bestärkendes Lernen (RL)

Autor und Herausgeber

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland (ORCID 0000-0001-7538-0990)
Editors: Konstantina Olioumtsevits (ORCID 0000-0003-2697-8659), Ekkehard Schnoor (ORCID 0000-0001-8252-1499), Aalto University, Espoo, Finland
*Corresponding author: alex.jung@aalto.fi

Zusammenfassung

Bestärkendes Lernen (RL) ist die Aufgabe, zu lernen, optimal innerhalb einer environment zu agieren. Die Aktionen werden von einem Agent basierend auf Beobachtungen über die environment ausgeführt. Das mathematische Standardmodell der Interaktion zwischen Agent und environment ist ein Markov decision process (MDP). Ein Markov decision process (MDP) umfasst eine transition function, die conditional probability distribution des nächsten Zustand gegeben den aktuellen Zustand und die Aktion. Eine weitere Komponente eines Markov decision process (MDP) ist eine Belohnung-Funktion, die jedes Paar aus Zustand und Aktion auf einen Belohnung-Wert abbildet, der als negativer Verlust-Wert interpretiert werden kann. RL unterscheidet sich vom überwachtes Lernen im Aufbau der Verlustfunktion: Bei jedem Zeitschritt wird die Belohnung nur für die ausgeführte Aktion beobachtet, und sie hängt von früheren Aktionen ab, was einen Kompromiss zwischen der Erkundung neuer Aktionen und der Ausnutzung der bisher besten erzwingt.

Definition

Man betrachte einen Roboter auf der 4×4 gridworld von Abb. 1: ein Gitter aus Zellen, wobei die Zelle in Zeile r und Spalte c als $\langle r, c \rangle$ geschrieben wird, mit einer Wand bei $\langle 2, 2 \rangle$ und einem Ziel bei $\langle 3, 3 \rangle$. Bei jedem Zeitschritt bewegt sich der Roboter nach oben, unten, links oder rechts. Eine Bewegung in die Wand oder die Begrenzung lässt ihn an Ort und Stelle. Jede Bewegung liefert eine Belohnung von -1, während die Bewegung, die das Ziel erreicht, eine Belohnung von $+1$ liefert und zu einem Stillstand führt. RL ist die Aufgabe, aus solcher Interaktion zu lernen, Aktionen zu wählen, die schließlich das Ziel erreichen. Dem Lernenden wird nicht gesagt, welche Aktion korrekt ist, sondern er muss die Aktionen entdecken, die die größte Belohnung liefern, indem er sie ausprobiert, und eine Aktion beeinflusst nicht nur die unmittelbare Belohnung, sondern auch den nächsten Zustand und dadurch alle späteren Belohnungen. Diese beiden Eigenschaften, die Versuch-und-Irrtum-Suche und die verzögerte Belohnung, unterscheiden RL von anderen Lernformen (Sutton and Barto, 2018, Sect. 1.1).

Das mathematische Standardmodell einer RL-Anwendung ist ein Markov decision process (MDP) (Sutton and Barto, 2018, Sect. 1.1): ein state space $\mathcal{S}$,

ein action space $\mathcal{A}$, die conditional probability distribution des nächsten Zustand und der Belohnung gegeben den aktuellen Zustand und die Aktion, und ein Diskontierungsfaktor $\gamma \in [0, 1]$. Die conditional probability distribution des nächsten Zustand allein ist die transition function. Ein Agent interagiert mit seiner environment in Zeitschriten $t = 1, 2, \ldots$: er beobachtet den Zustand s_t, wählt eine Aktion a_t und erhält die Belohnung r_{t+1} und den nächsten Zustand s_{t+1} (Sutton and Barto, 2018, Sect. 3.1). Die Aktionen werden durch eine Policy (bestärkendes Lernen) π gewählt, die für eine stochastic Policy (bestärkendes Lernen) eine conditional probability distribution $\mathbb{P}^{(a|s)}$ über den Aktionen gegeben den Zustand ist. Die zu maximierende Größe ist der Return, die diskontierte Summe der zukünftigen Belohnungen, $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$, wobei der Diskontierungsfaktor γ spätere Belohnungen weniger gewichtet (Sutton and Barto, 2018, Sect. 3.3).

Die state-value function einer Policy (bestärkendes Lernen),

$$v_\pi(s) := \mathbb{E}_\pi\big\{G_t \mid s_t = s\big\},$$

ist der erwartete Return aus einem Zustand, wenn die Aktionen durch π gewählt werden (Sutton and Barto, 2018, Sect. 3.5). Eine optimale Policy (bestärkendes Lernen) π_* ist eine, deren state-value function mindestens der jeder anderen Policy (bestärkendes Lernen) in jedem Zustand entspricht (Sutton and Barto, 2018, Sect. 3.6). In der gridworld mit $\gamma = 1$ bringt eine kürzeste Route von d Zügen -1 für jeden ihrer ersten $d - 1$ Züge und $+1$ für den letzten, sodass $v_{\pi_*}(s) = -(d-1) + 1 = 2 - d$.

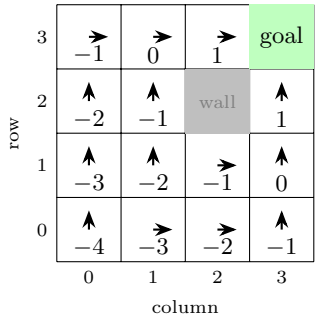


Abbildung 1: Die 4×4 gridworld des Eröffnungsbeispiels: eine Wand bei $\langle 2, 2\rangle$, das Ziel bei $\langle 3, 3\rangle$, Belohnung -1 pro Zug und $+1$ für den Zug, der das Ziel erreicht. Jede Zelle zeigt die optimale state-value function $v_{\pi_*}(s)$, die 2 minus der Anzahl der Züge zum Ziel entspricht, und den Pfeil der optimalen Policy (bestärkendes Lernen) π_*. Beide werden aus den Belohnungen der ausprobierten Züge gelernt, ohne dass mitgeteilt wird, wie ein Zug den Zustand ändert. Daten generiert von `pythondemos/reinforcementlearning.py`

Als Spezialfall des machine learning (ML) trainiert RL auch ein Modell auf Daten: eine Policy (bestärkendes Lernen) ist die Hypothese von RL. Die beiden

oben genannten Eigenschaften unterscheiden es vom überwachtes Lernen. Die erste, die Versuch-und-Irrtum-Suche, ist der Aufbau der Verlustfunktion. Eine überwachtes Lernen-Methode besitzt das Label jedes Datenpunkt und kann den Verlust jeder Kandidaten-Hypothese und ihrer Vorhersagen bewerten. Eine RL-Methode kann den Verlust, die negative Belohnung, nur für die Aktion bewerten, die sie ausgeführt hat, da die Belohnungen der nicht ausgeführten Aktionen nicht beobachtet werden. Die zweite, die verzögerte Belohnung, ist, dass der Verlust zum Zeitschritt t von den vorherigen Aktionen abhängt. RL ist auch kein unüberwachtes Lernen: Es maximiert ein Belohnung-Signal, anstatt Struktur in ungelabelten Datenpunkte zu finden (Sutton and Barto, 2018, Sect. 1.1).

Innerhalb von RL selbst unterscheiden sich die Methoden in der Menge der vorab benötigten Informationen über die environment, von einem bekannten MDP bis hin zu nichts außer den beobachteten Belohnungen (Sutton and Barto, 2018, Sect. 1.3). Wenn das MDP bekannt ist, berechnen Methoden wie die value iteration und die policy iteration eine optimale Policy (bestärkendes Lernen), bevor der Agent sie ausführt (Sutton and Barto, 2018, Ch. 4). Eine Policy (bestärkendes Lernen) ist gierig bezüglich einer value function v, wenn sie in jedem Zustand $s \in \mathcal{S}$ die erwartete unmittelbare Belohnung plus den diskontierten Wert des nächsten Zustand maximiert. Sie wählt daher eine Aktion in $\arg\max_{a \in \mathcal{A}} \mathbb{E}\{r_{t+1} + \gamma v(s_{t+1}) \mid s_t = s, a_t = a\}$ (Sutton and Barto, 2018, Sect. 4.2). Value iteration und policy iteration nutzen zwei Fakten. Die state-value function einer Policy (bestärkendes Lernen) ist der Fixpunkt ihres Bellman operator, den die iterative policy evaluation von jeder Start-value function aus erreicht (Sutton and Barto, 2018, Sect. 4.1). Und die Policy (bestärkendes Lernen), die gierig bezüglich v_π ist, ist mindestens so gut wie π (Sutton and Barto, 2018, Sect. 4.2).

Wenn das MDP nicht bekannt ist, muss eine RL-Methode aus der Interaktion selbst lernen, und jede Belohnung, die sie beobachtet, ist die einer Aktion, die sie ausgeführt hat. Dies erzwingt einen Kompromiss zwischen Exploration, dem Ausprobieren von Aktionen, deren Belohnung unbekannt ist, und Exploitation, der Wahl der bisher als besten befundenen Aktionen (Sutton and Barto, 2018, Sect. 1.1). Q-learning lernt die Aktion-value Funktion $q_{\pi_*}(s, a)$, den erwarteten Return des Ausführens von a in s und des anschließenden Folgens von π_*, aus den Belohnungen der Aktionen, die es ausprobiert, und aktualisiert nach jedem Zeitschritt (Sutton and Barto, 2018, Sect. 6.5). Andere Methoden lernen die Policy (bestärkendes Lernen) direkt (Sutton and Barto, 2018, Ch. 13) oder schätzen ein Modell der environment und planen damit (Sutton and Barto, 2018, Ch. 8).

Zwei Spezialfälle des MDP entfernen die verzögerte Belohnung und lassen nur den Kompromiss zwischen Exploration und Exploitation (Sutton and Barto, 2018, Sect. 2.1). Der erste hat einen einzelnen Zustand: Die transition function liefert diesen Zustand unabhängig von der Aktion, sodass der Return eine Summe von Belohnungen ist, die jeweils nur von der aktuellen Aktion abhängen. Der Einzel-Zustand-Fall ist der multiarmed bandit (MAB) (Sutton and Barto, 2018, Ch. 2). Ein Ad-Server, der eines von zehn Bannern anzeigt und die Klicks des angezeigten Banners beobachtet, aber nicht die der neun zurückgehaltenen,

steht vor diesem Fall (Bubeck and Cesa-Bianchi, 2012). Die upper confidence bound (UCB)-Methode handhabt dies, indem sie zum Stichprobenmittel jeder Aktion einen Konfidenzterm hinzufügt, der mit der Häufigkeit schrumpft, mit der die Aktion gewählt wurde (Auer et al., 2002, Thm. 1; Bubeck and Cesa-Bianchi, 2012).

Der zweite Fall behält mehrere Zustände bei, zieht aber s_t aus einer festen Wahrscheinlichkeitsverteilung, die nicht von den ausgeführten Aktionen abhängt. Die Aktion zum Zeitpunkt t ändert dann die Belohnung r_{t+1} und nichts anderes, und der Zustand wird als Kontext bezeichnet (Sutton and Barto, 2018, Sect. 2.9). Der zweite Fall ist der contextual bandit. Die Lernaufgabe ist es, eine Hypothese zu lernen, die einen Kontext auf eine Aktion abbildet, was die Form eines überwachtes Lernen-Problems hat, außer dass der Verlust immer noch nur für die ausgeführte Aktion beobachtet wird. Nachrichtenempfehlung ist ein Beispiel: Der Kontext ist das Besucherprofil, die Aktion ist der auf der Startseite platzierte Artikel, und die Belohnung ist der Klick (Li et al., 2010).

RL ist oft eine Form des Online-Lernen: Der Agent lernt, während er agiert, und die Datenpunkte, aus denen er lernt, kommen einzeln pro Zeitschritt an und hängen von seiner aktuellen Policy (bestärkendes Lernen) ab (Sutton and Barto, 2018, Sect. 1.1). Offline-RL lernt stattdessen eine Policy (bestärkendes Lernen) aus einem festen Datensatz aufgezeichneter Interaktionen, ohne während des Training in der environment zu agieren (Levine et al., 2020). Die Exploration wird dann von demjenigen festgelegt, der den Datensatz gesammelt hat.

Siehe auch: MDP, Policy (bestärkendes Lernen), Belohnung, Return, value function, state-value function, gridworld, Agent, environment, Bellman operator, Q-learning, policy iteration, value iteration, MAB, contextual bandit, UCB, Online-Lernen, überwachtes Lernen, unüberwachtes Lernen.

Querverweise

Literatur

1. Auer et al. (2002). Finite-time Analysis of the Multiarmed Bandit Problem. Machine Learning, 47, pp. 235–256.

2. Bubeck and Cesa-Bianchi (2012). Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. Found. Trends Mach. Learn., 5(1), pp. 1–122.

3. Levine et al. (2020). Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. URL: https://arxiv.org/abs/2005.01643.

4. Li et al. (2010). A Contextual-Bandit Approach to Personalized News Article Recommendation. In: Proc. International World Wide Web Conference, pp. 661–670.

5. Sutton and Barto (2018). Reinforcement Learning: An Introduction. 2nd ed., MIT Press, Cambridge, MA, USA.