Soft Q-learning

BY ZHIJUN ZENG

2023年9月18日

1 最大化熵方法

在强化学习中，我们的目标是寻找一个策略，能够最大化环境中得到的回报，数学表示为

$$\pi_{\mathrm{std}}^* = \mathrm{argmax}_\pi \sum_t \mathbb{E}_{(s_t, a_t) \sim \rho_\pi}[r(s_t, a_t)]$$

这里π_{std}^*是传统RL中的到的最优策略，ρ_π为基于策略π决定的状态与动作分布，r为回报

为了增加模型的探索能力exploration，使得模型能够更鲁棒和适应不同场景， 我们考虑设计这个最大化的目标不仅仅最大化获得的回报，还要考虑策略的分布特征， 即最大化每一步的熵值， 其中α是控温系数，用来调节熵的相对重要性。

$$\pi_{\mathrm{MaxEnt}}^* = \mathrm{argmax}_\pi \sum_t \mathbb{E}_{(s_t, a_t) \sim \rho_\pi}[r(a_t, s_t) + \alpha H(\pi(\cdot|s_t))]$$

这里π_{MaxEnt}^*为考虑熵的最大化之后的最优策略，其中熵的定义是

$$(\pi(\cdot|s_t)) = -\int_{\mathcal{A}} \pi(a_t|s_t) \log \pi(a_t|s_t) \mathrm{d}a_t = \mathbb{E}_{a_t \sim \rho_\pi(\cdot, s_t)}[-\log(\pi(a_t|\, s_t))]$$

2 Soft Q-function and Soft Value-function

我们假设我们想要寻找的策略分布以如下形式出现

$$\pi \propto \exp(-\mathcal{E}(s_t, a_t))$$

并且假设能量函数形式与Q function成反比

$$\mathcal{E}(s_t, a_t) = -\frac{1}{\alpha} Q_{\mathrm{soft}}(s_t, a_t)$$

这样一来，动作的选择与Q的值相关，Q越大概率越大，满足我们的要求。

由于回报函数中有了最大熵的加入，作者提出了该形式下对应的Q function 与 Value function

$$\begin{aligned} Q_{Soft}^* &= r_t + \mathbb{E}_{(s_{t+1}, \cdot) \sim \rho_\pi}\left[\sum_{l=1}^{\infty} \gamma^l \left(r_{t+l} + \alpha\, H\left(\pi_{M\alpha xEnt}^*(\cdot|\, s_{t+l})\right)\right)\right] \\ V_{Soft}^* &= \alpha * \log \int_{\mathcal{A}} \exp\left(\frac{1}{\alpha}\, Q_{Soft}^*(s_t, a')\right) d\, a' \end{aligned}$$

并且作者证明上述定义的Soft Q function与Soft Value function满足Bellman equation

$$Q_{\mathrm{soft}}^*(s_t, a_t) = r_t + \gamma \mathbb{E}_{s_{t+1} \sim p_s}[V_{\mathrm{soft}}^*(s_{t+1})]$$

定理 1. 根据上述定义的Soft Q function与Soft Value function，最优的策略为

$$\pi^*_{\mathrm{MaxEnt}}(a_t|s_t) = \exp\left(\frac{1}{\alpha}(Q^*_{\mathrm{soft}}(s_t, a_t) - V^*_{\mathrm{soft}}(s_t))\right)$$

3 Training via Soft Q-learning

定理 2. (Soft Q-iteration). 如果Q_{soft}与V_{soft} 有界，且$\int_{\mathcal{A}} \exp\left(\frac{1}{\alpha} Q^*_{Soft}(s_t, a')\right)$积分对所有$s_t$有界，且$Q^*_{\mathrm{soft}} < \infty$存在。则以下不动点迭代收敛于$Q^*_{\mathrm{soft}}, V^*_{Soft}$

$$Q_{\mathrm{soft}}\,(\mathbf{s}_t, \mathbf{a}_t) \leftarrow r_t + \gamma\, \mathbb{E}_{\mathbf{s}_{t+1} \sim p_{\mathbf{s}}}[V_{\mathrm{soft}}\,(\mathbf{s}_{t+1})], \forall \mathbf{s}_t, \mathbf{a}_t$$
$$V_{\mathrm{soft}}\,(\mathbf{s}_t) \leftarrow \alpha \log \int_{\mathcal{A}} \exp\left(\frac{1}{\alpha}\, Q_{\mathrm{soft}}\,(\mathbf{s}_t, \mathbf{a}')\right) d\mathbf{a}', \forall \mathbf{s}_t$$

3.1 Soft Q learning

我们使用神经网络来参数化soft Q function,并且引入重要性采样改写Soft Value function的定义

$$V^\theta_{\mathrm{soft}}(\mathbf{s}_t) = \alpha \log \mathbb{E}_{q_{\mathbf{a}'}}\left[\frac{\exp\left(\frac{1}{\alpha}\, Q^\theta_{\mathrm{soft}}(\mathbf{s}_t, \mathbf{a}')\right)}{q_{\mathbf{a}'}(\mathbf{a}')}\right]$$

其中$q_{a'}$是action space上的任意分布， 进而的到 soft Q function

$$Q^{\bar{\theta}}_{\mathrm{soft}}(s_t, a_t) = r_t + \gamma \mathbb{E}_{s_{t+1} \sim p_s}[V^{\bar{\theta}}_{\mathrm{soft}}(s_{t+1})]$$

这里用了double Q的想法，delay update Q

那么我们可以将soft Q-iteration 表示为以下minimization问题

$$J_Q(\theta) = \mathbb{E}_{s_t \sim q_{g_t}, a_t \sim q_{g_t}}\left[\frac{1}{2}\,(\hat{Q}^{\bar{\theta}}_{soft}(s_t, a_t) - Q^\theta_{soft}(s_t, a_t))^2\right]$$

3.2 采样Q值

对于离散动作空间我们可以直接通过状态输入得到每个动作的Q， 但是对于连续动作空间就不行了。这个时候我们需要一个采样器。

我们利用一个神经网络$a_t = f^\phi(\xi; s_t)$来选择动作，其中ξ是一个高斯噪声。为了使我们的策略分布与当前我们认为的最佳分布接近，我们利用KL散度来约束两者尽量接近

$$J_\pi(\phi; s_t) = D_{KL}\left(\pi^\phi\,(\cdot|\, s_t) \| \exp\left(\frac{1}{\alpha}\,(Q^\theta_{soft}(s_t, \cdot) - V^\theta_{soft})\right)\right)$$

这个KL散度可以利用stein variational gradient descent计算,下降方向为

$$\Delta\, f^\phi(\cdot; \mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi^\phi}[\kappa(\mathbf{a}_t, f^\phi(\cdot; \mathbf{s}_t))\, \nabla_{\mathbf{a}'}\, Q^\theta_{\mathrm{soft}}\,(\mathbf{s}_t, \mathbf{a}')|_{\mathbf{a}' = \mathbf{a}_t}$$
$$+ \alpha\, \nabla_{\mathbf{a}'}\, \kappa(\mathbf{a}', f^\phi(\cdot; \mathbf{s}_t))|_{\mathbf{a}' = \mathbf{a}_t}]$$

更新值为

$$\frac{\partial J_\pi(\phi;\mathbf{s}_t)}{\partial\phi} \propto \mathbb{E}_\xi\left[\Delta f^\phi(\xi;\mathbf{s}_t)\frac{\partial f^\phi(\xi;\mathbf{s}_t)}{\partial\phi}\right]$$

3.3 算法

Algorithm 1 Soft Q-learning

$\theta, \phi \sim$ some initialization distributions.
Assign target parameters: $\bar{\theta} \leftarrow \theta$, $\bar{\phi} \leftarrow \phi$.
$\mathcal{D} \leftarrow$ empty replay memory.

for each epoch **do**
  for each t **do**
    Collect experience
      Sample an action for $\mathbf{s}_t$ using f^ϕ:
        $\mathbf{a}_t \leftarrow f^\phi(\xi;\mathbf{s}_t)$ where $\xi \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.
      Sample next state from the environment:
        $\mathbf{s}_{t+1} \sim p_{\mathbf{s}}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$.
      Save the new experience in the replay memory:
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$.
    Sample a minibatch from the replay memory
      $\{(\mathbf{s}_t^{(i)}, \mathbf{a}_t^{(i)}, r_t^{(i)}, \mathbf{s}_{t+1}^{(i)})\}_{i=0}^N \sim \mathcal{D}$.
    Update the soft Q-function parameters
      Sample $\{\mathbf{a}^{(i,j)}\}_{j=0}^M \sim q_{\mathbf{a}'}$ for each $\mathbf{s}_{t+1}^{(i)}$.
      Compute empirical soft values $\hat{V}_{\mathrm{soft}}^{\bar{\theta}}(\mathbf{s}_{t+1}^{(i)})$ in (10).
      Compute empirical gradient $\hat{\nabla}_\theta J_Q$ of (11).
      Update θ according to $\hat{\nabla}_\theta J_Q$ using ADAM.
    Update policy
      Sample $\{\xi^{(i,j)}\}_{j=0}^M \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for each $\mathbf{s}_t^{(i)}$.
      Compute actions $\mathbf{a}_t^{(i,j)} = f^\phi(\xi^{(i,j)}, \mathbf{s}_t^{(i)})$.
      Compute Δf^ϕ using empirical estimate of (13).
      Compute empiricial estimate of (14): $\hat{\nabla}_\phi J_\pi$.
      Update ϕ according to $\hat{\nabla}_\phi J_\pi$ using ADAM.
  end for
  if epoch *mod* update_interval $= 0$ **then**
    Update target parameters: $\bar{\theta} \leftarrow \theta$, $\bar{\phi} \leftarrow \phi$.
  end if
end for