Übung zu Organic Computing II

Lernverfahren

David Pätzel

3. Juli 2019

Universität Augsburg
Institut für Informatik
Lehrstuhl für Organic Computing

Maschinelles Lernen

Übliche Einteilung

ORGANIC COMPUTING

- *Supervised Learning* (SL)
- *Reinforcement Learning* (RL)
- *Unsupervised Learning* (UL)

Übliche Einteilung

ORGANIC COMPUTING

- *Supervised Learning* (SL)
  - künstliche neuronale Netze
  - Support-Vector-Machines
  - ...
- *Reinforcement Learning* (RL)
- *Unsupervised Learning* (UL)

Übliche Einteilung

ORGANIC COMPUTING

- *Supervised Learning* (SL)
  - künstliche neuronale Netze
  - Support-Vector-Machines
  - ...
- *Reinforcement Learning* (RL)
  - Monte-Carlo-Methoden
  - Temporal-Difference-Learning
  - ...
- *Unsupervised Learning* (UL)

Übliche Einteilung

ORGANIC COMPUTING

- *Supervised Learning* (SL)
  - künstliche neuronale Netze
  - Support-Vector-Machines
  - ...
- *Reinforcement Learning* (RL)
  - Monte-Carlo-Methoden
  - Temporal-Difference-Learning
  - ...
- *Unsupervised Learning* (UL)
  - Clustering-Algorithmen
  - Autoencoder
  - ...

Übliche Einteilung

ORGANIC COMPUTING

- *Supervised Learning* (SL)
  - künstliche neuronale Netze
  - Support-Vector-Machines
  - ...
- *Reinforcement Learning* (RL)
  - Monte-Carlo-Methoden
  - Temporal-Difference-Learning
  - ...
- *Unsupervised Learning* (UL)
  - Clustering-Algorithmen
  - Autoencoder
  - ...

Übergänge sind z. T. fließend!

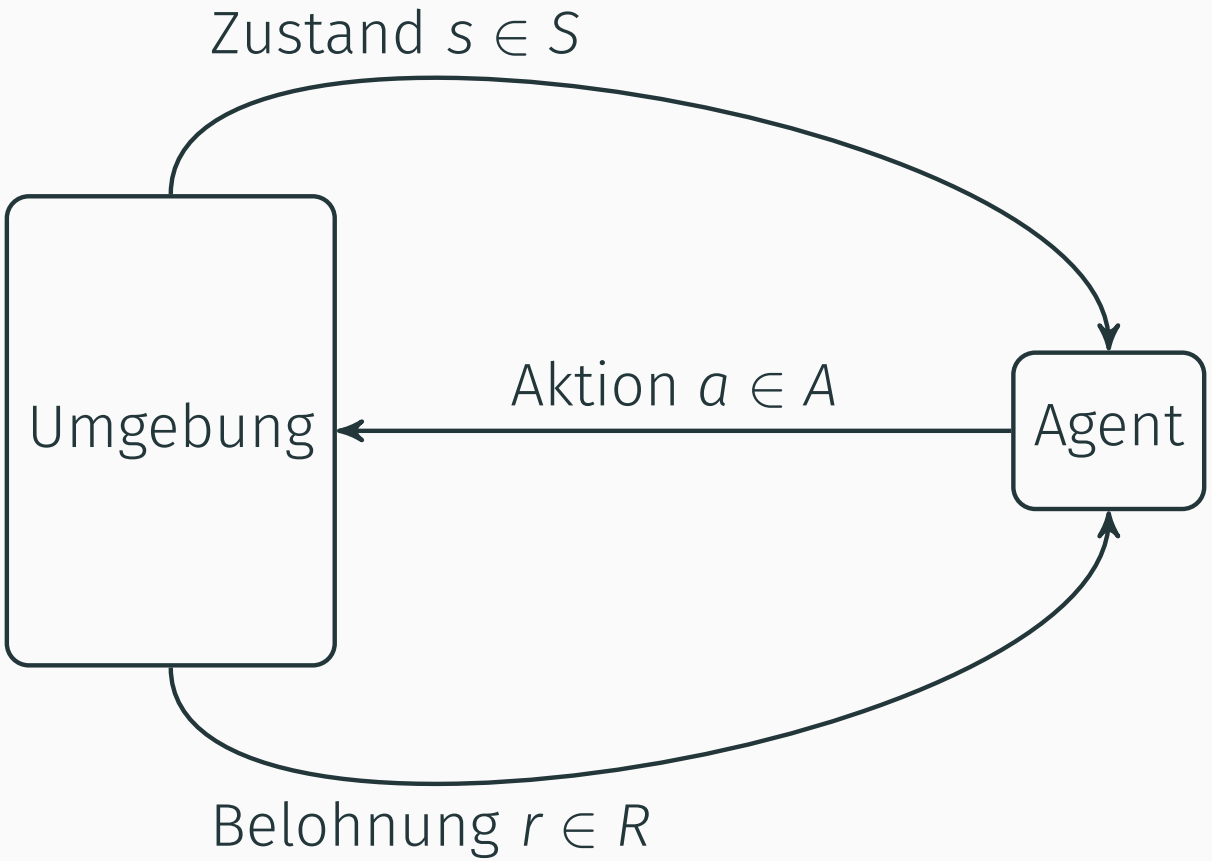
ML und OC

ORGANIC COMPUTING

- OC: „Design zur Laufzeit“ ermöglichen!
  $\Rightarrow$ Online-Learning oft sinnvoll
- viele Selbst-X-Eigenschaften setzen Lernen voraus
- insbesondere wichtig: Lernen durch Interaktion
  $\Rightarrow$ RL!
- aber: oft auch SL und UL relevant

Reinforcement Learning (RL)

Reinforcement-Learning-Modell



Endlichkeit

Meistens gilt (oder wird angenommen)

$$|S| < \infty \text{ und } |A| < \infty$$

Episodische vs. kontinuierliche RL-Probleme

ORGANIC COMPUTING

Definition (Episodische RL-Probleme)

... können in Episoden unterteilt werden.

Nach dem Ende einer Episode: Zurücksetzen der Umgebung in einen Startzustand, unabhängig davon, *wie* die Episode geendet hat.

Definition (Kontinuierliche RL-Probleme)

... sind alle nicht-episodischen RL-Probleme.

Gewinn (Return)

ORGANIC COMPUTING

Definition (Gewinn)

Die über die Zeit *insgesamt* ausgeschüttete Belohnung, diskontiert mit $\gamma \in [0, 1]$:

$$g_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

Ziel des Agenten: Gewinn maximieren!

Weitere Elemente des RL-Modells

ORGANIC COMPUTING

- Policy $\pi : S \to A$: Verhaltensweise des Agenten
- State-Value-Funktion v_π ordnet jedem Zustand den erwarteten Gewinn zu, wenn der Policy π gefolgt wird

$$v_\pi(s) = \mathbb{E}_\pi[g_t \mid s_t = s]$$

- Action-Value-Funktion v_π ordnet jedem Zustand und jeder darin möglichen Aktion den erwarteten Gewinn zu, wenn in diesem Zustand die Aktion gewählt wird und anschließend der Policy π gefolgt wird

$$q_\pi(s, a) = \mathbb{E}_\pi[g_t \mid s_t = s, a_t = a]$$

Lösung eines RL-Problems

ORGANIC COMPUTING

- eine optimale Policy (Verhaltensregel) $\pi_* : S \rightarrow A$
- optimal, wenn $v_\pi(s) = \max_\pi v_\pi(s)$

Was der RL-Agent dabei weiß

ORGANIC COMPUTING

Nichts.

Außer die Menge der Aktionen (er muss sie ja ausführen können) und vielleicht die Menge der Zustände.

Q-Learning

Grundlegendes

ORGANIC COMPUTING

- basierend auf **Tabelle** $Q : S \times A \to \mathbb{R}$
- Garantie: Q konvergiert zu q_{π_*} (gegeben genug Zeit etc.)
  $\Rightarrow$ Optimale Policy wird gefunden!
- Parameter
  - Schrittweite $\alpha \in (0, 1]$
  - Q-Learning-Diskontierungsfaktor $\gamma \in [0, 1]$
  - ggf. $\epsilon \in (0, 1]$

Initialisierung von Q

ORGANIC COMPUTING

- wichtig: $Q(\text{terminal}, a) = 0 \quad \forall a \in A$
- sonst beliebig
- z. B. $Q(s, a) = 0 \quad \forall s \in S, a \in A$

Ablauf

In allen Nicht-Endzuständen s:

1. Wähle Aktion a durch Policy basierend auf Q (z. B. ϵ-greedy).
2. Führe a aus.
3. Beobachte Belohnung r, neuen Zustand s'.
4. Update Q:

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma max_a Q(s',a) - Q(s,a))$$

In Endzuständen: Tue nichts, starte neue Episode.

ϵ-Greedy-Policy basierend auf Q

ORGANIC COMPUTING

- Aktion a mit maximalem $q_{\pi_*}(s, a)$: Bestmögliche Aktion im Zustand s!
- $Q(s, a)$ aktuelle Schätzung von $q_{\pi_*}(s, a)$
  $\Rightarrow \arg\max_a Q(s, a)$ aktuelle Schätzung der besten Aktion im Zustand s
- ϵ-Greedy-Policy basierend auf Q:
  - in $0 \leq \epsilon \leq 1$ Fällen: Wähle Aktion $\arg\max_a Q(s, a)$.
  - ansonsten: Wähle zufällige Aktion.

Q-Wert von Endzuständen

ORGANIC COMPUTING

- Wichtig: Der Q-Wert von Endzuständen bleibt 0!
- … klar, weil keine belohnungsgebende Aktion möglich

Übungsblatt 8

ORGANIC COMPUTING

- OpenAI-Gym
- Q-Learning

Los geht’s!