SAC(Soft Actor Critic)

SAC 是一种将 **极大化熵学习** 与 Actor-Critic 框架结合的 离线策略强化学习方法。SAC 不仅希望环境奖励的最大化，还希望策略的熵最大化。

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi}\left[\sum_t r(s_t, a_t) + \alpha H(\pi(\cdot|s_t))\right] \tag{1}$$

其中， α 是正则化系数，用来控制熵的重要程度。

Soft Policy Iteration

熵 的定义是：

$$\begin{aligned} H(\mathbf{p}) &= -\sum_i p_i \log p_i, (\text{离散随机分布}) \\ &= -\int p(x)\log p(x)\mathrm{d}x, (\text{连续随机分布}) \end{aligned}$$

在 SAC 中，希望找到一个策略，极大化奖励的同时极大化策略在每一个状态下动作分布的熵。因此，奖励函数被曾广为：

$$r_{soft}(s_t, a_t) = r(s_t, a_t) + \alpha \mathbb{E}_{s_{t+1}\sim p}[H(\pi(\cdot|s_{t+1}))] \tag{2}$$

其中, α 为熵温度系数，决定了对熵最大化的重视程度； p 为状态转移分布， $r(s_t, a_t)$ 为在 s_t 执行 a_t 能获得奖励的期望。bellman 方程：

$$Q(s_t, a_t) = r(s_t, a_t) + \mathbb{E}_{\substack{s_{t+1}\sim p \\ a_{t+1}\sim\pi}}[Q_{soft}(s_{t+1}, a_{t+1}) - \alpha\log(\pi(a_{t+1}|s_{t+1}))] \tag{3}$$

将公式（2）带入公式（3），有：

$$Q_{soft}(s_t, a_t) = r(s_t, a_t) + \mathbb{E}_{s_{t+1}, a_{t+1}}[Q_{soft}(s_{t+1}, a_{t+1}) - \alpha\log(\pi(a_{t+1}|s_{t+1}))] \tag{4}$$

因此，可以找到比当前更好的新策略 π_{new}：

$$\pi_{new} = \arg\min_{\pi'} \mathrm{D}_{\mathrm{KL}}\left(\pi'(\cdot|s_t) \parallel \frac{\exp\left(\alpha^{-1}Q_{soft}^{\pi_{old}}(s_t, \cdot)\right)}{Z^{\pi_{old}}(s_t)}\right) \tag{5}$$

公式（5）现将 $Q_{soft}^{\pi_{old}}$ 指数化为 $\exp\left(\alpha^{-1}Q_{soft}^{\pi_{old}}\right)$，再将其归一化为分布 $\frac{\exp\left(\alpha^{-1}Q_{soft}^{\pi_{old}}(s_t,\cdot)\right)}{Z^{\pi_{old}}(s_t)}$，其中 $Z^{\pi_{old}}(s_t) = \int \exp(\alpha^{-1}Q_{soft}^{\pi_{old}}(s_t, a_t))\mathrm{d}a_t$ 为归一化函数。希望找到一个策略 π' 在状态 s_t 下的动作分布与 $\frac{\exp\left(\alpha^{-1}Q_{soft}^{\pi_{old}}(s_t,\cdot)\right)}{Z^{\pi_{old}}(s_t)}$ 分布的 KL 散度 最小，即希望分布 $\pi'(\cdot|s_t)$ 与分布 $\frac{\exp\left(\alpha^{-1}Q_{soft}^{\pi_{old}}(s_t,\cdot)\right)}{Z^{\pi_{old}}(s_t)}$ 越相似越好。

Soft Actor Critic

为两个动作价值函数 Q（参数为 ω_1, ω_2） 和一个策略函数 π （参数为 θ）建模。基于 Double DQN 的思想，SAC 使用两个 Q 网络，但每次进挑选一个 Q 值较小的网络，从而缓解 Q 值过高估计的问题。对于任意一个函数 Q 的损失函数为：

$$
\begin{aligned}
L_Q(\omega) &= \mathbb{E}\left[\frac{1}{2}(Q_\omega(s_t,a_t) - (r_t + \gamma V_{\omega^-}(s_{t+1})))^2\right] \\
&= \mathbb{E}\left[\frac{1}{2}\left(Q_\omega(s_t,a_t) - \left(r_t + \gamma\left(\min_{j=1,2} Q_{\omega_j^-}(s_{t+1},a_{t_1}) - \alpha\log\pi(a_{t+1}|s_{t+1})\right)\right)\right)^2\right]
\end{aligned} \tag{6}
$$

其中，R 是策略过去收集的数据，因为 SAC 是一种离线策略算法。为了让训练更加稳定，这里使用了目标 Q 网络 Q_{ω^-}，同样是两个目标 Q 网络，与两个 Q 网络一一对应。SAC 中目标 Q 网络的更新方式与 DDPG 中的更新方式一样。

策略 π 的损失函数由 KL 散度得到，化简后为：

$$
L_\pi(\theta) = \mathbb{E}\left[\alpha\log(\pi_\theta(a_t|s_t)) - Q_\omega(s_t,a_t)\right] \tag{7}
$$

可以理解为最大化函数 V，因为有 $V(s_t) = \mathbb{E}\left[Q(s_t,a_t) - \alpha\log\pi(a_t|s_t)\right]$。

对连续动作空间的环境，SAC 算法的策略输出高斯分布的均值和标准差，但是根据高斯分布来采样动作的过程是不可导的。因此，我们需要用到**重参数化技巧**（reparameterization trick）。重参数化的做法是先从一个单位高斯分布 $\mathcal{N}$ 采样，再把采样值乘以标准差后加上均值。这样就可以认为是从策略高斯分布采样，并且这样对于策略函数是可导的。我们将其表示为 $a_t = f_\theta(\epsilon_t;s_t)$，其中 ϵ_t 是一个噪声随机变量。同时考虑到两个函数 Q，重写策略的损失函数：

$$
L_\pi(\theta) = \mathbb{E}\left[\alpha\log\pi_\theta(f_\theta(\epsilon_t;s_t)|s_t) - \min_{j=1,2} Q_{\omega_j}(s_t, f_\theta(\epsilon_t;s_t))\right] \tag{8}
$$

证明

公式（4）的证明：

$$
\begin{aligned}
Q_{soft}(s_t,a_t) &= r(s_t,a_t) + \alpha\mathbb{E}_{s_{t+1}}\left[H(\pi(\cdot|s_{t+1}))\right] + \mathbb{E}_{s_{t+1},a_{t+1}}\left[Q_{soft}(s_{t+1},a_{t+1})\right] \\
&= r(s_t,a_t) + \alpha\mathbb{E}_{s_{t+1},a_{t+1}}\left[\mathbb{E}_{a_{t+1}}(-\log\pi(a_{t+1}|s_{t+1}))\right] \\
&= r(s_t,a_t) + \mathbb{E}_{s_{t+1},a_{t+1}}\left[Q_{soft}(s_{t+1},a_{t+1}) - \alpha\log(\pi(a_{t+1}|s_{t+1}))\right]
\end{aligned}
$$