【David Silver强化学习公开课之五】Model-Free Control(
决未知Environment下的Control问题)

发表于 2016-07-26 | 分类于 project experience | | 2026

本文是David Silver强化学习公开课第五课的总结笔记。这一课主要讲了解决在未知policy未知environment的
下control问题的两种思路，即On-Policy和Off-Policy，在第四课的Monte-Carlo和Temporal Difference基础上
改进。

本文是David Silver强化学习公开课第五课的总结笔记。这一课主要讲了解决在未知policy未知environment的
下control问题的两种思路，即On-Policy和Off-Policy，在第四课的Monte-Carlo和Temporal Difference基础上
改进。

【转载请注明出处】chenrudan.github.io

本文是David Silver强化学习公开课第五课的总结笔记。这一课主要讲了解决在未知policy未知environment的
下control问题的两种思路，即On-Policy和Off-Policy，在第四课的Monte-Carlo和Temporal Difference基础上
改进。

本课视频地址:RL Course by David Silver - Lecture 5: Model Free Control。

本课ppt地址:http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/control.pdf

文章的内容是课程的一个总结和讨论，会按照自己的理解来组织。个人知识不足再加上英语听力不是那么好
有一些理解不准的地方，欢迎一起讨论。

建了一个强化学习讨论qq群，有兴趣的可以加一下群号595176373或者扫描下面的二维码。

1.内容回顾

上次课谈到了在给定policy的情况下求解未知environment的MDP问题，称之为Model-Free Prediction问题。
则是解决未知policy情况下未知environment的MDP问题，也就是Model-Free Control问题，这个问题实际
常见的强化学习问题。由于这种问题中未知policy，那么就有两种思路来获得policy，一种称为on-policy le
是基于某个policy做出一些action然后评估这个policy效果如何，一种称为off-policy learning是从一些已知的
中学习policy，比如机器人在学习走路时，可以从人控制机器人走路的sample中来学习，但不是完全的跟san
的action完全一样，在sample中尝试去走不同的一步看是否有更好reward。

2. On-Policy Monte-Carlo

文章目录 站点概览

On-Policy Monte-Carlo由policy evaluation + $\epsilon-$Greedy Policy Improvement组成。

在第三课的动态规划解决planning问题(已知environment)中提出的Policy iteration和value iteration，其中 iteration由policy evaluation和policy improvement组成。第四课中未知environment的policy evaluation是通过 卡洛方法求解，结合起来到本课可以得到第一个解决Model-Free control方法即先通过贪婪算法来确定当 policy，再通过蒙特卡洛policy evaluation来评估当前的policy好不好，再更新policy。

如果在已知environment情况下policy improvement更新方式是$\pi^{'}(s) = \underset{a \in A}{argmax} R_s^a + P_{ss'}^a V(s^{'})$，可以看出 解决方案是通过状态转移矩阵把所有可能转移到的状态得到的值函数都计算出来，从中来选择最大的，但 environment则没有状态转移矩阵，因此只能通过最大化动作值函数来更新policy即$\pi^{'}(s) = \underset{a \in A}{argmax} Q(s,a$ 于improvement的过程需要动作值函数，那么在policy evaluation的过程中针对给定的policy需要计算的$V(s$ 换成$Q(s,a)$。

但是greedy算法是存在一定的问题的，例如现在有两扇门，打开一扇门会有一定的奖励，经过一些开门试验 择能够获得奖励最大的门。假设第一次打开左边的门获得的immdiate reward是0，那么左边门的return 为$V(left) = 0$，第二次打开右边获得的immdiate reward是+1，右边门return更新$V(right) = 1$，此时如果 greedy算法，那么下一次肯定会选择右边的门，第三次选择右边门获得reward是+3，return 为$V(right) = 2$(蒙特卡洛方法平均了一下)，根据greedy算法第四次也会选择右边门。因此按照贪婪算法就 直选择右边门，但是其实我们并不清楚左边门到底是什么情况，我们只尝试了一次。从这个例子可以看出，需 执行每个action的结果都做比较充分的了解，才能说自己的policy是正确的。

因此提出改进算法在greedy基础上有一定概率选择一个随机action，即$\epsilon-$Greedy Exploration，假设有 action，那么有ϵ的概率随机选择一个action(包括greedy action)，从而可以得到更新的policy为(17:48开始证明 进算法算出的新policy比之前的好，此处略过)

$$\pi(a|s) = \begin{cases} \frac{\epsilon}{m} + 1 - \epsilon & \text{if } a^* = \underset{a \in A}{argmax} Q(s,a) \\ \frac{\epsilon}{m} & \text{otherwise} \end{cases} \quad (1)$$

如果每一个的episode都进行一次evaluation和improvement迭代，那么在第k次迭代时可以更新$\epsilon = \frac{1}{k}$。更新 法称之为GLIE Monte-Carlo Control。

3.Sarsa Algorithm

由第四课Temporal-difference方法能解决MC问题，即通过TD方法来求$Q(s,a)$，之前TD的值函数更新 为 $V(S_t) = V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$，那么动作值函数更新公 为$Q(S,A) = Q(S,A) + \alpha(R + \gamma Q(S^{'}, A^{'}) - Q(S,A))$。

基于TD的policy迭代是每走一个step执行一个action都会更新一次。具体的On-Policy Control流程如下:

- 初始化$Q(s,a)$
- for each episode:
- ==初始化一个状态S
- ==基于某个策略Q和当前状态S选择一个动作A
- ==for each step of one episode:
- ====执行一个动作A，得到反馈的immdiate reward为R，和新的状态S'
- ====基于当前策略Q和状态S'选择一个新动作A'
- ====更新策略: $Q(S,A) = Q(S,A) + \alpha[R + \gamma Q(S', A') - Q(S,A)]$
- ====更新状态$S = S'$
- ==直到S到达终止状态

同样的，上面是TD(0)的更新方式，扩展到step为n时更新动作值函数的公式如下:

$$q_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+n}) \quad (2)$$

文章目录 站点概览

$$Q(S_t,A_t)=Q(S_t,A_t)+\alpha(q_t^{(n)}-Q(S_t,A_t)) \quad (3)$$

4. Off-Policy Learning

之前说过Off-Policy Learning是在某个已知策略(behaviour policy)$\mu(a|s)$下来学习目标策略(target policy)$\pi($
这样就能够从人的控制或者其他表现的比较好的agent中来学习新的策略。如果把两个策略当成两
布$P(X),Q(X)$，并且假设reward函数为$f(X)$，两种分布中reward期望为:$E_{X\ P}[f(X)]=E_{X\ Q}[\frac{P(X)}{Q(X)}f($
因此可以从μ中来估计π获得的return，这个方法称为Importance Sampling。

Off-Policy Monte-Carlo是在第四课的Monte-Carlo Learning上的改进，即更新值函数的公式变为

$$G_t^{\frac{\pi}{\mu}}=\frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}\frac{\pi(A_{t+1}|S_{t+1})}{\mu(A_{t+1}|S_{t+1})}\cdots\frac{\pi(A_T|S_T)}{\mu(A_T|S_T)}G_t \quad (4)$$

$$V(S_t)=V(S_t)+\alpha(G_t^{\frac{\pi}{\mu}}-V(S_t)) \quad (5)$$

同样的Off-policy TD也是改变了更新值函数公式，改变的这一项相当于给TD target加权，这个权重值代表
策略和已知策略匹配程度，代表了是否能够信任目标policy提出的这个action。

$$V(S_t)=V(S_t)+\alpha(\frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}(R_{t+1}+\gamma V(S_{t+1}))-V(S_t)) \quad (6)$$

5.Off-Policy Q-Learning

针对未知policy，即本课面临的问题，Off-policy的解决方案是Q-Learning，更新动作值函数。根据$\mu(\cdot|S_t)$
时刻选择的action是A_{t+1}，根据$\pi(\cdot|S_t)$下一个时刻action是A'，更新
为$Q(S_t,A_t)=Q(S_t,A_t)+\alpha(R_{t+1}+\gamma Q(S_{t+1},A')-Q(S_t,A_t)$，也就是说在某个已知策略下选择了
时刻的动作A_{t+1}，以及下一个时刻的状态S_{t+1}和奖赏R_{t+1}，将目标策略选择的动作A'替换到更新公式中。
方法不同的是，可以同时更新π和μ，且π是greedy的方式，而μ是采用了ϵ-greedy方式。Q-Learning的
为$R_{t+1}+\gamma Q(S_{t+1},A')=R_{t+1}+\gamma Q(S_{t+1},\underset{a'}{argmax}Q(S_{t+1},a'))=R_{t+1}+\underset{a'}{max}\gamma Q(S_{t+1},a')$，这里
选择使Q最大的a'，所以后面的式子就是在求最大的Q。

小结:本课主要是在第四课基础上解决了未知policy未知environment的MDP问题，课程最后有对动态规划
Temporal Difference方法的对比。

‹ 【David Silver强化学习公开课之四】Model-Free Learning(解决未知Environment下的Prediction问题)

【David Silver强化学习公开课之六】求解近似