Paper Critique

Anirud N, CE21B014

Course: DA7400, Fall 2024, IITM
Paper: [MOReL: Model-Based Offline Reinforcement Learning]
Date: [9 August 2024]
Make sure your critique Address these following points:
1. The problem the paper is trying to address
2. Key contributions of the paper
3. Proposed algorithm/framework
4. How the proposed algorithm addressed the described problem
Note: Be concise with your explanations. Unnecessary verbosity will be penalized. Please don't exceed 2 pages.

---

1 Problem Statement

Offline RL uses already available data, so it works on data-driven learning of policies. This emerged especially due to demand for RL, where exploration is not always possible or is possible under proper supervision, such as in medical fields, for safety concerns.
Offline RL faces unique challenges, such as deviation between the state visitation distribution between the candidate and logging policies. This leads to a **distribution shift**. Function approximation errors added to this shift lead to poor results.
The paper aims to address this main issue of offline RL.

2 Key Contributions

The main contribution is the development of MOReL: Model-Based Offline Reinforcement Learning (nearly min-max optimal). It uses P-MDPs (Pessimistic MDPs) that lower-bound the actual environment. This helps achieve the best performance under worst-case scenarios. P-MDPs are partitioned into known and unknown states—so a heavy penalty is given for visiting unknown states. The paper establishes the upper bound for MOReL and the lower bound for any offline RL algorithm. The paper finally evaluates and compares the algorithm with other benchmarks and existing algorithms.

3 Proposed Algorithm/Framework

The paper does not assume that the logging policies are known. Given the dataset $\mathcal{D}$, the goal is to produce π_{out} with minimal suboptimality. The algorithm is as follows in the next page - Algorithm 1.

4 Advantages and Conclusions of the Algorithm

- In terms of results, the proposed algorithm achieved SOTA in 12 /20 combinations of the data.
- The paper observed that the agent performed better when it learned from the dataset generated by partially trained model over a random model generated dataset
- The value of a policy in PMDP cannot exceed the actual value - preventing over-estimation and model exploitation.

Algorithm 1 Model-Based Offline Reinforcement Learning

1: Given dataset $\mathcal{D}$

2: Construct approximate dynamic models $\hat{P} : S \times A \to S$ using $\mathcal{D}$.

3: Gaussian dynamic model is used to learn the dynamic model:
$\hat{P}(\cdot|s,a) \equiv \mathcal{N}\left(f_\phi(s,a), \Sigma\right)$, with mean $f_\phi(s,a) = s + \sigma_\Delta \cdot \text{MLP}_\phi\left(\frac{s-\mu_s}{\sigma_s}, \frac{a-\mu_a}{\sigma_a}\right)$,
where $\mu_s, \sigma_s, \mu_a, \sigma_a$ are the mean and standard deviations of states/actions in $\mathcal{D}$; σ_Δ is the standard deviation of state differences, i.e., $\Delta = s' - s$, $(s, s') \in \mathcal{D}$. MLE with mini-batch Stochastic optimisation is used to find the MLP parameters.

4: Partition the dataset into Known and Unknown.
Partition the state-action space into **Known** and **Unknown**. The method learns multiple models $\{f_{\phi_1}, f_{\phi_2}, \ldots\}$ with different initializations of weights and biases. Define discrepancy as:
$\text{disc}(s,a) = \max_{i,j} \left\| f_{\phi_i}(s,a) - f_{\phi_j}(s,a) \right\|_2$,
where f_{ϕ_i} and f_{ϕ_j} are members of the ensemble.

$$U_{\text{practical}}(s,a) = \begin{cases} \text{FALSE (i.e., Known)} & \text{if } \text{disc}(s,a) \leq \text{threshold} \\ \text{TRUE (i.e., Unknown)} & \text{if } \text{disc}(s,a) > \text{threshold} \end{cases} \tag{1}$$

5: Construct P-MDP $\mathcal{M}_p = \{S \cup \text{HALT}, A, r_p, \hat{P}_p, \hat{\rho}_0, \gamma\}$.
This PMDP must penalize policies that have actions or state pairs outside the explored set to prevent distribution shifts.
The P-MDP is described by := $\{S \cup \text{HALT}, A, r_p, \hat{P}_p, \hat{\rho}_0, \gamma\}$. HALT is an additional absorbing state we introduce into the state space of . $\hat{\rho}_0$ is the initial state distribution learned from the dataset . The modified reward and transition dynamics are given by:

$$\hat{P}_p(s'|s,a) = \begin{cases} \delta(s' = \text{HALT}) & \text{if } U^\alpha(s,a) = \text{TRUE} \\ & \text{or } \ s = \text{HALT} \\ \hat{P}(s'|s,a) & \text{otherwise} \end{cases} \qquad r_p(s,a) = \begin{cases} -k & s = \text{HALT} \\ r(s,a) & \text{otherwise} \end{cases} \tag{2}$$

$\delta(s' = \text{HALT})$ is the Dirac delta function, which forces the MDP to transition to the absorbing state HALT. For unknown state-action pairs, we use a reward of $-k$, while all known state-actions receive the same reward as in the environment. The P-MDP heavily punishes policies that visit unknown states, thereby providing a safeguard against distribution shift and model exploitation.

6: $\pi_{\text{out}} \leftarrow \text{PLANNER}(\mathcal{M}_p = \hat{\pi}_b)$

7: **Return** π_{out}

- The suboptimal bound given by this algorithm works best so far in worst-case scenarios
- The performace of P-MDP is close to the performance in the actual environment and that P-MDP gives a monotonous learning curve as expected due ti monotonic improvement theory from policy gradient methods.
- This algorithm ensures that the agent does not drift to unknown states when the agent cannot predict accurately using the static dataset.