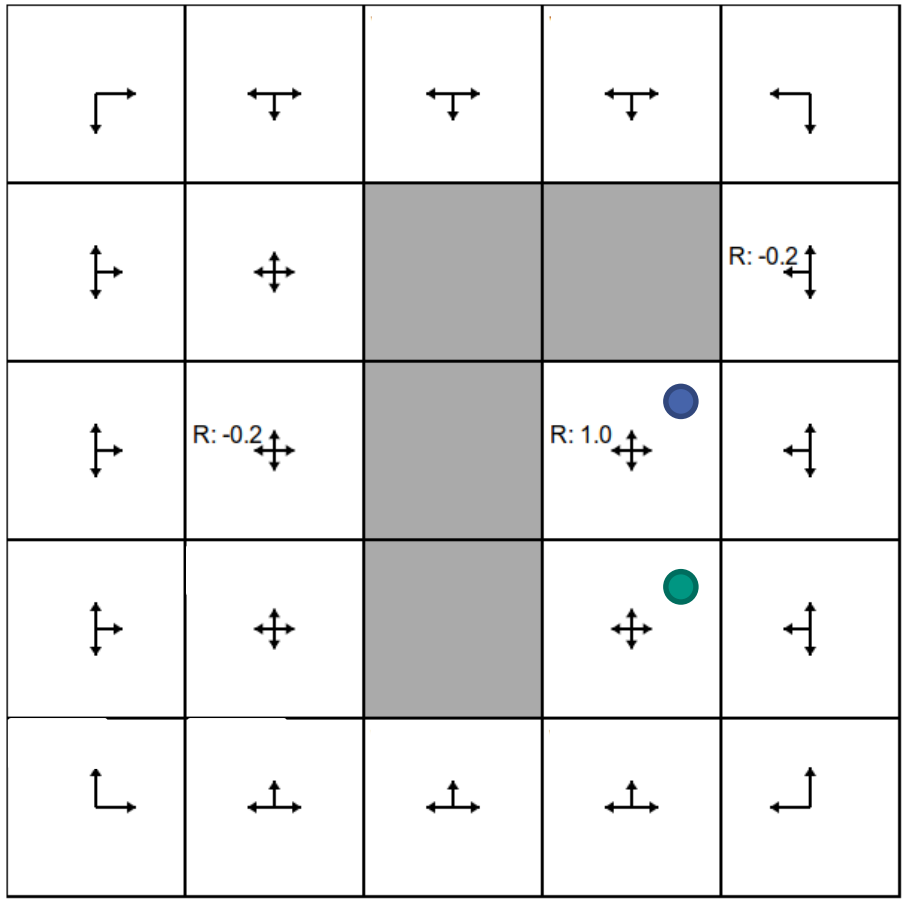
$Q(\mathrm{s},\mathrm{a}) \leftarrow Q(\mathrm{s},\mathrm{a}) + \alpha[\mathrm{T} - Q(\mathrm{s},\mathrm{a})]$

Temporal Difference Error

ϵ-greedy

$$a = \begin{cases} \arg\max_a Q(s,a), & x \sim U[0,1] < \epsilon \\ a \in A(s), & x \sim U[0,1] \geq \epsilon \end{cases}$$

Q-Learning



Wähle a von s mit Strategie abgeleitet von Q (z.B. ϵ-greedy)

Nimm Aktion a, beobachte r, s'

$$T = r + \gamma \max_{a'} Q(s', a')$$

$$Q(s, a) \leftarrow Q(s, a) + \alpha[T - Q(s, a)]$$

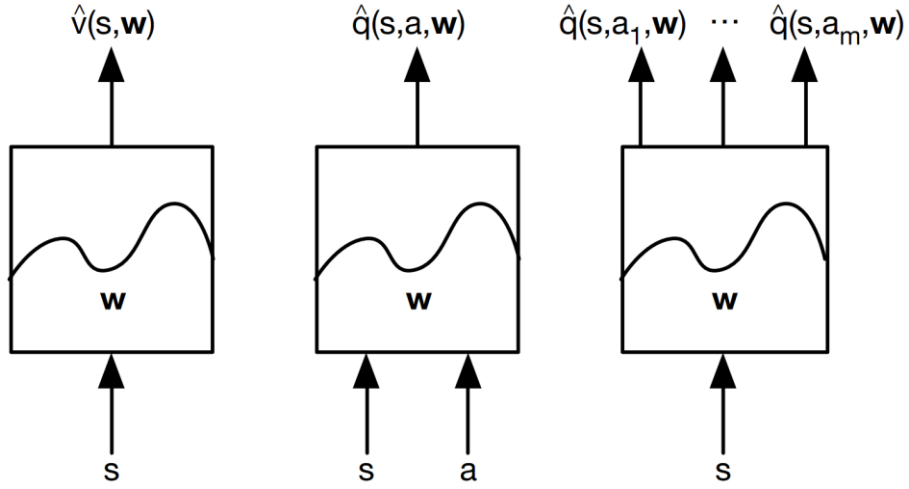
$$T = 0 + 0.9 * 0$$
$$Q(g, up) \leftarrow 0 + 0.1[T - 0]$$

$$T = 1 + 0.9 * 0$$
$$Q(b, left) \leftarrow 0 + 0.1[T - 0]$$

$$T = 0 + 0.9 * 0.1$$
$$Q(g, up) \leftarrow 0 + 0.1[T - 0]$$

RL mit Funktionsapproximation – Deep RL

- Warum?
  - Reinforcement Learning wird häufig auf komplexen Problemen angewandt
    - Backgammon: 10^{20} Zustände
    - Go: 10^{170} Zustände
    - StarCraft II: quasi kontinuierlich
  - Unmöglich alle Werte in Tabelle zu speichern
- Wie?
  - Linearkombination von Merkmalen
  - Neurale Netze



Quelle: Silver2015

Q-Learning mit Funktionsapproximation

Initialisiere $Q_\theta(s,a)\ \forall\ s \in S, a \in A(s)$ zufällig

Wiederhole (für jede Episode):

Initialisiere s

Wiederhole (für jeden Schritt der Episode):

Wähle a von s mit Strategie abgeleitet von Q (z.B. ϵ-greedy)

Nimm Aktion a, beobachte r, s′

$\mathrm{T} = \mathrm{r} + \gamma \max_{a'} Q_\theta(\mathrm{s}', \mathrm{a}')$

$\theta \leftarrow \theta - \alpha \nabla_\theta \mathbb{E}_{s' \sim \mathcal{P}^a_{ss'}}[(\mathrm{T} - Q_\theta(s,a))^2]$

nicht stationär

Korrelation

ϵ-greedy

$$a = \begin{cases} \arg\max_a Q(s,a), & x \sim U[0,1] < \epsilon \\ a \in A(s), & x \sim U[0,1] \geq \epsilon \end{cases}$$

Target Network

- Das Target (Sollwert) wird nicht mit den aktuellen Aktionswerten des Netzes berechnet, sondern mit einer niederfrequent aktualisierten Kopie des Netzes
  - Periodische Aktualisierung des Target Network

$$\mathrm{T} = R + \gamma \max_{a'} Q_{\theta'}(\mathrm{s}', \mathrm{a}')$$

$$\theta \leftarrow \theta - \alpha \nabla_\theta \mathbb{E}_{s' \sim \mathcal{P}_{ss'}^{a}, [}\left(\mathrm{T} - Q_\theta(s, a)\right)^2]$$

- Stabileres Training
  - Bricht die Korrelation von Aktionswert (Q) und Target
  - Resultiert in einer statischeren Verteilung der Labels

Experience Replay

- Ein endlicher Erfahrungsspeicher aus welchem zufällige Erfahrungen (s_t, a_t, r_t, s_{t+1}) gezogen werden um zu lernen
  - Erfahrungen werden zunächst im Erfahrungsspeicher gespeichert
  - Um zu lernen werden Minibatches aus zufälligen Erfahrungen aus dem Erfahrungsspeicher erstellt

- Stabileres Training
  - Bricht die Korrelation von Erfahrungen welche durch Trajektorien entstehen
  - Resultiert in einer statischeren Verteilung der Daten
  - Führt zu besserer Dateneffizienz

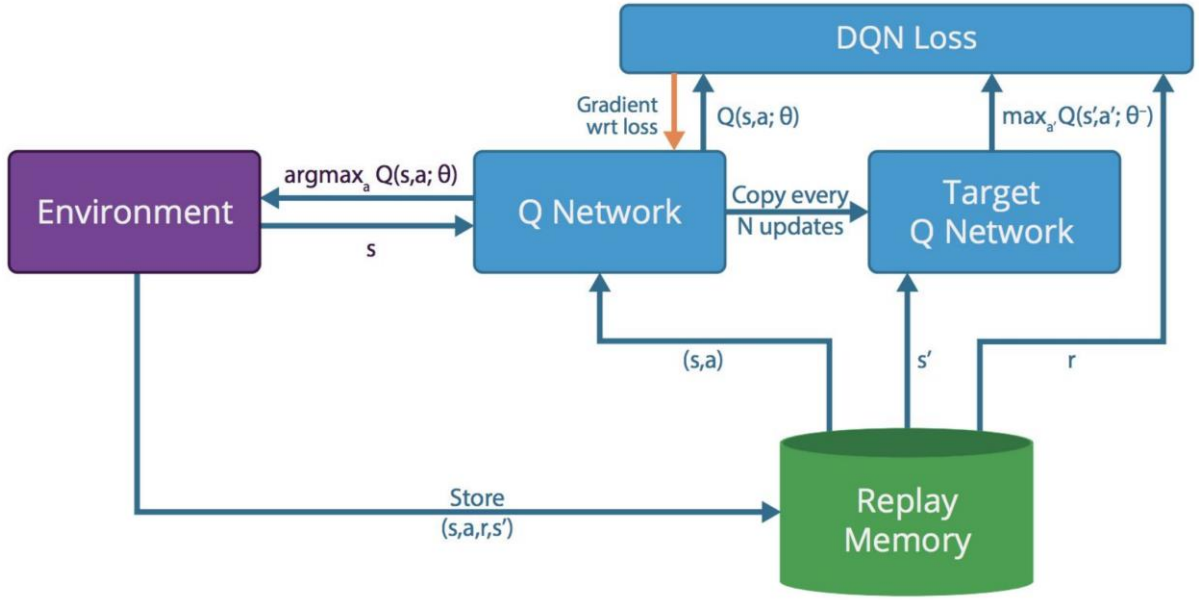
Deep Q-Learning

Initialisiere Erfahrungspuffer D mit Kapazität N
Initialisiere Q_θ mit zufälligen Gewichten θ
Initialisiere $Q_{\theta'}$ mit $\theta' = \theta$
Wiederhole (für jede Episode):
    Initialisiere s
    Wiederhole (für jeden Schritt der Episode):
        Wähle a von s mit Strategie abgeleitet von Q
        Nimm Aktion a, beobachte r, s′
        Speicher Erfahrung (s, a, r, s') in D
        Ziehe eine zufällige Stichprobe von Erfahrungen aus D

$$\mathrm{T} = \begin{cases} r & \textit{Falls die Episode im nächsten Schritt terminiert} \\ \mathrm{r} + \gamma \max_{a'} Q_{\theta'}(\mathrm{s}', \mathrm{a}') & \textit{ansonsten} \end{cases}$$

$$\theta \leftarrow \theta - \alpha \nabla_\theta \mathbb{E}_{s' \sim \mathcal{P}^a_{ss'}} \left[\left(\mathrm{T} - Q_\theta(s, a) \right)^2 \right]$$

        Alle C Schritte $Q_{\theta'} = Q_\theta$



Charakteristika – Q-Learning

- Strategie wird nur implizit gelernt/verbessert
- Lernt deterministische Strategien
- Generell weniger stabil (siehe Erweiterungen)
- Dateneffizient (Off-Policy)
- Diskrete Aktionsräume