# 第13章 SAC 算法

本章开始介绍最后一种经典的策略梯度算法，即 Soft Actor-Critic 算法，简写为 SAC 。SAC 算法是一种基于最大熵强化学习的策略梯度算法，它的目标是最大化策略的熵，从而使得策略更加鲁棒。SAC 算法的核心思想是，通过最大化策略的熵，使得策略更加鲁棒，经过超参改良后的 SAC 算法在稳定性方面是可以与 PPO 算法华山论剑的。注意，由于 SAC 算法理论相对之前的算法要复杂一些，因此推导过程要多很多，但是最后的结果还是相对简洁的，因此读者可以根据自己的需求选择性阅读，只需要关注伪代码中变量的涵义以及结果公式即可。

## 13.1 最大熵强化学习

由于 SAC 算法相比于之前的策略梯度算法独具一路，它走的是最大熵强化学习的路子。为了让读者更好地搞懂什么是 SAC ，我们先介绍一下最大熵强化学习，然后从基于价值的 Soft Q-Learning 算法开始讲起。我们先回忆一下确定性策略和随机性策略，确定性策略是指在给定相同状态下，总是选择相同的动作，随机性策略则是在给定状态下可以选择多种可能的动作，不知道读者们有没有想过这两种策略在实践中有什么优劣呢？或者说哪种更好呢？这里我们先架空实际的应用场景，只总结这两种策略本身的优劣，首先看确定性策略：

优势：**稳定性且可重复性**。由于策略是确定的，因此可控性也比较好，在一些简单的环境下，会更容易达到最优解，因为不会产生随机性带来的不确定性，实验也比较容易复现。

劣势：**缺乏探索性**。由于策略是确定的，因此在一些复杂的环境下，可能会陷入局部最优解，无法探索到全局最优解，所以读者会发现目前所有的确定性策略算法例如 DQN 、DDPG 等等，都会增加一些随机性来提高探索。此外，面对不确定性和噪音的环境时，确定性策略可能显得过于刻板，无法灵活地适应环境变化。

再看看随机性策略：

优势：**更加灵活**。由于策略是随机的，这样能够在一定程度上探索未知的状态和动作，有助于避免陷入局部最优解，提高全局搜索的能力。在具有不确定性的环境中，随机性策略可以更好地应对噪音和不可预测的情况。

劣势：**不稳定**。正是因为随机，所以会导致策略的可重复性太差。另外，如果随机性太高，可能会导致策略的收敛速度较慢，影响效率和性能。

不知道读者有没有发现，这里字里行间都透露着随机性策略相对于确定性策略来说存在碾压性的优势。为什么这么说呢？首先我们看看确定性策略的优点，其实这个优点也不算很大的优点，因为所有可行的算法虽然可能不能保证每次的结果都是一模一样的，但是也不会偏差得太过离谱，而且我们一般也不会对可复现性要求那么高，一定要精确到每个小数点都正确，因此容易复现本身就是个伪命题。其次，这里也说了在一些简单的环境中更容易达到最优解，简单的环境是怎么简单呢？可能就是在九宫格地图里面寻找最短路径或者石头剪刀布的那种程度，而实际的应用环境是不可能有这么简单的场景的。

再看看随机性策略的缺点，其实也不算是什么缺点，因为在随机性策略中随机性是我们人为赋予的，换句话说就是可控的，反而相对来说是可控的稳定性。结合我们实际的生活经验，比如在和别人玩游戏对战的时候，是不是通常会觉得招式和套路比较多的人更难对付呢？因为即使是相同的情况，高手可能会有各种各样的方式来应对，反之如果对方只会一种打法，这样会很快让我们抓住破绽并击败对方。在强化学习中也是如此，我们会发现实际应用中，如果有条件的话，我们会尽量使用随机性策略，诸如 A2C 、PPO 等等，因为它更加灵活，更加鲁棒，更加稳定。

然而，最大熵强化学习认为，即使我们目前有了成熟的随机性策略，即 Actor-Critic 一类的算法，但是还是没有达到最优的随机。因此，它引入了一个信息熵的概念，在最大化累积奖励的同时最大化策略的熵，使得策略更加鲁棒，从而达到最优的随机性策略。我们先回顾一下标准的强化学习框架，其目标是得到最大化累积奖励的策略，如式 (13.1) 所示。

而最大熵强化学习则是在这个基础上加上了一个信息熵的约束，如式 (13.2) 所示。

其中 是一个超参，称作温度因子（ temperature ），用于平衡累积奖励和策略熵的比重。这里的 就是策略的信息熵，定义如式 (13.3) 所示。

它表示了随机策略 对应概率分布的随机程度，策略越随机，熵越大。后面我们可以发现，虽然理论推导起来比较复杂，但实际实践起来是比较简单的。

## 13.2 Soft Q-Learning

前面小节中我们引入了带有熵的累积奖励期望，接下来我们需要基于这个重新定义的奖励来重新推导一下相关的量。后面我们会发现虽然推导起来比较复杂，但用代码实现起来是比较简单的，因为几乎跟传统的 Q-Learning 算法没有多大区别。因此着重于实际应用的同学可以直接跳过本小节的推导部分，直接看后面的算法实战部分。

现在我们开始进行枯燥地推导过程了，首先是 Q 值函数和 V 值函数，我们先回顾一下传统的最优 Q 值函数的定义，如式 (13.4) 所示。

相应地，最优的 V 值函数只需要根据最优 Q 值函数取 argmax 即可，如式 (13.5) 所示。

在最大熵强化学习中，我们需要重新定义最优 Q 值函数需要改成式 (13.6) 。

即增加了一个熵的项，但是对应地最优 V 值函数就不能简单地取 argmax 了。但是根据贝尔曼方程，我们可以得到如式 (13.7) 所示的关系。

注意 Q 和 V 之间存在着线性关系，如式 (13.8) 所示。

根据这个关系就可以得到式 (13.9) 。

进一步地，可以得到式 (13.10) 。

可能这一步的求解读者们理解起来有点困难，其实是这样的，在 V 值到达最优即最大的时候，也就是 ，相当于 也是最大的，我们设这个最大值为 c，那么就可以得到式 (13.11) 。

两边都对动作 积分之后，就可以得到式 (13.12) 。

这里策略分布对动作的积分其实就是等于值函数，去掉一些无关的常数项，就是上面我们得到的最终式子了。注意到，这里求解出来的 V 函数其实是一种 softmax 的形式，如式 (13.13) 所示。

意思就是需要平滑地取最大值，而当 的时候，就退化成了传统的 Q 函数，对应的 softmax 也就变成了 hardmax，这也是为什么叫 soft 的原因。这其实就是玻尔兹曼分布的一种形式，即给所有动作赋予一个非零的概率，让智能体倾向于学习到能够处理相应任务的所有行为，注意是所有。针对这个玻尔兹曼分布，原论文提出了一个 energy-based policy 的模型，感兴趣的读者可以深入了解，但由于对我们理解 SAC 算法并不是很重要，因此这里就不展开讲解了。

另外注意，其实这里的 是不太好算的，因为需要对整个动作空间积分然后进行 softmax，这个计算量是非常大的，因此也可以借助于 PPO 中用到的重要性采样，将其转化为一个包含 $Q$ 函数的期望的形式，如式 (13.14) 所示。

其中 一般用 Categorica 分布来表示，最后我们就可以得到 Soft Q-Learning 的损失函数了，如式 (13.15) 所示。

其中 表示目标网络的 Q 值。

这里还有一个问题，就是理论上直接从当前策略分布 中采样也是比较困难的，此时我们可以借助一个额外的采样网络。首先构建一个参数为 的 ，这个网络可以在状态 下，将一个服从任意分布的噪声样本 映射到 ，将 的动作分布记为 ，那么参数 的更新目标就是让 尽可能接近 ，这种方式很容易想到用 KL 散度来做，如式 (13.16) 所示。

其梯度如式 (13.17) 所示。

## 13.3 SAC

实际上 SAC 算法有两个版本，第一个版本是由 Tuomas Haarnoja 于 2018 年提出来的[[1]](#footnote-1)，，第二个版本也是由 Tuomas Haarnoja 于 2019 年提出来的[[2]](#footnote-2)，一般称作 SAC v2。第二个版本主要在前一版本的基础上做了简化，并且实现了温度因子的自动调节，从而使得算法更加简单稳定。

我们先讲讲第一版，第一版的 SAC 算法思想基本上是和 Soft Q-Learning} 是特别相似的，只是额外增加了两个 $V$ 值网络（即包含目标网络和当前网络）来估计价值。$V$ 网络的目标函数定义如式 (13.18) 所示。

其中 主要来自经验回放中的样本分布，其梯度如式 (13.19) 所示。

Soft Q 函数的目标函数定义如式 (13.20) 所示。

对应梯度如式 (13.21) 所示。

策略的目标函数相比于 Soft Q-Learning 要更简洁一些，如式 (13.22) 所示。

其梯度如式 (13.23) 所示。

## 13.4 自动调节温度因子

本小节主要讲解如何推导出自动调节因子的版本，整体推导的思路其实很简单，就是转换成规划问题，然后用动态规划、拉格朗日乘子法等方法简化求解，只关注结果的读者可以直接跳到本小节最后一个关于温度调节因子 的梯度下降公式即可。

注意第一个版本和 Soft Q-Learning 都存在一个温度因子 的超参，并且这个超参是比较敏感的，第二个版本则设计了一个可以自动调节温度因子的方法。首先回顾一下累积奖励期望公式，如式 (13.24) 所示。

其中左边项是奖励部分，右边项是熵部分，我们的目标是同时最大化这两个部分，但是由于这两个部分的量纲不同，因此需要引入一个温度因子 来平衡这两个部分。第二版 SAC 的思路就是，我们可以把熵的部分转换成一个约束项，即只需要满足式 (13.25) 即可。

换句话说，我们只需要让策略的熵大于一个固定的值 即可，这样就可以不需要温度因子了。这样一来我们的目标就变成了在最小期望熵的约束条件下最大化累积奖励期望，如式 (13.26) 所示。

可以看出，当我们忽略掉这个约束条件时，就变成了一个标准的强化学习问题。这里将策略 拆分成了每一时刻的策略 ，以便于求解每一时刻 t 的最优温度因子 。由于每一时刻 t 的策略只会对未来奖励造成影响（马尔可夫性质），因此可以利用动态规划的思想，自顶向下（从后往前）对策略进行求解，即分解为式 (13.27) 。

这样一来我们只需要求出第一个子问题，即式 (13.28) 。

这个问题可以通过拉格朗日乘子法求解，首先我们做一个简化，如式 (13.29) 所示。

进而原问题简化为式 (13.30) 。

通过乘上一个拉格朗日乘子 （也叫做对偶变量，也相当于温度因子），我们可以得到式 (13.31) 。

当 时，可得到 ，当 时，可得到 ，一般来说 ， 因此 。

我们的目标是最大化 ，如式 (13.32) 所示。

因此原问题可以转化为相对易解的对偶问题，如式 (13.33) 所示

首先固定住温度因子 ，就能够得到最佳策略 使得 最大化，如式 (13.34) 所示。

求出最佳策略之后，就可以求出最佳的温度因子 $\alpha\_T^\*$，如式 (13.35) 所示。

这样回到第一个子问题，即在时刻 $T$ 下的奖励期望，如式 (13.36) 所示。

接下来我们就可以求解第二个子问题，即在时刻 T-1 下的奖励期望。回顾一下 Soft Q 函数公式，如式 (13.37) 所示。

代入第一个子问题的最优策略 之后，就可以得到式 (13.38) 。

同样地，利用拉格朗日乘子法，就能得到第二个子问题即 T-1 下的奖励期望最大化为式 (13.39) 。

进而得到式 (13.40) 。

我们会发现第二个子问题的答案形式其实和第一个子问题是一样的，依此类推，我们就可以得到温度因子的损失函数，如式 (13.41) 所示。

这样一来就能实现温度因子的自动调节了。这一版本由于引入了温度因子的自动调节，因此不需要额外的 V 值网络，直接使用两个 Q 网络（包含目标网络和当前网络）来作为 Critic 估计价值即可。

1. Haarnoja T , Zhou A , Abbeel P ,et al.Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor.2018[2023-08-02].DOI:10.48550/arXiv.1801.01290. [↑](#footnote-ref-1)
2. Haarnoja T , Zhou A , Hartikainen K ,et al.Soft Actor-Critic Algorithms and Applications.2018[2023-08-02].DOI:10.48550/arXiv.1812.05905. [↑](#footnote-ref-2)