AraCom IT Services AG

#1 FACH ARTIKEL

WWW.ARACOM.DE

Reinforcement Learning

April 29, 2020

„Verschiedene Kausalitäten beobachten und aus der Konsequenz Verhaltensweisen erlernen. Das ist die Grundlage, um auch bevorstehende unbekannte Probleme zu lösen. Die Natur als Grundlage der Maschine – das ist Intelligenz!“

Reinforcement Learning ist eine Methode, um Software-Agenten das Meistern intellektueller Aufgaben durch Erlernen bestimmter Verhaltensweisen zu ermöglichen. Als Grundlage des Lernens dient die Lernmethode „Versuch und Irrtum“ aus der Natur – diese natürliche Methode wird maschinell nachgebildet. Der Lernprozess ermöglicht Agenten auch zukünftig vorerst unbekannte Probleme durch erlernte Verhaltensweisen zu lösen.

Reinforcement Learning (RL) oder auch Verstärkendes Lernen ist ein Teilgebiet des Machine Learning. Es stellt einer der drei grundlegenden Paradigmen (neben Supervised Learning und Unsupervised Learning) des maschinellen Lernens dar und beschäftigt sich mit der Frage, wie Software-Agenten in einer Umgebung (Environment), in einer spezifischen Situation, agieren (Action) sollten, um die kumulierte Belohnung (Reward) zu maximieren. Das erlernte Verhalten bzw. die Verhaltensregel wird als Policy bezeichnet und dementsprechend strebt Reinforcement Learning stets eine optimale Policy an.

INHALTSVERZEICHNIS

1. Reinforcement Learning vs Supervised Learning

Der Schwerpunkt des Reinforcement Learning liegt auf dem Finden eines Gleichgewichts von der Erforschung neuer Umgebungen und Nutzung des aktualisierten Wissens (State), was das Lösen eines Optimierungsproblems mit Hilfe eines Algorithmus impliziert. Im Gegensatz dazu benötigt das Supervised Learning die Kennzeichnung expliziter Input- und Output-Paare. Ergänzend werden unvorteilhafte Aktionen manuell korrigiert. Somit bedingt Reinforcement Learning das Lernen aus Erfahrungen, welche durch Erforschung sowie Interaktion mit der Umgebung entstehen und nicht aus vorher festgelegten Datensätzen.

REINFORCEMENT LEARNING	VS	SUPERVISED LEARNING
Sequentielle Entscheidungsfindung Der Output hängt von dem Zustand des aktuellen Inputs ab und der nächste Input hängt von dem Output des vorherigen Inputs ab.		**Abhängigkeit der Entscheidungsfindung durch den ersten Input**
Abhängigkeit der Entscheidungen Es werden Sequenzen von abhängigen Entscheidungen gekennzeichnet.		**Unabhängigkeit der Entscheidungen** Jede Entscheidung wird einzeln gekennzeichnet.
Beispiel: Brettspiel Go (siehe AlphaGo Zero)		**Beispiel:** Objekterkennung

Modelle des Supervised Learning bieten somit im Gegensatz zum Reinforcement Learning nur einen geringen Beitrag zu selbstlernenden und autonomen Algorithmen.

2. Funktionsweise des Reinforcement Learning

Der Begriff Reinforcement Learning beinhaltet unterschiedliche Einzelmethoden. In jeder Einzelmethode existiert ein Agent, der selbstständig eine Überlebens- bzw. Erfolgsstrategie erlernt. Dies erfolgt durch den Agenten, welcher eine Reihe von Zuständen und mehrere mögliche Aktionen pro Zustand hat. Die Durchführung einer Aktion während eines Zustandes bietet dem Agenten eine Belohnung. Diese Belohnung kann negativ oder positiv sein. Die Belohnung stellt damit einen Richtwert da, welcher dem Agent zeigt, ob das Verhalten (Policy) in einer bestimmten Situation vorteilhaft oder unvorteilhaft war. Ziel ist, wie im oberen Absatz erwähnt, den kumulierten Wert der Belohnungen zu maximieren.

3. Markow Entscheidungsprozess im Reinforcement Learning

Das Modell der Umgebung wird meistens als Markow Entscheidungsprozess (markov decision process | kurz: MDP) formuliert. Dabei handelt es sich um einen diskreten stochastischen Kontrollprozess. Der mathematische Rahmen ist nützlich zur Untersuchung von Optimierungsproblemen, die durch Reinforcement Learning gelöst werden können. Reinforcement Learning basiert auf Algorithmen der dynamischen Programmierung. Bei der Programmierung entsteht so eine optimale Policy (optimale Strategie bzw. Verhaltensregel), wenn durch Modellierung der Umgebung ein MDP entsteht (siehe Optimalitätsprinzip von Bellman).

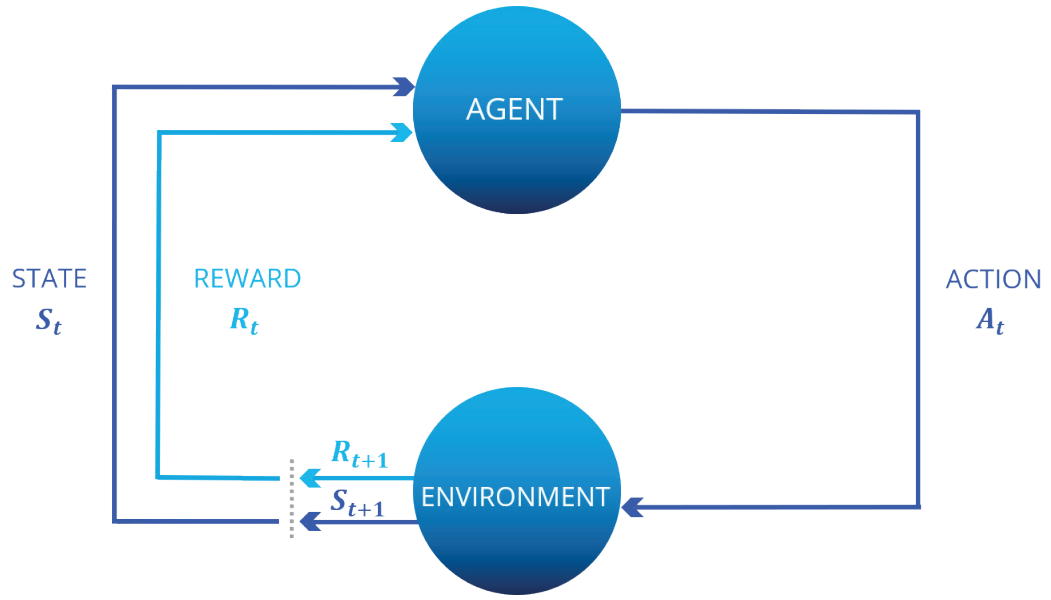

Abbildung: Markow Entscheidungsprozess [2]

4. Modellbasiertes vs. Modellfreies Reinforcement Learning

Im modellbasierten Reinforcement Learning existiert ein Agent, der versucht die modellierte Umgebung zu verstehen und entsprechend zu agieren. Das modellfreie Reinforcement Learning beinhaltet im Umkehrschluss nicht das Erlernen eines Modells, stattdessen wird eine Policy, durch die Nutzung von Algorithmen, wie beispielweise Q-Learning, erlernt.

In folgenden Beispielen wird als Grundlage die optimale Fahrroute zur Arbeit hinzugezogen.

BEISPIEL: MODELLBASIERT	VS	BEISPIEL: MODELLFREI
Es werden Karten und das Stauaufkommen analysiert und daraufhin die Fahrroute von X geplant.		**X hat durch wiederholtes zufälliges Probieren gelernt, dass die A7 morgens stark befahren ist, also erfolgt die Fahrt über die Landstraßen.**

Das Problem wird somit im modellbasierten Reinforcement Learning geplant und im modellfreien Reinforcement Learning lernt der Agent selbst, wie das Problem gelöst wird, ohne das Problem im Vorfeld zu kennen.

5. On-Policy vs Off-Policy im Reinforcement Learning

Policy kann allgemein als Regelwerk bezeichnet werden, welche sowohl bei On-Policy und Off-Policy Algorithmen vorhanden ist. Die Unterscheidung erfolgt durch die Art und Weise, wie Aktionen ausgewählt werden. Bei On-Policy wird sowohl der nächste Zustand als auch die aktuelle Aktion bei der Auswahl berücksichtigt. Bei Off-Policy wird nur der nächste Zustand berücksichtigt und der Algorithmus wählt eine Greedy-Action (Aktion) aus. Greedy: Wählt die Aktion, die auf Basis des momentanen Wissens in der aktuellen Situation am Besten abgeschnitten hat.

ON-POLICY	VS	OFF-POLICY
Algorithmen berücksichtigen den nächsten Zustand und die aktuelle Aktion.		**Algorithmen berücksichtigen den nächsten Zustand und wählen eine greedy Action.**

6. Reinforcement Learning Algorithmus: Q-Learning

Q-Learning ist ein modellfreier Off-Policy Reinforcement Algorithmus, um Regeln zu erlernen. Dies erfolgt iterativ, indem einem Agenten Aktionen in bestimmten Situationen gelehrt werden. Wie bereits erwähnt, ist hierfür kein Modell der Umgebung nötig. Der Algorithmus basiert auf dem Temporal Difference Learning.

Der Algorithmus beinhaltet eine Funktion, welche den Wert der Qualität (Q) der Zustand-Aktion Kombination aktualisiert – als Indikator für die kumulierte Belohnung – ferner ist die Basis des Algorithmus eine Gleichung nach dem Optimalitätsprinzip von Bellman:

$$Q^{neu}(S_t, A_t) \leftarrow \underbrace{Q(S_t, A_t)}_{\text{ALTER WERT}} + \overbrace{\alpha}^{\text{LERNRATE}} \cdot \overbrace{\Big(\underbrace{R_t}_{\text{BELOHNUNG}} + \overbrace{\gamma}^{\text{DISKONTIERUNGSFAKTOR}} \cdot \underbrace{\max_A Q(S_{t+1}, A)}_{\text{GESCHÄTZTER OPTIMALER ZUKÜNFTIGER WERT}} - Q(S_t, A_t)\Big)}^{\text{ZEITLICHER UNTERSCHIED}}$$

Abbildung: Q-Learning Funktion

Das **Q** wird im Vorfeld von dem Programmierer mit einem fixen Wert initialisiert. Dies erfolgt noch vor dem tatsächlichen Lernen. Das Lernen beginnt und zum Zeitwert **t** wählt der Agent eine Aktion **A(t)** und erhält so seine Belohnung **R(t)**. Daraufhin befindet sich der Agent im neuen Zustand **S(t+1)**, woraufhin das **Q** aktualisiert wird **Q(neu)**. Der neue Zustand hängt normalerweise von dem vorherigen Zustand **S(t)** und der gewählten Aktion **A(t)** ab. Da es sich bei Q-Learning um einen iterativen Algorithmus handelt, wiederholt sich dieser Vorgang so lange, bis das Lernen gestoppt wird.

7. Reinforcement Learning Algorithmus: Sarsa

Sarsa ist ein modellfreier On-Policy Reinforcement Algorithmus. Der Algorithmus ist die On-Policy Variante des Q-Learning Algorithmus. Wie der Q-Learning Algorithmus basiert Sarsa auf dem Temporal Difference Learning.

Die Funktion zur Auswertung des Q-Values von Sarsa ist daher nur geringfügig abweichend zu der Q-Learning Funktion:

$$Q^{neu}(S_t, A_t) \leftarrow \underbrace{Q(S_t, A_t)}_{\text{ALTER WERT}} + \overbrace{\alpha}^{\text{LERNRATE}} \cdot \overbrace{\Big(\underbrace{R_t}_{\text{BELOHNUNG}} + \overbrace{\gamma}^{\text{DISKONTIERUNGSFAKTOR}} \cdot \underbrace{Q(S_{t+1}, A_{t+1})}_{\text{GESCHÄTZTER ZUKÜNFTIGER WERT}} - Q(S_t, A_t)\Big)}^{\text{ZEITLICHER UNTERSCHIED}}$$

Abbildung: SARSA Funktion

8. Anwendungsbereiche des Reinforcement Learning

Aktuell wird insbesondere im Spiele-Bereich oder Denksport das Reinforcement Learning genutzt, da die Trainings in diesem Fall keine enormen finanziellen Risiken bergen. Dabei werden auch viele unterschiedliche Methoden miteinander kombiniert, um das bestmögliche Ergebnis zu erzielen.

Im Folgenden werden drei Anwendungsbereiche für Reinforcement Learning vorgestellt – diese sind lediglich ein kleiner Ausschnitt der Vielzahl von möglichen Anwendungsbereichen.

8.1 AlphaGo und AlphaGo Zero

AlphaGo und AlphaGo Zero sind Computerprogramme, welche das japanische Brettspiel „Go" spielen. Aufgrund der maximalen Feldgröße von 19x19 bietet Go eine enorme Anzahl an gültigen Konstellationen – bedeutend mehr als bei Schach. Mit der enormen Anzahl an gültigen Spielzügen ist es für die derzeitige Technologie unmöglich den kompletten Spielbaum von Go zur Verfügung zu stellen, wodurch neben Supervised Learning auch das Reinforcement Learning zum Trainieren genutzt wurde. Die Weiterentwicklung von AlphaGo zu AlphaGo Zero, mit Hilfe des reinen Reinforcement Learning Ansatzes, ermöglichte es dem Programm das mehrfache Schlagen des Go-Weltmeisters und weiterer Profi-Go-Spieler. Reinforcement Learning ermöglichte dem Programm so das Erlernen vorerst unbekannter Spielzüge und das Trainieren mit sich selbst [3].

8.2 Optimierung chemischer Reaktionen

Mit einem Modell zur Optimierung chemischer Reaktionen mit Hilfe von Reinforcement Learning befassten sich Zhou, Li und Zare. Ihr Modell zeigt, dass ein Reinforcement Learning Algorithmus einen enormen Mehrwert in der Chemie haben könnte. Der Agent des Algorithmus optimierte die chemische Reaktion mit dem Markow Entscheidungsprozess. Der Zustand beinhaltete die Menge unterschiedlicher chemischer Bedingungen, wie der pH-Wert oder die Temperatur. Die Aktionen hingegen die Menge aller potenziellen Aktionen, welche die Bedingungen aus dem Zustand ändern könnten. Dieses Anwendungsbeispiel zeigt, wie Reinforcement Learning den Zeitaufwand von Versuch und Irrtum-Arbeiten reduzieren kann [4].

8.3 Ampel-Steuerung

Zur Lösung von Stauproblemen haben Forscher eine Ampel-Steuerung entworfen, welche in einer simulierten Umgebung getestet wurde. Um ein Verkehrsnetz abzubilden wurde ein Multi-Agenten System entwickelt. Mit Hilfe von Reinforcement Learning konnte so das Zusammenspielen der Agenten Verkehrssignale so steuern, dass der Verkehr signifikant flüssiger war. Als Aktionen standen den Agenten acht Möglichkeiten zur Verfügung, welche die Phasenkombinationen darstellten. Zur Belohnung wurde der Wert der Verringerung der Verspätungen genutzt [5] [6].

9. Grundvoraussetzung zur Verwendung von Reinforcement Learning

Damit Reinforcement Learning angewendet werden kann, müssen folgende Kriterien erfüllt sein:

Verständnis für das zu lösende Problem

Reinforcement Learning ist nicht die Antwort auf jedes Problem. Manche Probleme lassen sich einfacher lösen und andere können mit RL gar nicht gelöst werden.

- Lässt sich das Problem mit der Versuch und Irrtum Methode lösen?
- Könnte der Agent durch Interaktion mit der Umgebung Verhaltensstrukturen erlernen?
- Lässt sich das Verhalten belohnen?
- Kann das Problem zu einem Markowschen Entscheidungsprozess modelliert werden?

Simulierte Umgebung

Eine reale Umgebung sollte sich als abstraktes Modell simulieren lassen können.

Qual der Wahl: Der Algorithmus

Es gibt eine Vielzahl unterschiedlicher Reinforcement Learning Algorithmen. Soll der Algorithmus On-Policy oder Off-Policy sein? Sollte der Algorithmus modellbasiert oder modellfrei sein? Q-Learning oder doch Sarsa? Welcher Algorithmus löst mein Problem am effizientesten?

10. Fazit zu Reinforcement Learning

Reinforcement Learning zeigt, dass es viele potenzielle Anwendungsbereiche gibt. Aktuell sind für diese Methode viele Ressourcen nötig, jedoch wird die Zukunft zeigen, dass auch hier der Ressourcenverbrauch und dementsprechend die Kosten verringert werden können. Wir sind gespannt, welchen Mehrwert das Reinforcement Learning noch für unsere Gesellschaft birgt.

Quellen:

1. Bengio Y., Courville C., Goodfellow I. (2016). Deep Learning.
2. Barto A.G., Sutton R.S. (2014). Reinforcement Learning: An Introduction. PDF
3. Antonoglou I., Baker L., Bolton A., Chen Y., Graepel T., Guez A., Hassabis D., Huang A., Hubert T., Hui F., Lai M., Lillicrap T., Schrittwieser J., Sifre L., Silver, D., Simonyan K., van den Driessche, K. (2017). Mastering the game of Go without human knowledge.
4. Li X., Zare R.N., Zhou Z. (2017). Optimizing Chemical Reactions with Deep Reinforcement Learning.
5. Arel I., Kohls A.G., Liu C., Urbanik T. (2010). Reinforcement learning-based multi-agent system for network traffic signal control.
6. Sommer M. (2019). Resilient Traffic Management.