强化学习理论之无模型的方法

周家豪

前瞻跨媒体实验室
计算技术研究所 ICT
中国科学院大学

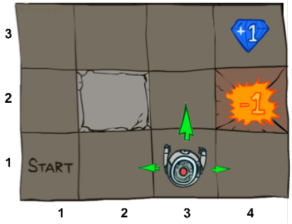
jackhzhou@hotmail.com

2019 年 4 月 14 日

目录

形式化-奖励

"Reinforcement Learning focused on **goal-directed** learning from **interaction**."



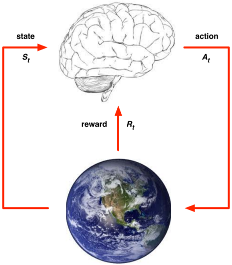
Goal-directed - 奖励假设

"All goals can be described by the **maximisation** of **expected cumulative reward**."

目标可以表示为**奖励**的累积。

形式化-状态，行为

"Reinforcement Learning focused on **goal-directed** learning from **interaction**."

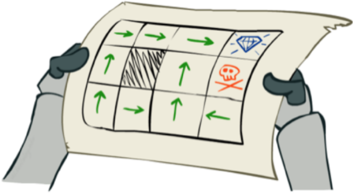


Interaction - 交互建模

智能体看到环境**状态** S^t，做出**行为** A^t，得到 **奖励** R^t，并到新环境状态 S^{t+1}，；
奖励度量智能体在状态下行为的好坏。

形式化-策略

"Reinforcement Learning focused on **goal-directed** learning from **interaction**."



π

Solution - 求解目标

策略 $\pi(s) = a$ 是智能体在状态 s 选择行为 a 能得到最大累积奖励 $\sum_t R^t$。

马尔科夫决策过程

Markov Decision Process

Markov Decision Process(MDP) is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$.

Markov Property

$P(S^{t+1}|S^t, S^{t-1}, \ldots) = P(S^{t+1}|S), S \in \mathcal{S}$

1. $\mathcal{S}$ 环境的状态空间，
2. $\mathcal{A}$ 智能体的行为空间，
3. $\mathcal{P}$ 状态转移函数 $\mathcal{P}^a_{ss'} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$，
4. $\mathcal{R}$ 奖励函数 $\mathcal{R}^a_s = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$，
5. γ 折扣因子 $\gamma \in [0, 1]$.

模型 - 马尔科夫决策过程的状态转移函数 $\mathcal{P}$ 和奖励函数 $\mathcal{R}$。

状态值函数

最优策略 $\rightarrow$ 定义状态值函数评估最优策略。

状态值函数

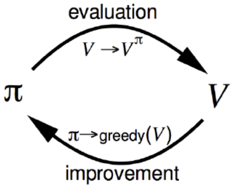
$$v_\pi(s) = \mathbb{E}_\pi[\sum_t \gamma^t R^t | S_0 = s]$$

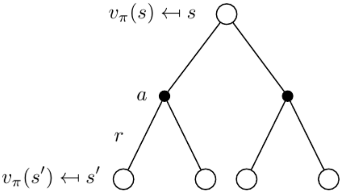
最优策略

策略偏序 - $\pi \geq \pi'$if $v_\pi(s) \geq v'_\pi(s), \forall s$
最优策略等价性 - $v_{\pi^*}(s) = v^*(s)$

策略迭代方法

策略迭代 - 估计 $+$ 控制 (提升) $\rightarrow$ 迭代策略求解。





$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_0 = s](\text{贝尔曼期望方程})$$
$$= \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a V_\pi(s')$$

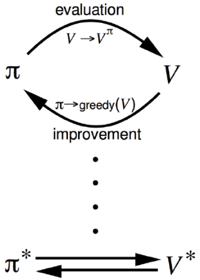
策略迭代算法

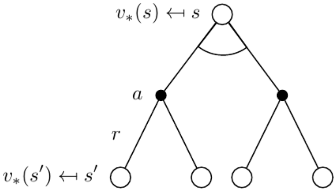
Data: MDP$\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$
Result: $\pi^*(s)$ for all s
For each state s, initialize $\pi(s)$ randomly;
repeat
  For each state s, initialize $v_\pi(s) \leftarrow 0$;
  /* 值估计 */
  repeat
    foreach s **do**
      $v_\pi(s) \leftarrow \mathcal{R}_s^{\pi(s)} + \sum_{s'} \gamma \mathcal{P}_{ss'}^{\pi(s)} v_\pi(s')$
    end
  until *until* $v_\pi(s)$*'s converge*;
  /* 策略提升（控制） */
  foreach s **do**
    $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} \mathcal{R}_s^a + \sum_{s'} \gamma \mathcal{P}_{ss'}^a v_\pi(s')$
  end
until *until* $\pi(s)'s$ *converge*;

Algorithm 1: 策略迭代

值迭代方法

值迭代 - 最优值函数 (等价最优策略) $\rightarrow$ 迭代值函数求解。





$v^*(s) = \max_a \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v^*(s')$(贝尔曼最优方程)

值迭代算法

贝尔曼最优方程 $\rightarrow$ 值函数迭代

Data: MDP$\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

Result: $\pi^*(s)$ for all s

For each state s, initialize$v^*(s) \leftarrow 0$;

/* 值迭代 */

repeat

  foreach *step k* **do**

    $v^*_{k+1}(s) = \max_{a \in \mathcal{A}}(\mathcal{R}^a_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} v^*_k(s'))$

  end

until *until $v^*(s)'s$ converge*;

/* 最优策略等价最优值函数 */

foreach s **do**

  $\pi^*(s) \leftarrow \max_{a \in \mathcal{A}}(\mathcal{R}^a_s + \gamma \sum_{s' \in \mathcal{S}} P^a_{ss'} v^*(s'))$

end

Algorithm 2: 值迭代

未知模型

未知模型

状态转移函数 $\mathcal{P}_{ss'}^{a}$ 计算复杂或者未知
奖励函数 $\mathcal{R}_{s}^{a}$ 计算复杂或者未知
上述两种情况融合

状态转移函数 $\mathcal{P}$，奖励函数 $\mathcal{R}$ 未知 -> 能否估计 $\mathcal{P}, \mathcal{R}? \rightarrow$ Model-based RL

基于模型的方法 Model-based RL

1. 抽样估计状态转移概率和奖励函数 $\rightarrow$ 例如：蒙特卡洛估计 (Monte Carlo Estimation)
   $\hat{\mathcal{P}}^a_{ss'} = \frac{N^a_{ss'}}{N^a_s}$
   $\hat{\mathcal{R}}^a_s = \frac{1}{M}\sum_{i=1}^{M} R_s^{a(i)}$
2. 利用估计的转移概率函数和奖励函数求解最优策略

能否直接估计值函数? $\rightarrow$ 无模型的方法。

策略迭代可行性

策略迭代方法在无模型是否可行？

1. 估计 $v_\pi(s) \leftarrow \mathcal{R}_s^{\pi(s)} + \sum_{s'} \gamma \mathcal{P}_{ss'}^{\pi(s)} v_\pi(s')$
   $v_\pi(s) = \mathbb{E}[\sum_t \gamma^t R^t | S^0 = s; \pi]$ 可以估计 $v_\pi(s)$，例如蒙特卡洛估计 (MC)
2. 控制 $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} \mathcal{R}_s^a + \sum_{s'} \gamma \mathcal{P}_{ss'}^a v_\pi(s')$
   需要状态转移函数和奖励函数求解 $\rightarrow$ 动作值函数

动作值函数

$q_\pi(s,a) = \mathcal{R}_s^a + \sum_{s'} \gamma \mathcal{P}_{ss'}^a v_\pi(s')$
$q_\pi(s,a) = \mathcal{R}_s^a + \sum_{s'} \gamma \mathcal{P}_{ss'}^a q_\pi(s', \pi(s'))$贝尔曼期望方程

策略迭代 (Q 值)

Data: MDP$\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$
Result: $\pi^*(s)$ for all s
For each state s, initialize$\pi(s)$ randomly ;
repeat
 ~~For each state s, initialize $v_\pi(s) \leftarrow 0$~~ Initialize
 $q_\pi(s,a) = 0, \forall s, a$;
 repeat
  foreach s *and* a **do**
   $\cancel{v_\pi(s) \leftarrow \mathcal{R}_s^{\pi(s)} + \sum_{s'} \gamma \mathcal{P}_{ss'}^{\pi(s)} v_\pi(s')}$
   $q_\pi(s,a) \leftarrow \mathcal{R}_s^a + \sum_{s'} \gamma P_{ss'}^a q_\pi(s', \pi(s'))$
  end
 until *until* $\cancel{v_\pi(s)}$*'s* $q_\pi(s,a)$*'s converge*;
 foreach s **do**
  $\cancel{\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}} \mathcal{R}_s^a + \sum_{s'} \gamma \mathcal{P}_{ss'}^a v_\pi(s')}$
  $\pi(s) \leftarrow \arg\max_a q_\pi(s,a)$
 end
until *until* $\pi(s)'s$ *converge*;

Algorithm 3: 策略迭代 (Q 值)

Q 值估计

1. Monte Carlo Estimation
   $q_\pi(s,a) = \mathbb{E}[\sum_t \gamma^t R^t | S^t = s, A^t = a]$
   $q_\pi(s,a) \leftarrow q_\pi(s,a) + \eta(\hat{q}_\pi(s,a) - q_\pi(s,a)), \hat{q}_\pi(s,a) = \frac{1}{M}\sum_{i=1}^{M} q_\pi(s,a)^{(i)}$ 样本更新 (低效)
2. Temporal Difference Estimation
   $q_\pi(s,a) = \mathbb{E}[R_t + \gamma q_\pi(s',a') | S^t = s, A^t = a]$
   $q_\pi(s,a) \leftarrow q_\pi(s,a) + \eta((R_s^a + \gamma q_\pi(s', \pi(s'))) - q_\pi(s,a))$ 行为更新 (高效)

SARSA 算法 1

Data: $\langle \mathcal{S}, \mathcal{A}, \gamma \rangle$
Result: $\pi^*(s)$ for all s
For each state s and a, initialize $q_\pi(s, a)$ randomly ;
foreach *episode* **do**
  Set s to initial state; **repeat**
    Take action $a \leftarrow \arg\max_a q_\pi(s, a)$ Observe R, s' Choose action $a' \leftarrow \arg\max_{a'} q_\pi(s', a')$;
    $q_\pi(s, a) = q_\pi(s, a) + \eta[(R + \gamma q_\pi(s', \pi(s'))) - q_\pi(s, a)]$;
    $s \leftarrow s'$;
  until *until s is terminal state*;
end

Algorithm 4: SARSA 算法 1

SARSA 收敛条件

收敛条件一 GLIE

π is GLIE(Greedy in the limit with infinite exploration);

1. Greedy in the limit: **收敛到贪心策略**;
   Take action $a \leftarrow \arg\max_a q_\pi(s, a)$ **满足**;
2. Infinite exploration: **所有** (s, a) **被抽样无穷次**;
   Take action $a \leftarrow \arg\max_a q_\pi(s, a)$ **不满足**;

状态空间不能穷尽，陷入局部最优 $\rightarrow$ **需要探索。**

SARSA 收敛条件

开发和探索

开发 - 根据已经探索的空间 (抽样数据) 计算最优策略，例如：策略迭代；
探索 - 获取未曾探索的状态行为空间，例如：ϵ-Greedy；
ϵ-Greedy 以 ϵ 的概率选择随机行为，以 $1-\epsilon$ 的概率选择贪心行为 $\rightarrow$ 保证所有 (s,a) 被抽样无穷次；
ϵ 逐渐减小，例如 $\frac{1}{k}$ $\rightarrow$ 保证最终收敛到贪心策略。

收敛条件二 - Robbins–Monro

Robbins-Monro algorithm is a methodology for solving a root(θ^*) finding problem, where the function $M(\theta)=y$ is represented as an expected value $\mathbb{E}[N(\theta)=M(\theta)]$.
$\theta_{t+1}=\theta_t-\eta_t(N(\theta)-y), \lim_{t\rightarrow\infty}\theta_t\rightarrow\theta^*$
需要 $\sum_t \eta^t=\infty, \sum_t(\eta^t)^2<\infty$，例如：$\eta^t=(\frac{1}{t})^p, p>1$

SARSA 算法 2

Data: $\langle \mathcal{S}, \mathcal{A}, \gamma \rangle$
Result: $\pi^*(s)$ for all s
For each state s and a, initialize $q_\pi(s, a)$ randomly ;
foreach *episode* k **do**
  Set s to initial state;
  repeat
    Take action $\cancel{a \leftarrow \arg\max_a q_\pi(s, a)}$ a by ϵ-Greedy,
    Observe R, s' Choose action $\cancel{a' \leftarrow \arg\max_{a'} q_\pi(s', a')}$ a' by ϵ-Greedy;
    $q_\pi(s, a) = q_\pi(s, a) + \eta[(R + \gamma q_\pi(s', \pi(s'))) - q_\pi(s, a)]$;
    $s \leftarrow s'$;
  until *until s is terminal state*;
end

Algorithm 5: SARSA 算法 2

值迭代方法可行性

值迭代方法在无模型时是否可行?

v 值迭代

1. 迭代 $v^*(s) = \max_{a\in\mathcal{A}}(\mathcal{R}_s^a + \gamma\sum_{s'\in\mathcal{S}}\mathcal{P}_{ss'}^a v^*(s'))$ 直到收敛;
难以直接估计 $v^*(s)$
2. 求解 $\pi^*(s) \leftarrow \max_{a\in\mathcal{A}}(\mathcal{R}_s^a + \gamma\sum_{s'\in\mathcal{S}} P_{ss'}^a v^*(s'))$
需要模型 $\mathcal{P},\mathcal{R}$ 求解。

Q 值迭代

1. 迭代 $Q^*(s,a) = R(s,a) + \sum_{s'}\mathcal{P}_{ss'}^a \max_{a'}\gamma Q^*(s',a')$
$q^*(s,a) \leftarrow q^*(s,a) + \eta[(R_s^a + \gamma max_a' q^*(s',a')) - q^*(s,a)]$
可以估计，例如 TD 估计
2. 求解 $\pi^*(s) \leftarrow \arg\max_a Q^*(s,a)$
无需模型求解。

Q-Learning 算法

Data: $\langle \mathcal{S}, \mathcal{A}, \gamma \rangle$
Result: $\pi^*(s)$ for all s
For each state s and a, initialize $q^*(s, a)$ randomly ;
foreach *episode* **do**
  Set s to initial state;
  repeat
    Take action a by ϵ-Greedy, Observe R, s' ~~Choose action a' by ϵ-Greedy~~Choose action a' by Greedy;
    $\cancel{q_\pi(s, a) = q_\pi(s, a) + \eta[(R + \gamma q_\pi(s', \pi_(s'))) - q_\pi(s, a)]}$;
    $q^*(s, a) \leftarrow q^*(s, a) + \eta[(R_s^a + \gamma max_a' q^*(s', a')) - q^*(s, a)]$;
    $s \leftarrow s'$;
  until *until s is terminal state*;
end

Algorithm 6: Q-Learning 算法

SARSA VS Q-Learning

强化学习报告

周家豪

强化学习基础理论
强化学习的形式化
马尔科夫决策过程
基于模型的方法
策略迭代方法
值迭代方法
无模型的方法 (*)
基于策略迭代 - SARSA
基于值迭代-Q-Learning
总结

1. SARSA 源于策略迭代算法在无模型上的扩展。基于策略迭代的 SARSA 的两次行为 (a, a') 来自于同样的策略，定义为在线策略学习。
   若扩展 SARSA，使两次行为 (a, a') 来源于不同的策略，则为离线策略学习 (以不同的分布更新期望，需要进行重要性采样 (在不同分布下估计期望的方法))。
2. Q-Learning 源于值迭代算法在无模型上的扩展。基于值迭代的 Q-Learning 的两次行为 (a, a') 来自不同的策略 (ϵ-Greedy 和 Greedy 策略)，是离线策略学习。
   Q-Learning 虽然是离线策略学习学习，但不需要重要性采样即可保证其收敛性 (探索策略 + 学习率的选定)。

能否直接估计最优策略？ $\rightarrow$ 策略梯度。

Thank You!