强化学习总览

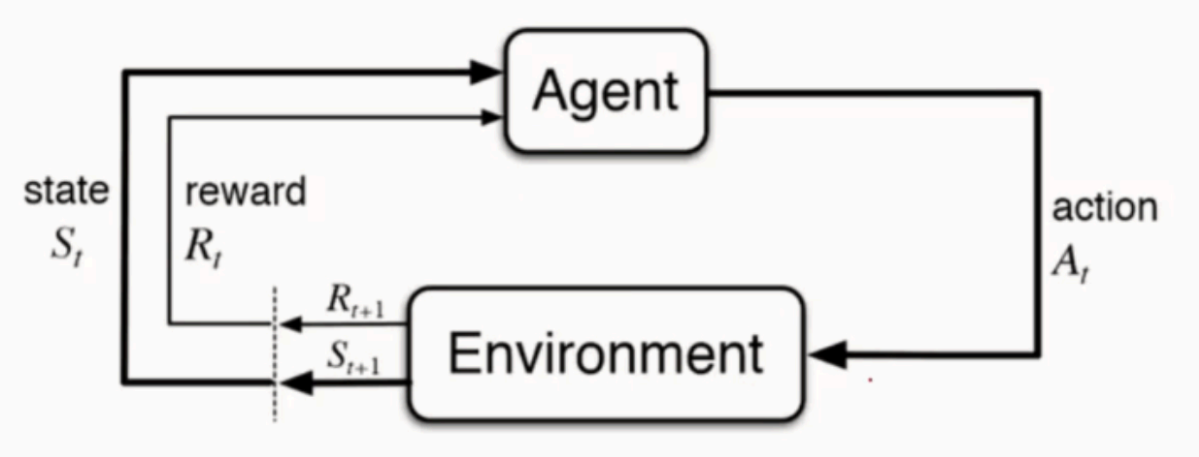
首先需要明确的是强化学习的定义、目标、分类

- **定义**：所有强化学习的框架都是基于MDP定义的
- **目标**：所有强化学习的目标都是求解最佳的策略 π^*
- **分类**：我的理解是通过有无Model的存在和强化学习的解法类别（Control）这两个正交的维度将强化学习进行分类，具体可以如下分为5类：

	value	**policy**	**both**
model-based	Value-Iteration	Policy-Iteration	-
model-free	Value-Based	Policy-Based	Actor-Critic

这样分类的好处在于使我们清楚的认知自己学习、研究的大方向在哪里，应该要解决什么核心问题

一、MDP



- 强化学习的基本作用图：Agent在Environment中探索，Environment根据Agent的action改变state并给予Agent相应的reward，Agent根据state和reward做出action以获得最多的reward
- MDP用一个 $tuple(S, A, P, R, \gamma)$ 对上述情形进行数学建模
  S：state，t时刻Agent和Environment的状态 S_t
  A：action，t时刻Agent采取的动作 A_t
  P：状态转移矩阵，揭示Agent在 S_t 下选择执行动作 A_t 之后状态成 S' 的概率
  R：reward function，一个根据Agent在t时刻的状态 S_t 以及采取的动作 a_t 来给予Agent相应reward的函数
  γ：discount factor，Agent倾向于获得短期reward，对于t时刻之后的reward需要“打折扣”

1. Policy function

- 什么是policy：
  policy可以看作是一系列action的集合，action是policy的具体表现
- 什么是policy function：
  poclicy function $\pi(a|s)$ 接收一个状态 s_t ，返回在这个状态下能够采取的所有action的概率，是一个概率分布函数
  比如现在进入到了 s_1 状态下，我采取动作 a_1 的概率为0.6，采取动作 a_2 的概率为0.4
- decision making：
  显然，我们要在policy function输出的一堆【action：概率】中选择一个作为我们 s_t 时刻的真正action，decision making就会告诉我们如何进行选择
  常见的有greedy的方式，即只选择最大概率的action作为真正的action；
  以及stochastic的方式，即严格按概率来随机选择

2. Value function

- 什么是value：
  value通过量化每个state在策略 π 下的长期期望汇报来反映该策略的好与坏
- Return G_t ：
  根据value的定义，我们需要定义另外一个东西来进行更加确切的reward的描述，这个描述需要考虑到当前能够立刻获得的reward以及未来能获得的reward（也就是定义中的长期回报），Return的公式如下：
  $G_t = R_t + \gamma * R_{t+1} + \gamma^2 * R_{t+2} + \ldots + \gamma^T * R_T$ ，其中 γ 是discount factor
- 什么是value function：
  value function $v(s)$ 接收一个状态state返回这个状态的价值，其中 $v^\pi(s_t) = E_\pi[Gt|S = s_t]$ 翻译为在策略 π 下，一个状态的价值等于其Return的期望
- 什么是q function：
  q function学名叫做action-value function（相对应的前面的value function也被称作state-value function），他表征的是在状态s下（遵循策略 π）采取特定动作a所能获得的value，他是value function的更细化表达
  $q^\pi(s_t, a_t) = E_\pi[G_t|S = s_t, A = a_t]$

二、Bellman Equation

Bellman Equation在MDP模型里扮演着很重要的角色，它通过纯数学的推导，能够揭示policy、state-value、action-value之间的关系，为求解MDP提供了坚实的基础

1. Bellman Expectation Equation（BEE）

展示的是在固定策略 π 下，state-value、action-value、policy之间的神奇关系

（1）state-function版本：

$$V^{\pi}(s)=\sum_{a\in A}\pi(a|s)*Q^{\pi}(s,a)$$

- 理解：在策略 π 下，状态 s 的价值等于他能做出的所有动作 a 的动作价值（action-value）按照这个动作发生的概率（policy）的加权期望。

（2）action-value版本：

$$Q^{\pi}(s,a)=R(s,a)+\gamma*\sum_{s'\in S}P(s'|s,a)*V^{\pi}(s')$$

- 理解：在策略 π 下，处在状态 s 时，采取动作 a 的价值等于其在 s 时刻能获得的即时奖励 $R(s,a)$ 与在下一时刻 s' 能获得的折扣奖励之和
- 注意，动作 a 是选定的

（3）合并版本：

$$V^{\pi}(s)=\sum_{a\in A}\pi(a|s)*[R(s,a)+\gamma*\sum_{s'\in S}P(s'|s,a)*V^{\pi}(s')]$$

- 它巧妙的揭示了状态 s 和 s' 的关系

2. Bellman Optimal Equation（BOE）

展示的是在最优策略时（而不是固定策略）

（1）state-value版本：

$$V^{*}(s)=max_{a\in A}Q^{*}(s,a)$$

- 理解：最优的state-value是在所有可能采取的动作中action-value最大的那个

（2）action-value版本：

$$Q^{\pi}(s,a)=R(s,a)+\gamma*\sum_{s'\in S}P(s'|s,a)*V^{*}(s')$$

- 理解：与BEE类似

（3）合并版本：

$$V^{*}(s)=max_{a\in A}[R(s,a)+\gamma*\sum_{s'\in S}P(s'|s,a)*V^{*}(s')]$$

3. 附赠上述式子的推导过程

$$
\begin{aligned}
Q^{\pi}(S_t, a_t) &= E_{\pi}[G_t \mid S = s_t, A = a_t] \\
&= E_{\pi}[R_t + \gamma G_{t+1} \mid S = s_t, A = a_t] \\
&= R(s,a) + \gamma E_{\pi}[G_{t+1} \mid S = s_t, A = a_t] \\
&= R(s,a) + \gamma \sum_{s' \in S} P(s' \mid s,a) V^{\pi}(s')
\end{aligned}
$$

$$
\begin{aligned}
V^{\pi}(s_t) &= E_{\pi}[G_t \mid S = s_t] \\
&= E_{\pi}[Q^{\pi}(s_t, a_t) \mid s_t = s] \\
&= \sum_{a \in A} \pi(a \mid s_t) Q^{\pi}(s_t, a_t)
\end{aligned}
$$

三、MDP的解法

MDP的解法就是RL中的“通用解法”，先了解一下大概有哪些思路，再具体研究
MDP的解涉及两个最关键的问题，分别是Prediction和Control

1. Prediction

- 如何评估一个策略 π 的好坏?
- model-based中：Bellman Equation直接算
- model-free中：采用MC或者TD的方法，通过采样估计得到

2. Control

- 如何找到最优的策略 π^* ？

（1）Model-based中：

Policy Iteration

- 随机init一个 π
- 用这个 π 通过BEE来求得当前的value function $v^{\pi}(s)$
- 用greedy的方式通过 $v^{\pi}(s)$ 来更新策略 π
- 如此重复直到达到某个阈值或标准

Value Iteration

- 利用BOE迭代，使策略 $v(s)$ 与最佳 $v^*(s)$ 接近
- 找到最佳value function后提取出对应的最佳策略 π^*

（2）Model-free中：

新方法层出不穷、蓬勃发展

Value Based

- MC-ϵ greedy
- SARSA、Q-learning
- DQN

Policy Based

- Policy Gradient等

Actor Critic

- classic
- advantage