Multi-Task Offline Reinforcement Learning with Heuristic Conservative Soft Actor-Critic

Project of AI3601 Reinforcement Learning, 2024 Spring, SJTU

Group 1: Xiangyuan Xue[*†], Kailing Wang[*], Jiazi Bu[*]

June 4, 2024

[*]Equal contribution [†]Project Leader & Slides Presenter

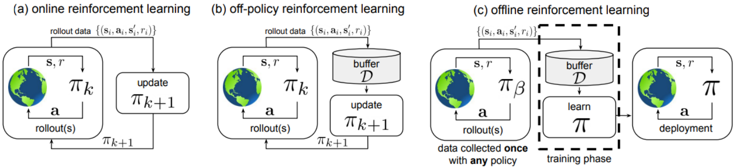
Introduction

Motivation

- Online RL requires the agent to frequently interact with the environment to gather samples, but such interaction can be extremely expensive.
- Collecting real-world samples can hardly be parallelized, which makes the process remarkably slow.
- The agent may encounter catastrophic failures while training, which is not affordable in some scenarios.
- In the era of large models, scaling up and generalizing the agent to various tasks is promising.

Introduction

Problem Definition



- Offline RL requires the agent to learn from a fixed dataset without interacting with the environment.
- Multi-task RL requires the agent to learn a satisfactory policy that can generalize to various tasks.
- We aim to combine offline RL with multi-task RL.

Challenges

- In the offline setting, bootstrap methods unavoidably introduce distribution shifts, leading to overestimation of Q-values, which makes the traditional algorithms fail to learn a satisfactory policy.
- Most of the proposed multi-task RL methods are designed for the online setting, so they will suffer from the same issues when combined with offline RL and fail to achieve reliable performance.

Method

Method

Preliminaries

- Markov decision process (MDP): $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$
- Bellman equation:

$$V^*(s) = \max_{\pi} \mathbb{E}_{a \sim \pi(\cdot|s)} \left[Q^\pi(s, a) \right]$$

$$Q^*(s, a) = R(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\max_{a'} Q^*(s', a') \right]$$

- Offline dataset: $\mathcal{D} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^{N}$
- Multi-task dataset: $\mathcal{D} = \cup_{i=1}^{M} \mathcal{D}_i$

Method

Soft Actor-Critic (SAC)

- SAC maximizes the entropy of the policy to avoid suboptimal policies. The objective function is:

$$J(\pi) = \sum_{t=0}^{\infty} \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} \left[R(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t)) \right]$$

- The corresponding modified Bellman operator is:

$$\mathcal{T}^\pi Q(s, a) = R(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\mathbb{E}_{a' \sim \pi(\cdot|s')} \left[Q(s', a') - \alpha \log \pi(a'|s') \right] \right]$$

- Update policy by minimizing the KL divergence between the current policy and the softmax of Q-values:

$$\pi'(\cdot|s) = \arg\min_{\mu} D_{\mathrm{KL}} \left(\mu(\cdot|s) \,\middle\|\, \frac{\exp(Q^\pi(s, \cdot))}{Z^\pi(s)} \right)$$

Method

Soft Actor-Critic (SAC)

- Reparameterization trick is required for Gaussian policy.
- Two Q-networks are used to avoid the overestimation of Q-values, where soft target updates are applied.
- Automating entropy adjustment for maximum entropy:

$$\max_{\pi} \sum_{t=0}^{\infty} \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} \left[R(s_t, a_t) \right]$$
$$\text{s.t. } \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} \left[-\log \pi(a_t | s_t) \right] \geq \mathcal{H}_0, \forall t$$

- Solve the optimization problem by Lagrange duality:

$$\alpha_t^* = \arg\min_{a_t} \mathbb{E}_{a_t \sim \pi(\cdot|s_t)} \left[-\alpha_t \log \pi(a_t|s_t) - \alpha_t \mathcal{H}_0 \right]$$

Conservative Q-Learning (CQL)

- The key idea of CQL is to discourage the out-of-distribution Q-values while encouraging the Q-values in the dataset.
- The original form of CQL is formulated as:

$$\min_{Q} \max_{\mu} \alpha \left(\mathbb{E}_{s\sim\mathcal{D},a\sim\mu(\cdot|s)}\left[Q(s,a)\right] - \mathbb{E}_{s\sim\mathcal{D},a\sim\pi_{\beta}(\cdot|s)}\left[Q(s,a)\right]\right) \\ + \frac{1}{2}\mathbb{E}_{(s,a,s')\sim\mathcal{D}}\left[\left(Q(s,a) - \mathcal{B}^{\pi_k}Q^k(s,a)\right)^2\right] + \mathcal{R}(\mu)$$

- Note that α is the temperature parameter, $\mathcal{B}$ is the Bellman operator, and $\mathcal{R}$ is a general regularizer.

Conservative Q-Learning (CQL)

- If $\mathcal{R}$ is the KL divergence between the current policy and the uniform distribution, it can be rearranged into:

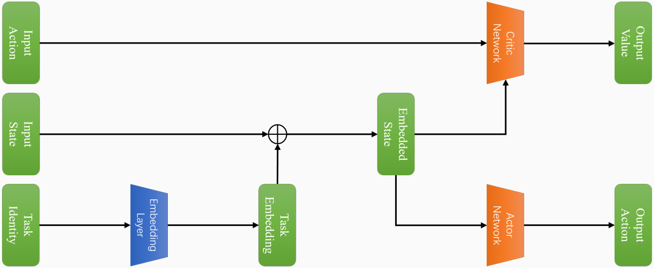
$$\min_{Q} \alpha \mathbb{E}_{s\sim\mathcal{D}} \left[\log \sum\nolimits_a \exp(Q(s,a)) - \mathbb{E}_{a\sim\pi_\beta(\cdot|s)} \left[Q(s,a) \right] \right] + \frac{1}{2} \mathbb{E}_{(s,a,s')\sim\mathcal{D}} \left[\left(Q(s,a) - \mathcal{B}^{\pi_k} Q^k(s,a) \right)^2 \right]$$

- In continuous action spaces, the log-sum-exp operation should be approximated by importance sampling:

$$\begin{aligned} \log \sum\nolimits_a \exp(Q(s,a)) &= \log \left(\frac{1}{2} \sum\nolimits_a \exp(Q(s,a)) + \frac{1}{2} \sum\nolimits_a \exp(Q(s,a)) \right) \\ &= \log \left(\frac{1}{2} \mathbb{E}_{a\sim U(\mathcal{A})} \left[\frac{\exp(Q(s,a))}{U(a)} \right] + \frac{1}{2} \mathbb{E}_{a\sim\pi(\cdot|s)} \left[\frac{\exp(Q(s,a))}{\pi(a|s)} \right] \right) \end{aligned}$$

Method

Task Embedding



- We assign a unique identity k to each task. The identity will be converted into the task embedding e and added onto the state s as the task-aware state z.

Method

Heuristic Relabeling

- We share the transitions between tasks by relabeling dataset, where out-of-distribution samples are discarded.
- The threshold is defined by conservative Q-values:

$$\tau(\mathcal{D}_i) = \{Q_i(s,a) | (s,a) \in \mathcal{D}_i\}_{k\%}$$

- We relabel the transition only if its conservative Q-value is no smaller than the threshold, namely:

$$\Delta(s,a) = Q_i(s,a) - \tau(\mathcal{D}_i) \geq 0$$

Heuristic Relabeling

Algorithm 1 Heuristic Relabeling

Require: The multi-task offline dataset $\mathcal{D} = \cup_{i=1}^{M} \mathcal{D}_i$.

Initialize the active dataset $\mathcal{D}^* \leftarrow \mathcal{D}$.
for $k \leftarrow 1, 2, \cdots, L$ **do**
  Train the agent using samples from $\mathcal{D}^*$ to obtain the latest critic network Q.
  for $i \leftarrow 1, 2, \cdots, M$ **do**
    Initialize the active dataset $\mathcal{D}_i^* \leftarrow \mathcal{D}_i$.
    Compute the relabeling threshold $\tau(\mathcal{D}_i) \leftarrow \{Q(s,a) | (s,a) \in \mathcal{D}_i\}_{k\%}$.
    for $j \leftarrow 1, 2, \cdots, M$ where $j \neq i$ **do**
      Extend dataset by relabeling $\mathcal{D}_i^* \leftarrow \mathcal{D}_i^* \cup \{(s,a,r,s') \in \mathcal{D}_j | Q_i(s,a) \geq \tau(\mathcal{D}_i)\}$.
    end for
  end for
  Merge the active dataset $\mathcal{D}^* \leftarrow \cup_{i=1}^{M} \mathcal{D}_i^*$.
end for

- The algorithm is as the above pseudocode.

Experiments

Experiment Setup

- Behavioral cloning (BC): weak baseline.
- Conservative soft actor-critic (CSAC): strong baseline.
- Multi-task conservative soft actor-critic (MTCSAC): Apply task embedding based on CSAC.
- Heuristic conservative soft actor-critic (HCSAC): Apply heuristic relabeling based on MTCSAC.
- All the critic and actor networks are implemented as MLPs with 3 hidden layers and 1024 hidden units.

Experiment Setup

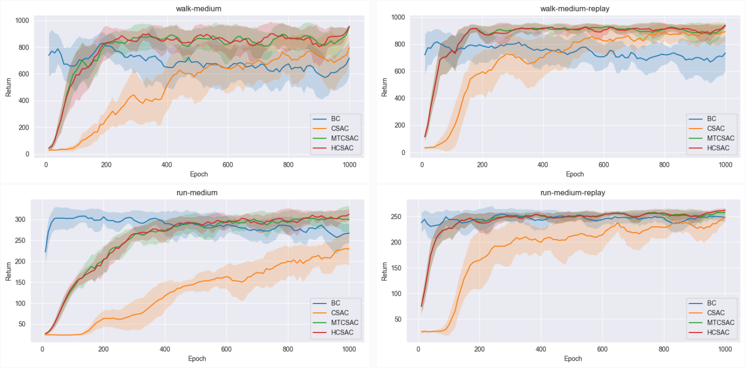
- The experiments are divided into two groups (i.e. `medium` and `medium-replay`) according to the dataset.
- In each group, BC and CSAC are trained on the two tasks (i.e. `walk` and `run`) separately, while MTCSAC and HCSAC are trained on the union of the datasets for the two tasks.
- Train 1000 epochs with a batch size of 1024. Online evaluation is conducted every 10 epochs. Learning rate is set to 3×10^{-4} for the critic and 1×10^{-4} for the actor.

Experiment Setup

- The soft target update rate is set to 0.01.
- The discount factor is set to 0.99.
- The weight of the CQL loss term is set to 10.0.
- We sample 10 actions each from uniform distribution and policy distribution to estimate the log-sum-exp term.
- For HCSAC, heuristic relabeling is conducted every 100 epochs, where the quantile is set to 5%.

Experiments

Performance Evaluation



- Our methods perform better in terms of training curves.

Performance Evaluation

- MTCSAC and HCSAC achieve higher average returns with lower standard deviations compared to BC and CSAC.
- Task embedding and heuristic relabeling in general bring superior performance and improve the robustness.

Dataset	Task	BC	CSAC	MTCSAC	HCSAC
medium	walk	551.52 ± 125.24	653.57 ± 233.83	870.06 ± 92.99	$\mathbf{907.38 \pm 67.39}$
	run	$\mathbf{317.26 \pm 9.46}$	213.67 ± 49.56	306.16 ± 20.43	308.38 ± 7.64
	average	434.39 ± 67.35	433.62 ± 141.70	588.11 ± 56.71	$\mathbf{607.88 \pm 37.52}$
replay	walk	829.63 ± 74.64	812.80 ± 146.82	$\mathbf{931.37 \pm 9.21}$	927.43 ± 14.76
	run	244.80 ± 33.35	234.81 ± 22.11	254.31 ± 11.60	$\mathbf{256.79 \pm 10.13}$
	average	537.22 ± 53.00	523.81 ± 84.47	$\mathbf{592.84 \pm 10.41}$	592.11 ± 12.44

Conclusion

Contribution

- We propose heuristic conservative soft actor-critic to solve the offline multi-task reinforcement learning problem.
- We conduct a series of experiments on the MuJoCo benchmark to verify the effectiveness of our method.
- The results show that task embedding and heuristic relabeling in general bring superior performance and improve the robustness of the agent.

Conclusion

Discussion

- The tasks `walk` and `run` both rely on low-level locomotion skills. Sharing the policy may be unnecessary because we expect the agent can learn from a large number of tasks to acquire those essential skills in real-world scenarios.
- In MuJoCo, the vectorized observations already contain all the information needed for robotic control, so the multi-task learning methods cannot be fully exploited.
- Our methods may work even better in more complex environments, where the agent can utilize diverse transitions to learn better representations.

Thank You!