蒙特卡洛方法&Q-Learning & TD —— 强化学习基础算法

关键词：

#蒙特卡洛方法
#Q-Learning
#时序差分算法

本文介绍两种强化学习基础算法。

- 蒙特卡洛方法:
  这个算法十分简单，我们的目标是维护一个 Q-Table。Q-Table中储存着在任意一个状态 s 下，采取动作 a 的价值 Q(s，a)。

我们令 代理 与 环境 互动，直到终止状态，得到一条轨迹。通过这条轨迹，我们可以 得到 状态 s 下，采取动作 a 的一个带折扣因子的奖励值 G_t 。

我们令 $Q(s,a)$ 朝 G_t 步进，即$Q(s,a) \leftarrow Q(s,a)+\alpha[G_t-Q(s,a)]$。

经过大量的互动，我们最终得到的 Q-Table 可以趋近真实值（期望值）。

根据 Q-Table，我们就可以在不同状态下，选择合适的行为。

然而，这种方法存在许多不足。**我们只有到达中止状态才可以开始更新 Q-Table，并且对于很多问题，Q-Table 是极度庞大的，我们很难去维护这么一个极大的表。**

- 时序差分算法:
  与蒙特卡洛方法不同的是，我们不再尝试用 G_t 更新 Q-Table，这样可以解决 蒙特卡罗方法必须走完一整条轨迹才能更新的缺点。
  我们用 $r+\gamma*Q(s',a')$ 来代替蒙特卡洛方法中的 G_t 。可以发现，我们用到了当前的 Q 表来更新 Q 表，这实际上是 广义策略迭代的思想，即策略提升可以在策略评估未完全确定的情况下进行。

值得一提的是，代理与环境互动的时候，我们往往选择 $\epsilon-greedy$ 这类减小偶然性的策略。

- Q-Learning:
  同样是维护一个 Q 表，更新式子如下:
  Q(s,a)=(1−α)∗Q(s,a)+α∗(r+γ∗max′aQ(s′,a′))
  可以发现，Q-Learning 与 时序差分算法 的不同点在于我们在 s' 状态下，贪婪地选择最佳动作对应的 Q 值（贪婪策略，但是只在更新公式中使用贪心法，代理与环境互动的时候并不贪心）。

范式

- 随机初始化 Q-Table
- 代理与环境互动
- 更新 Q-Table
  后两步不停地循环。

与深度学习融合

Q-Table 很难维护，很多时候状态数和动作数相当庞大。

我们用一个函数代替Q-Table，深度学习中，这个函数就是 神经网络 ， 可以将 状态 映射到动作的价值，更新方式就是通过损失反向传播。